Sapienza Università di Roma

Dipartimento di Informatica e Sistemistica A. Ruberti

Dottorato di Ricerca in Ingegneria Informatica

XIX Ciclo

Modeling and Reasoning about Semantic e-services in Cooperative Information Systems

Luigi Dragone

Anno Accademico 2006/2007

ADDRESS:	Luigi Dragone
	CM Sistemi S.p.A.
	via Simone Martini, 126
	I-00142, Roma, Italy
E-MAIL:	`luigi.dragone@gruppocm.it`
WWW:	`http://www.gruppocm.it/`

Abstract

We propose a new framework for the analysis of functional properties of e-services supporting the development of cooperative information systems.

The framework aims at extending and integrating different approaches from planning, automatic verification and database theory, providing both a rich domain specification and a suitable operational semantics of the e-service contract, on which we define functional consistency and adequacy properties. From the modeling side, the proposed approach allows for specifying complex e-services based on the IOPE paradigm (Input, Output, Preconditions, and Effects), in which the static properties of the modeled system are specified using a Description Logic knowledge base, as assumed in Semantic Web applications. Moreover, our framework enforces a minimal-change semantics for the axiomatization of the update operator, supporting also for the specification of conditional and non-deterministic e-service behaviors. It also includes the ability to reason about update repairing strategies w.r.t. the domain constraints, thus allowing for incomplete service specification.

On this foundation, we are able to formally define several interesting properties of services, involving accessibility, validity, and repairability of services. From the computational side, we prove that, while checking the above formal properties is undecidable in general expressive settings, in our framework such properties are decidable. We prove this result by reducing the above tasks to reasoning in the two-variable function-free fragment of first-order logic.

Moreover, we give special regards to service-specific features, such as, for example, the emphasis on black-box encapsulation, the characterization of the scope of service applicability (e.g., served user community), and the concept of functional similarity, abstracting both from the actual service provider and the concrete interface specification.

The proposed approach is especially suited for the design of integration solutions among mutually autonomous organizations that agree upon the domain specification language, e.g., e-government and business-to-business scenarios. It enforces knowledge sharing about available functionalities and enables the implementation of dynamic binding mechanisms for failure recovery and execution customization, and the aggregation of functionalities by interaction mediators.

Acknowledgments

After a so long adventure, there are lots of people I would like to thank...

Firstly, I would like to thank my supervisor, Prof. Riccardo Rosati. I could not have imagined having a better advisor and mentor for my PhD, he was promptly available for me and without his expert guidance and knowledge I would never have finished.

I wish to thank to faculty members of this department and to teachers that have held PhD courses which I have attended, as well as the program coordinator, Prof. Maurizio Lenzerini, and teachers of the ESSLLI 2004 summer school. A big thank also goes to the faculty and staff of the Dipartimento di Informatica e Sistemistica.

Thus I gladly express my gratitude to my thesis committee members and to external reviewers for their observations and concerns. My thanks go out also to anonymous reviewers that have refereed my papers and that have provided me very useful insights and suggestions to improve my research.

I would like to express my gratitude to my colleagues and to my friends that have supported (and *sopportato*!) me during these years. In particular, I must mention Fabio Vernacotola, Luca Cianfarani, and Mario Escalona for their invaluable help.

I am forever indebted to my parents and to my brother Paolo for their understanding, endless patience and encouragement when it was most required.

Finally, this dissertation would not have been possible without the endorsement of Paolo Naggar. He has given me the fantastic opportunity of earning the PhD working on research and development projects at CM Sistemi. I only hope to have achieved results that come up to the expectation: applying formal theories to software engineering issues is somewhat puzzling, but it is also an exciting challenge! For the same reason, I'd like to thank also Andrea Rastellini and Bruno Barthe.

Of course, despite all the assistance provided by Prof. Rosati and others, I alone remain responsible for the content of the following, including any errors or omissions which may unwittingly remain, as for forgetting to mention someone.

Rome,
december 2007

To my parents

Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

If, says Zeno, everything is either at rest or moving when it occupies a space equal to itself, while the object moved is in the instant, the moving arrow is unmoved.

Aristotle, *Physics* VI:9, 239b5

1.1 Problem Studied

The *Service-Oriented Computing* (SOC) paradigm [ACKM04] has gained in recent years a lot of interest from the industrial and scientific communities in the field of information technology, in general, and in the design and implementation of information systems, in particular. This paradigm is based upon the metaphor of service (or *e-service*) as a mean to totally encapsulate software application features, in order to make them openly available to highly decoupled clients, implementing a flexible machine-to-machine interaction and building a complex network of dynamically interacting actors (*service providers* and *requesters*). Such a kind of network is the milieu on which new applications are built by assembling/composing available services (*service orchestration* and *synthesis*) and users look for services suitable to their needs (*service discovery* and *dynamic binding*)[1].

A service provided in such a way should not only hide implementation details, but also aim at offering a higher level of abstraction to the client, closer to the end-user's perception in terms of granularity of the system representation. In

[1]The most relevant implementation in this field is, of course, represented by the so-called *web-service* (WS) communication protocol stack: it is an initiative mainly managed by the World Wide Web Consortium (W3C) and many software vendors ([WS, BHM+04]). The resulting specifications are essentially based on the XML language ([BPSM+06]) and associated standards and Internet technologies, but the e-service metaphor can be easily adopted using different middleware like CORBA ([OMG04]).

other words, an e-service should be a software system that offers functionalities oriented to the management of “real-world” resources (even using computer-based tools), instead of a remotely accessible component that manages computing resources (e.g., database, documents, communication links, etc.). On the other hand, there is a strong requirement on the well-foundedness of the service contract specification between involved actors, like in other composition-oriented software frameworks.

This paradigm covers many application scenarios coming out from the wide availability of networking and distributed computing technologies, due to the success of the Internet and the World Wide Web. In particular, it easily addresses integrated and cooperative information systems in the area of *e-business* and *e-government*, but it is also well suited for mobile network users (*m-service*).

In order to adequately support the design and the implementation of a service-oriented architecture, we need to devise formal tools or, more precisely, to tune existing ones to cope with peculiar modeling issues of the e-service paradigm.

Several approaches proposed in literature model e-services by reducing them to their specific primitive constructs, in order to make it feasible to apply the various developed techniques: however, such approaches often ignore the characteristic features of e-services. In particular, they generally do not consider that an e-service can be described at:

an “intensional” level where e-services are characterized on the basis of their functionalities;

an “extensional” level where e-services are characterized on the basis of their applicability contexts (i.e., the user population that can access to the provided service).

If the former is the typical aspect addressed in the field of computational semantics and system/protocol verification, the extensional characterization is a distinguishing feature of such a kind of technologies.

On the industrial side, we have noticed that current proposals are mainly focused on non-functional aspects, but such approaches lead to authoritative constraints that prevent from a fully exploitation of these solutions, raising a market barrier. In fact, they are well-suitable only for such (large) subjects having the ability of imposing their own models and protocols to the community (*de facto* or *de jure*). This fact prevents other operators from investing into original solutions, since they are not enough isolated w.r.t. external factors. Moreover, the lack of a precise functional characterization, especially in a highly distributed and autonomous environment, hides the equivalence/similarity relation among elements, inducing to a useless proliferation of artifacts. It results into a high level of complexity in the development and management of the system.

Consequently, we aim to address the implementation of system integration solutions, especially suited for the fields of e-government and virtual-enterprise, devising a model that allows the definition of e-service properties and capabilities in a complex domain, having a well-founded semantic model characterizing the enactment of an e-service.

Such a kind of framework should be abstract enough in order to capture only functional aspects, leaving out details deriving from implementation and communication issues, but taking into account the organizational elements related to the extensional characterization.

In particular we are addressing a general problem in the characterization and analysis of evolving complex structures: in other words, as we will show in the following chapter, we need to cope with a dynamic system which state and transition representation is often very complex. From a general perspective this is a classical issue addressed in many fields from the database update to the knowledge-based planning, and many results have been already applied to the development of service-oriented applications, but, nowadays, we observe the lack of a suitable setting allowing a system designer to analyze and verify some foundational and useful properties.

In fact, the adoption of a highly expressive language as the *First-Order Predicate Logic* (FOL) for modeling dynamic systems easily leads to face very hard (or even unsolvable) problems in terms of automatic verification of formal properties of e-services. On the other hand, less expressive formalisms, like for example the ones based on propositional logic, are too weak to model successfully real-world e-service scenarios. So, in order to devise a more expressive language for e-service functional modeling, while preserving the computability of the associated reasoning problems, we turn our attention to the family of *Description Logics*[2] (DL). Such a class of logics (which are mainly fragments of first-order logic) has been explicitly defined with the main aim of constituting an optimal trade-off between representational abilities and computational properties of reasoning ([Var96]). We basically adopt expressive description logic languages (e.g., $\mathcal{ALCQI}$ and $\mathcal{ALCQIO}$) to describe the static world properties, but since we need to cope with some dynamic, and non-deterministic, features like updates, we need to extend the language adequately in order to formulate our problems in terms of computable reasoning tasks, which are mainly focused on static world descriptions. In particular, our proposal relies upon the decidable fragment of first-order logic $\mathcal{C}^2$: function-free first-order predicate logic with at most two variables and counting quantifiers.

In order to adequately support the system design and implementation with semantically founded tools, we need to address the verification of several properties (from correctness to equivalence) in a dynamic setting in the presence of complex domain languages and some source of incompleteness. These problems are, in general, intractable from a formal perspective (i.e., undecidable), despite removing some constraint can lead to an effectively addressable issue (e.g., complex static settings with incomplete specifications), but, at the same time, not completely satisfactory: hence we need to investigate the hypothesis of devising a sort of eventually approximated solution that is able to cope with the whole user requirement set.

1.2 Contribution

The main goal of this work is to design an e-service modeling framework that allows for analyzing functional properties at a high level of abstraction, based on a formal semantic characterization, in order to devise development and execution support tools in the construction of service-oriented solutions.

[2]See [BCM+03] for an introduction.

More specifically, in this work we present the following main contributions:

1. the analysis of semantic based properties of e-services, which turn out as relevant in the development of a service-oriented solution in the field of e-government and cooperative information systems, with a special attention to correctness and equivalence analysis;

2. from the semantic side, the formalization of the dynamic model underlying a suitable minimal-change semantics of e-services according to an operational paradigm, also including side-effects. Such a formalization is based on a decidable fragment of first-order logic;

3. from the computational side, the results on the decidability and complexity of the automatic verification of the above properties, even in the presence of incomplete specifications;

4. from the logic side, we present some extensions of classical expressive Description Logics in order to deal with complex role and dynamic constraints, introducing a conservative extension of a knowledge-base formalization framework in the presence of updates.

The modeling approach proposed is able to deal with many dynamic features of e-services, since it has been devised to capture as much as possible the properties of an update operator acting on a complex world description. In fact, many approaches proposed in the literature, as we show in the following, are focused on the semantic characterization of e-services, but, they lack a suitable operational semantics, even though they rely on powerful modeling languages, as the ones used in the *Semantic Web*. This limitation shows up in the case of many DL-based frameworks, currently adopted as foundation for semantic service inventory and discovery (e.g., WSML [dBFK$^+$05], OWL-S [MPM$^+$04]). In other words, while such approaches adopt expressive logics as formal foundation, they usually apply a strong *reification*, so the interpretation structures of such formalizations are indeed mapped on the possible activation bindings. Roughly speaking, many approaches aim to logically describe the service invocation, rather than the service execution. These static approaches result not enough expressive for the analysis of service enactment properties, since they ignore many aspects of the update operations carried out by a service, that are *ipso-facto* a dynamic concern.

In this respect, we have devised a formal toolkit that in our aims should support the system integrator performing typical design tasks, according to a formal system validation paradigm. In other words, we have deeply analyzed the formal correctness properties of a semantically annotated e-service, in terms of its own contract. In our vision, a service is liable if it is always able to honor its own obligation w.r.t. the client without violating any domain constraint, considering both static and dynamic rules (the former ones restrict the class of legal system states, while the latter ones limit the class of admissible system state transitions), assuming that the client itself is operating correctly. Despite this is a very common problem addressed in various investigation fields, no completely satisfactory solutions are available, since it is very hard to adequately tune the expressive modeling power w.r.t. the tractability of problem instances. Moreover, in this scenario, some additional issues related to the incompleteness of information or specification generally arise. So, the approach devised is able to keep into account also an update repair strategy, and, differently from similar proposals, it is decidable even in the presence of arbitrary domain specifications (in the adopted language).

The analysis of formal consistency of a service specification has been employed to build a more complex framework that is able to support also other typical activities in the construction of a service-oriented architecture. In particular, we have addressed the analysis of the adequacy of a service to accomplish a given task, expressed by means of a user goal, as tool for semantic matchmaking. Hence, we have analyzed the problem of service equivalence w.r.t. the ability of a service to replace another one: we notice that this property is quite useful in the implementation of a fault-tolerant system. We have also devised an approach to service equivalence that is able to take into account also user constraints.

Besides, we have extended the equivalence or similarity analysis in order to deal also with extensional concerns, that we have remarked as a specific aspect of e-services w.r.t. other component-oriented software approaches. In particular, we are able to compare the capabilities of e-services within a community introducing some degree of abstractness regarding their coverage and, also, to cluster similar services into templates.

The devised framework can express a significant fragment of the current concrete Semantic Web languages employed in the specification of *Semantic Web-Service* (SWS) applications and also in the design of complex information systems as Object-Oriented and Entity-Relationship paradigms.

1.3 Thesis Outline

The rest of the document is organized as follow: in Chapter 2 we start defining our reference scenario and analyzing some interesting related works among the number of approaches proposed to deal with the SOC and SWS. In Chapter 3, once formal tools employed in the rest of the work has been briefly introduced, we provide the basic definitions of the various notions subsequently employed to lay out the framework. In Chapters 4 and 5 we present the axiomatizations and the related results of several kinds of e-services that we are able to model, progressively increasing the specification language expressiveness. In Chapter 6 we point out some interesting properties arising in the construction of a service-oriented computing solution such as matchmaking, discovering, compatibility analysis and so on. In Chapter 7 we devise some language extensions in order to deal with more complex and expressive constraint classes, involving also dynamic aspects, extending results shown so far also to these new languages. Finally, in Chapter 8 we illustrate our future investigation plan.

An extended survey of the state of the art in the field of Service-Oriented Computing and related technologies is reported in appendix, including an index of employed definitions and notations.

CHAPTER 2

Service-Oriented Computing

In this chapter we shortly discuss about the reference computing model developed on the idea of e-service and the more relevant approaches employed in the specification, analysis and implementation of this kind of software system.

We pay a special attention to the modeling aspect of dynamic features in order to characterize a suitable formal semantic and emphasize tools devised to support the system design and integration.

2.1 The e-service computing paradigm

As observed in Chapter 1 there is a lot of interest, both from the research and industry communities, on technologies based on the metaphor of "service", since, from many sides, this paradigm has been pointed out as a sort of foundational tool to deal with the never-solved software integration problem.

It is worth noticing that any technology can be the "silver bullet" in a such complex and heterogeneous scenario, but we can also observe that from proposal devised in this field some interesting solutions and new issues have been raised. Nevertheless, while the easy network accessibility has greatly increased the number and variety of integration challenges and issues (opportunities ?), extending also to small scale development contexts, it is trivial to observe that the integration has represented so far the more critical, in terms of cost, time and quality, aspect in the implementation of ICT solutions. Hence, as many vendors and development communities are now almost committed in the definition of feasible standard integration tools (e.g., languages, protocols, specifications and so on), there is the need of many contributions from the research community to cope with new issues raised (e.g., dependability, trading, on-the-fly agreement).

In fact, in this context many contributions from several research areas, in particular Artificial Intelligence (Knowledge Representation and Planning), Database Theory, Distributed Computing and Formal Validation, have come together leading to innovative interesting approaches to the development and management of complex software applications and large-scale cooperative information system by the integration of loosely coupled elements.

The Service-Oriented Computing can be seen as a computational model ([PG03]) based on the service metaphor as a mean to denote a functional element that can be easily accessed from a suitable environment without any significant limitation deriving from heterogeneity of implementation technology or organizational structure. In this context a software application or an information system can be treated as a community of interacting services that cooperate in order to provide more complex services to other applications or to the end-users. The computation is highly distributed in terms of physical and organizational location of devices.

Some authors, as [RK03, Hof03, Sta03], denote as *e-service* the business model characterized by the selling of, not necessarily related to information and communication technologies (ICT), services by means of ICT infrastructure (e.g., Internet/WWW) as an evolution of the e-commerce model ([LSAS00]), assumed as a new family of factors of production. Literally, it denotes a service delivered/accessed by electronic media.

In [MPC01], the term e-service has also been employed to denote a software system element (e.g., a software component) that can be easily integrated into a more complex system. Generally speaking, an e-service is an information system element (possibly built by different components, depending on the granularity of the analysis) characterized by the following key properties:

- it implements a specific business function or manages a specific business domain object (i.e., *by-function modularization* and *by-object modularization* according to [CLNS97]);
- it is agnostic w.r.t. the implementation technology/architecture, since it realizes a strong and complete encapsulation of one or more (possibly) pre-existing functionalities;
- it can be easily integrated into complex solutions as a basic building block;
- it can be employed to define intra and extra-organizational integration links;
- it exposes its behavior as a kind of atomic task (possibly).

More specifically, we denote as *web services* or *XML web services* ([WS]) the stack of open middleware standards ([BHM+04]) based on the XML language ([BPSM+06]) and other Internet-related protocols (mainly related to HTTP [FGM+99]), independent from a particular vendor, strongly stratified (despite some foundational elements are not completely defined yet), that enable us to build a flexible integration of heterogeneous software systems, implementing a typical case of e-service technology.

As web service is usually denoted also a web or grid (meaning an open computational environment/community) of accessible and customizable services ([ACKM04]), by the virtues both of open flexible interfacing protocols (i.e., XML web service stack) and of the availability of enhanced annotation, search and management tools.

From this point of view, web services can be also considered as the last evolution of the wrapping technologies addressing the encapsulation of end-user services available on the World Wide Web (i.e., HTML form over HTTP) into a software component enabling the replacement of the interacting client (i.e., the web browser) with another software system[1]. According to [HS04], the main goal is to switch from the flexible human-machine interaction implemented by the WWW to a flexible machine-machine interaction[2]. As consequence, there is a strong need not only regarding the adoption of open and "light" interaction protocol (w.r.t. other middleware technologies), but also the re-engineering of application structure and development process.

In particular, the *service-oriented architecture* (SOA) is an approach to the design and implementation of integrated enterprise applications (EAI) that leverages on the e-service model. Generally speaking, it is an evolution of the *component*-based development models or, in other words, of a new kind of *mega-programming* technique ([TBB03, Sta02]). Following this perspective, the development of software application does not only rely on the composition of pre-existent modules (components or services), but, more specifically, on the adoption of high level composition specification language (e.g., *work-flow* definition language) that allows to model in a more natural way business processes supported by the *enterprise information system* (EIS), in order to reduce the conceptual distance between specification and implementation.

Among various application contexts of this kind of technology, and in particular in the field of Cooperative Information Systems (autonomous and heterogeneous information systems that cooperate in order to achieve a common business goal), a special role is covered by the business-to-business (B2B) integration ([KAJR03]). In this case and in its specializations (e.g., Business Partner Integration, Virtual Enterprise, e-government), different organizations (i.e., economic operators) adopt the service paradigm so that their own business process can be encapsulated and they need to expose only the cooperation interfaces to involved business partners. In this setting, the XML web service stack acts as a sort of *lingua franca* among actors for both the specification and the implementation of business protocols/conversations.

Another special class of service is represented by so-called mobile service (*m-service*), that are strongly characterized by aspect related to user/provider location and mobility properties[3]. As for the e-service, the term is also employed to denote the related business model.

In the following, we consider the e-service as an integrable self-describing software component/tool that is aiming to menage "real-world" resources (*web-service as paradigm*), rather than simply an open-protocol integrable software components (*web-service as protocol*).

A deeper analysis on service-oriented applications, the development process and modeling issues is reported in Appendix A.

[1]As matter of facts many of large-scale publicly available web services are nowadays the wrapping version of some existing end-user tools or legacy applications.

[2]Moreover, the ability to integrate heterogeneous software systems on the Web is a key factor in the development of the so-called Web 2.0, that according to its inventors should became a (new) tools for flexible human-human interaction. Please refer to [AKTV07] for a short introduction and an analysis from the Semantic Web perspective.

[3]Moreover, we notice that stressing the concept of mobility of elaboration facilities leads to the *Grid Computing* model. The relationships between these computational paradigms is discussed in [MKF02].

2.2 Formal approaches to service-oriented computing

In this section we will analyze and discuss the most relevant approaches proposed in literature to address problems concerning the modeling, design and implementation of cooperative software systems based on service-oriented computing paradigm.

We pay attention, in particular, to approaches oriented to the analysis of dynamic features and integration issues, possibly keeping into account complex domain specifications. Some other relevant approaches are discussed in Appendix B and C.

2.2.1 Petri Networks

The Petri (or place/transition) network language family ([Pet62]) has been extensively employed to model business process (both intra and extra-organizational) and to analyze their properties in particular adopting the work-flow modeling paradigm ([vdA98]). In fact, this class of languages can easily deal with typical constructs related to distributed/parallel task execution (e.g., *join/fork*, non-deterministic behavior) and there exists a number of algorithms to check formal properties of modeled system in terms of safety and liveness (e.g., presence of *deadlock*, presence of *race condition*, termination) as well expected delivered performance, despite most expressive versions are undecidable ([Jan95, Esp96]).

In particular, has been remarked that a service modeled according to DAML-S and OWL-S methodologies (so-called *service profile* and *service process* specification components) can be instead easily interpreted in terms of process work-flow and axiomatized using Petri Nets. On this foundation, a complex service analysis formal toolkit has been proposed in [NM02]. Roughly speaking, the devised approach translates the service process of the integrated service and the service profiles of involved services into an equivalent Petri Net specification, on which it is possible to formally check composition properties. The analysis reports the computational complexity of property analysis problems w.r.t. the process modeling language constructs allowed and it also provides a simulation based technique to estimate expected performance delivered by the integrated service implementation from declared SLAs of partner services.

A similar proposal is also presented in [Mar03], but in this case a more complex completely decentralized architectural model is adopted and assuming that every service is a module of a more complex cooperative process. A service module is represented as a fragment of Petri Net s.t. its places are classified as internal or external (input/output): the latter are also denoted by the type of exchanged message (received or transmitted). A composition is specified giving a connection structure that accordingly links input and output places of various modules involved. A syntax-based compatibility relation is devised: the compatibility is achieved if linked input and output places can deal the same message type. Also a semantic-based compatibility property is devised and analyzed: a service (i.e., process module) is *usable* in a composition if it does not compromise the liveness and safety process properties.

A quite similar proposal is also discussed in [HB03], but a special attention is payed to the modeling and implementation of typical work-flow patterns ([vdAtHKB04]) as, e.g., *selection* and *discriminator* that are rather relevant in cooperative processes involving many distributed organizations using unreliable communication channels. In order to characterize the behavioral equivalence of services, a specific notion of *bisimulation* has also been devised: this approach is also able to enforce a strong encapsulation level, since every service can expose a public process specification that is bisimilar to the actual implementation (and hence equivalent from the point of view of external actors) without providing additional information about the private part.

2.2.2 Activity and Statechart Diagrams

The *activity diagram* and the *statechart diagram* ([HG97]) are very popular approaches to the formalization of work-flow and business process as well as to the analysis and specification of behavioral requirements (i.e., they are both extensively used in UML [OMG07]). In fact, these languages can easily employed also by business domain analyst without strong technical skills, they support a convenient incremental specification design since they allow for hierarchical decomposition of problems and they also allow to model the organizational impact (i.e., task assignment/responsibility) of depicted process.

Starting from these observations, in [WW97], a general approach to partition a state-chart diagram into orthogonal components, suitable of independent, parallel and distributed execution, has been analyzed. A single central orchestration process has been hence replaced by several distributed coordination processes that preserves the exposed behavior (at least in case of reliable communication links). The key point of the proposed approach is that the decomposition of the given complex process state-chart into orthogonal components: such elements can be executed concurrently and hence can be distributed to different providers, while the mapping between the original state-chart and the one obtained from the integration of orthogonal components preserves the homomorphism property w.r.t. the state-chart composition algebra operators.

Despite the approach has been devised to support business process reengineering projects and distributed work-flow management system named as MENTOR ([WWWD96]), it can be also employed as a tool to split a typical hub-based orchestration process as in BPEL into a more distributed cooperative protocol. In fact, the system SELF-SERV, described in [SBD03], leverages on this principle to deploy the process orchestration deriving from the composition of different atomic services among a community of coordination nodes employing some routing tables that are able to specify also state transition conditions and not only message delivery paths.

This model has also been enriched in [MBM03] in order to deal with problems introduced by the mobility aspect of m-services. The extension is based on the distinction among *business sites*, as the organizations that deliver services, and *execution sites*, denoting places where the computation is actually carried out (i.e., the user smart-phone), and data binding.

Starting from this modeling approach, a planning strategy has been introduced in [ZBD$^+$03]: this solution essentially relies on the enumeration of possible execution plans and the selection of the optimal ones (w.r.t. a given cost criterion) using Mathematical Programming techniques.

A service customization/personalization model based on user characteristics/preferences is presented in [SBM$^+$04]: it relies on the state-chart paradigm to specify the process schema, extending it introducing the possibility to specify also for every task/activity the input/output parameter signature and data dependencies. This approach assumes that information is stored in suitable XML documents, defined using XML Schema, and that process condition/data binding expressions are defined using a language derived from XPath ([CD99]). Also the user profile is represented using an XML document. The orchestration system is defined on this foundation: it is fully distributed and customizable in the sense that it is able to adapt the enactment according to user preferences and to assign the execution of different tasks to various service providers assuming that both the client and the coordinator can not be continuously accessible (i.e., in the case of mobile accessing user). In order to implement the devised solution, the employ of *control tuples*, derived from the Linda language ([ACG86]), has been proposed to represent the process fragment of pertinence of each actor: these tuples can be generated from a template process and inserted into the *tuple space* of each involved service.

2.2.3 Web Service Componentization

The service paradigm is closely related to software modularization and reuse, so that different approached leveraging on component-based design solution has been adapted also to service-oriented application implementation.

In [YPvdH02, Yan03], is discussed the problem of componentization/modularization of web services w.r.t. both the application design and the reuse and extension/customization of implementation elements. A language suitable to model the service composition is introduced, in particular, it allows the designer to specify:

1. the operation execution order;
2. the managed data dependencies;
3. alternative execution paths.

The model is also able to distinguish between service composition completely specified in the design phase, partially specified allowing for dynamic service provider binding and completely dynamic, generated according to the actual enactment. Dependencies among atomic activities can be expressed both in terms of control and data dependencies.

The model can assign a type to each service and assess the service compatibility w.r.t. the similarity/compatibility of respective types.

In order to support the reuse of service specifications and implementations, the model also introduces a formalism, derived from object-oriented languages, that is able to describe a service in terms of messages, operations and process structure[4], but it is also able to model inheritance relations among them.

Relying on this representation language it is possible to describe abstract or partially specified services and also to formalize the reuse and the specialization of different constructs (e.g., message, execution process): the specification language both allows for the extension of inherited class and the overriding of superclass properties.

[4]It is a quite similar in the spirit to the mixture of BPEL, WSDL and XML Schema languages currently employed in many applications.

In [MPC01], using a similar approach, is addressed the problem of analyzing the behavioral compatibility among services into a SOA: each service is defined as a component based on a finite state automaton specified by a state transition diagram annotated with guard conditions. Each transition is also characterized by its input and output message structured types. In this scenario, a service is compatible according to an external observer to each other that is able to produce the same execution logs in terms of observable automaton state transitions and exchanged messages.

2.2.4 Process Algebras

The *process algebras* and the π-calculus ([Mil89, Hoa85, Mil93]) are a large family of formal languages essentially developed to describe communication protocols among concurrently running processes and to verify their properties.

Such kind of formalism relies upon the notion of *behavioral type system* ([IK01]) applied to model concurrent processes: this is an approach similar to classical type systems generally used to describe formal data structures, but it is able to deal with dynamic features of modeled object, providing a synthetic abstract description. Roughly speaking, a process algebra allows to formally describe the behavior of a class of (software) process in terms of the observable input/output message flow on suitable communication channels with other concurrents processes. From this description, several deductive procedures have been defined to check formal properties of these behavioral types and their compositions[5]. The definition of complex/composed process is based on an algebraic notation based on a set of suitable behavioral primitives (e.g, send or receive a message, perform an action). As a classical type system, it is also possible to assert containment relationships and to define generalization hierarchies.

Originally developed to support the design of complex communication protocols, the π-calculus and its variations has been successfully adapted to service-oriented computing, assuming the various interacting actors are a kind of concurrent process. In particular, this model has been adopted in [MB03] as formal base of an extended service interface description language derived from XLANG, since, differently from other approaches that generally need to face complexity and decidability issues, it allows to easily verify various formal properties (in particular of composed services, intended as a community of interacting concurrent processes) providing a way to extend the operation signature with behavioral attributes (e.g., constraints on operation activation sequences). In this scenario, some limitations of this formalism, in particular the difficulty in the description of internal behavior/properties, has turned out as a very interesting feature, since it enables an approach based on a strong encapsulation, distinguishing between “what” a process (i.e., a service provider) does and “how” it works.

In [CRR02], the π-calculus is employed to check formal properties, expressed using Linear Temporal Logic (LTL) ([Pnu77]), of cooperative programs using model checking techniques ([McM93]). In particular, it is introduced and discussed a notion of *sub-typing* based on the possibility of a process to simulate (i.e., replace) another one.

A more specific approach is presented in [KvB03]: a language, named as BPE-calculus, is designed to capture dynamic features of BPEL language using

[5] A process inherits the properties of its own class.

a suitable behavioral type system. This type system has been encoded using a process algebra that, as many other approaches in this field, simply ignores details regarding message data structures. The proposed approach is able to verify interesting properties (i.e., safety, liveness) expressed using LTL model checking. As other approaches, also in this case the process equivalence is defined w.r.t. the bisimulation. But differently from other similar proposals, the effort devolved to reduce the conceptual distance between the formal and the business languages should help the process analysis and re-engineering.

2.2.5 Concurrent Transaction Logic

This is a kind of modal logic expressly introduced in [DKRR98] to model work-flow processes and to check their properties, that stems from an extension of predicate calculus devised to reason about actions and updates ([BK93]).

This language is essentially based on the first-order predicate logic enriched with two additional connectors ($\otimes$ and $|$) used to represent serial and parallel events and two modal operators ($\odot$ and $\diamond$) used to denote the execution of an atomically isolated action or a conditional action. The system state is represented as a relational structure (a ground term database) that is updated during a work-flow enactment. In the CTR logic it is possible to define a special kind of Horn clauses and, consequently, to introduce a logic programming paradigm based on the SLD resolution adapted to this special language, following an approach quite similar to Prolog. In this context, a work-flow is defined by means of a logic programming goal without recursion that updates the state of the system database. Given a work-flow specification accordingly encoded, it is possible to check whether or not it satisfies several temporal/dynamic constraints (e.g., the possibility of a certain event occurrence, the temporal ordering between different events): these techniques can be employed to verify the (absolute) consistency of a specification w.r.t. a set of domain constraints or the consistency of a given enactment and to find an action plan possibly satisfying the constraints. In the case of a non-recursive goal specification s.t. also the *unique event property* holds (i.e., a significant event must occur at most once in any admissible enactment), the CTR verification algorithms are more efficient than corresponding ones developed for standard symbolic model checking ([McM93]), since they do not suffer from the exponential blow-up of system states.

In [DKP+99], an application of the CTR language to a *virtual enterprise* scenario is discussed: it is a particular kind of B2B integration characterized by extemporaneous and contingent agreements among actors. Consequently, in order to implement an effective business process integration using the ICT infrastructure, as well as the availability of suitable middleware, it is also required to define a formal agreement among the cooperating parts and to check whether such a contract is compatible with each partner policy (i.e., it does not break any security/privacy constraints). Adopting the CTR language, this requirement can be implemented modeling the integration process work-flows and checking the consistency w.r.t. the union of constraint sets of every actor.

The language has been further extended in [DKR04], allowing for explicitly modeling also operations performed by the business partner and its own decisions. The resulting language (CTR-S) has been, in fact, designed to address problems concerning the negotiation among potential partners, since each one has its own goals and it is possible that two actors have conflicting ones. This language can be interpreted in term of Game Theory, assuming that a cooperation work-flow is consistently defined if a global strategy exists and it is compatible with agent constraints.

A further extension is presented in [RK07]: the reasoning approach is able to deal with both control and data flows, supporting the service contracting (i.e., checking whether is possible to achieve a goal) and orchestration (i.e., computing an operation scheduling to satisfy a constraint set).

2.2.6 Relational Transducers

Typical e-commerce applications, as many other“services” currently available through the WWW, are a particular class of data-driven web applications, in other words, the navigational structure of the web is strongly tied to the system data model (i.e., the relational schema) and user operations can be described in terms of update of the system database contents (CRUD primitives). This model can be generalized to the case of e-service applications (i.e., service provider) considering the application itself as a complex service exposing several primitive operations: the client uses these operations to access the service.

In order to model and analyze the behavior of this class of software systems, the notion of *relational transducer* has been introduced in [AVFY98]: it is a special kind of automaton s.t. its inputs, outputs and internal state are represented in terms of relational schemas and not as an alphabet as in the case of traditional finite state automata. A relational transducer reads its input as a set of tuples, updates accordingly the state of the internal database (i.e., transaction), writes the output also as a set of tuples and maintains a log of performed operations. The behavior of a relational transducer is defined using a set of rules and it is quite similar to an *active database* (i.e., trigger). In particular, using a special type named as *SPOCUS transducer* (semi-positive outputs and cumulative state), showing nice computational properties, it is possible to define various interesting problems as: the verification of the equivalence/containment between automata, the log validity checking w.r.t. a given specification, the reachability of a system state and the verification of a set of temporal properties (i.e., the satisfiability of domain temporal constraints).

In the general setting, these problems turn to be semi-decidable, however ([Spi00, DSV04]), in the case of SPOCUS transducer, limiting the expressive power introducing some additional restrictions (e.g., number of variables used in the rule specification, maximum size of the input message queue) it is possible to check properties defined using a temporal logic as LTL or Computational-Tree Logic (CTL) ([BAMP81]).

2.2.7 Multi Agent Protocols

In the field of multi-agent software systems the e-service paradigm can be considered as a way to specify the interaction protocol among agents or, in other words, an abstract behavioral specification implemented by means of a software agent ([BHM$^+$04]).

While the traditional XML web service model assimilates the interaction among components/actors as remote procedure calling, in this case it is more convenient to assume that a conversation among autonomous entities, sharing knowledge, resources and abilities, takes place exchanging messages. Generally, comparing agent-related specification languages and XML web service specification standards, it turns out that the latter ones are not enough expressive to adequately model the service/agent behavior.

In [Wal04], a multi-agent protocol, named as MAP, is introduced: it is employed to specify the conversation specification among cooperating services adapting the model defined in [FIP99]. This language, essentially, allows to describe the computing process implemented by each agent and the interaction with other ones in terms of messages sent and received. It also allows to discriminate roles covered by agents in different conversations and offers primitive constructs that are able to model parallel execution flows, decision point, iteration and sub-routing call. A procedure to translate an arbitrary specification in this language into the PROMELA language is also devise: PROMELA is the input language used by the model checker SPIN ([Hol04]) that implements various verification algorithms and it is able to verify if a given system specifications is consistent w.r.t. a set of constraints expressed in LTL: in particular, in this case, the problem of protocol consistency is addressed in terms of conversation termination ([CS01]). Adopting this approach it is possible to ensure the absence of deadlock, starvation, infinite recursions and synchronization errors. In order to make the problem feasible in terms of computational complexity, a simplification of the input is required: essentially the structure of exchanged data is ignored.

Another interesting approach based on the application of model checking techniques to the verification of formal properties of software agents implementing web services is devised in [PR04]. Agents are described as extended finite state automata that allow for internal state variables and complex guard-condition on transitions: they can be reduced to programs in transactional logic having as models sequences of states, represented as databases. Properties are formalized in a generalized linear temporal logic (GLTL), that is able to predicate about both path and state characteristics, as the specification of the composed process work-flow: it is essentially encoded as a set of additional constraints that restricts the domain of admissible computations. The proposed approach is also able to deal with multiple concurrent instances of modeled processes and to verify properties of an enactment set.

The multi-agent paradigm has been also applied to the definition and the implementation of the middleware and network environment in order to provide a family of meta-services (intended as service that manage the service-oriented application environment itself). In [MKY04], an architecture for the deployment of service-oriented applications is presented. It relies on three kinds of software agents:

master service agent, delegated to the management of available computing resources, to the instantiation of required elements, to the control of accesses and authorizations;

composite service agent, that coordinates orchestrations among different service agents served by the master service agent;

service agent, that implements capabilities involved in the provisioning of a specific service.

Speech Act

In [AGP03], an approach based on the *speech-act* paradigm has been proposed in order to enrich the interface description language of XML web services. As previously remarked, the standard language for the specification of web service interface WSDL lacks many features required to adequately state properties of conversational services or, in other words, that implement an enactment through a complex message flow between client and server. Roughly speaking, WSDL has been initially devised for design simple stateless services with simple message pattern (e.g., request/response, asynchronous request) and it is unable to express complex constraints involving multiple message flows/operations. On the other hand, the standard multi-agent conversational specification model ([FIP99]) allows to formalize as flow diagram exchanged message sequences between agents during a conversation session in order to specify only admissible conversation patterns. Since this approach cannot be directly employed on the standard XML web service architecture, because of its design restriction, an enhanced version of such services is devised: essentially the server node is able to keep track of the conversation state and it dynamically replies to the client specifying the set of admissible reply messages, limiting the search space accordingly. Generally it is required a degree of user intervention to control the client behavior: since the number of alternatives is minimized, it became more feasible than in the traditional approach.

Moreover, choreographic-based modeling approaches as WSCI ([AAF$^+$02]) provides a quite similar mechanism to describe the service access protocol, but in the latter case the protocol specification is statically defined as the interface structure at design-time, while in the former it can dynamically evolve during a conversation.

Commitment Protocols

Another technique deriving from solutions devised for multi-agent systems is presented in [XS03]: it is based on the application of *commitment protocols* that are able to characterize, in the context of a multi-agent interaction, the commitments that every part has taken. More specifically, these protocols are able to model that a part/agent is committed w.r.t. another one to assure that a given condition is satisfied. In this scenario, an agent can play a role if it is able to take a specific set of obligations. According to this modeling paradigm, it is possible to define some relevant commitment patterns that can be used to classify general interaction kinds among agent (e.g., acceptation, notification, resign, solicitation). These patterns can, hence, encoded into suitable CTL axioms obtaining a constraint set that a legal interactions must satisfy. Assuming that the agent behavior is represented using a statechart diagram, it is also possible to reduce the conversation consistency verification problem to the model checking of this structure w.r.t. the CTL axiomatization.

2.2.8 Conversational Specification

In [BFHS03], a *conversational specification* model has been devised allowing for the description of the behavior of a community of interacting services. In this approach, a composition schema (*ec-composition* and *ec-schema*) is essentially the description of cooperating processes (*peer*) connected by means of unidirectional asynchronous communication channels that route messages to network peers. Every peer in this model has an input queue that stores incoming messages from other network peers: when a message is peek from the queue it is processed according to the peer application rules updating the automaton state and eventually emitting new messages to other peers. In order to characterize the global system behavior, a special node, acting as an observatory (so-called *watcher*), has been introduced: it is able to monitor every communication channel and to log exchanged messages. According to this approach, the specification of a service composition is a language on the alphabet of message types that denote the set of admissible message sequences observed by the watcher.

Assuming that peers are implemented as Mealy automata ([HU79]), it is also possible to state the following properties:

1. given any arbitrary composition structure and automata specification the resulting language is context-sensitive;

2. if the automata input queue size is bounded, the resulting language is regular;

3. given any regular language, it is always possible to define a composition that realizes it;

4. the verification of arbitrary properties, encoded in LTL, of ec-compositions is undecidable assuming unbounded queue size ([HBCS03]).

Since this model does not require any central orchestration node (excepting from the watcher, but it is merely a formalization element and it can be implemented as a distributed system too) it is very suitable for peer-to-peer integration architecture.

In [FBS04a], the conversational specification model is applied to the analysis of XML web services integrated using a BPEL orchestration specification. Moreover, the peer representation language has been also extended, introducing guard conditions on automaton transition expressed using variables bounded to internal state and message contents. This approach, enriched with techniques presented in [FBS04b], is able to model the service composition keeping into account also stored and exchanged data and not only the type specification. In particular, there is the assumption that exchanged data messages are based on XML, while branching/guard conditions are expressed in XPath. The BPEL/WSDL specification is translated using a provided algorithm into an equivalent PROMELA specification: it can be processed using a model checker to verify formal properties expressed in LTL[6]. An interesting *synchronization* property among peers is also discussed: in the case of synchronizable peers the verification of arbitrary properties also using unbounded length queues is decidable since the composition schema can be reduced to an equivalent one without queues (*sc-configuration*). A set of sufficient conditions that ensure the synchronization of the composition given properties of involved peers is also devised.

[6]A similar approach, but based on WSFL, a service orchestration specification language superseded by BPEL, and restricted only on data dependencies, ignoring data processing issues, is described in [Nak02].

An approach to the automatic composition synthesis is discussed in [BCD+03]. The adopted representation model is based upon the finite state automata paradigm to describe both the available services, that can be included in the composition (i.e., service community), and the required composition target. The bind among different automata (services) is expressed on the base of a common message alphabet that represents possible actions: every agent agrees on the meaning of these atomic operations (i.e., the operational semantics). An automaton can both require and perform an action: a requiring-only automaton is a client. In this framework, given an extended Mealy automaton specification as composition target and a service community, it is possible to compute a broker Mealy automaton using an algorithm based on the Deterministic Propositional Dynamic Logic (DPDL) inference. This automaton is able to interact with a client described as the target automaton, sending and receiving messages with other ones during the client request processing activity[7]. In other words, the target specification is the description of the client behavior: the synthesized broker implements this protocol using available services in a coherent manner both w.r.t. the client and service specifications (e.g., when the broker is in a final state also involved services are in corresponding final states). An extended model that is also able to deal with incomplete client specification (i.e., a non-deterministic FSA with don't-care transitions) has been devised in [BCD04]. Moreover, in [BCD+05], a characterization of atomic operation semantics is also provided: it is based upon the notion of database update. In this case the service community must share a common domain model expressed using a relational schema expressing available operations in terms of update statements.

2.2.9 Situation Calculus

The *situation calculus* ([Rei91, MH69]) is a foundational approach in the area of the knowledge-based planning and it can be employed to solve a number of task in the implementation of service-oriented applications as matchmaking and composition.

The Golog ([LRL+97]) is a logic programming language based on the situation calculus paradigm: it has been extended in [MS02] to deal with the complex service composition as an *incomplete information planning* problem. According to this proposal, available services are essentially atomic operations that the planner can use to define an action plan able to achieve the user goal In particular, preconditions and effects of actions are represented using the DAML-S model, of which a suitable axiomatization in situation calculus has been devised. The composition problem is defined according to the following assumptions:

1. there exists a collection of generalized program templates that describe a family of possible processes;

2. there exists a community of services provided by different agents described as (conditional) operations with preconditions and effects;

3. the user selects a program template and submits the composition target as a set of constraint on the resulting action plan and on the final system state;

[7]The reference architecture model is essentially hub-and-spoke.

4. the Golog interpreter runs over the specification executing the given template, binding the abstract actions (i.e., service activation) to actual provider according to enactment data, user input and information obtained by sensing.

The Golog interpreter is employed two times: as off-line planner and as composition orchestrator, differently from other approaches that require an on-line planning. This solution is feasible under the assumption that information gathered during the off-line planning are enough stable and that performed actions cannot alter the global system state:

1. during the off-line planning the world-altering actions are merely simulated, while activable sensing actions are performed in order to acquire information needed to instantiate the program template into a suitable (conditional) action plan;

2. the conditional plan is hence executed monitoring the preconditions satisfaction and the stability of assumed conditions, in case they are no more valid the on-line planner is reactivated to look for a new suitable scheduling if any.

In this context, two properties of resulting Golog programs are introduced:

knowledge auto-consistent programs, that cannot fail because it is impossible to obtain required information to accomplish the task by sensing actions;

physically auto-consistent programs, that cannot fail because it is impossible to execute an action required to achieve the goal.

We point out that this solution is able to support a complex service orchestration but not the synthesis, since it assumes that the program structure (the template) has been *a priori* defined.

An alternative, and general, approach is instead based on the Process Specification Language (PSL) ([S+99]). It is a very expressive process specification language integrating both the first-order predicate logic and the situation calculus: it is extensively used as a design/documentation tool, but it is too expressive to be employed using (nearly) automated reasoning techniques.

2.2.10 Action Description Language

The problem of the validation of a work-flow specification in case of incomplete information is addressed in [GTL00]. This is typical scenario of intra-organization cooperation (i.e., B2B integration) that is also interesting in the case of e-services, since the strong encapsulation approach limits the available information.

In particular, it is impossible to analyze at design-time every admissible execution scenarios, since many information can be accessible only during the execution itself. To cope with these issues, the adoption of an *action description language* derived from $\mathcal{A}$ ([GL93]) is devised: the system state is described using a predicate alphabet, while world-altering and knowledge-producing actions are distinguished according to an approach common in the planning system design.

The language is also able to model asynchronous events and exceptions. On this foundation, it is possible to define a work-flow specification as a collection of rules and a user goal as a collection of constraints on the system state and to check if the work-flow is compatible with the goal achieving. In order to deal with incomplete information, a three-value logic is employed.

2.2.11 Hierarchical Task Networks

In [WSH+03], a system designed to build composed e-services based on hierarchical task network is described.

In fact, this paradigm is quite similar to the process-model defined in the standard DAML-S and OWL-S, as well as in BPEL, and it also possible to reduce the composition problem to a hierarchical planning instance. In this case, the action plan is computed considering the concept of task/activity instead of system state. The atomic service descriptions are used to define invocation preconditions and execution effects of basic operators, while tasks or complex methods can derive their constraints in a bottom-up way considering properties of their composing elements. Given such a kind of specification, the hierarchical planner is able to coordinate atomic services implementing the required complex service.

Compared with other planning systems, hierarchical planners as, e.g., SHOP2 ([NMAC+01]), despite their reduced expressive power (i.e., concurrent actions and conditional outcomes), exhibit generally better performance with limited computing resources.

In [Ler04], the hierarchical task decomposition paradigm is employed to implement a process analysis and verification suite. In particular, the process work-flow is expressed using the Little-JIL language: it is essentially based on the hierarchical task decomposition and on the attribution of activities to a community of interacting agents. Given this process representation, it can be translated into a collection of finite state automata (Finite State Process) to verify liveness and safety properties using the model checker system LTSA ([KM06a]).

2.2.12 Estimated Regression Planning

As other planning techniques, also the *estimated-regression* planning approach has been extensively and successfully employed to address many issues arising in the implementation of service-oriented applications.

For example, in [McD02], a reduction of composition problem to planning is presented: an algorithm is devised to translate a service composition problem instance allowing for processing it using a *partial-order planner* ([RN95]). The planner is employed to generate a collection of conditional action-plan relaxing the closed world assumption. The problem formalization essentially gives an axiomatization of the message exchange among agents, while sensing actions are represented using a reification of modal operator K. However, the model employed to represent both the system state and actions in this family of tools, as the PDDL ([McD98]) language, are not generally enough expressive to deal with complex data structures involved in the e-service specification (i.e., XML Schema). In order to cope with contingent events, an on-line planner is also introduced to compute alternative scheduling when needed. The system has been prototyped implementing a translator from the DAML-S specification of available and target services into a PDDL specification.

An alternative approach is discussed in [Pee03b]: the solution devised to the computer-aided service synthesis is based on the following steps:

1. the application domain is defined using a theory in PDDL;
2. a subset of relevant services are selected from a catalog using techniques derived from the component-based computer-aided software design;
3. selected services are encoded according the PDDL domain specification;
4. the initial state and the goal are specified;
5. the planning problem instance is defined and solved using the planner obtaining a suitable strategy in terms of available services.

Differently from other planning-based approaches, in this case there is a complete-information assumption: in other words, sensing actions are ignored.

A more interesting alternative, based upon a non-deterministic version of PDDL, is presented in [APY+02]. In this case, the non-determinism is employed to define action-plan that are able to deal with erroneous events. This approach requires that the composition target is expressed as a planning problem instance annotated with additional execution constraints in a version of CTL according to user preferences. Given this specification, the planner is employed to compute some feasible scheduling, that are bound to available service provider and filtered according to additional constraints.

The service composition problem is addressed in [CST03] as a planning application assuming that available operators are only partially instantiated, since they are specified in terms of data domain and not in terms of values (i.e., service operation signatures), while the system state is described in terms of exchanged messages. According to the devised approach, the composition goal is expressed in terms of achieving a suitable message flow w.r.t. a set of constraints. The problem instance is solved using a backward-chaining planning algorithm enriched with type-checking operations to enforce compatibility policies on exchanged messages.

2.2.13 Description Logics

The family of Description Logics has been applied to deal with several aspects in the service-oriented computing, since they exhibit a high expressive power, nice computational properties and they are also the formal foundation of a relevant fragment of the Semantic Web ([BLHL01]).

In particular, in this scenario, the language DAML-S, and its evolution OWL-S and also alternative proposals as WSMO/WSML ([RKL+05, dBFK+05]), is the main tool employed to describe a semantically enriched service according to the notion of *service profile* and *service model*. The former annotates the operation signature according to the IOPE paradigm expressing service input, outputs preconditions and effects w.r.t. a domain ontology, while the latter allows to specify the process model of a complex service in terms of atomic ones. This language has been widely adopted in different solutions to the service composition problems (e.g., planning-based ones), but it has became the main reference model in matchmaking applications, since, adequately encoding service discovery requests, it is possible employ DL inference algorithms to check if in a service catalog there exists a suitable candidate ([PKPS02]).

Generally speaking, languages underlying the Semantic Web (i.e., $\mathcal{SHOIN}(\mathbf{D})$ or $\mathcal{SHIF}(\mathbf{D})$) are indeed very expressive and corresponding inference problems are quite complex, despite in the average cases as well as in simulation and test scenarios description logic reasoners perform in acceptable manner. However, it is also possible to employ less expressive language as CLASSIC ([DNDSDM03]) s.t. the inference problem can be solved in polynomial time[8]. Another approach is the system LARKS ([SKWL99]) that improves the retrieval performances adding a preliminary filtering step done using linguistic arguments (i.e., classification taxonomy).

In [BK02], an alternative approach is presented: it is essentially based on an entity-relationship model enriched with the role transitive closure used to express the service model and to perform the service matchmaking given user requirements. The matchmaking is implemented in terms of evaluation of logic programs expressed in Datalog$^{\neg}$

In [PC03], a formalization approach to the service model is presented: it is based upon a first translation from the π-calculus to a modal logic and hence a second translation to an description logic extended with parameterized roles. The devised system enable to separate aspects related to the service description from ones related to service interaction using specific process algebra constructs as message and communication channel. It is suitable for solve matchmaking problems as well as to verify process formal properties as safety and liveness.

Another class of modeling approaches based upon DLs essentially oriented to address matchmaking problems is presented in [GMP04]. As previously discussed ones, this is essentially a static approach since it is focused on the description of service entailments not on the world states (i.e., the interpretation structure is mapped upon possible actual instantiation of service activations) but it allows for a domain specific knowledge in the sense of Semantic Web. The most interesting contribution is the distinction between service variety (i.e., the extension of service instance set) due to *incomplete knowledge* (i.e., multiple models of a given theory) and to *intended diversity* (i.e., multiple instances in a given model). While the latter is a goal of a service publisher (i.e, describing a service that is able to accept multiple client requests) the former is generally a consequence of poorly defined domain specification. Stemming from these concepts, also service *availability* and *coverage* concept has been introduced to denote the applicability of a service to deal a given class of requests. Matchmaking problems are reduced to various forms of reasoning tasks, despite some kind of constraints/features are not actually addressable since using only a DL or a two-variable fragment of first-order logic would seem to be to restrictive. Moreover also a feasible specialization ordering relation among services, given a client requirement, is provided. Also this relation is reduced to an inference problem in expressive DL.

This approach has been extended in [GM05, GMP06], in order to deal with the incomplete knowledge specification issues by means of an auto-epistemic extension of DL implementing a *local closed world* reasoning (i.e., assuming

[8]In most interesting applications analyzed so far, the problem size is essentially related to the schema size, while extensional aspects are ignored.

that for a “local” portion of the model the closed-world assumption holds).

Another extension has been devised in [d’A07], introducing a partition among constraints: *hard constraints* are features that a service must necessarily have, while *soft constraints* are features that a service should preferably have. Starting from this distinction a more refined ranking algorithm, also leveraging on DL reasoning, is provided.

In [BLM$^+$05a], a general approach to the update of DL extensional knowledge bases is devised: it relies upon a suitable extension of the language including some operators from the Hybrid Logics ([ABG$^+$07]). Despite it is complete action formalism, it has been applied to the analysis of properties of semantics e-services, in particular considering the role of the domain knowledge and incomplete specification. The proposed approach, in fact, introduce an update repair mechanism that is able to complete the service effects with additional updates in order to obtain a resulting world state that is consistent with a set of constraints (i.e., a TBox). In presence of an expressive description logic language, the problem is decidable only restricting the specification to some special classes of TBoxes (i.e., acyclic or definitorial). It has been employed to reason about formal properties of e-services in [BLM$^+$05b].

Another approach, essentially stemming from [BLM$^+$05b], has been devised in [WL06]: while the restriction on acyclic TBoxes is preserved, it is allowed for a more expressive DL language. In particular the action reasoning framework is extended to the $\mathcal{SHOIN}^+(\mathbf{D})^*$ language ([HPS03]) that is able to specify complex role assertions and also to deal with *concrete domains*. On this foundation, a matchmaking problem, restricted to preconditions and effects, is defined considering different matching conditions and providing an inference-based algorithms. An extension to the whole IOPE paradigm is feasible integrating other approaches.

CHAPTER 3

DEFINITIONS

In this chapter we present and discuss the preliminary assumptions on which the devised approach relies, and formally introduce the primitive constructs employed in the framework, assuming that the reader is familiar with first-order logic and computational complexity analysis.

3.1 Preliminary Assumptions

As discussed in Chapter 2, despite many works dealing with the problem of modeling and managing e-services, both from the technical perspective and from the formal one, the approaches proposed so far are not entirely satisfactory, since they generally ignore the semantic problem or delegate it to the specific application [LH03]. Such an approach could lead to inconsistency problems, as recently pointed out in the related field of Semantic Web in [HPPSH05], on which the main formal e-service languages rely. Given a knowledge representation language (i.e., a Semantic Web language), it is always possible to build a classification model of available services (a so-called service ontology), since we can define, using language constructs, the various service classes (e.g., payment services, reservation services, information services) specifying their relationships (e.g., generalization, dependency, refinement, instantiation). It is worth noticing that such a kind of ontology is interpreted over the universe of possible services without specifying their own semantics, unless "suggesting" an intuitive meaning through element names. There are different approaches (e.g., [MBE03, DNDSDM03, PKPS02, SdF03, LH03, GMP06]) based on this assumption, which rely upon knowledge representation tools, including Description Logics, to deal with concept taxonomies, while essentially ignoring dynamic features. Generally speaking, the "semantics" of these approaches is based on the instantiation of a service activation (i.e., a service enactment is a set of possibly ground assertions or facts). While this assumption can be quite adequate to support matchmaking and discovering applications, it is infeasible for more complex tasks as formal validation or automatic synthesis.

The considered computing environment is a community of distributed software agents that provide or request functionalities exposed by means of e-services in order to implement cooperative integration, as it is generally assumed in service-oriented architectures (e.g., [ACKM04, PG03]). As previously observed, this is a general integration model that is applicable in many business scenarios like:

- automatic service customization for mobile and extemporary users;
- cooperative information systems for *e-government* solutions [CNI05b, CNI05a, CNI05c, Peo05, G4B04];
- service-oriented integration in enterprise information systems (EAI);
- business partner information system integration and e-business cooperative solutions (e.g., B2B, virtual enterprise network, digital districts).

Such a kind of system integration pattern is generally denoted as *service-oriented integration* (SOI), since it leverages on service encapsulation properties to decouple the implementations. Services are generally intended as transactional operations (not the in database sense, however), managing the system state according to requestor directives and provider decisions.

Given a service-oriented computing environment, we are mainly interested in the following areas, pointed out in various development and execution scenarios:

1. the semantic specification and inventory of available services (*service directory*), supporting service matchmaking upon semantic specification (*service discovery*), tracing service specific features;
2. the analysis of general semantic services features like: consistency w.r.t. knowledge domain/constraints, instantiability, functional equivalence, replaceability, etc.;
3. the analysis of specific service features relevant to a client or a mediator aiming at achieving a particular goal or offering a new enhanced functionality to the community by aggregating available services into a complex process.

In order to deploy solutions suitable for end-user's needs in such kind of scenarios, exploiting features of loose coupling composition approaches as proposed in [Kay03], among various technology enablers (e.g., agreed standards, reliable middleware, open protocols), we need an abstract way to describe structure and relevant features of the world and the semantics of provided services.

The e-service approach should provide abstraction from the implementation by means of a strong wrapping metaphor (*black-box*), but the representation of the public component's interface does not automatically ensure that the formalization is abstract enough to express only functional properties. On the other side, this modeling strategy can result too weak to capture intended semantics.

In this work, the design problem is addressed abstracting from the interaction protocol between actors involved into a service enactment, since the functional semantics of a service should be independent not only from its implementation (how the function is realized), but also from the interaction protocol (how the function is accessed). Notably, XML web-service standards (e.g., WSDL, SOAP) are aiming to achieve network-protocol independence, but many approaches proposed in literature, like [MB03, CIJ$^+$00, DKR04, SBS04], are focused on the design and analysis of abstract messaging protocols. Despite that such kind of details are strictly necessary, in order to deal with semi-automatic/automatic synthesis or verification issues (more significant proposals are discussed in [MS02, BFHS03, APY$^+$02, Sri02, DSV04, McD02, BCD$^+$03]), we are interested in the analysis of the expressiveness of a framework representing e-services as substantially atomic or, at least, self-consistent operations[1].

Starting from this assumption, there is a general agreement upon the adoption of a modeling paradigm based on the elicitation of which (not how) information is exchanged during the enactment between requestor and provider, which are the admissible states of the world before the enactment and which are the possible world states after the correct completion of the service execution. In other words, in order to characterize an e-service we need to specify its own *inputs*, *outputs*, *pre-conditions* and *effects* (IOPE). This is a general approach deriving substantially from the foundational papers on Computing Semantics (i.e., Hoare's logic [Hoa69]) and the AI Planning literature ([RN95, EFL$^+$04]), but widely used in other fields, like database update theory ([Win90]) and active databases ([AHV95a, AVFY98]), despite with some adjustments, and adopted by the semantic web-services community in the definition of modeling languages and related standards (e.g., OWL-S, DAML-S, WSMO) and generally assumed in the most approaches in this field.

Despite this large agreement, formal assumptions done by different authors can vary not negligibly on substantial modeling aspects (e.g., reliability degree, complete or incomplete information, concurrency level), thus the approaches are not easily comparable in a direct way. More specifically, most of these approaches lack the formalization of some intuitive and, in our opinion, interesting notions, like, for example:

- consistency and realizability of available services: is a given service consistently defined w.r.t. the domain knowledge/constraints?
- functional equivalence/replaceability: are two or more services acting in a quite similar manner? are they doing the "same" thing? can a service replace another faulting one?
- functional similarity w.r.t. the invocation context: are two or more services similar, abstracting from the invocation scope? E.g., given two tax payment services, are they actually the "same" service, despite they are serving different user communities, have potentially been specified independently and do not expose the same interfaces/contract? Assuming that the service contract is as detailed and complete as possible, in the case of similar services, they must differ at least for the fragment denoting the service coverage.

In other words, while the classical (i.e., in terms of software components) functional equivalence is assessed considering the whole specification, the functional similarity is related only to "intrinsic" intensional specification fragment, ignoring the extensional one.

[1]We point out that in several situations the service interaction protocol (e.g., synchronous or asynchronous) depends only upon organizational constraints (i.e., the service provider implements part of service process in terms of *human work-flow*).

How to (formally) characterize these elements is generally an open issue, but, on the other hand, we can easily conclude that the ability to deal with these properties from a formal perspective will greatly help the design and implementation process, improving the quality of delivered solutions. Formal design approaches, given the costs induced by their implementation, can be effectively employed only if they are able to provide some useful insights by means of automated (even partially) procedures, helping to prevent or to solve issues that possibly raise during the system life-cycle.

We aim at devising a set of reasoning tasks about available services and user's goals, in order to check some of these properties. Considering the design issues arising in the implementation of a service-oriented architecture, the ability to detect service replaceability or equivalence on a functional base allows to easily realize automatic on-line service discovering/binding policies. In fact, it is possible to narrow the query range to locate a specific service provider in a given class (i.e., the class of service compatible with a temporary unavailable one) and enforcing the replacement according to a behavioral comparison.

In our intentions, the service specification should be intended as the formalization of the service contract. Under a fairness assumption, we suppose that agents act according to domain constraints, which means that they prevent inconsistent evolutions of the world's state and that they enforce service contracts. Roughly speaking, the service preconditions are concerning the commitment of service requestor, while the service effects the commitment of the service provider. Moreover, we consider services that have inherently non-deterministic effects, allowing arbitrary internal process logic.

We also assume that the information concerning the world available to an agent that reasons about the service community and related elements can be incomplete, although it is assumed to be correct. In other words, the inference process is done according to a general *open-world assumption* (OWA), while most of the approaches deriving from database applications are based upon a *closed-world assumption* (CWA).

The latter is applicable in the case of integration implemented into a closed and controlled environment (e.g., EIS), but generally we need to cope with partial domain models, adopting a sort of hybrid setting.

Remark 1. *As pointed out in several observations, the specification of some extensional elements is a characteristic feature of service-oriented applications: since this specification is* ipso facto *only partial, we strongly need to deal with any possible world that is compatible with it.*

Despite the network environment is concurrently accessed by a number of independent agents in a non-cooperative manner (each one aims at achieving its own goal, possibly clashing with other agent's ones), we ignore such kind of problems. In fact, keeping them into account, we should safely conclude only that every operation can eventually fail due to a clash. On the other hand, we can observe that the (total) world model should be large enough to allow every agent to act in a quite isolated manner, since a conflict is statistically unlikely and distributed transaction technologies provide (limited) locking capabilities allowing the sequencing of enactments. Therefore, we assume that only an enactment is active at a time or, equivalently, we ignore concurrent client sessions.

3.2 Formal Tools

In the following we briefly present the main formal tools employed in the present work. In particular, we present an interesting fragment of first-order predicate logic for which a decidable proof algorithm exists and the family of Description Logics, widely employed in the Semantic Web and Semantic Web Services applications.

Roughly speaking, we focus our attention on a class of Description Logics such that reasoning tasks about model update properties in the description logic can be reduced to decidable inference problems in first-order logic.

3.2.1 First-Order Logic with Counting Quantifiers

Among various fragments of the well-known first-order predicate logic we consider the function-free fragment with at most two variables with *counting quantifiers*, denoted as $\mathcal{C}^2$. Roughly speaking, such a kind of construct can express conditions as "there exist at most k elements such that ..." or "there exist at least k elements such that ...", generally expressed in first-order logic using equality.

Let $\mathbf{V} = \{x, y\}$ be the set of variable names, a $\mathcal{C}^2$-term is an occurrence of a variable name. Let $\mathbf{P}$ be a finite set of unary and binary predicate names, a $\mathcal{C}^2$-atomic formula set is defined as followings:

- if P is an unary predicate name and t is a term, then $P(t)$ is an atomic formula;
- if P as a binary predicate name t_1 and t_2 are terms, then $P(t_1, t_2)$ is an atomic formula;
- $\top$ and $\bot$ are atomic formulas.

Finally, the set of $\mathcal{C}^2$ well-founded formulas is inductively defined as:

- if A is an atomic formula, then A is a formula;
- if ϕ is a formula, then (ϕ) is a formula;
- if ϕ is a formula, then $\neg\phi$ is a formula;
- if ϕ and ψ are formulas, then $\phi \wedge \psi$, $\phi \vee \psi$, $\phi \to \psi$, $\phi \equiv \psi$ are also formulas;
- if ϕ is a formula and $v \in \mathbf{V}$ is a variable name, then $\forall v.\phi$ is a formula;
- if ϕ is a formula, $v \in \mathbf{V}$ is a variable name, $n \in \mathbb{N}$ is a positive natural number and $\bowtie \in \{=, <, >, \leq, \geq\}$, then $\exists^{\bowtie n} v.\phi$ is a formula.

As shorthands, if we have a multiple occurrence of the same quantifier Q as $Qx.Qy.\phi$ we simply use the notation $Qx, y.\phi$.

The underlying semantics is the standard semantics of first-order logic considering the additional case of counting quantifiers that can be expressed according the following rule: let $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ be an interpretation structure for the alphabet $\mathbf{P}$, σ an assignment for variables $\mathbf{V}$ and ϕ a well-founded formula, then $\langle \omega, \sigma \rangle \models \exists^{\geq n} v.\phi$ iff there exists a set $D \subseteq \Delta^\omega$ s.t. $\|D\| \geq n$ and for each $d \in D$ we have that $\omega \models \phi\,(\sigma\,[v/d])$.

The semantics can be also defined reducing the language to first-order logic with equality according to the fact that:

$$\exists^{\geq n} v.\phi(v) \Longleftrightarrow \exists x_1, \ldots, x_n. \bigwedge_{i \neq j} x_i \neq x_j \wedge \bigwedge_i \phi(x_i)$$

Other counting quantifiers can be defined analogously using the language closure under boolean operators.

Notably, the presence of equality does not increase the expressive power of the language, since it can be simulated introducing a binary predicate eq and the following axioms: $\forall x.\mathsf{eq}\,(x, x)$ and $\forall x.\exists^{=1} y.\mathsf{eq}\,(x, y)$.

A remarkable property of this language is that the satisfiability and finite satisfiability problems are decidable ([GOR97, HSG04]). For more details about the two-variable fragment of first-order logic and counting quantifiers see [PH05].

3.2.2 Expressive Description Logics

The description logic language $\mathcal{ALCQI}$ is defined in terms of concept expressions according to the following syntax:

$$\begin{aligned} C, C' &\longrightarrow A \mid \neg C \mid C \sqcap C' \mid (\bowtie n\, R\, C) \\ R &\longrightarrow P \mid R^- \end{aligned}$$

where A and P denote, respectively, atomic concept and atomic role names, C and R denote, respectively, arbitrary concepts and roles, $n \in \mathbb{N}$ and $\bowtie \in \{=, <, >, \leq, \geq\}$ is a generic relational operator on natural numbers. The language $\mathcal{ALCQIO}$ enriches the concept expression of $\mathcal{ALCQI}$ language with the nominal construct:

$$C, C' \longrightarrow A \mid \neg C \mid C \sqcap C' \mid (\bowtie n\, R\, C) \mid \{o\}$$

Other operators (e.g., $\sqcup$, $\forall$, $\exists$) can be defined in terms of primitive ones. The concepts $\top$ and $\bot$ denote the interpretation universe and the empty set. The semantics of such a language is defined w.r.t. an interpretation structure $\mathcal{I}$ s.t. concept names are interpreted as subsets of the domain $\Delta^{\mathcal{I}}$ and roles and object names, respectively, as binary relations and elements over $\Delta^{\mathcal{I}}$, as follows:

$$\begin{aligned} A^{\mathcal{I}} &\subseteq \Delta^{\mathcal{I}} \\ (\neg C)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\ (C \sqcap C')^{\mathcal{I}} &= C^{\mathcal{I}} \cap C'^{\mathcal{I}} \\ (\bowtie n\, R\, C)^{\mathcal{I}} &= \big\{ o \in \Delta^{\mathcal{I}} \,|\, \big\| \big\{ o' \in \Delta^{\mathcal{I}} | \langle o, o' \rangle \in R^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}} \big\} \big\| \bowtie n \big\} \\ \{o\}^{\mathcal{I}} &= \big\{ o^{\mathcal{I}} \big\} \\ P^{\mathcal{I}} &\subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \\ (R^-)^{\mathcal{I}} &= \big\{ \langle o, o' \rangle \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} | \langle o', o \rangle \in R^{\mathcal{I}} \big\} \end{aligned}$$

A *knowledge base* (KB) is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$ formed of a *terminological box* or TBox $\mathcal{T}$ and an *assertional box* or ABox $\mathcal{A}$. The former contains concept inclusion axioms in the form $C \sqsubseteq D$, the latter object membership axioms or assertional sentences in the forms $o : C$ and $(o, o') : R$. An interpretation $\mathcal{I}$ satisfies, or is a *model* of, the knowledge base iff, for each $C \sqsubseteq D \in \mathcal{T}$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, for each $o : C \in \mathcal{A}$, $o^{\mathcal{I}} \in C^{\mathcal{I}}$ and for each $(o, o') : R \in \mathcal{A}$, $\langle o^{\mathcal{I}} o'^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$. Generally, the TBox is considered as the *intensional specification* of the knowledge base, while the ABox is the *extensional one*. The $\mathcal{ALCQIO}$ language allows also for another kind of axioms, i.e., *cardinality restrictions* which have the form $\sharp(C) \bowtie n$: an interpretation $\mathcal{I}$ is a model for the above axiom iff $\|C\|^{\mathcal{I}} \bowtie n$. Such kinds of Description Logics also have a *standard semantics* in terms of first-order logic fragment function-free with counting quantifiers $\mathcal{C}^2$ presented in the previous section, as follows:

$$\begin{aligned}
\pi_x(A) &\triangleq A(x) \\
\pi_x(\neg C) &\triangleq \neg \pi_x(C) \\
\pi_x(C \sqcap C') &\triangleq \pi_x(C) \wedge \pi_x(C') \\
\pi_x(\bowtie n\ P\ C) &\triangleq \exists^{\bowtie n} y.(P(x,y) \wedge \pi_y(C)) \\
\pi_x(\bowtie n\ P^-\ C) &\triangleq \exists^{\bowtie n} y.(P(y,x) \wedge \pi_y(C)) \\
\pi_y(C) &\triangleq \pi_x(C)[x/y, y/x] \\
\pi(\sharp(C) \bowtie n) &\triangleq \exists^{\bowtie n} x.\pi_x(C) \\
\pi(C \sqsubseteq D) &\triangleq \forall x.\pi_x(C) \rightarrow \pi_x(D)
\end{aligned}$$

where each object name o has been replaced with a singleton concept name O and cardinality restrictions $\sharp(O) = 1$. The *unique name assumption* can be easily enforced adding additional axioms the impose the disjointness among different singleton sets.

We remark that such a class of knowledge representation languages is providing a formal foundation to the Semantic Web but also to many design paradigms employed in the specification of information systems as UML structural models (i.e., class and component diagrams) and Entity-Relationship diagrams ([BCN92]).

For more details and related results please refer to [BCM+03] and [Tob00].

3.2.3 Computational Complexity

In this work we adopt the notation for computational complexity classes as presented in [Pap94].

In particular, we will analyze the time-complexity of problems using a deterministic or non-deterministic Turing Machine (TM). Since most of the problems that we study are decision problems, we generally consider a TM that is able, given a problem instance accordingly encoded on the input tape, to check whether it belongs to a language s.t. the target property holds for its productions (and only for these ones), writing the boolean result on the output tape.

Some very complex problems are, indeed, solvable using TMs that are also able to access an *oracle*: a computational device that is able to recognize a production of a language of a given complexity class in constant time (excluding the time required to encode/decode the input and the output), according to the approach to *relative complexity classes* introduced in [LL76]. We are also assuming that this oracle is accessed at most once in every non-deterministic computation branch.

The most relevant complexity classes used in the rest are reported in Table 3.1.

Class	Description
EXP	Class of languages that can be recognized using a deterministic TM that operates in time exponential in the input length
NEXP	Class of languages that can be recognized using a non-deterministic TM that operates in time exponential in the input length
coNEXP	Class of languages that are complement of languages in NEXP
NEEXP	Class of languages that can be recognized using a non-deterministic TM that operates in time double-exponential in the input length
$\mathsf{NP}^{\mathsf{NEXP}}$	Class of languages that can be recognized using a non-deterministic TM that operates in time polynomial in the input length, using an oracle for a language in NEXP (or its complement)
$\mathsf{coNP}^{\mathsf{NEXP}}$	Class of languages that are complement of languages in $\mathsf{NP}^{\mathsf{NEXP}}$

Table 3.1: Complexity class definitions

3.3 System Specification

In this section, we introduce the model employed to specify system properties, starting from general assumptions adopted in knowledge representation applications and in the Semantic Web field.

Roughly speaking, in this context, we suppose that the system can be formally described using an alphabet of symbols of various kinds denoting objects, groups and links between them, in order to provide a formal axiomatization in a kind of logics, s.t. interpretations of the resulting logic theory are the possible system states. Since we are interested in evolving systems or, in other terms, into changing-state systems, we also need to describe such transitions and reasoning about them.

More technically, we assume that an infinite countable universe $\mathfrak{U}$ is given and that the system is described using an alphabet composed of a finite set of unary predicate names (or concept names) $\mathbf{A}$, a finite set of binary predicate names (or role names) $\mathbf{P}$ and a finite set of constant names (or object names) $\mathbf{O}$. Let object names be constantly interpreted according to the *standard-names assumption* on a finite subset $\mathfrak{O} \subset \mathfrak{U}$ of the given universe, i.e., by a bijective function $\cdot^{\mathcal{I}} : \mathbf{O} \mapsto \mathfrak{O}$.

We also assume that the modeled system, or, in other words, the application domain, can be described in terms of static properties/constraints using an expressive knowledge base formalism (i.e., a description logic), which allows both to define complex data structures and to easily include also extensional specification elements that will turn useful in the following.

Definition 1 (Domain specification)**.** *A domain specification is composed of three finite mutually disjoint sets:*

- *a concept alphabet* ($\boldsymbol{A}$)*;*
- *a role alphabet* ($\boldsymbol{P}$)*;*
- *an object alphabet* ($\boldsymbol{O}$)*.*

A system state (or world state) is described using an interpretation of the alphabet on the universe.

Definition 2 (World state)**.** *A world state* $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ *is an interpretation s.t.:*

- $\mathfrak{O} \subseteq \Delta^\omega \subseteq \mathfrak{U}$ *is the interpretation (or active) domain;*
- $\cdot^\omega$ *is a function that maps each concept name* $A \in \boldsymbol{A}$ *to a set* $A^\omega \subseteq \Delta^\omega$*, each role name* $P \in \boldsymbol{P}$ *to a set* $P^\omega \subseteq \Delta^\omega \times \Delta^\omega$ *and each object name as the* $\cdot^\mathcal{I}$ *function.*

Since generally not every interpretation can be considered to be a legal system state representation, we introduce the ability to restrict the state space to the valid ones by means of a constraint set, expressed using a suitable language.

Definition 3 (World specification)**.** *A world specification* $\mathcal{W}$ *is a knowledge base* $\langle \mathcal{T}_\mathcal{W}, \mathcal{A}_\mathcal{W} \rangle$ *expressed on the alphabet* $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$ *using the expressive description logic* $\mathcal{ALCQI}$*.*

Definition 4 (Legal world state)**.** *Given a world specification* $\mathcal{W}$*, a world state* ω *is valid w.r.t. the specification iff it is a model of the description logic theory:*

$$\omega \models \mathcal{W}$$

Definition 5 (Consistent world specification)**.** *A world specification* $\mathcal{W}$ *is consistent iff there exists at least a legal world state w.r.t.* $\mathcal{W}$*.*

Given a domain specification, assuming, w.l.o.g., that Top and New are new concept names, we define a knowledge base $\tilde{KB}$ composed by the instantiation of the axiom schema[2] reported in Table 3.2 for any concept name $A \in \mathbf{A}$, role name $P \in \mathbf{P}$ and object name $o \in \mathbf{O}$.

Remark 2. *We use* axiom schemas *as a useful notation shorthands: given an alphabet, the instantiated theory is obtained by replacing name placeholders (e.g.,* A*,* P*,* o*) with any compatible name and parameter name with assigned value, evaluating any translation or name mapping function, as shown in the following.*

[2]For the sake of simplicity, even when we are defining first-order language productions, we adopt a description logic syntax, at least when it is expressive enough, switching to the FOL syntax when it is needed. We also omit common schema arguments when they can inferred from the context (e.g., domain or world specifications).

Table 3.2: The basic axiom schema

$$\begin{aligned} \top &\sqsubseteq \mathsf{Top} \sqcup \mathsf{New} \\ \mathsf{Top} \sqcap \mathsf{New} &\sqsubseteq \bot \\ A &\sqsubseteq \mathsf{Top} \\ \top &\sqsubseteq \forall P.\mathsf{Top} \\ \top &\sqsubseteq \forall P^{-}.\mathsf{Top} \\ o &: \mathsf{Top} \end{aligned}$$

Remark 3. *We assume that name mapping functions are always injective, and when we employ different functions at the same time we also assume that their codomains are mutually disjoint. We assume also that name sets of different primitive constructs (e.g., concepts, roles, variables) are always mutually disjoint.*

We inductively define a translation function τ over the concept expressions of the description logic language $\mathcal{ALCQIO}$, from the alphabet $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$ to the new alphabet $\langle \mathbf{A} \cup \{\mathsf{Top}\}, \mathbf{P}, \mathbf{O} \rangle$, as follows:

$$\begin{aligned} \tau(A) &\triangleq A \\ \tau(C \sqcap C') &\triangleq \tau(C) \sqcap \tau(C') \\ \tau((\bowtie\ n\, R\, C)) &\triangleq (\bowtie\ n\, R\, \tau(C)) \\ \tau(\{o\}) &\triangleq \{o\} \\ \tau(\neg C) &\triangleq \mathsf{Top} \sqcap \neg\tau(C) \end{aligned}$$

Now, we start to introduce our approach to deal with reasoning tasks generally concerning dynamic features using a "traditional" logic language, in the sense that it does not provide native temporal primitives. The basic idea is to embed a system state transition, described in terms of initial and final states, parameter assignments, etc., into a single interpretation structure on which we solve some reasoning tasks (satisfiability or entailment) obtained by accordingly encoding the e-service checking problem into a suitable set of axioms. The link between original and "working" interpretation structures is caught by the following definition: it will be extended in the following as we go along, so that we are able to cope with various modeling refinements.

Definition 6 (Embedding relation)**.** *Let $\omega = \langle \Delta^{\omega}, \cdot^{\omega} \rangle$ be an arbitrary world state defined on an interpretation domain $\Delta^{\omega} \subseteq \mathfrak{U}$, and let $\hat{\omega} = \langle \mathfrak{U}, \cdot^{\hat{\omega}} \rangle$ be any interpretation over the alphabet $\langle \boldsymbol{A} \cup \{\mathsf{Top}\}, \boldsymbol{P}, \boldsymbol{O} \rangle$. The world state is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$) iff the following conditions hold:*

$$\begin{aligned} \Delta^{\omega} &= \mathsf{Top}^{\hat{\omega}} \\ N^{\omega} &= N^{\hat{\omega}} \\ o^{\omega} &= o^{\hat{\omega}} \end{aligned}$$

for any $N \in \boldsymbol{A} \cup \boldsymbol{P}$ and for any $o \in \boldsymbol{O}$.

We can easily generalize the provided definition introducing a name mapping function that embeds the structure using different concept or role names (since objects are always interpreted using the *unique name assumption*, we do not need an object mapping function).

Definition 7 (Generalized embedding relation)**.** *Let $\omega = \langle \Delta^{\omega}, \cdot^{\omega} \rangle$ be an arbitrary world state defined on an interpretation domain $\Delta^{\omega} \subseteq \mathfrak{U}$, let m be a function that maps each concept (resp. role) name A (resp. P) into a new one concept name $m(A)$ (resp. role name $m(P)$) and let Top_m be a new concept name. Given any interpretation $\hat{\omega} = \langle \mathfrak{U}, \cdot^{\hat{\omega}} \rangle$ over the alphabet $\langle m(\boldsymbol{A}) \cup \{\mathsf{Top}_m\}, m(\boldsymbol{P}), \boldsymbol{O} \rangle$, the world state ω is embedded into the interpretation ($\omega \rightsquigarrow_m \hat{\omega}$), w.r.t. the mapping m and the embedded top name Top_m, iff the following conditions hold:*

$$\begin{aligned} \Delta^{\omega} &= \mathsf{Top}_m^{\hat{\omega}} \\ N^{\omega} &= m(N)^{\hat{\omega}} \\ o^{\omega} &= o^{\hat{\omega}} \end{aligned}$$

for any $N \in \boldsymbol{A} \cup \boldsymbol{P}$ and for any $o \in \boldsymbol{O}$.

Remark 4. *We notice that using different mapping functions, which means having mutually disjoint co-domains, and possibly different embedded top names, distinct arbitrary world states can be embedded into an interpretation built over the union of mapped alphabets.*

Remark 5. *The properties of the embedding relation shown above can be extended to the generalized case, keeping into account accordingly the name mapping function m and the top name Top_m.*

Lemma 1. *Let be ω and $\hat{\omega}$ be respectively a world state and an arbitrary interpretation s.t. the world state is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$) w.r.t. a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$. Then, ω is embedded into $\hat{\omega}$ for any other domain specification $\langle \boldsymbol{A'}, \boldsymbol{P'}, \boldsymbol{O'} \rangle$ s.t.:*

$$\begin{aligned} \boldsymbol{A'} &\subseteq \boldsymbol{A} \\ \boldsymbol{P'} &\subseteq \boldsymbol{P} \\ \boldsymbol{O'} &\subseteq \boldsymbol{O} \end{aligned}$$

Proof. Trivial. □

Lemma 2. *Let be ω and $\hat{\omega}$ be respectively a world state and an arbitrary interpretation s.t. the world state is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$). Then, $\hat{\omega} \models \tilde{KB}$.*

Proof. We analyze various axioms of the knowledge base, showing that the provided structure is a model according to the standard semantics. Since $\Delta^{\omega} = \mathsf{Top}^{\hat{\omega}}$ and $\mathfrak{U} \setminus \Delta^{\omega} = \mathsf{New}^{\hat{\omega}}$, we can easily conclude that:

$$\begin{aligned} \hat{\omega} &\models \top \sqsubseteq \mathsf{Top} \sqcup \mathsf{New} \\ \hat{\omega} &\models \mathsf{Top} \sqcap \mathsf{New} \sqsubseteq \bot \end{aligned}$$

Since $A^\omega = A^{\hat{\omega}}$ and $A^\omega \subseteq \Delta^\omega = \mathsf{Top}^{\hat{\omega}}$ we can also infer that:

$$\hat{\omega} \models A \sqsubseteq \mathsf{Top}$$

Using a similar argument, we can derive that also the other axioms hold in $\hat{\omega}$. □

Lemma 3. *Let $\hat{\omega}$ be a model of $\tilde{KB}$, then there exists exactly one world state ω that is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$).*

Proof. The axioms introduced into the knowledge base force the interpretation of named concepts, roles and objects to be contained into the extension of the concept Top, which means that the projected structure built restricting the interpretation domain Δ^ω to $\mathsf{Top}^{\hat{\omega}}$ is well-founded, since the codomain of the interpretation function $\cdot^\omega$ is actually defined over Δ^ω.

To show that there is only one world state embedded into the interpretation we point out that the projection is completely deterministic (there is any decision point) and that the projected structure (ω) is completely defined once the model is provided. □

Theorem 1. *Let be ω and $\hat{\omega}$ be respectively a world state and an arbitrary interpretation s.t. the world state is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$), then:*

$$R^\omega = R^{\hat{\omega}}$$

for any $\mathcal{ALCQIO}$ role expression R built over the domain specification alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$.

Proof. Since the embedding relation imposes that $P^\omega = P^{\hat{\omega}}$, we can easily conclude that also $[P^-]^\omega = [P^-]^{\hat{\omega}}$ and that the claim follows. □

Theorem 2. *Let be ω and $\hat{\omega}$ be respectively a world state and an arbitrary interpretation s.t. the world state is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$), then:*

$$C^\omega = [\tau(C)]^{\hat{\omega}}$$

for any $\mathcal{ALCQIO}$ concept expression C built over the domain specification alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$.

Proof. We prove the theorem by induction over the expression language.

1. $A^\omega = [\tau(A)]^{\hat{\omega}}$ where $A \in \mathbf{A}$. By the definition of translation function τ:

   $$[\tau(A)]^{\hat{\omega}} = A^{\hat{\omega}}$$

   By the definition of the embedding relation between the structures:

   $$A^{\hat{\omega}} = A^\omega$$

2. $[\{o_1, \ldots, o_n\}]^\omega = [\tau(\{o_1, \ldots, o_n\})]^{\hat{\omega}}$ where $\{o_1, \ldots, o_n\} \subseteq \mathbf{O}$. By the definition of translation function τ:

   $$[\tau(\{o_1, \ldots, o_n\})]^{\hat{\omega}} = [\{o_1, \ldots, o_n\}]^{\hat{\omega}}$$

According to the standard semantics:

$$[\{o_1, \ldots, o_n\}]^{\hat{\omega}} = \bigcup_{i=1}^{n} o_i^{\hat{\omega}}$$

By the definition of the embedding relation between the structures:

$$\bigcup_{i=1}^{n} o_i^{\hat{\omega}} = \bigcup_{i=1}^{n} o_i^{\omega}$$

According to the standard semantics:

$$\bigcup_{i=1}^{n} o_i^{\omega} = [\{o_1, \ldots, o_n\}]^{\omega}$$

3. $[C \sqcap C']^{\omega} = [\tau(C \sqcap C')]^{\hat{\omega}}$. By the definition of translation function τ:

$$[\tau(C \sqcap C')]^{\hat{\omega}} = [\tau(C) \sqcap \tau(C')]^{\hat{\omega}}$$

According to the standard semantics:

$$[\tau(C) \sqcap \tau(C')]^{\hat{\omega}} = \tau(C)^{\hat{\omega}} \cap \tau(C')^{\hat{\omega}}$$

By the inductive hypothesis:

$$\tau(C)^{\hat{\omega}} \cap \tau(C')^{\hat{\omega}} = C^{\omega} \cap C'^{\omega}$$

According to the standard semantics:

$$C^{\omega} \cap C'^{\omega} = [C \sqcap C']^{\omega}$$

4. $[\neg C]^{\omega} = [\tau(\neg C)]^{\hat{\omega}}$. By the definition of translation function τ:

$$[\tau(\neg C)]^{\hat{\omega}} = [\mathsf{Top} \sqcap \neg\tau(C)]^{\hat{\omega}}$$

According to the standard semantics:

$$[\mathsf{Top} \sqcap \neg\tau(C)]^{\hat{\omega}} = \mathsf{Top}^{\hat{\omega}} \cap [\neg\tau(C)]^{\hat{\omega}}$$

$$\mathsf{Top}^{\hat{\omega}} \cap [\neg\tau(C)]^{\hat{\omega}} = \mathsf{Top}^{\hat{\omega}} \cap (\mathfrak{U} \setminus [\tau(C)]^{\hat{\omega}})$$

By the inductive hypothesis:

$$\mathsf{Top}^{\hat{\omega}} \cap (\mathfrak{U} \setminus [\tau(C)]^{\hat{\omega}}) = \mathsf{Top}^{\hat{\omega}} \cap (\mathfrak{U} \setminus C^{\omega})$$

By the definition of the embedding relation between the structures:

$$\mathsf{Top}^{\hat{\omega}} \cap (\mathfrak{U} \setminus C^{\omega}) = \Delta^{\omega} \cap (\mathfrak{U} \setminus C^{\omega})$$

Since $\Delta^{\omega} \subseteq \mathfrak{U}$ and $C^{\omega} \subseteq \mathfrak{U}$:

$$\Delta^{\omega} \cap (\mathfrak{U} \setminus C^{\omega}) = \Delta^{\omega} \setminus C^{\omega}$$

According to the standard semantics:

$$\Delta^{\omega} \setminus C^{\omega} = [\neg C]^{\omega}$$

5. $[(\geq\ n\, R\, C)]^{\omega} = [\tau((\geq\ n\, R\, C))]^{\hat{\omega}}$, where R is an arbitrary role expression ($R \rightarrow P|P^{-}$, $P \in \mathbf{P}$). By the definition of translation function τ:

$$[\tau((\geq\ n\, R\, C))]^{\hat{\omega}} = [(\geq\ n\, R\, \tau(C))]^{\hat{\omega}}$$

According to the standard semantics:

$$[(\geq\ n\, R\, \tau(C))]^{\hat{\omega}} = \left\{\alpha|\ \left\| S^{\hat{\omega}}_{\tau(C),R}(\alpha) \right\| \geq n\right\} \tag{3.1}$$

where $S^{\hat{\omega}}_{\tau(C),R}(\alpha)$ is the set of R-successors of element α belonging to $\tau(C)$ defined as:

$$S^{\hat{\omega}}_{\tau(C),R}(\alpha) \triangleq \{\beta | \beta \in \tau(C)^{\hat{\omega}}, \langle \alpha, \beta \rangle \in R^{\hat{\omega}}\}$$

But, since the inductive hypothesis and Theorem 1, we can establish that:

$$S^{\hat{\omega}}_{\tau(C),R}(\alpha) = \{\beta | \beta \in C^{\omega}, \langle \alpha, \beta \rangle \in R^{\omega}\}$$

In other words, we can conclude that:

$$S^{\hat{\omega}}_{\tau(C),R}(\alpha) = S^{\omega}_{C,R}(\alpha)$$

Applying such result to Eq. 3.1, we have that:

$$[(\geq\ n\, R\, \tau(C))]^{\hat{\omega}} = \left\{\alpha|\ \left\| S^{\omega}_{C,R}(\alpha) \right\| \geq n\right\}$$

Observing, according to the standard semantics, that:

$$[(\geq\ n\, R\, C)]^{\omega} = \left\{\alpha|\ \left\| S^{\omega}_{C,R}(\alpha) \right\| \geq n\right\}$$

we have proved the claim.

□

Remark 6. *We notice that other language constructs can be defined in terms of primitive ones:*

$$\begin{aligned} \exists R.C &\triangleq (\geq 1 R C) \\ \forall R.C &\triangleq \neg \exists R.\neg C \\ C \sqcup C' &\triangleq \neg(\neg C \sqcap \neg C') \end{aligned}$$

Given a world state $\omega = \langle \Delta^{\omega}, \cdot^{\omega} \rangle$ we define an embedding function μ that maps such an interpretation into another interpretation $\hat{\omega}$ s.t.:

- the interpretation domain is the whole universe ($\Delta^{\tilde{\omega}} = \mathfrak{U}$);
- the interpretation of concepts, roles and object names is preserved ($N^{\omega} = N^{\tilde{\omega}}$);
- the interpretation of Top is the active domain of ω ($\mathsf{Top}^{\tilde{\omega}} = \Delta^{\omega}$);
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$.

The function π computes the inverse of μ, projecting out from an interpretation $\hat{\omega}$ a world state ω, and it is defined only for structures that are models of the knowledge base $\tilde{KB}$.

We remark that the provided definitions follow the construction used to define the embedding relation among structures. It is easy to observe that the following results hold:

Lemma 4. *Given a world state ω, let $\hat{\omega} = \mu(\omega)$, then $\omega \rightsquigarrow \hat{\omega}$.*

Lemma 5. *Given a model $\hat{\omega}$ of $\tilde{KB}$, let $\omega = \pi(\hat{\omega})$, then $\omega \rightsquigarrow \hat{\omega}$.*

Let $KB = \langle \mathcal{T}, \mathcal{A} \rangle$ be an arbitrary knowledge base built over the domain specification (i.e., a world specification $\mathcal{W}$), we define a new knowledge base $\tau(KB)$ over the extended alphabet s.t., for each general inclusion assertion $C \sqsubseteq D$ in the TBox $\mathcal{T}$, it includes a new axiom of the form:

$$\tau(C) \sqsubseteq \tau(D)$$

for each ABox assertion $o : C$ in $\mathcal{A}$, it includes a new axiom of the form:

$$o : \tau(C)$$

and, for each ABox assertion $(o, o') : R$ in $\mathcal{A}$, it includes a new axiom of the form:

$$(o, o') : \tau(R)$$

Theorem 3. *If a world state ω is a model of the knowledge base KB, then the structure $\hat{\omega} = \mu(\omega)$ is a model of the knowledge base $\tilde{KB} \wedge \tau(KB)$.*

Proof. The mapping function μ computes a new structure $\hat{\omega}$ that embeds the model ω. By Lemma 2, in such a structure the axioms of the knowledge base $\tilde{KB}$ hold, so, in order to prove that $\hat{\omega}$ is a model for the whole knowledge base we need to prove that also other axioms, obtained applying the τ function to concept expression, are satisfied. By contradiction, we assume that there is an axiom $\tau(C) \sqsubseteq \tau(D)$ s.t.:

$$\tau(C)^{\hat{\omega}} \nsubseteq \tau(D)^{\hat{\omega}}$$

According to Theorem 2, we have that $\tau(C)^{\hat{\omega}} = C^{\omega}$ and $\tau(D)^{\hat{\omega}} = D^{\omega}$, and we can conclude that:

$$C^{\omega} \nsubseteq D^{\omega}$$

contradicting the hypothesis that:

$$\omega \models C \sqsubseteq D$$

We can prove an analogous result regarding the ABox assertions, keeping into account that objects are always interpreted according to the unique name assumption (each structure agrees upon their interpretation). □

Theorem 4. *If $\hat{\omega}$ is a model of the knowledge base $\tilde{KB} \wedge \tau(KB)$, then the interpretation $\omega = \pi(\hat{\omega})$ is a world state that satisfies the knowledge base KB.*

Proof. According to Lemma 3 the structure $\omega = \pi(\hat{\omega})$ is an interpretation of the knowledge base KB. In order to show that it is also a model, we need to prove that all assertions hold. By contradiction, we assume that there is an axiom $C \sqsubseteq D$ s.t.:

$$C^{\omega} \nsubseteq D^{\omega}$$

According to Theorem 2 we have that $\tau(C)^{\hat{\omega}} = C^{\omega}$ and $\tau(D)^{\hat{\omega}} = D^{\omega}$, thus we can conclude that:

$$\tau(C)^{\hat{\omega}} \nsubseteq \tau(D)^{\hat{\omega}}$$

contradicting the hypothesis that:

$$\hat{\omega} \models \tau(C) \sqsubseteq \tau(D)$$

As in the previous theorem we can prove an analogous result regarding the ABox assertions. □

From the previous theorems we can easily conclude that:

Corollary 1. *The knowledge base KB is satisfiable on an arbitrary interpretation domain $\Delta \subseteq \mathfrak{U}$ iff the knowledge base $\tilde{KB} \wedge \tau(KB)$ is satisfiable on $\mathfrak{U}$.*

Theorem 5. *Given a world specification $\mathcal{W}$, the problem of checking if it is consistent is in* NEXP.

Proof. According to the proposed definition and to Corollary 1 in order to check whenever the given specification is consistent, we need to solve a satisfiability problem for a $\mathcal{C}^2$ sentence. As shown in [PH05] the SAT$\mathcal{C}^2$ problem is in NEXP, w.r.t. the length of the sentence, even allowing for succinct coding of number restrictions, while the devised construction is clearly linear in the size of the input, so there is a polynomial reduction from which the proposition follows. □

We notice that we consider the satisfiability of the (original) knowledge base KB w.r.t. some interpretation domain Δ^ω chosen among subsets of the universe $\mathfrak{U}$. Now, if the knowledge base is satisfiable in a restricted domain, is it also satisfiable on the whole universe? In general the answer is no, but if we adopt a language for which the *disjoint union model property* holds[3], like $\mathcal{ALCQI}$, the answer turns to be positive.

Lemma 6. *Given an $\mathcal{ALCQI}$-knowledge base, if it admits a model over a finite subset Δ^ω of the interpretation universe $\mathcal{U}$, then it admits also a model on the whole universe.*

Proof. Since the finite cardinality when can be an infinite countable number of disjoint replicas of the model using elements in $\mathcal{U} \setminus \Delta^\omega$: these replicas satisfy TBox assertions, while ABox assertions are already satisfied by the original one. The union of such a kind of models cover the whole universe and by the disjoint union model property it is also a model of the theory. □

Lemma 7. *Given an $\mathcal{ALCQI}$-knowledge base, if it admits a model over an infinite subset Δ^ω of the interpretation universe $\mathcal{U}$, then it admits also a model on the whole universe.*

Proof. We distinguish two cases, if the cardinality of the complement of the interpretation domain w.r.t. the universe $\delta = \mathfrak{U} \setminus \Delta^\omega$ is finite or not.

If $\|\delta\| = \aleph_o$, then there is a bijective function between Δ^ω and δ, and we can build a replica ω' of the model ω using the set δ as interpretation domain. Clearly, ω' is also a model of the TBox, while ω is a model of the whole knowledge base. Since the domains are disjoint, their union, that covers the whole universe,

[3]According to such a property, if $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ and $\omega' = \langle \Delta^{\omega'}, \cdot^{\omega'} \rangle$ are two model of a theory, the interpretation $\omega \biguplus \omega' = \langle \Delta^\omega \biguplus \Delta^{\omega'}, \cdot^\omega \uplus \cdot^{\omega'} \rangle$ is also a model. It is a typical property of modal logics that turns also useful for Description Logic applications ([CD03, De 95]). The operator $\uplus$ is defined as $A \uplus B \triangleq (A \setminus B) \cup (B \setminus A) = (A \cup B) \setminus (A \cap B)$.

for the disjoint model property is also a model of the TBox and of the ABox too, because the interpretation function $\cdot^{\omega \uplus \omega'}$, restricted to the domain Δ^{ω} is equals to $\cdot^{\omega}$.

If the set δ is finite, we can select a subset of $\delta' \subset \Delta^{\omega}$ of the same cardinality. Using the bijective function between these sets, swapping elements in δ and δ' and accordingly adjusting their links, we can derive a new model ω' of the TBox having as interpretation domain the set $\mathfrak{U} \setminus \delta'$. According to the disjoint union model property the structure ω'' having as interpretation domain the set $\delta \cup \delta' = (\mathfrak{U} \setminus \delta) \uplus (\mathfrak{U} \setminus \delta')$ is also a model of the TBox. Using the same construction adopted in the proof of the previous claim, we can derive a new model ω''' having as its interpretation domain the whole universe $\mathfrak{U}$, but this structure is not necessarily also a model of the ABox too. Employing the disjoint union model property, we can derive another new model ω'''' s.t. the interpretation domain is δ, since:

$$\mathfrak{U} \uplus \Delta^{\omega} = \delta$$

Combining this model with ω we obtain finally a structure that since the domains are disjoint, their union covers the whole universe, for the disjoint model property is also a model of the TBox and of the ABox too, because the interpretation function $\cdot^{\omega \uplus \omega''''}$, restricted to the domain Δ^{ω} is equals to $\cdot^{\omega}$. □

Theorem 6. *Given an $\mathcal{ALCQI}$-knowledge base, if it admits a model over an arbitrary subset Δ^{ω} of the interpretation universe $\mathcal{U}$, then it admits also a model on the whole universe.*

The result proved in previous claims does not hold if the language allows for using nominal constructs or more general cardinality axioms, as the following example shows, since they can restrict the satisfiability only to finite interpretation domains.

Example 1. *The following $\mathcal{ALCQIO}$ TBox allows only for models s.t. the interpretation domain has size* 1*:*

$$\top \sqsubseteq \{o\}$$

where o is a generic object name.

In other words, restring our attention only to $\mathcal{ALCQI}$ world specification we can reduce at least the complexity of the consistency check problem.

Lemma 8. *Given an $\mathcal{ALCQI}$ world specification $\mathcal{W}$, the resulting knowledge base $\tilde{KB} \wedge \tau(KB)$ is also in $\mathcal{ALCQI}$*

Proof. The claim follows easily from the definition of translation function and additional axioms: if there is any nominal-related constructs, the knowledge base can be expressed using the $\mathcal{ALCQI}$ language. □

From the previous results, the following property follows.

Theorem 7. *Given an $\mathcal{ALCQI}$ world specification $\mathcal{W}$, the problem of checking if it is consistent is* EXP*-complete.*

Proof. The proof of the membership claim is quite similar to Theorem 5, but applying Lemma 8 we obtain the reduction to a satisfiability problem instance for the $\mathcal{ALCQI}$ language that is EXP-complete, according to [CD03, CDLN01].

Moreover, applying Theorem 6 we can reduce the problem of satisfiability checking for an arbitrary $\mathcal{ALCQI}$ knowledge base to the consistency checking of the corresponding world specification (that is essentially a knowledge base). Hence, if the world specification admits as a model a subset of the interpretation universe, then knowledge base is satisfiable,otherwise the knowledge base is unsatisfiable. The reduction is clearly polynomial since it is the identity function. □

3.4 Object Instantiation

In the previous section we have discussed about basic properties of world states in the spirit of knowledge representation systems, but since we are in a dynamic setting, we need to introduce some update operators. In this section we start introducing the notion of object instantiation and related properties, as well as a first example of the approach devised to cope with implied verification problems related to them. We will extend this model in further chapters.

The system can dynamically evolve from a state to another one, generally as the result of the execution of some actions. These actions can essentially alter the extensional level or, in other words, perform an update of the system state specification. In present framework, world-altering actions can be only carried out by means of provided e-services, which we will describe in the following, but, w.l.o.g., we can assume that a service enactment can essentially perform the following kinds of tasks:

- create new objects, which means adding elements that are not included in the active domain of the initial state (assuming that $\mathfrak{U} \setminus \Delta^\omega \neq \oslash$) to the active domain of the resulting state, and that can be viewed as new;
- add or remove elements to a concept extension;
- add or remove links between elements.

For the sake of symmetry we can also consider the ability to destroy an existing element, which means throwing it out from the active domain, however we are not interested in such an ability[4].

While the extensional level can be altered by performed actions, the intensional level, built essentially by the system specification and constraints, must be assumed as immutable. In other words, the proposed approach investigates knowledge update related problems, which are somehow more general than knowledge revision ones. On the other hand, we must be able to analyze the actions under the perspective of soundness w.r.t. a given constraint set.

Definition 8 (Finite room world state)**.** *A world state ω has room for at least a finite strictly positive integer number n of newly created objects iff the cardinality of the set $\mathfrak{U} \setminus \Delta^\omega$ is at least n.*

[4]The proposed framework can be easily adjusted with some minor modifications in order to prevent that “destroyed” elements can come back as newly “created” ones.

Given an integer number $n \in \mathbb{N}$, assuming, w.l.o.g., that Top and New are new concept names, aux is a new role name and spy is a new object name, we define a knowledge base $\tilde{KB}^n$, as extension of the knowledge base $\tilde{KB}$ previously defined, adding the axioms obtained by the instantiation of the axioms shown in Table 3.3 w.r.t. the parameter n.

Table 3.3: The axiom schema $\Delta KB^n(\mathsf{spy}, \mathsf{aux})$

$$\top \sqsubseteq \forall \mathsf{aux}.\mathsf{New}$$
$$\top \sqsubseteq \forall \mathsf{aux}^-.\mathsf{New}$$
$$\mathsf{spy} : \mathsf{New} \sqcap (\geq\ n\, \mathsf{aux})$$

Given a world state $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ we extend the definition of embedding function μ, and corresponding embedding relation, provided at page 37, adding the following specifications:

- the object spy is assigned to an element of $\mathsf{New}^{\tilde{\omega}}$, if any;
- the role aux is interpreted as $\left\{\mathsf{spy}^{\tilde{\omega}}\right\} \times \mathsf{New}^{\tilde{\omega}}$.

As in the previous case, the function π computes the inverse of μ, while the mapping function is actually a multiple-result or a non-deterministic function, since it can embed the world state into different structures given the choice of the spy point. However, these structures are *isomorphic*, as we prove in the following, and actually indistinguishable by the languages employed. The construction of the interpretation using the embedding function is depicted in Figure 3.1.

Theorem 8. *If a world state ω has room for at least a finite strictly positive number n of newly created objects, then the interpretation $\tilde{\omega} = \mu(\omega)$ is a model of the knowledge base $\tilde{KB}^n$.*

Proof. Since the world state has room for at least n newly created element, then $\|\mathfrak{U} \setminus \Delta^\omega\| \geq n$, that means that also the extension of the concept New contains at least n element. We peek an arbitrary element x_1 as the interpretation of the object spy and since the interpretation of the role aux is $\left\{\mathsf{spy}^{\tilde{\omega}}\right\} \times \mathsf{New}^{\tilde{\omega}}$, then x_1 has n aux-successors (including itself) $\{x_1, x_2, \ldots, x_n\}$.

Other axioms are also satisfied using the construction performed by the mapping function, since the partition constraints follow from the definition of world state and qualified restrictions are enforced by construction itself. $\square$

Theorem 9. *If $\tilde{\omega}$ is a model of the knowledge base $\tilde{KB}^n$, then the interpretation $\omega = \pi(\tilde{\omega})$ is a world state having room for at least a finite strictly positive integer number n of newly created objects.*

Proof. Each model $\tilde{\omega}$ of the knowledge base must interpret the concept New to a set having at least n elements, since the constraint on the aux-successors of the object spy and the qualified restrictions over the role aux.

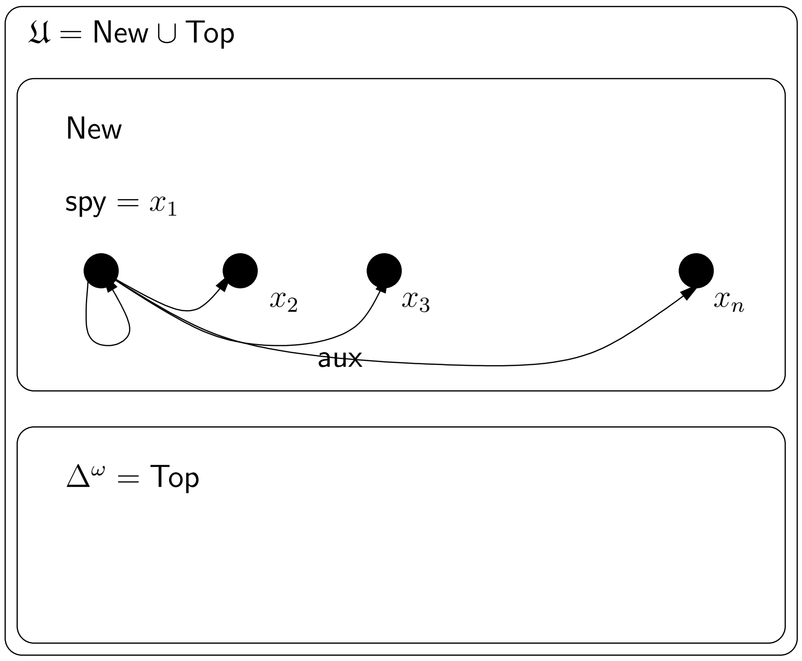

Figure 3.1: An example of the embedded model structure

The axioms over the concept New and Top also ensure that they realize a complete partition of the universe $\mathfrak{U}$ as required from the world state definition. □

Remark 7. *It is worth noticing, that such claims also extend to the case of $n = 0$ (no instantiation), and, hence, we can apply them to any $n \in \mathbb{N}$.*

3.5 Variables and Queries

In order to cope with complex domain specifications, we now include in our framework the typical constructs employed in the formalization of information systems, like variables and queries. The adopted approach is in the spirit of [Bor94], but other query language classes are also addressable using a suitable encoding: e.g., in [CDV05] is devised an encoding for union of conjunctive queries on DL knowledge bases.

Definition 9 (Parameterized query)**.** *Given a domain specification and a finite set of variable names $\boldsymbol{V}$, distinct from domain alphabets, a parameterized query $Q(\boldsymbol{V})$ is an arbitrary $\mathcal{ALCQIO}$-concept expression built on the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{V}, \boldsymbol{P}, \boldsymbol{O}\rangle$.*

Definition 10 (Variable assignment)**.** *Given a set of variable names $\boldsymbol{V}$ and a subset Δ of the universe $\mathfrak{U}$, an assignment $\sigma_{\boldsymbol{V}}$ on Δ is a function mapping names in $\boldsymbol{V}$ to elements of Δ.*

An assignment binds each variable name to a domain element, and each variable can be considered essentially as a singleton (or a nominal)[5].

Definition 11 (Extended interpretation)**.** *Given a model, i.e., a world state, $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ and a variable assignment $\sigma_{\boldsymbol{V}}$ on Δ^ω, the extended interpretation $\omega \triangleleft \sigma_{\boldsymbol{V}}$ is a new structure s.t.:*

- *the interpretation domain is the same as ω;*
- *the interpretation function $\cdot^{\omega \triangleleft \sigma_{\boldsymbol{V}}}$ is the same as ω for each name not in $\boldsymbol{V}$, while each $V \in \boldsymbol{V}$ it is interpreted as:*

$$V^{\omega \triangleleft \sigma_{\boldsymbol{V}}} = \sigma_{\boldsymbol{V}}(V)$$

Let $\mathbf{X}$ and $\mathbf{Y}$ be two distinct variable name sets, and let $\sigma_{\mathbf{X}}$ and $\sigma_{\mathbf{Y}}$ be two assignment on the same domain Δ^ω, given a model ω on such domain, we can combine the extended interpretations obtaining a new model $(\omega \triangleleft \sigma_{\mathbf{X}}) \triangleleft \sigma_{\mathbf{Y}}$ where each variable names in $\mathbf{X} \cup \mathbf{Y}$ is bound to a domain element. Since the name sets are distinct, the application order of the operator $\triangleleft$ is irrelevant.

Lemma 9. *Given a model $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ and two variable assignments $\sigma_{\boldsymbol{X}}$ and $\sigma_{\boldsymbol{Y}}$ on the same domain Δ^ω, if $\boldsymbol{X}$ and $\boldsymbol{Y}$ are distinct, then:*

$$(\omega \triangleleft \sigma_{\boldsymbol{X}}) \triangleleft \sigma_{\boldsymbol{Y}} = (\omega \triangleleft \sigma_{\boldsymbol{Y}}) \triangleleft \sigma_{\boldsymbol{X}}$$

Definition 12 (Parameterized query evaluation)**.** *Given a query $Q(\boldsymbol{V})$, a world state $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ and a variable assignment σ on Δ^ω the evaluation of the query is the set of elements of Δ^ω assigned to the concept Q in the extended interpretation $\omega \triangleleft \sigma_{\boldsymbol{V}}$:*

$$Q^\omega(\sigma_{\boldsymbol{V}}) = \{\alpha | \omega \triangleleft \sigma_{\boldsymbol{V}} \models \alpha : Q\}$$

Among queries we need to distinguish those that realize an access function, i.e., select at most an element as result.

Definition 13 (Access function)**.** *Given a world specification $\mathcal{W}$, a query $Q(\boldsymbol{V})$, is an access function iff for each legal world state ω and for each assignment $\sigma_{\boldsymbol{V}}$ on Δ^ω the cardinality of its evaluation is at most 1:*

$$\|Q^\omega(\sigma_{\boldsymbol{V}})\| \leq 1$$

Given a domain specification $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$ and the corresponding embedding knowledge base $\tilde{KB}$, and given a set $\mathbf{V}$ of variable names, we define a new knowledge base $\tilde{KB}^{\mathbf{V}}$ on the alphabet $\langle \mathbf{A} \cup \mathbf{V} \cup \{\mathsf{Top}, \mathsf{New}\}, \mathbf{P}, \mathbf{O} \rangle$ adding to $\tilde{KB}$ axioms obtained instantiating the schema $\Delta KB^{\mathbf{V}}$, shown in Table 3.4, where V is any variable name in $\mathbf{V}$, introduced as a new concept name in the alphabet.

We extend also the definition of the μ function in order to deal with variable assignments too, using the notion of extended interpretation previously devised. We define a new mapping function $\mu^{\mathbf{V}}(\omega, \sigma)$ s.t.:

[5]It has been shown [Tob00] that cardinality constraints, as in $\mathcal{C}^2$ logics, and nominal constructs, as in $\mathcal{ALCQIO}$, are, roughly speaking, equivalent, despite the latter language strictly contains the former one in terms of expressive power [Tob99].

Table 3.4: The axiom schema $\Delta KB^{\mathbf{V}}$

$$V \sqsubseteq \mathsf{Top}$$
$$\sharp(V) = 1$$

- the interpretation domain is the whole universe ($\Delta^{\tilde{\omega}} = \mathfrak{U}$);
- the interpretation of concepts, roles and objects is preserved ($N^{\omega} = N^{\tilde{\omega}}$);
- the interpretation of Top is the active domain of ω ($\mathsf{Top}^{\tilde{\omega}} = \Delta^{\omega}$);
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;
- the interpretation of variable auxiliary concepts is defined according to the assignment ($\sigma(V) = V^{\tilde{\omega}}$).

We can also define an inverse projection function $\pi^{\mathbf{V}}$ that given a valid structure $\tilde{\omega}$ for the new knowledge base returns a pair $\langle \omega, \sigma \rangle$ s.t. $\tilde{\omega} = \mu^{\mathbf{V}}(\omega, \sigma)$.

Lemma 10. *Given a consistent pair ω, σ, then:*

$$\mu^{\boldsymbol{V}}(\omega, \sigma) = \mu(\omega \triangleleft \sigma)$$

Proof. The result holds since, according to the provided definition of extended interpretation, variable and concept name sets are distinct, so that we can deal with variable as auxiliary concepts (even having an additional cardinality restrictions). □

Combining the previous result with Lemmas 4 and 2 we obtain also the following:

Theorem 10. *Given a consistent pair ω, σ, s.t. $cod(\sigma) \subseteq \Delta^{\omega}$, let $\tilde{\omega} = \mu^{\boldsymbol{V}}(\omega, \sigma)$, then:*

$$\omega \triangleleft \sigma \rightsquigarrow \tilde{\omega}$$

and:

$$\tilde{\omega} \models \tilde{KB}^{\boldsymbol{V}}$$

Proof. The definition of variable assignment and extended interpretation ensure that each variable concept name is interpreted as a singleton in the world state domain, enforcing the cardinality restriction axioms. □

Theorem 11. *Given a model $\tilde{\omega}$ of the knowledge base $\tilde{KB}^{\boldsymbol{V}}$ and a pair $\langle \omega, \sigma \rangle$ s.t. $\langle \omega, \sigma \rangle = \pi^{\boldsymbol{V}}(\tilde{\omega})$, then:*

$$\omega \triangleleft \sigma \rightsquigarrow \tilde{\omega}$$

over the concept alphabet $\boldsymbol{A} \cup \boldsymbol{V}$ and then:

$$\omega \rightsquigarrow \tilde{\omega}$$

over the concept alphabet $\boldsymbol{A}$.

Proof. This is an extension of Lemma 5: analogously to the previous case the result holds because concept and variable names are distinct sets and variables can be managed as auxiliary concept names. The only issue is related to the definition of the variable assignment, that forces each variable to be bound to exactly one element at time, but such a kind of constraint is realized by cardinality restriction axioms. □

Theorem 12. *Given a world specification $\mathcal{W}$, a query $Q(\boldsymbol{V})$ is an access function iff the following implication holds:*

$$\tilde{KB}^{\boldsymbol{V}} \wedge \tau(\mathcal{W}) \models \sharp(\tau(Q(\boldsymbol{V}))) \leq 1$$

Proof. Assuming that the implication holds, but by contradiction that $Q(\mathbf{V})$ is not an access function, let ω be a legal world state and let σ be a consistent assignment on it s.t., the evaluation of the query contains more than one element.

Since Theorem 3, using the function $\mu^{\mathbf{V}}$ we can embed the world model into a structure that is surely a model for the $\tau(\mathcal{W}) \wedge \tilde{KB}$ axioms except the newly introduced ones ($\tilde{KB}^{\mathbf{V}} \setminus \tilde{KB}$). So that we can prove the claim, we need to show that also other axioms hold. Since the assignment σ is consistent, each variable V is mapped to an element of Δ^{ω}, but since $\mathsf{Top}^{\tilde{\omega}} = \Delta^{\omega}$, where $\tilde{\omega} = \mu^{\mathbf{V}}(\omega, \sigma)$, we have that:

$$\mu^{\mathbf{V}}(\omega, \sigma) \models V \sqsubseteq \mathsf{Top}$$

Furthermore, since each variable is assigned to only one element, given the definition of the mapping function:

$$\mu^{\mathbf{V}}(\omega, \sigma) \models \sharp(V) = 1$$

We have proved that $\mu^{\mathbf{V}}(\omega, \sigma)$ is a model of the knowledge base. By hypothesis the implication holds, so:

$$\mu^{\mathbf{V}}(\omega, \sigma) \models \sharp(Q(\mathbf{V})) \leq 1$$

But according to Theorem 10 we have that $\omega \triangleleft \sigma \rightsquigarrow \tilde{\omega}$ and, applying Theorem 2, we can conclude that:

$$Q(\mathbf{V})^{\omega \triangleleft \sigma} = \tau(Q(\mathbf{V}))^{\tilde{\omega}}$$

or, in other words, that:

$$\omega \triangleleft \sigma \models \sharp(Q(\mathbf{V})) \leq 1$$

Since the definition of query evaluation we have that:

$$\|Q^{\omega}(\sigma_{\mathbf{V}})\| \leq 1$$

Assuming that $Q(\mathbf{V})$ is an access function, but by contradiction, that the implication does not hold, let ω' be a model of the knowledge base s.t.:

$$\omega' \not\models \sharp(\tau(Q(\mathbf{V}))) \leq 1$$

or, in other words, that:

$$\left\| \tau(Q(\mathbf{V}))^{\omega'} \right\| > 1$$

Applying the $\pi^{\mathbf{V}}$ project function, we obtain a pair $\langle \omega, \sigma \rangle$ s.t.

- σ is consistently defined, since $\omega' \models V \sqsubseteq \mathsf{Top}, \sharp(V) = 1$;
- according to Theorem 11 and to Theorem 4, since $\omega' \models \tau(\mathcal{W}) \wedge \tilde{KB}$ the world state ω is legal;
- applying result of Theorem 2 we can conclude also than:

$$Q(\mathbf{V})^{\omega \triangleleft \sigma} = \tau(Q(\mathbf{V}))^{\omega'}$$

Since the query is an access function, the evaluation of the query provided the pair ω, σ is s.t.:

$$\|Q^{\omega}(\sigma)\| \leq 1$$

Applying the definition of query evaluation, we can establish that:

$$\left\|\tau(Q(\mathbf{V}))^{\omega'}\right\| \leq 1$$

proving the claim. □

Proposition 1. *Given an implication $\phi \models \psi$ where ϕ and ψ are $\mathcal{C}^2$ sentences, the problem of checking whether the implication holds is in* coNEXP.

Proof. Since the language is closed under negation, the implication problem can be easily reduced to an instance of the corresponding unsatisfiability problem of the sentence:

$$\phi \wedge \neg\psi$$

that is solvable in coNEXP, since according to [PH05] the satisfiability problem is in NEXP, even allowing for succinct number encoding. □

Now, we can provide a basic complexity result regarding the problem of deciding whether a query is an access function.

Theorem 13. *Given a world specification $\mathcal{W}$ and a query $Q(\boldsymbol{V})$, the problem of checking if the query is an access function is in* coNEXP.

Proof. In order to solve the problem, we can apply the property shown in Theorem 12, reducing it to a reasoning task in $\mathcal{C}^2$ logics.

The encoding is clearly linear in the size of the input, while, the corresponding implication problem is in coNEXP (see Proposition 1), since we are looking for a counterexample as a model of the sentence ϕ defined as:

$$\begin{aligned}\phi \triangleq & \bigwedge_{C \sqsubseteq D \in \tilde{KB}^{\mathbf{V}} \cup \tau(\mathcal{W})} \forall x.\pi_x(C)(x) \rightarrow \pi_x(D)(x) \\ & \wedge \bigwedge_{o:C \in \tilde{KB}^{\mathbf{V}} \cup \tau(\mathcal{W})} \pi_x(C)(o) \wedge \exists^{>1} x.\pi_x(\tau(Q(\mathbf{V})))(x)\end{aligned}$$

Where π_x is a standard translation function from Description Logics expressions to first-order predicate logic formulas in the variable x introduced in Section 3.2.2. In other words, we need to check that the sentence ϕ is not satisfiable, hence we have to solve an instance of UNSAT$\mathcal{C}^2$. □

So far we have presented and analyzed the properties of the query language of the framework and the usage of variable symbols. Now, we have also to show how they can be used in order to deal with object instantiation: since newly created objects are essentially anonymous, we need to name them so that we are able to referring them in subsequent operations. So we introduce a special kind of variable, whose values are bound to newly instantiated objects.

Definition 14 (Instantiation assignment)**.** *Given a set of variable names $\boldsymbol{V}$ and domain Δ, s.t. $\|\boldsymbol{V}\| \leq \|\mathfrak{U} \setminus \Delta\|$, an instantiation assignment $\sigma'_{\boldsymbol{V}}$ consistent w.r.t. the provided domain is an injective function from $\boldsymbol{V}$ to the set $\mathfrak{U} \setminus \Delta$.*

Lemma 11. *Given a set of variable names $\boldsymbol{V}$ and world state ω, an instantiation assignment for $\boldsymbol{V}$ exists iff the world state has room for at least $\|\boldsymbol{V}\|$ newly created elements.*

Proof. Let n be the cardinality of the set $\delta = \mathfrak{U} \setminus \Delta$, if $n \geq \|\mathbf{V}\|$ then it is possible to assign to each variable name V a distinct element in δ.

If $\sigma'_{\mathbf{V}}$ is an instantiation assignment the codomain contains exactly $n = \|\mathbf{V}\|$ elements (it is an injective function), but by the definition we have also that $cod(\sigma'_{\mathbf{V}}) \subseteq \delta$. □

Like ordinary variables, also instantiation variables can be accommodated as singleton concept names, but their assignment is outside the current active domain. Since each instantiation variable is a new distinct object, the assignment function must map different names to different instances, i.e., must be an injective function. Essentially, the new interpretation domain must grow in order to include at least the new elements defined, but since we need to enforce a minimal-change semantics, we need that the expansion is as smallest as possible, in terms of set cardinality.

Definition 15 (Extended interpretation with instantiation)**.** *Given a model, i.e., a world state, $\omega = \langle \Delta^{\omega}, \cdot^{\omega} \rangle$, a variable assignment $\sigma_{\boldsymbol{X}}$ on Δ^{ω}, and a consistent instantiation assignment $\sigma'_{\boldsymbol{Y}}$, s.t. $cod(\sigma'_{\boldsymbol{Y}}) \cap \Delta^{\omega} \subseteq \emptyset$ and $\boldsymbol{X} \cap \boldsymbol{Y} \subseteq \emptyset$, the extended interpretation $\omega \lhd \langle \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ is a new structure s.t.:*

- *the interpretation domain $\Delta^{\omega'}$ is the smallest subset of $\mathfrak{U}$ s.t.:*

$$\Delta^{\omega} \cup cod(\sigma'_{\boldsymbol{Y}}) \subseteq \Delta^{\omega'}$$

- *the interpretation function $\cdot^{\omega'}$ is the same as ω for each name not in $\boldsymbol{X} \cup \boldsymbol{Y}$, while each $X \in \boldsymbol{X}$ is interpreted as:*

$$X^{\omega'} = \sigma_{\boldsymbol{X}}(X)$$

and each $Y \in \boldsymbol{Y}$ is interpreted as:

$$Y^{\omega'} = \sigma'_{\boldsymbol{Y}}(Y)$$

Lemma 12. *Given a model, i.e., a world state, $\omega = \langle \Delta^{\omega}, \cdot^{\omega} \rangle$, a variable assignment $\sigma_{\boldsymbol{X}}$ on Δ^{ω}, a consistent instantiation assignment $\sigma'_{\boldsymbol{Y}}$, s.t. $cod(\sigma'_{\boldsymbol{Y}}) \cap \Delta^{\omega} \subseteq \emptyset$ and $\boldsymbol{X} \cap \boldsymbol{Y} \subseteq \emptyset$. Then, the domain of the extended interpretation is:*

$$\Delta^{\omega'} = \Delta^{\omega} \cup cod(\sigma'_{\boldsymbol{Y}})$$

Proof. According to the definition $\Delta^{\omega} \cup cod(\sigma'_{\mathbf{Y}}) \subseteq \Delta^{\omega'}$, therefore we need to show only that $\Delta^{\omega'} \subseteq \Delta^{\omega} \cup cod(\sigma'_{\mathbf{Y}})$. By contradiction, if there exists an element x s.t. $x \in \Delta^{\omega'}$ and $x \notin \Delta^{\omega} \cup cod(\sigma'_{\mathbf{Y}})$, we can remove it from ω' obtaining a new structure ω'' where all properties of extended interpretation holds but that has smaller interpretation domain than ω'. $\square$

Lemma 13. *Given a model, i.e., a world state, $\omega = \langle \Delta^{\omega}, \cdot^{\omega} \rangle$, a variable assignment $\sigma_{\boldsymbol{X}}$ on Δ^{ω}, a consistent instantiation assignment $\sigma'_{\boldsymbol{Y}}$, s.t. $\boldsymbol{X} \cap \boldsymbol{Y} \subseteq \emptyset$, then there exists only one extended interpretation.*

Proof. Due the previous lemma we can observe that either the extension of the interpretation domain and the extension of the interpretation function are defined in a completely deterministic way[6]. $\square$

Theorem 14. *Given a world state ω and a variable assignment $\sigma_{\boldsymbol{X}}$, let σ'_1 and σ'_2 be two distinct instantiation assignments for the same variable names $\boldsymbol{Y}$, s.t. $\boldsymbol{X} \cap \boldsymbol{Y} \subseteq \emptyset$. Then, the two structures ω'_1 and ω'_2 obtained as extended interpretations are isomorphic.*

Proof. In order to prove the claim we need to provide a bijective function $h : \Delta^{\omega'_1} \mapsto \Delta^{\omega'_2}$ s.t.:

- for each unary predicate name $p \in \mathbf{A} \cup \mathbf{X} \cup \mathbf{Y}$ and for each element $x \in \Delta^{\omega'_1}$, $x \in p^{\omega'_1} \leftrightarrow h(x) \in p^{\omega'_2}$;
- for each binary predicate name $r \in \mathbf{P}$ and for each element pair $\langle x, y \rangle \in \Delta^{\omega'_1} \times \Delta^{\omega'_1}$, $\langle x, y \rangle \in r^{\omega'_1} \leftrightarrow \langle h(x), h(y) \rangle \in r^{\omega'_2}$;
- for each object name $o \in \mathbf{O}$ and for each element $x \in \Delta^{\omega'_1}$, $x = o^{\omega'_1} \leftrightarrow h(x) = o^{\omega'_2}$.

Such a function can be defined as:

$$h(x) = \begin{cases} x & x \in \Delta^{\omega} \\ \sigma'_2(Y) & \sigma'_1(Y) = x \end{cases}$$

This function is well-defined since it is total ($dom(h) = \Delta^{\omega'_1} = \Delta^{\omega} \cup cod(\sigma'_1)$) and it maps each element of its domain to exactly one element of its codomain.

It is also a surjective function, in fact, assuming that exists an element $y \in \Delta^{\omega'_2}$ s.t. $y \notin cod(h)$ we can distinguish two cases:

- $y \in \Delta^{\omega}$, but by definition $h(y) = y$, then $y \in cod(h)$;
- $y \in cod(\sigma'_2)$, then exists a variable name Y s.t. $\sigma'_2(Y) = y$, but there also exists an element x s.t. $\sigma'_1(Y) = x$, hence $y = h(x)$.

To show that the function is also injective we consider three main cases (there are four cases, but one can be ignored for symmetry). By contradiction we assume that exist two distinct elements x_1 and x_2 s.t. $h(x_1) = h(x_2)$:

1. both elements belong to Δ^{ω}: the function h restricted to this domain is the identity function, $x_1 \neq x_2$ and, consequently, $h(x_1) = x_1 \neq x_2 = h(x_2)$;

[6]The argument is quite similar to this one used in the proof of Lemma 3.

2. both elements belong to $\Delta^{\omega'_1} \setminus \Delta^{\omega}$: since $x_1 \neq x_2$ then there exists two variable names Y_1 and Y_2 s.t. $\sigma'_1(Y_i) = x_i$, but, since σ'_2 is also injective $h(x_1) = \sigma'_2(Y_1) \neq \sigma'_2(Y_2) = h(x_2)$;

3. $x_1 \in \Delta^{\omega}$ and $x_2 \in \Delta^{\omega'_1} \setminus \Delta^{\omega}$: by definition $h(x_1) \in \Delta^{\omega}$, while $h(x_2) \in \Delta^{\omega'_2} \setminus \Delta^{\omega}$, then $h(x_1) \neq h(x_2)$.

We have shown that the function h is a bijection between the two interpretation domains, therefore we need to prove that it is also an homomorphism.

Let o be an object name interpreted as $o^{\omega} \in \Delta^{\omega}$, since the definition of extended interpretation we have also that:

$$o^{\omega} = o^{\omega'_1} = o^{\omega'_2}$$

Since $o^{\omega} \in \Delta^{\omega}$, then $h(o^{\omega}) = o^{\omega}$.

Let r be a binary relation name in $\mathbf{P}$, also in this case, due the definition of extended interpretation, we can establish that:

$$r^{\omega} = r^{\omega'_1} = r^{\omega'_2}$$

For each pair $\langle x, y\rangle \in \Delta^{\omega} \times \Delta^{\omega}$ we have that $\langle x, y\rangle = \langle h(x), h(y)\rangle$ and since $r^{\omega} \subseteq \Delta^{\omega} \times \Delta^{\omega}$, we can conclude that $\langle x, y\rangle \in r^{\omega'_1} \leftrightarrow \langle h(x), h(y)\rangle \in r^{\omega'_2}$. For other element pairs, we remark that:

- $((\Delta^{\omega'_1} \times \Delta^{\omega'_1}) \setminus (\Delta^{\omega} \times \Delta^{\omega})) \cap r^{\omega'_1} \subseteq \emptyset$ and $((\Delta^{\omega'_2} \times \Delta^{\omega'_2}) \setminus (\Delta^{\omega} \times \Delta^{\omega})) \cap r^{\omega'_2} \subseteq \emptyset$,
- given a pair $\langle x, y\rangle \in (\Delta^{\omega'_1} \times \Delta^{\omega'_1}) \setminus (\Delta^{\omega} \times \Delta^{\omega})$, the corresponding pair $\langle h(x), h(y)\rangle$ is contained in $(\Delta^{\omega'_2} \times \Delta^{\omega'_2}) \setminus (\Delta^{\omega} \times \Delta^{\omega})$.

In other words, the other pairs cannot be contained into the extension of the relation as their images through the mapping h. The argumentation is exemplified in Figure 3.2.

For unary predicate names in $\mathbf{A} \cup \mathbf{X}$, a similar argument holds, while for unary predicate names in $\mathbf{Y}$ the homomorphism property is enforced by the definition of the mapping function h itself. □

We have proved that given a world state and a variable assignment, the instantiation assignment is quite irrelevant, since all valid extended interpretations are isomorphic at least w.r.t. the domain alphabet. In other words, every interpretation is indistinguishable from each other using the $\mathcal{ALCQI}$ language: as shown in [IL90] and related works, $\mathcal{C}^2$ is invariant under the *2-pebble game with counting*, which is a weaker condition than isomorphism. Moreover, since the object names are restricted to the common domain subset, these structures are also indistinguishable using nominal-enriched description logic or first-order languages.

Given a domain specification $\langle \mathbf{A}, \mathbf{P}, \mathbf{O}\rangle$, a set of variable names $\mathbf{X}$ and the corresponding knowledge base $\tilde{KB}^{\mathbf{X}}$, let $\mathbf{Y}$ be a set of instantiation variable names, s.t. $\mathbf{X} \cap \mathbf{Y} \subseteq \emptyset$, we define a new knowledge base $\tilde{KB}^{\mathbf{X},\mathbf{Y}}$ on the alphabet $\langle \mathbf{A} \cup \mathbf{X} \cup \mathbf{Y} \cup \{\mathsf{Top}, \mathsf{Top}', \mathsf{New}\}, \mathbf{P}, \mathbf{O}\rangle$ adding axioms resulting from the instantiation of axiom templates in the schema $\Delta KB^{I}(\mathsf{Top}')$ presented in Table 3.5.

We accordingly extend also the definition of $\mu^{\mathbf{V}}$, introducing the new mapping function $\mu^{\mathbf{X},\mathbf{Y}}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$ s.t.:

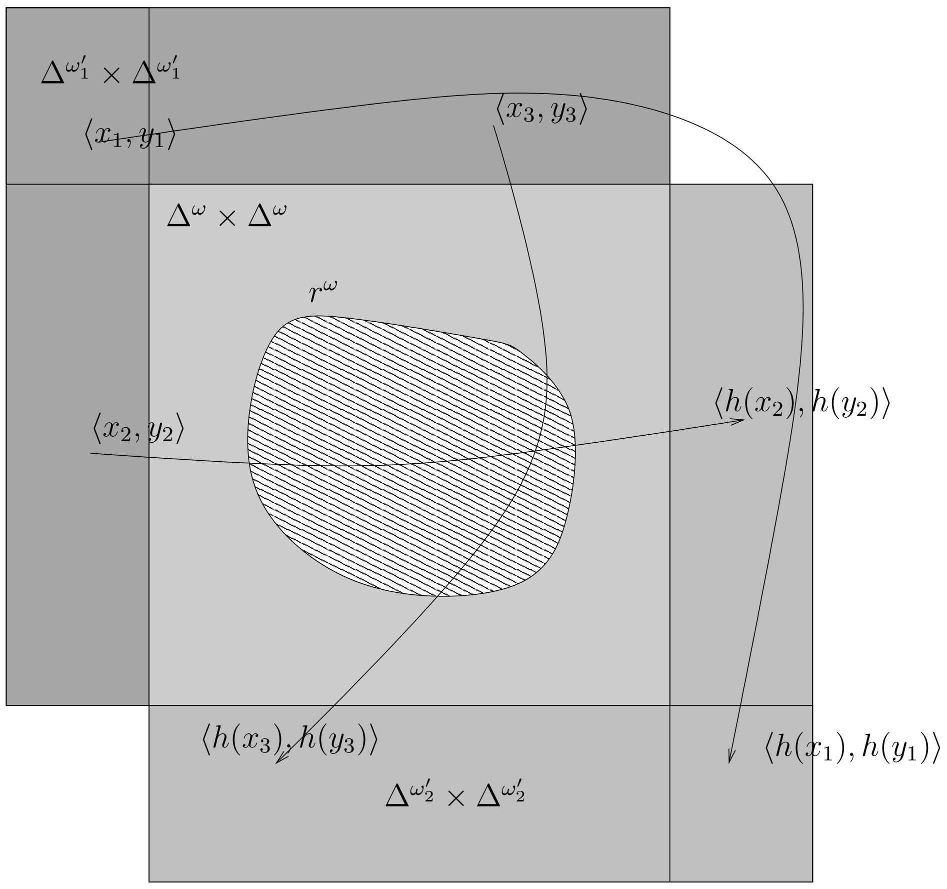
$\Delta^{\omega'_1} \times \Delta^{\omega'_1}$
$\langle x_1, y_1 \rangle$
$\langle x_3, y_3 \rangle$
$\Delta^{\omega} \times \Delta^{\omega}$
r^{ω}
$\langle x_2, y_2 \rangle$
$\langle h(x_2), h(y_2) \rangle$
$\langle h(x_3), h(y_3) \rangle$
$\langle h(x_1), h(y_1) \rangle$
$\Delta^{\omega'_2} \times \Delta^{\omega'_2}$

Figure 3.2: An example of the isomorphism h applied to a binary relation r

Table 3.5: The axiom schema $\Delta KB^I(\mathsf{Top}')$

$$\mathsf{Top}' \equiv \mathsf{Top} \sqcup \bigsqcup_{Y \in \mathbf{Y}} Y$$

$$\bigwedge_{Y \in \mathbf{Y}} Y \sqsubseteq \mathsf{Top}' \sqcap \neg \mathsf{Top}$$

$$\bigwedge_{Y \in \mathbf{Y}} \sharp(Y) = 1$$

$$\bigwedge_{Y \in \mathbf{Y}, Y' \in \mathbf{Y}, Y \neq Y'} Y \sqcap Y' \sqsubseteq \bot$$

- the interpretation domain is the whole universe ($\Delta^{\tilde{\omega}} = \mathfrak{U}$);
- the interpretation of concepts, roles and objects is preserved ($N^{\omega} = N^{\tilde{\omega}}$);
- the interpretation of Top is the active domain of ω ($\mathsf{Top}^{\tilde{\omega}} = \Delta^{\omega}$);
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;
- the interpretation of Top' is the interpretation domain of ω', where ω' is the extended interpretation $\omega \lhd \langle \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$;
- the interpretation of variable auxiliary concepts is defined according to the assignment ($\sigma(V) = V^{\tilde{\omega}}$).

We can also define an inverse projection function $\pi^{\mathbf{X},\mathbf{Y}}$ that, given a structure $\tilde{\omega}$ that is a model for the new knowledge base, returns a triple $\langle \omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$ s.t. $\tilde{\omega} = \mu^{\mathbf{X},\mathbf{Y}}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$.

Theorem 15. *Given a world state ω having room for at least $\|\boldsymbol{Y}\|$ new elements, and a consistent variable assignment $\sigma_{\boldsymbol{X}}$, then for any consistent instantiation assignment $\sigma'_{\boldsymbol{Y}}$, s.t. $cod(\sigma'_{\boldsymbol{Y}}) \cap \Delta^{\omega} \subseteq \emptyset$ and $\boldsymbol{X} \cap \boldsymbol{Y} \subseteq \emptyset$, the interpretation $\mu^{\boldsymbol{X},\boldsymbol{Y}}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$ is a model of the knowledge base $\tilde{KB}^{\boldsymbol{X},\boldsymbol{Y}}$.*

Proof. Given the hypothesis of enough room in the set $\mathfrak{U} \setminus \Delta^{\omega}$, such an instantiation assignment exists. Let $\sigma'_{\mathbf{Y}}$ be such an instantiation assignment: according to Theorem 3, using the function μ we can embed the world model into a structure that is surely a model for the $\tilde{KB} \subset \tilde{KB}^{\mathbf{X},\mathbf{Y}}$ axioms. In order to prove the claim we need to show that also other axioms hold. Since the assignment σ is consistent, each variable $X \in \mathbf{X}$ is mapped to an element of Δ^{ω}, but since $\mathsf{Top}^{\tilde{\omega}} = \Delta^{\omega}$, where $\tilde{\omega} = \mu^{\mathbf{X},\mathbf{Y}}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$, we have that:

$$\tilde{\omega} \models X \sqsubseteq \mathsf{Top}$$

Furthermore, since each variable is assigned to only one element, given the definition of the mapping function:

$$\tilde{\omega} \models \sharp(X) = 1$$

We notice, given the definition of mapping function, that:

$$\mathsf{Top}'^{\tilde{\omega}} = \Delta^{\omega'}$$

Applying Lemma 12, we obtain that:

$$\mathsf{Top}'^{\tilde{\omega}} = \Delta^{\omega} \cup cod(\sigma'_{\mathbf{Y}})$$

According to the definition of the mapping function for the concept Top:

$$\mathsf{Top}'^{\tilde{\omega}} = \mathsf{Top}^{\tilde{\omega}} \cup cod(\sigma'_{\mathbf{Y}}) \tag{3.2}$$

Since the definition of codomain of a function, we have that:

$$cod(\sigma'_{\mathbf{Y}}) = \bigcup_{Y \in \mathbf{Y}} \{\sigma'_{\mathbf{Y}}(Y)\}$$

Applying the definition of mapping function for variable names, we establish that:

$$cod(\sigma'_{\mathbf{Y}}) = \bigcup_{Y \in \mathbf{Y}} Y^{\tilde{\omega}}$$

In other words, replacing back into Eq. 3.2:

$$\mathsf{Top}'^{\tilde{\omega}} = \mathsf{Top}^{\tilde{\omega}} \cup \bigcup_{Y \in \mathbf{Y}} Y^{\tilde{\omega}}$$

Hence, applying the standard semantics, we obtain that:

$$\tilde{\omega} \models \mathsf{Top}' \equiv \mathsf{Top} \sqcup \bigsqcup_{Y \in \mathbf{Y}} Y$$

Since also the instantiation assignment is valid, each variable $Y \in \mathbf{Y}$ is assigned to a distinct element, thus we have that:

$$\tilde{\omega} \models \sharp(Y) = 1$$

and, given the injectivity property of the assignment:

$$\tilde{\omega} \models Y \sqcap Y' \sqsubseteq \bot$$

for any pair $Y \in \mathbf{Y}, Y' \in \mathbf{Y}$ s.t. $Y \neq Y'$. We point out that the result does not rely on the specific instantiation assignment, so it can be generalized to any consistent one, demonstrating the claim. $\square$

Corollary 2. *Given a world state ω having room for at least $\|\boldsymbol{Y}\|$ new elements, and a consistent variable assignment $\sigma_{\boldsymbol{X}}$, the knowledge base $\tilde{KB}^{\boldsymbol{X},\boldsymbol{Y}}$ is satisfiable.*

Theorem 16. *Let $\tilde{\omega}$ be a model of the knowledge base $\tilde{KB}^{\boldsymbol{X},\boldsymbol{Y}}$ and let $\langle \omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ be a triple s.t. $\tilde{\omega} = \mu^{\boldsymbol{X},\boldsymbol{Y}}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then:*

$$\omega \rightsquigarrow \tilde{\omega}$$

on the concept alphabet $\boldsymbol{A}$, and:

$$\omega \lhd \sigma_{\boldsymbol{X}} \rightsquigarrow \tilde{\omega}$$

on the concept alphabet $\boldsymbol{A} \cup \boldsymbol{X}$, and:

$$\omega \lhd \langle \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle \rightsquigarrow \tilde{\omega}$$

Proof. Since $\tilde{\omega}$ is a model for the knowledge base $\tilde{KB}^{\mathbf{X},\mathbf{Y}}(\mathsf{Top}')$ and it strictly contains the corresponding knowledge base $\tilde{KB}^{\mathbf{X}}$ we can apply Theorem 11 to prove the claim. We point out that axioms related to cardinality and disjointness enforce the interpretation of variable names to be suitable as instantiation variable assignment, since each concept Y is interpreted as distinct singleton element over the domain $\mathsf{Top}'^{\tilde{\omega}} \setminus \mathsf{Top}^{\tilde{\omega}}$. In order obtain such result we need also to show that:

$$\langle \omega, \sigma_{\mathbf{X}} \rangle = \pi^{\mathbf{X}}(\tilde{\omega})$$

but, given the definition of mapping and corresponding projection functions $\pi^{\mathbf{X}}$ and $\pi^{\mathbf{X},\mathbf{Y}}$, they are identical excluding the part involving instantiation variables, which in this case can be ignored.

To prove the last claim, since the assignment are well-founded, we need only to show that the interpretation domain is also well-founded. According to Lemma 12 the following constraint must hold:

$$\Delta^{\omega'} = \Delta^{\omega} \cup cod(\sigma'_{\mathbf{Y}})$$

but since:

$$\tilde{\omega} \models \mathsf{Top}' \equiv \mathsf{Top} \sqcup \bigsqcup_{Y \in \mathbf{Y}} Y$$

hence, applying the mapping function definition ($\mathsf{Top}'^{\tilde{\omega}} = \Delta^{\omega'}$, $\mathsf{Top}^{\tilde{\omega}} = \Delta^{\omega}$ and $\sigma(X) = X^{\tilde{\omega}}$), it follows that:

$$\Delta^{\omega'} = \Delta^{\omega} \cup \bigcup_{Y \in \mathbf{Y}} \{\sigma'_{\mathbf{Y}}(Y)\}$$

proving the claim. □

3.6 Conclusions

In this chapter we have introduced the basic elements on which the framework is built.

In particular, we have provided the definition of the meta-model in terms of syntax and semantics that allows to setup a problem instance related to the analysis of a service community in a given context by the means of domain and world specification. Moreover, the adopted semantics is the standard foundation of Description Logics and it is mainly compatible with the approaches adopted in the Semantic Web, while it is also able to deal with incomplete information, since it is based on the open-world assumption. We have also introduced primitive language constructs that we will employ in the following to define e-services and their properties.

CHAPTER 4

Simple e-services

In this chapter we introduce a basic type of semantic e-service in order to present main properties related to the analysis of behavioral consistency starting from the model depicted in Chapter 3.

A simple e-service is essentially an atomic update operator, described following the IOPE (inputs, outputs, preconditions and effects) paradigm, that alters the world state into a new one according to its own definition.

In a more general setting, an e-service can declare multiple possible effects, which are non-deterministically selected in order to mimic the black-box behavior of the service provider. In this simple version, we assume that an e-service has only one possible effect defined in its contract[1], which is realized once the service is invoked in a consistent way.

The model based approach to the update, combined with the minimal-change semantics, allows to easily cope with the frame problem in a fashion quite similar to other approaches adopted in the literature. For simple e-services, we exclude the possibility of any side-effect (i.e., an indirect effect implied by the world specification), that can potentially interfere with previous assumptions. Such effects are admitted in several other proposals, generally stemming from [Win90], where, in order to ensure that the system state is always legal, even after an update, some model repairs are transparently performed by the agent. However, we will accordingly remove this limitation in the following, allowing for some kind of update repair strategy: we aim at obtaining both a suitable (e.g., restricting the repair search space to a neighborhood of the update operator in terms of underlying models) and decidable solution. As we show in the rest of the chapter, these properties are tightly bounded.

In order to reason about e-services specification employing the first-order fragment $\mathcal{C}^2$, introduced in Section 3.2.1, we need to express the formalization of the update semantics in terms of a suitable logic theory. We start from a domain definition based upon a DL knowledge base that describes properties and constraints of an admissible state, intended in terms of extensions of the predicate symbols, and we aim at reasoning about the effects of e-service exe-

[1]However, some minor forms of non-determinism are also allowed in simple e-services.

cutions also denoted in terms of predicate interpretation extensions. In other words, it is possible to reify the update introducing new concepts and relations whose extensions correspond to the ones in the state resulting from the update. In order to achieve this result, we rely upon the notion of embedding a world state structure, or a world state transition specification, into an interpretation of an *ad hoc* built theory, stating a correspondence between the semantic properties (satisfiability or entailment) of such a kind of logical formalization and the state transition system which is typically used to describe the semantics of an e-service.

4.1 Syntax

In this section we briefly present the definition of the syntactical primitives, which we will employ to define e-services in the rest of the work.

Definition 16 (Simple e-service)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a simple e-service specification S is a quadruple formed by:*

- *a (possibly empty) finite set of input variable names $\boldsymbol{X}_S$;*
- *a (possibly empty) finite set of output or instantiation variable names $\boldsymbol{Y}_S$;*
- *a (possibly empty) finite set of invocation precondition constraints $\mathcal{P}_S$;*
- *a simple effect E_S.*

Informally, according to the IOPE paradigm, a service is defined specifying the values required for its execution, the values resulting from the execution itself[2], the conditions under which it can be requested by a client, and the updates performed, if any.

Definition 17 (Service community)**.** *A service community is a set $\mathcal{S}$ of e-service specifications built over the same domain specification.*

Only services belonging to the same community can be directly compared or composed, otherwise, since different domain specifications can employ different names or assign different meanings to the same name, a preliminary reconciliation of the knowledge bases is required in order to combine them (e.g., object renaming, inter-schema assertion definition). In other words, a community of services agrees upon the semantics of shared specifications.

Definition 18 (Atomic precondition term)**.** *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$. An atomic precondition term, which is suitable for such a service, is a pair $\langle s, Q(\boldsymbol{X}) \rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the precondition (positive or negative);*
- *$Q(\boldsymbol{X})$ is a parameterized query over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

[2]Since we are specifically addressing world-altering rather than information-gathering services, the output values are related only to the action performed.

Definition 19 (Precondition constraint). *Let S be a service having the set $\boldsymbol{X}_S$ as its input variable names. An invocation precondition constraint P is any finite arbitrary set of atomic precondition terms on such input variables.*

Roughly speaking, a precondition constraint is a conjunction of positive (resp. negative) atomic conditions that are satisfied if the query result is not empty (resp. is empty) given an input variable assignment and a world state. A set of precondition constraints is interpreted as a disjunction of such constraints: the service invocation preconditions hold if at least a constraint is satisfied. In other words, the service preconditions are expressed in a kind of *disjunctive normal form*.

In the rest, we can employ also the following alternative syntax for the specification of conditions (e.g., preconditions):

$$C, C' \longrightarrow C \;\texttt{and}\; C' \mid C \;\texttt{or}\; C' \mid \texttt{not}\; C' \mid Q(\mathbf{X})$$

A condition is represented as an ordinary boolean expression having DL concept expressions as atoms, that is converted in the DNF[3].

Definition 20 (Positive effect argument). *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, and let $\boldsymbol{Y}_S$ be the corresponding output variable names. A positive effect argument is any element $Y \in \boldsymbol{Y}_S$ or any parameterized query $Q(\boldsymbol{X})$ over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

Definition 21 (Negative effect argument). *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$. A negative effect argument is any parameterized query $Q(\boldsymbol{X})$ over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

Definition 22 (Atomic concept effect). *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, and let $\boldsymbol{Y}_S$ be the corresponding output variable names. An atomic concept effect, that is suitable for such a service, is a triple $\langle s, A, a \rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the effect (insert or delete);*
- *$A \in \boldsymbol{A}$ is the target concept name;*
- *a is the argument of the update (positive or negative) according to the sign of the effect.*

Definition 23 (Atomic role effect). *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, and let $\boldsymbol{Y}_S$ be the corresponding output variable names. An atomic role effect, that is suitable for such a service, is a quadruple $\langle s, P, l, r \rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the effect (insert or delete);*
- *$P \in \boldsymbol{P}$ is the target role name;*
- *l and r are the arguments of the update (positive or negative) according to the sign of the effect.*

[3]We assume that the impact on the input specification size of this encoding is negligible. Whatever, provided complexity results are assuming the normal form encoding.

Definition 24 (Simple effect)**.** *A simple effect E for a service S is an arbitrary set of atomic concept and role effects built according to its variable names and domain specification.*

Roughly speaking, a simple effect specifies which elements (or pair of elements) are inserted or removed from a set (or from a binary relation). Generally each atomic effect can affect more elements since its domain is denoted using queries[4].

For the sake of notational simplicity we suppose that only a service S is taken into account, so it can be omitted from formulas when it is clear from the context.

Moreover, an atomic effects can be more written also using the notation:

$$+A(a), -A(a), +P(l, r), -P(l, r)$$

An idempotent effect is simply expressed as an empty effect set.

Example 2. *Let $\mathcal{W}$ be an axiomatization of a simple domain, where people interact with e-services provided by public administrations. $\mathcal{W}$ contains the assertions reported in Table 4.1, where concept and role names have the intuitive meaning. The same domain is also depicted as conventional Entity-Relationship diagram (ERD) in Figure 4.1.*

Table 4.1: An example world specification $\mathcal{W}$

$\exists \mathsf{residentIn}^-.\top$	$\sqsubseteq$	Town			
$\exists \mathsf{residentIn}.\top$	$\sqsubseteq$	$\mathsf{Citizen}$	$\mathsf{Citizen} \sqcap \mathsf{Town}$	$\sqsubseteq$	$\bot$
$\exists \mathsf{authorizedFor}.\top$	$\sqsubseteq$	$\mathsf{Citizen}$	$\mathsf{Citizen} \sqcap \mathsf{Good}$	$\sqsubseteq$	$\bot$
$\mathsf{Citizen}$	$\sqsubseteq$	$(= 1\ \mathsf{residentIn}\ \top)$	$\mathsf{Good} \sqcap \mathsf{Town}$	$\sqsubseteq$	$\bot$
$\exists \mathsf{locatedIn}^-.\top$	$\sqsubseteq$	Town	$\mathsf{Shop} \sqcap \mathsf{Town}$	$\sqsubseteq$	$\bot$
Shop	$\sqsubseteq$	$(= 1\ \mathsf{locatedIn}\ \top)$	$\mathsf{Shop} \sqcap \mathsf{Good}$	$\sqsubseteq$	$\bot$
$\exists \mathsf{registeredIn}^-.\top$	$\sqsubseteq$	Town	$\mathsf{Shop} \sqcap \mathsf{Citizen}$	$\sqsubseteq$	$\bot$
$\mathsf{Vehicle}$	$\sqsubseteq$	$(= 1\ \mathsf{registeredIn}\ \top)$	$\mathsf{Vehicle}$	$\sqsubseteq$	Good
$\exists \mathsf{owner}^-.\top$	$\sqsubseteq$	$\mathsf{Citizen}$	town_1	:	Town
Shop	$\sqsubseteq$	$(\leq 1\ \mathsf{owner}\ \top)$	town_2	:	Town
Good	$\sqsubseteq$	$(\leq 1\ \mathsf{owner}\ \top)$			

According to the given axiomatization, each good has an owner, while vehicles must be registered to the local administrative department. Suppose, for example, that there exists a service S that allows a citizen to change its own residence and to specify the new one. It can be modeled as:

$$\boldsymbol{X} = \{x_1, x_2\}$$
$$\boldsymbol{Y} = \emptyset$$
$$\mathcal{P} = x_1 \sqcap \mathsf{Citizen}\ \mathsf{and}\ x_2 \sqcap \mathsf{Town}$$
$$E = \big\{-\mathsf{residentIn}(x_1, \exists \mathsf{residentIn}^-.x_1), +\mathsf{residentIn}(x_1, x_2)\big\}$$

[4]Intuitively the approach aims at mimicking the `UPDATE ... WHERE ...`, `DELETE ... WHERE ...` and `SELECT ... INTO ...` primitives of the SQL language.

The input parameters x_1 and x_2 denote, respectively, the citizen who is asking for the change and the new residence town. This service is accessible by every citizen, and allows to select any town as the new residence location. The town town_1 *provides also an enhanced version to its inhabitants that ask for a residence change. Such enriched e-service S_1 is restricted only to citizens of* town_1*, but it is capable also to accordingly change the registration of vehicles belonging to the requestor, in the sense that the vehicles belonging to the requestor will be registered to the authority of the new town. Formally speaking, the service is defined as:*

$$\begin{aligned}
&\boldsymbol{X} = \{x_1, x_2\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x_1 \sqcap \exists \mathsf{residentIn}.\{\mathsf{town}_1\} \text{ and } x_2 \sqcap \mathsf{Town}\\
&E = \left\{-\mathsf{residentIn}(x_1, \exists \mathsf{residentIn}^{-}.x_1), +\mathsf{residentIn}(x_1, x_2)\right\}\\
&\quad \cup \{-\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, \{\mathsf{town}_1\})\}\\
&\quad \cup \{+\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, x_2)\}
\end{aligned}$$

As in the previous case, the update effect is obtained by combining insert and delete primitives.

4.2 Semantics

In this section, we define the semantics of e-service primitives previously presented and the definition of the system state transition resulting from a service enactment. We also analyze the accessibility of a service and its correctness in terms of fulfillment of service contract w.r.t. the enforcing of world constraints.

We start defining the service access condition and the related decision problem, reducing it to the satisfiability check of a $\mathcal{C}^2$ theory.

Definition 25 (Precondition satisfaction). *Let S be a service, let ω be a world state and $\sigma_{\boldsymbol{X}}$ an input assignment. We say that the service preconditions of S hold in ω w.r.t. $\sigma_{\boldsymbol{X}}$, iff there exists at least a precondition $P \in \mathcal{P}$ s.t. for each atomic precondition $\langle s, Q(\boldsymbol{X})\rangle \in P$:*

- *if $s = +$, then the evaluation of the query Q in ω using σ is not empty;*
- *or, if $s = -$, then the evaluation of the query Q in ω using σ is empty.*

We can denote this condition as $\omega \models \mathcal{P}(\sigma_{\boldsymbol{X}})$.

Definition 26 (Accessible e-service). *A service S is accessible, w.r.t. a world specification $\mathcal{W}$, in a legal world state ω iff:*

- *there is room enough for newly created objects, i.e., ($\|\boldsymbol{Y}_S\| \leq \|\mathfrak{U} \setminus \Delta^{\omega}\|$);*
- *there exists at least an input assignment $\sigma_{\boldsymbol{X}}$ s.t. service preconditions are satisfied in the initial state ω w.r.t. the input variable assignment $\sigma_{\boldsymbol{X}}$.*

Given a service specification S, we can build a knowledge base KB^P combining axioms of knowledge bases $\tilde{KB}^n$ and $\tilde{KB}^{\mathbf{V}}$, where n is the size of the set $\mathbf{Y}_S$ and $\mathbf{V} = \mathbf{X}_S$, adding the following boolean axioms:

$$\bigvee_{P \in \mathcal{P}_S} \bigwedge_{p \in P} \gamma(p)$$

where γ is a function defined as:

$$\gamma(\langle s, Q(\mathbf{X})\rangle) \triangleq \begin{cases} \alpha_p : \tau(Q) & \text{if } s = + \\ \tau(Q) \sqsubseteq \bot & \text{if } s = - \end{cases} \tag{4.1}$$

being α_p a new fresh constant name not appearing elsewhere.

Remark 8. *We notice that we have built a knowledge base mixing freely TBox and ABox axioms: indeed, we consider such a knowledge base as a first-order predicate logic theory, so we are using the DL notation only as a more compact syntax for first-order sentences. That is meaning, as previously shown, that a TBox GCI assertion $C \sqsubseteq D$ must be read as the FOL sentence $\forall x.\pi_x(x) \rightarrow \pi_x(D)(x)$, where π_x as the standard translation function from Description Logics expression language to firs-order, and an ABox assertion $o : C$ must be read as the sentence $\pi_x(C)(o)$ or $\exists x.\pi_x(C)(x)$, if the object name appears nowhere in the rest of the theory.*

Theorem 17. *Given a state ω and an assignment σ, a service S is accessible from ω using σ iff there exists an interpretation ω^S s.t. $\langle \omega, \sigma\rangle = \pi^{\mathbf{V}}(\omega^S)$ and $\omega^S \models KB^P$*

Proof. Let ω^S be a model of the knowledge base KB^P: since $KB^P \supset \tilde{KB}^n$, according to Theorem 9, the structure ω obtained using the projection π is a world state with room enough for new objects. Since $KB^P \supset \tilde{KB}^{\mathbf{V}}$, we can establish also that $\omega^S \models \tilde{KB}^{\mathbf{V}}$ and the variable assignment σ is well-founded, according to Theorem 11. In order to complete the proof we need to show that also preconditions hold. Let $P^* \in \mathcal{P}_S$ be a set of atomic preconditions s.t.:

$$\omega^S \models \bigwedge_{p \in P^*} \gamma(p)$$

Since we have assumed that ω^S is a model of the knowledge base at least an element satisfying these constraints must exist. For each $p \in P^*$, if the sign is positive ($s_p = +$) we have that:

$$\omega^S \models \alpha : \tau(Q_p)$$

which means that $[\tau(Q_p)]^{\omega^S} \neq \emptyset$; otherwise, if $s_p = -$ we can conclude that:

$$\omega^S \models \tau(Q_p) \sqsubseteq \bot$$

which means that $[\tau(Q_p)]^{\omega^S} = \emptyset$. Applying results of Theorems 11 and 2, we obtain that there exists a precondition P^* s.t. for each atomic precondition $\langle s, Q(\mathbf{X})\rangle \in P^*$:

- if $s = +$ the evaluation of the query is not empty, i.e. $Q^\omega(\sigma) = Q^{\omega \lhd \sigma} = [\tau(Q_p)]^{\omega^S} \neq \emptyset$;

- otherwise, if $s = -$, the evaluation of the query is empty, i.e., $Q^{\omega}(\sigma) = Q^{\omega \lhd \sigma} = [\tau(Q_p)]^{\omega^S} = \emptyset$.

Let ω be a world state and σ an assignment, s.t. service S is accessible from ω using σ, extending the mapping function $\mu^{\mathbf{V}}$ accordingly in order to keep into account also the interpretation of the object spy and of the role aux: then, we obtain a structure ω^S that, according to Theorems 8 and 10, satisfies the axioms of both knowledge bases $\tilde{KB}^n$ and $\tilde{KB}^{\mathbf{V}}$.

Since preconditions hold, there exists at least an element P^* s.t. for each atomic precondition $\langle s, Q(\mathbf{X})\rangle \in P^*$:

- if $s = +$ the evaluation of the query is not empty, i.e. $Q^{\omega}(\sigma) \neq \emptyset$;
- or, if $s = -$ the evaluation of the query is empty, i.e., $Q^{\omega}(\sigma) = \emptyset$.

Given the definition of query evaluation in terms of extended interpretation we have that:

$$Q^{\omega}(\sigma) = Q^{\omega \lhd \sigma}$$

Applying Theorem 10 and Theorem 2, we have that:

$$Q^{\omega \lhd \sigma} = [\tau(Q_p)]^{\omega^S}$$

since $\omega \lhd \sigma \rightsquigarrow \omega^S$. Consequently, for each constraint $p \in P^*$ having a positive sign, interpreting $\alpha_p^{\omega^S}$ as any element of the $[\tau(Q_p)]^{\omega^S}$ and, since it is not empty, such an element exists, we have that:

$$\omega^S \models \alpha_p : \tau(Q_p)$$

Analogously, for each constraint having a negative sign, since the interpretation is empty we can establish that:

$$\omega^S \models \tau(Q_p) \sqsubseteq \bot$$

Hence, ω^S is a model of KB^P. □

Given the previous result and the provided definition of accessible service, the following properties follow.

Corollary 3. *A service S is accessible w.r.t. a world specification $\mathcal{W}$ iff the knowledge base $KB^P \wedge \tau(\mathcal{W})$ is satisfiable.*

Theorem 18. *Given a world specification $\mathcal{W}$ and a simple e-service S, the problem of checking if S is accessible is in* NEXP.

Proof. According to Corollary 3 we can reduce the check of service accessibility to the satisfiability of a $\mathcal{C}^2$ sentence having a length linear in the size of the input specification (world and service). The result follows from the observation, already employed in showing other complexity results, that satisfiability check problem for this language is solvable non-deterministic exponential time. □

Now, we introduce the formal definitions related to the semantics of the state update adopted in our framework. Roughly speaking, we denote sets of elements or element pairs affected by an update specification defined using the syntax previously introduced. Assuming a minimal-change semantics, such definitions are employed in the following to provide the definition of the system state transition relation.

Definition 27 (Concept insert set)**.** *Let E be a simple service effect specification, $A \in \boldsymbol{A}$ a concept name, ω a world state, $\sigma_{\boldsymbol{X}}$ an input variable assignment consistently defined w.r.t. ω, the object insert set for the concept A is defined as:*

$$A^{+}(\omega, \sigma_{\boldsymbol{X}}) = \left(\bigcup_{\langle +, A, Q(\boldsymbol{X}) \rangle \in E} Q^{\omega}(\sigma_{\boldsymbol{X}}) \right) \setminus A^{\omega}$$

Definition 28 (Concept insert set with instantiation)**.** *Let E be a simple service effect specification, $A \in \boldsymbol{A}$ a concept name, ω a world state, $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ respectively an input and output variable assignment consistently defined w.r.t. ω. The object insert set for the concept A is defined as:*

$$A^{+}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) = \bigcup_{\langle +, A, Y \rangle \in E} \{\sigma'_{\boldsymbol{Y}}(Y)\} \cup A^{+}(\omega, \sigma_{\boldsymbol{X}})$$

The concept insert set denote the extension of the model that is affected by the update as element that will be added to the interpretation of the concept name A. Analogously, we can define the set of removed elements and affected pairs.

Definition 29 (Concept delete set)**.** *Let E be a simple service effect specification, $A \in \boldsymbol{A}$ a concept name, ω a world state, $\sigma_{\boldsymbol{X}}$ an input variable assignment consistently defined w.r.t. ω. The object delete set for the concept A is defined as:*

$$A^{-}(\omega, \sigma_{\boldsymbol{X}}) = \left(\bigcup_{\langle -, A, Q(\boldsymbol{X}) \rangle \in E} Q^{\omega}(\sigma_{\boldsymbol{X}}) \right) \cap A^{\omega}$$

Definition 30 (Role insert set)**.** *Let E be a simple service effect specification, $P \in \boldsymbol{P}$ a concept name, ω a world state, $\sigma_{\boldsymbol{X}}$ an input variable assignment consistently defined w.r.t. ω. The link insert set for the role P is defined as:*

$$P^{+}(\omega, \sigma_{\boldsymbol{X}}) = \left(\bigcup_{\langle +, P, Q(\boldsymbol{X}), Q'(\boldsymbol{X}) \rangle \in E} Q^{\omega}(\sigma_{\boldsymbol{X}}) \times Q'^{\omega}(\sigma_{\boldsymbol{X}}) \right) \setminus P^{\omega}$$

Definition 31 (Role insert set with instantiation)**.** *Let E be a simple service effect specification, $P \in \boldsymbol{P}$ a concept name, ω a world state, $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ respectively an input and output variable assignment consistently defined w.r.t. ω. The link insert set for the role P is defined as:*

$$\begin{aligned} P^{+}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) = & \bigcup_{\langle +, P, Y, Y' \rangle \in E} \{\langle \sigma'_{\boldsymbol{Y}}(Y), \sigma'_{\boldsymbol{Y}}(Y') \rangle\} \\ \cup & \bigcup_{\langle +, P, Y, Q(\boldsymbol{X}) \rangle \in E} \{\sigma'_{\boldsymbol{Y}}(Y)\} \times Q^{\omega}(\sigma_{\boldsymbol{X}}) \\ \cup & \bigcup_{\langle +, P, Q(\boldsymbol{X}), Y \rangle \in E} Q^{\omega}(\sigma_{\boldsymbol{X}}) \times \{\sigma'_{\boldsymbol{Y}}(Y)\} \cup P^{+}(\omega, \sigma_{\boldsymbol{X}}) \end{aligned}$$

Definition 32 (Role delete set)**.** *Let E be a simple service effect specification, $P \in \boldsymbol{P}$ a role name, ω a world state, $\sigma_{\boldsymbol{X}}$ an input variable assignment consistently defined w.r.t. ω, the link delete set for the role P is defined as:*

$$P^{-}(\omega, \sigma_{\boldsymbol{X}}) = \left(\bigcup_{\langle -, P, Q(\boldsymbol{X}), Q'(\boldsymbol{X}) \rangle \in E} Q^{\omega}(\sigma_{\boldsymbol{X}}) \times Q'^{\omega}(\sigma_{\boldsymbol{X}}) \right) \cap P^{\omega}$$

The provided definitions cover all cases that can occur in the specification of service effects using the previously defined language mixing different kind of update arguments. In order to provide a consistent definition of service effects, we need also to verify that, for every concept or role, the insert and delete sets are always distinct, as done in other similar approaches (e.g., [BLM+05a]) in the definition of consistent service specification. This constraint ensures that a service can never state that an element (or a link between elements) is, at the same time, added to a set and removed from it. In a more declarative fashion, we are interested only in services that have no contradicting effects, although alternative approaches can be devised imposing an ordering relation to effect specifications that are interpreted in a more procedural fashion (i.e., update statements). Since instantiation variables are taken into account only in insert statements (positive effect) and they cannot induce any clash, they can be safely ignored at this stage.

Definition 33 (Consistent simple e-service effect)**.** *Given a service S with a simple effect E, let $\mathcal{W}$ be a world specification and let $\mathcal{P}_S$ be the service invocation preconditions, S is consistently defined iff for each legal world state ω and for each consistent assignment, there is no element or element pair that belongs both to the insert set and the delete set of some concept or role.*

Example 3. *Both services introduced in Example 2 are not consistently defined: in fact their specification does not prevent the ambiguous case when the current and new towns are the same one. This is a typical case of* idempotent *operation that is not allowed in our framework, unless it is explicitly stated using an empty effect set, since it can potentially lead to semantic inconsistencies.*

So in order to provide a consistent service effect specification w.r.t. the domain constraints, we need to adjust the service preconditions introducing another negative atomic precondition as $\texttt{not}\ x_2 \sqcap \exists \mathsf{residentIn}^{-}.x_1$ *and* $\texttt{not}\ x_2 \sqcap \{\mathsf{town}_1\}$.

Given the knowledge base KB^P for a service S and an effect specification E for the same service, we define a new knowledge base KB^E adding axioms obtained by the instantiation for each concept $A \in \mathbf{A}$ and role name $P \in \mathbf{P}$ of the axiom schema ΔKB^E, presented in Table 4.2, where $A_E^+, A_E^-, P_E^+, P_E^-$ are resp. the names of the new auxiliary concepts and roles introduced for every concept A or role P in the domain specification.

Theorem 19. *Given a world specification $\mathcal{W}$ a service effect E is consistently defined iff for each concept name $A \in \boldsymbol{A}$ we have that:*

$$KB^E \wedge \tau(\mathcal{W}) \models A_E^+ \sqcap A_E^- \sqsubseteq \bot$$

and for each role name $P \in \boldsymbol{P}$:

$$KB^E \wedge \tau(\mathcal{W}) \models \neg \exists x, y. P_E^+(x, y) \wedge P_E^-(x, y)$$

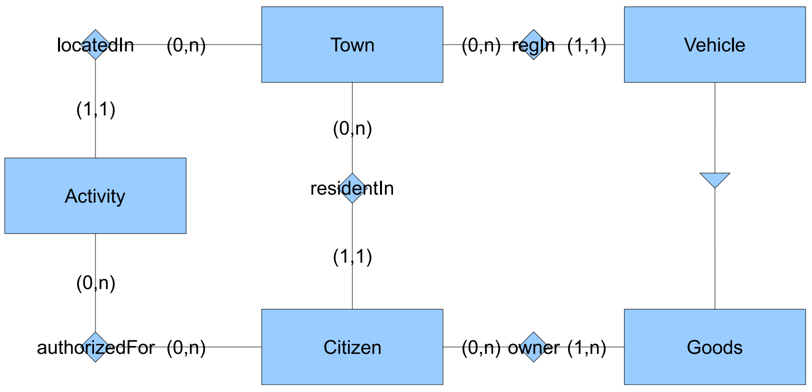

Figure 4.1: An example world specification $\mathcal{W}$ as ERD

Table 4.2: The axiom schema ΔKB^E

$$
\begin{aligned}
A^+_E &\equiv \left(\bigsqcup_{\langle +, A, Q(\mathbf{X}) \rangle \in E} \tau(Q) \right) \sqcap \neg A \\
A^-_E &\equiv \left(\bigsqcup_{\langle -, A, Q(\mathbf{X}) \rangle \in E} \tau(Q) \right) \sqcap A \\
\forall x, y. P^+_E(x, y) &\leftrightarrow \left(\bigvee_{\langle +, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E} \tau(Q)(x) \wedge \tau(Q')(y) \right) \wedge \neg P(x, y) \\
\forall x, y. P^-_E(x, y) &\leftrightarrow \left(\bigvee_{\langle -, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E} \tau(Q)(x) \wedge \tau(Q')(y) \right) \wedge P(x, y)
\end{aligned}
$$

Proof. Assuming that the service is accessible, otherwise the claim is easily proved, and that the effect is consistently defined, but, by contradiction, that the implication does not hold. In other words, there exists at least a model $\hat{\omega} \models KB^E \wedge \tau(\mathcal{W})$ s.t. there is an element $x \in \Delta^{\hat{\omega}}$ s.t. for some concept A we obtain that:

$$\hat{\omega} \models x : A_E^+ \sqcap A_E^-$$

or that there is a pair $\langle x, y\rangle \in \Delta^{\hat{\omega}} \times \Delta^{\hat{\omega}}$ s.t. for some role P:

$$\hat{\omega} \models P_E^+(x, y) \wedge P_E^-(x, y)$$

W.l.o.g. we prove the claim in the first case, the latter can be easily obtained applying the same argument. Given Theorem 11, applying the projection function $\pi^{\mathbf{X}}$ we obtain a model ω and an input variable assignment σ s.t. their extended interpretation is embedded into $\hat{\omega}$. Since $\hat{\omega} \models x : A_E^+$, given the definition axioms of the concept A_E^+ we have the an atomic effect $\langle +, A, Q\rangle$ s.t. $\hat{\omega} \models x : \tau(Q)$ must exist. Analogously we can prove that also another effect $\langle -, A, Q'\rangle$ s.t. $\hat{\omega} \models x : \tau(Q')$ must exist. But applying Theorem 2 we obtain also that:

$$\omega \lhd \sigma \models x : Q, x : Q'$$

in other words, that $x \in Q^{\omega}(\sigma)$ and $x \in Q'^{\omega}(\sigma)$. According to definition of insert and delete set (ignoring possibly instantiated objects), we can conclude that:

$$x \in A^+(\omega, \sigma) \cap A^-(\omega, \sigma)$$

which means that exists a pair ω, σ, consistent w.r.t. world specification and service invocation preconditions that violates the service consistency assumption.

Now we assume that the implication holds, but, by contradiction, that the service effect is not consistently defined. In other words, at least a pair ω, σ s.t. their components are consistent with world specification $\mathcal{W}$ and service invocation precondition $\mathcal{P}_S$ but that there exists an element x s.t.:

$$x \in A^+(\omega, \sigma) \cap A^-(\omega, \sigma)$$

or there exists a pair $\langle x, y\rangle$ s.t.:

$$\langle x, y\rangle \in P^+(\omega, \sigma) \cap P^-(\omega, \sigma)$$

Applying the mapping function $\mu^{\mathbf{X}}$ we build a new structure ω' that is a model for the axioms of the knowledge base KB^P, since Theorem 17. We add to the structure also the interpretation of new concept and role names applying the following construction:

$$\begin{array}{rcl}
[A_E^+]^{\hat{\omega}} & = & A^+(\omega, \sigma) \\
[A_E^-]^{\hat{\omega}} & = & A^-(\omega, \sigma) \\
[P_E^+]^{\hat{\omega}} & = & P^+(\omega, \sigma) \\
[P_E^-]^{\hat{\omega}} & = & P^-(\omega, \sigma)
\end{array}$$

defining a new interpretation $\hat{\omega}$ that is yet a model of KB^P, the interpretation of other names is untouched, but that is also a model of other axioms. For example, considering the axiom:

$$\forall x, y. P_E^-(x, y) \leftrightarrow \bigvee_{\langle -, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E} \tau(Q)(x) \wedge \tau(Q')(y) \wedge P(x, y)$$

and let $\langle x^*, y^* \rangle$ be a pair s.t. $\langle x^*, y^* \rangle \in [P_E^-]^{\hat{\omega}}$. According to the definition of the model we have that:

$$\langle x^*, y^* \rangle \in P^-(\omega, \sigma)$$

Given the definition of the role delete set there must exist an effect $\langle -, P, Q, Q' \rangle$ s.t.

$$\langle x^*, y^* \rangle \in Q^\omega(\sigma) \times Q'^\omega(\sigma) \cap P^\omega$$

Since the structure ω and the assignment σ are embedded into ω' and $\hat{\omega}$, we have also that:

- $\langle x^*, y^* \rangle \in P^{\hat{\omega}}$ since $\langle x^*, y^* \rangle \in P^\omega$;
- $x^* \in [\tau(Q)]^{\hat{\omega}}$ since $x^* \in Q^\omega(\sigma)$;
- $y^* \in [\tau(Q)]^{\hat{\omega}}$ since $y^* \in Q^\omega(\sigma)$;

According to standard first-order semantics, we have that:

$$\hat{\omega} \models \tau(Q)(x^*) \wedge \tau(Q')(y^*) \wedge P(x^*, y^*)$$

and, hence, that:

$$\hat{\omega} \models \forall x, y. P_E^-(x, y) \rightarrow \bigvee_{\langle -, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E} \tau(Q)(x) \wedge \tau(Q')(y) \wedge P(x, y)$$

Let $\langle x^*, y^* \rangle$ be a pair s.t. there exists an effect $\langle -, P, Q, Q' \rangle$ s.t.

$$\hat{\omega} \models \tau(Q)(x^*) \wedge \tau(Q')(y^*) \wedge P(x^*, y^*)$$

According to standard first-order semantics, we establish that: $x^* \in [\tau(Q)]^{\hat{\omega}}$, $y^* \in [\tau(Q')]^{\hat{\omega}}$ and $\langle x^*, y^* \rangle \in P^{\hat{\omega}}$. Since the structure ω with the assignment σ are embedded into ω' and $\hat{\omega}$, those statements imply that: $\langle x^*, y^* \rangle \in P^\omega$, $x^* \in Q^\omega(\sigma)$ and $y^* \in Q^\omega(\sigma)$.

Applying the definition of role delete set we can conclude that:

$$\langle x^*, y^* \rangle \in P^-(\omega, \sigma)$$

So, given the construction applied to build the structure $\hat{\omega}$, we have also that:

$$\hat{\omega} \models P_E^-(x^*, y^*)$$

So we have also proved that:

$$\hat{\omega} \models \forall x, y. P_E^-(x, y) \rightarrow \bigvee_{\langle -, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E} \tau(Q)(x) \wedge \tau(Q')(y) \wedge P(x, y)$$

Extending such argumentation to other kind of update construct, we can derive that the interpretation $\hat{\omega}$ is a model for the knowledge base KB^E too.

Since the hypothesis, the built model is s.t., for each concept name $A \in \mathbf{A}$ we have that:

$$\hat{\omega} \models A^+_E \sqcap A^-_E \sqsubseteq \bot$$

and for each role name $P \in \mathbf{P}$:

$$\hat{\omega} \models \neg\exists x, y.P^+_E(x, y) \wedge P^-_E(x, y)$$

But, as contradiction hypothesis, we have assumed that exists at least an element $x \in A^+(\omega, \sigma) \cap A^-(\omega, \sigma)$ for some A, but given the construction we have also that:

$$x \in [A^+_E]^{\hat{\omega}} \cap [A^-_E]^{\hat{\omega}}$$

which means that:

$$\hat{\omega} \not\models A^+_E \sqcap A^-_E \sqsubseteq \bot$$

contradicting the initial hypothesis. □

Theorem 20. *Given a world specification $\mathcal{W}$ and a service effect E of a service S, the problem of checking if the effect is consistently defined is in* coNEXP*.*

Proof. As done for other cases previously analyzed, we can solve the problem applying the property proved in Theorem 19, polynomially reducing it to a reasoning task in $\mathcal{C}^2$ logics, that according to Proposition 1 can be solved in coNEXP. □

Remark 9. *If we assume the world specification $\mathcal{W}$ as fixed for a given service community $\mathcal{S}$, the problem reduction is only linear in the size of the e-service specification: such a property holds for most of subsequent reductions.*

Given the previous definitions, we can finally introduce dynamic aspects, defining the transition relation between system states resulting from the enactment of a service. As stated in the preliminary assumption, we are ignoring any other source of change of the system state: it cannot evolve autonomously and there is any other interacting agent. Despite this is a quite limiting assumption in planning applications (e.g., multi-agent systems or robotics), it is widely adopted in the field of cooperative information systems, since the system state generally evolves as a consequence of an application request, that can be modeled as an e-service, and different requesting agents are not clashing on the same domain objects[5].

Definition 34 (Simple e-service successor relation)**.** *Given a pair of world states ω and ω', an input and output variable assignments $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ consistently defined w.r.t. ω, we say that ω' is a (potential) successor state of ω, resulting from the execution of an e-service S, that realizes the effect E and instantiating the set $\boldsymbol{Y}$, iff:*

- *the interpretation domain $\Delta^{\omega'}$ of the successor state is the smallest subset of $\mathfrak{U}$ s.t.:*

$$\Delta^{\omega} \cup cod(\sigma'_{\boldsymbol{Y}}) \subseteq \Delta^{\omega'}$$

[5]Notably, most of the proposals [NRF06, NRLW06, NRLH06, BTP04] for introducing transaction management capabilities in web-services middleware lack of strong locking capabilities, since the network is highly asynchronous at least from the application perspective (*long running processes*).

- *the interpretation of object names is preserved ($o^{\omega} = o^{\omega'}$);*
- *for each concept name $A \in \boldsymbol{A}$, the insert set is included in the successor state interpretation:*
$$A^{\omega'} \supseteq A^{+}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$$
and the delete set is excluded:
$$A^{\omega'} \cap A^{-}(\omega, \sigma_{\boldsymbol{X}}) \subseteq \emptyset$$
- *for each role name $P \in \boldsymbol{P}$, the insert set is included in the successor state interpretation:*
$$P^{\omega'} \supseteq P^{+}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$$
and the delete set is excluded:
$$P^{\omega'} \cap P^{-}(\omega, \sigma_{\boldsymbol{X}}) \subseteq \emptyset$$

The set of possible successor states obtainable from a state ω, applying the effect E of a service using the assignment $\sigma_{\mathbf{X}}$ and $\sigma'_{\mathbf{Y}}$ is denoted as:

$$\Omega_E(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

where $\mathbf{Y}$ is the set of newly instantiated objects.

Remark 10. *Given a pair $\langle \boldsymbol{Y}, E\rangle$, the output variable occurrences in effect definition must belong to the set $\boldsymbol{Y}$.*

Remark 11. *In case of simple e-service the instantiation name set is always $\boldsymbol{Y}_S$.*

From the previous definition we can be easily shown that:

Lemma 14. *The interpretation domain of the successor state ω' is the same as the extended interpretation $\omega \triangleleft \langle \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}\rangle$.*

Among the potential successor states resulting from the execution of a service that realizes its effects, we are interested in the ones that minimally differ from the initial state according to a notion of minimal-change semantics. In particular, we adopt a structure-distance metric based on the number of elements whose interpretation changes from a structure to another: a similar approach in data reconciliation is adopted, for example, in [LM96]. Since the interpretation of object names is always the same, they can be thrown out from the definition.

Definition 35 (Symmetric difference distance)**.** *Given two interpretation structures ω and ω' of a description logic alphabet $\langle \boldsymbol{A}, \boldsymbol{P}\rangle$, their distance is defined as:*
$$d(\omega, \omega') \triangleq \sum_{A \in \boldsymbol{A}} \left\| A^{\omega} \uplus A^{\omega'} \right\| + \sum_{P \in \boldsymbol{P}} \left\| P^{\omega} \uplus P^{\omega'} \right\|$$
where $\uplus$ denotes the set symmetric difference or disjoint union operator.

The provided distance measure induces a metric space over the set of possible world states since:

- it is non-negative ($d(\omega, \omega') \geq 0$);

- it ensures the identity of indiscernible ($d(\omega, \omega') = 0$ iff $\omega = \omega'$);
- it is symmetric ($d(\omega, \omega') = d(\omega', \omega)$);
- it obeys the triangular inequality ($d(\omega, \omega') \leq d(\omega, \omega'') + d(\omega'', \omega')$).

Such properties for interpretation structures follow directly from the ones of set symmetric difference operator $\uplus$. We are ignoring objects $\mathbf{O}$, since they are assumed to be immutable through state transitions and constantly interpreted in any world state.

We can now provide the definition of the transition relation between system states resulting from the execution of a simple e-service.

Definition 36 (Simple e-service transition relation)**.** *Let ω and ω' be a pair of world states, s.t. the latter is resulting from the execution of an e-service S in the state defined by the former, realizing the effect E. Given an input and an output variable assignments $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ consistently defined, there exists a system state transition from ω to ω' using the specified e-service effect iff:*

- *ω' is a (potential) successor state of ω w.r.t. the given assignments;*
- *there does not exist any other potential successor state ω'' of ω, w.r.t. the same assignments and service effect, s.t. it is closer to ω than ω' according to the symmetric difference distance (which means that $d(\omega, \omega') \leq d(\omega, \omega'')$ for each $\omega'' \in \Omega_E(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$).*

The concept of service enactment can be formally defined as the following. Using this concept we are able to easily denote the service outcomes in terms of both resulting system state and output variable assignment[6].

Definition 37 (Simple e-service enactment)**.** *Let ω be a world state and $\sigma_{\boldsymbol{X}}$ a consistent input variable assignment. Given a simple e-service S, the set of possible enactments contains all the pairs $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle$ s.t.:*

- *$\sigma'_{\boldsymbol{Y}}$ is a consistent instantiation assignment w.r.t. ω;*
- *ω and ω' are in transition relation w.r.t. the assignment and the service effect.*

The service enactment can be denoted as $S(\omega, \sigma_{\boldsymbol{X}})$.

Also the embedding relation, introduced at page 33, needs to be extended to this case since now we need to take into account both input and output variable assignments and initial and final system states.

Definition 38 (Simple e-service enactment embedding relation)**.** *Given a pair of world states $\omega = \langle \Delta^{\omega}, \cdot^{\omega} \rangle$ and $\omega' = \langle \Delta^{\omega'}, \cdot^{\omega'} \rangle$, defined on an interpretation domain that is a subset of $\mathfrak{U}$, an input assignment $\sigma_{\boldsymbol{X}}$ and an instantiation assignment $\sigma'_{\boldsymbol{Y}}$ both consistent w.r.t. ω, let m be a function that maps each concept (resp. role) name A (resp. P) into a new one $m(A)$ (resp. $m(P)$), let* Top *and* Top_m *be new concept names and let $\hat{\omega} = \langle \mathfrak{U}, \cdot^{\hat{\omega}} \rangle$ be an interpretation over the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{X} \cup \boldsymbol{Y} \cup m(\boldsymbol{A}) \cup \{\mathsf{Top}, \mathsf{Top}_m\}, \boldsymbol{P} \cup m(\boldsymbol{P}), \boldsymbol{O} \rangle$. The quadruple*

[6]Given the non-determinism in the object instantiation step, also in this simple case, we generally have multiple possible enactments.

$\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ is embedded into the interpretation $\hat{\omega}$ iff the following conditions hold:

$$\begin{aligned}
\Delta^{\omega} &= \mathsf{Top}^{\hat{\omega}} \\
\Delta^{\omega'} &= \mathsf{Top}_m^{\hat{\omega}} \\
N^{\omega} &= N^{\hat{\omega}} \\
N^{\omega'} &= m(N)^{\hat{\omega}} \\
\sigma_{\boldsymbol{X}}(X) &= X^{\hat{\omega}} \\
\sigma'_{\boldsymbol{Y}}(Y) &= Y^{\hat{\omega}} \\
\|V^{\hat{\omega}}\| &= 1 \\
o^{\omega} &= o^{\hat{\omega}}
\end{aligned}$$

for each $N \in \boldsymbol{A} \cup \boldsymbol{P}$, $o \in \boldsymbol{O}$, $X \in \boldsymbol{X}$, $Y \in \boldsymbol{Y}$ and $V \in \boldsymbol{X} \cup \boldsymbol{Y}$.

Remark 12. *The last definition aims to provide a way to embed a system state transition into a structure suitable to logically check dynamic system properties. Intuitively, the structure ω is the initial world state, the structure ω' is the enactment resulting structure, while the assignments are respectively the input provided to the service and the set of objects instantiated by the service itself (if any).*

Example 4. *We consider a simple domain specification s.t., let $\boldsymbol{A} = \{\mathsf{C}, \mathsf{D}\}$ and $\boldsymbol{P} = \{\mathsf{r}, \mathsf{s}\}$ be resp. the concept and role alphabets, the world state ω, depicted in Figure 4.2 is given as initial condition, where $\{x, y, z, w\}$ are some elements of the interpretation domain $\mathfrak{U}$.*

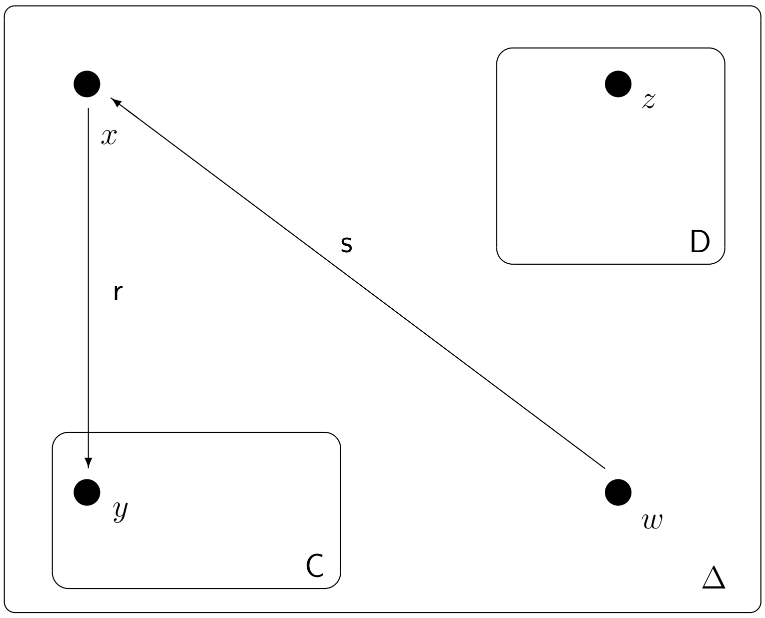

Figure 4.2: An example world state ω

For example, we assume that the following update statements have been issues during a service enactment:

$$-\mathsf{r}(x, \mathsf{C} \sqcap \exists \mathsf{r}^-. \{x\}), +\mathsf{r}(x, z), -\mathsf{D}(z), +\mathsf{C}(\neg\mathsf{C} \sqcap \exists \mathsf{s}. \{x\})$$

Considering the structure ω we can easily evaluate DL-query expressions concluding that:

$$\left[\mathsf{C} \sqcap \exists \mathsf{r}^-. \{x\}\right]^\omega \ni y$$
$$\left[\neg\mathsf{C} \sqcap \exists \mathsf{s}. \{x\}\right]^\omega \ni w$$

According to the provided semantics, the resulting system state ω' must be described as in Figure 4.3.

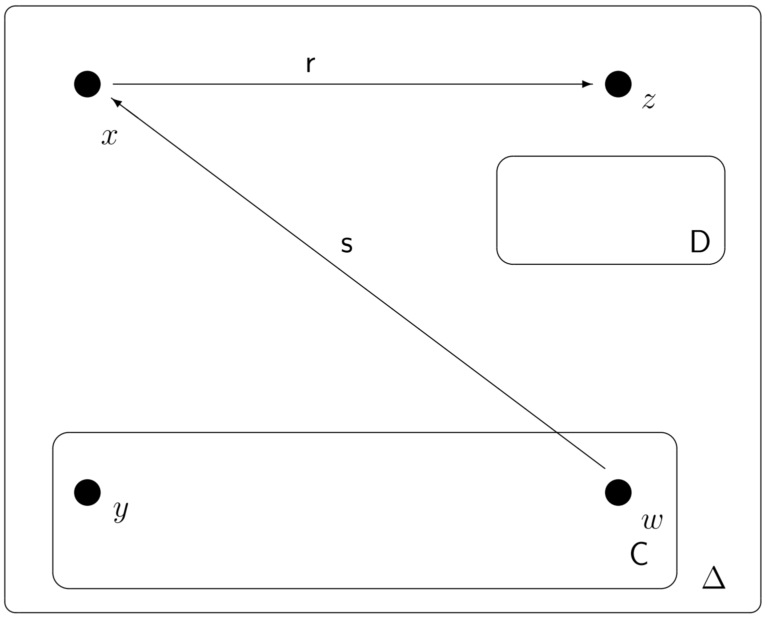

Figure 4.3: An example world state ω' resulting from updating the state ω

In other words, in order to enforce the update semantics, the ω' state must satisfy, among others, the following inter-model assertions:

$$\mathsf{D}^{\omega'} \subseteq \mathsf{D}^\omega \setminus \{z\}$$
$$\mathsf{C}^{\omega'} \subseteq \mathsf{C}^\omega \cup \{w\}$$
$$\mathsf{r}^{\omega'} \subseteq (\mathsf{r}^\omega \cup \{\langle x, z\rangle\}) \setminus \{\langle x, y\rangle\}$$

They cannot be evaluated using standard DL approaches, but if we embed both structures into a new suitable interpretation $\hat{\omega}$ on the same domain as depicted in Figure 4.4, we can easily conclude that these assertions can be encoded using a FOL theory interpreted on this ad-hoc built structure.

As we show in the following, since $\omega \rightsquigarrow \hat{\omega}$ and $\omega' \rightsquigarrow_m \hat{\omega}$, we are able to check relevant features of the enactment $\omega \rightarrow \omega'$ on $\hat{\omega}$ by means of logical inference in $\mathcal{C}^2$.

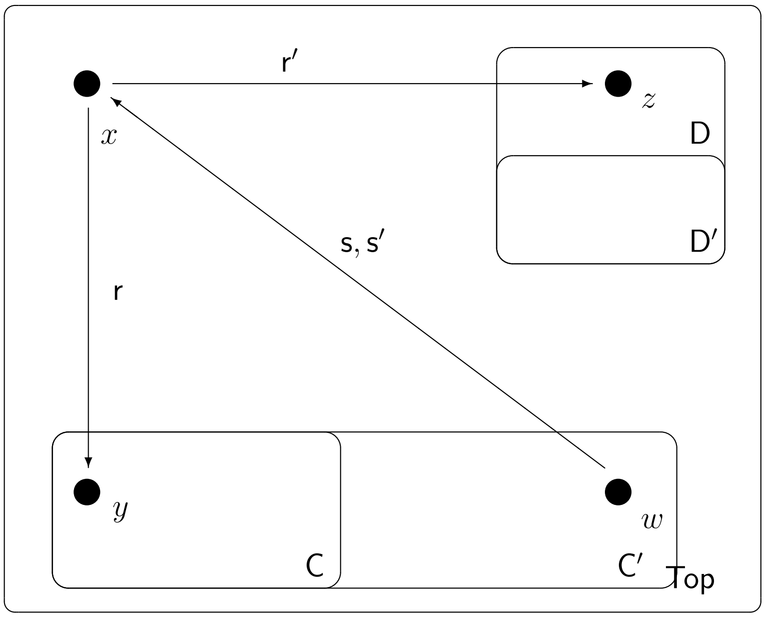

Figure 4.4: An example structure $\hat{\omega}$ embedding the transition $\omega \rightarrow \omega'$

From the definitions of world state embedding relation and generalized embedding relations we can easily derive the following property.

Lemma 15. *Given a quadruple $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ and an interpretation $\hat{\omega}$, s.t. the quadruple is embedded into it, then:*

- *the extended interpretation of the initial world state with the assignment is embedded into the structure $\hat{\omega}$:*

$$\omega \lhd \langle \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle \rightsquigarrow \hat{\omega}$$

- *the final world state is embedded, according to the mapping m, into the structure $\hat{\omega}$:*

$$\omega' \rightsquigarrow_m \hat{\omega}$$

Usually, name mapping functions are based upon the service/effect name, in order to allow for the construction of interpretation structures that capture multiple enactments at the same time: this will turn useful in order to deal with non-deterministic services. In the case of simple e-service the default mapping function is defined as $m(x) = x'$ while $\mathsf{Top}_m = \mathsf{Top}'$. In case of multiple service effects, the default mapping function for an effect E is defined as $m(x) = x_E$ while $\mathsf{Top}_m = \mathsf{Top}_E$.

Given a service S and an effect specification E for the same service, we define a new knowledge base KB^U_m adding to the knowledge base $KB^P \wedge \tilde{KB}^{\mathbf{X},\mathbf{Y}}(\mathsf{Top}_m)$ the axioms obtained by instantiating the schemas $\Delta KB^U(m)$, reported in Table 4.3, for each concept $A \in \mathbf{A}$ or role name $P \in \mathbf{P}$.

Table 4.3: The axiom schema $\Delta KB^U(m)$

$$m(A)^+ \equiv \left(\bigsqcup_{\langle +,A,Q(\mathbf{X})\rangle \in E} \tau(Q) \right) \sqcap \neg A \tag{4.2}$$

$$m(A)^- \equiv \left(\bigsqcup_{\langle -,A,Q(\mathbf{X})\rangle \in E} \tau(Q) \right) \sqcap A \tag{4.3}$$

$$\begin{aligned} \forall x, y.m(P)^+(x,y) \leftrightarrow & \left(\bigvee_{\langle +,P,Q(\mathbf{X}),Q'(\mathbf{X})\rangle \in E} \tau(Q)(x) \wedge \tau(Q')(y) \right) \\ & \wedge \neg P(x,y) \end{aligned} \tag{4.4}$$

$$\begin{aligned} \forall x, y.m(P)^-(x,y) \leftrightarrow & \left(\bigvee_{\langle -,P,Q(\mathbf{X}),Q'(\mathbf{X})\rangle \in E} \tau(Q)(x) \wedge \tau(Q')(y) \right) \\ & \wedge P(x,y) \end{aligned} \tag{4.5}$$

$$m(A)^* \equiv m(A)^+ \sqcup \bigsqcup_{\langle +,A,Y\rangle \in E} Y \tag{4.6}$$

$$\begin{aligned} \forall x, y.m(P)^*(x,y) \leftrightarrow & \; m(P)^+(x,y) \vee \bigvee_{\langle +,P,Y,Y'\rangle \in E} Y(x) \wedge Y'(y) \\ & \vee \bigvee_{\langle +,P,Y,Q(\mathbf{X})\rangle \in E} Y(x) \wedge \tau(Q)(y) \\ & \vee \bigvee_{\langle +,P,Q(\mathbf{X}),Y\rangle \in E} \tau(Q)(x) \wedge Y(y) \end{aligned} \tag{4.7}$$

$$m(A)^* \sqsubseteq m(A) \tag{4.8}$$

$$m(A)^+ \sqcap m(A)^- \sqsubseteq \bot \tag{4.9}$$

$$m(A)^- \sqsubseteq A \tag{4.10}$$

$$m(A) \sqcap \neg m(A)^* \equiv A \sqcap \neg m(A)^- \tag{4.11}$$

$$\forall x, y.m(P)^*(x,y) \rightarrow m(P)(x,y) \tag{4.12}$$

$$\forall x, y.\bot \leftarrow m(P)^+(x,y) \wedge m(P)^-(x,y) \tag{4.13}$$

$$\forall x, y.m(P)^-(x,y) \rightarrow P(x,y) \tag{4.14}$$

$$\forall x, y.m(P) \wedge \neg m(P)^*(x,y) \leftrightarrow P(x,y) \wedge \neg m(P)^-(x,y) \tag{4.15}$$

The knowledge base KB_m^U is defined over the alphabet $\langle A', P', O' \rangle$, where concept names are:

$$A' = \mathbf{A} \cup m(\mathbf{A}) \cup m(\mathbf{A})^* \cup m(\mathbf{A})^+ \cup m(\mathbf{A})^- \cup \mathbf{X} \cup \mathbf{Y} \cup \{\mathsf{Top}, \mathsf{New}, \mathsf{Top}_m\}$$

role names are:

$$P' = \mathbf{P} \cup m(\mathbf{P}) \cup m(\mathbf{P})^* \cup m(\mathbf{P})^+ \cup m(\mathbf{P})^- \cup \{\mathsf{aux}\}$$

and object names are:

$$O' = \mathbf{O} \cup \{\mathsf{spy}\}$$

Remark 13. *We notice that constructs (names and axioms) provided by the definition of the knowledge base* $\tilde{KB}^n(\mathsf{aux}, \mathsf{spy})$ *are relevant iff* $n = \| \mathbf{Y} \| > 0$.

Remark 14. *The axioms defined in Equations 4.2, 4.3, 4.4, and 4.5 are syntactical extension of corresponding ones in the knowledge base* KB^E *used in Theorem 19.*

As done in previous cases, we define an embedding function that, given an enactment, builds a new structure that (as we will show in the following) embeds the enactment itself and is a model of the provided knowledge base. While previously defined μ-functions take as argument a world state and a possible variable assignment, the new one requires also the interpretation corresponding to the resulting world state.

Given a quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$, we define an embedding function μ that maps such structures into another interpretation $\hat{\omega}$, extending the definition provided at page 50, s.t.:

- the interpretation domain is the whole universe ($\Delta^{\hat{\omega}} = \mathfrak{U}$);
- the interpretation of concepts, roles and objects in the starting state is preserved ($N^{\omega} = N^{\hat{\omega}}$);
- the interpretation of concepts, roles and objects in the final state is preserved ($N^{\omega'} = m(N)^{\hat{\omega}}$);
- the interpretation of Top is the active domain of ω ($\mathsf{Top}^{\hat{\omega}} = \Delta^{\omega}$);
- the interpretation of Top_m is the active domain of ω' ($\mathsf{Top}_m^{\hat{\omega}} = \Delta^{\omega'}$);
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;
- the interpretation of variable auxiliary concepts is defined according to the assignment ($\sigma_{\mathbf{X}}(X) = X^{\hat{\omega}}$ and $\sigma'_{\mathbf{Y}}(Y) = Y^{\hat{\omega}}$);
- the update-defining concepts and roles are interpreted according to the corresponding concept (resp. role) insert or delete set ($[m(N)^+]^{\hat{\omega}} = N^+(\omega, \sigma_{\mathbf{X}})$, $[m(N)^-]^{\hat{\omega}} = N^-(\omega, \sigma_{\mathbf{X}})$, $[m(N)^*]^{\hat{\omega}} = N^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$);
- the object spy is assigned to an element of $\mathsf{New}^{\hat{\omega}}$, if any;
- the role aux is interpreted as $\{\mathsf{spy}^{\hat{\omega}}\} \times \mathsf{New}^{\hat{\omega}}$.

The function π computes the inverse of μ, projecting out from an interpretation $\hat{\omega}$ a quadruple, representing a possible enactment between the world states ω and ω' given the variable assignments, and is defined only for structures that are models of the knowledge base KB^U_m.

We, now, show some properties of these functions w.r.t. the embedding relation.

Lemma 16. *Given a quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$, and structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu(\omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$, then the quadruple is embedded into $\hat{\omega}$.*

Proof. The claim follows from the observation that the function μ simply apply the definition of the embedding relation itself. We point out that, since there is a choice step, essentially regarding the selection of the spy-point element, the function is non-deterministic or it can be considered as a multi-function. □

Lemma 17. *Given a model $\hat{\omega}$ of KB^U_m, let $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$ be a tuple s.t.:*

$$\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle = \pi(\hat{\omega})$$

then it is embedded into $\hat{\omega}$.

Proof. Since the structure $\hat{\omega}$ is a model of the knowledge base KB^U_m, the cardinality axioms ensures that from the interpretation of variable related auxiliary concepts it is always possible to build consistent variable assignments as observed in the proof of Theorem 16. The embedding relation among the quadruple and the structure follows from the definition of the projection function π and the previous lemma. □

On this base, we can present a first main result about embedded structures and knowledge bases previously defined, showing that the latter ones are able to caught the enactment update semantics.

In other words, an interpretation structure that is a model of a suitable knowledge base, obtained from the instantiation of axiom schemas on a service specification, if it actually embeds an enactment of the given service according to the adopted semantics. Using this result, we have a main tool to reason about dynamic properties of e-services, assumed that they can be encoded in a compatible manner. Now, we show how to deal with various kind of semantic properties in following chapters.

Theorem 21. *Given an enactment $\langle \omega', \sigma'_{\mathbf{Y}} \rangle \in S(\omega, \sigma_{\mathbf{X}})$ of a consistently defined simple service, let $\hat{\omega}$ be a structure s.t. $\hat{\omega} = \mu(\omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$, then:*

$$\hat{\omega} \models KB^U_m$$

Proof. Let $\hat{\omega}$ be the structure built applying the function μ to the enactment: since such a function is an extension of the mapping function used in the proof of Theorem 17, we can use the same argument to prove that $\hat{\omega} \models KB^P$. On the other hand, the mapping function, assuming $\mathsf{Top}_m = \mathsf{Top}'$, is also an extension of the mapping function used in the proof of Theorem 15, so we can conclude that $\hat{\omega} \models \tilde{KB}^{\mathbf{X},\mathbf{Y}}(\mathsf{Top}_m)$.

In order to complete the proof we need to show that other axioms also hold. Regarding definition axioms introduced by in Equations 4.2, 4.3, 4.4, and 4.5, we point out that the construction adopted is an extension of which used in the proof of Theorem 19, so we can adopt the same argument also here. They need some adjustment in order to keep into account also instantiation variables (Equations 4.6 and 4.7).

Given an element $x \in [m(A)^*]^{\hat{\omega}}$, since the construction we have that:

$$x \in A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

which, according to insert set definition, means that:

- $x \in A^+(\omega, \sigma_{\mathbf{X}})$
- or exists an effect $\langle +, A, Y\rangle \in E$ s.t. $x = \sigma'_{\mathbf{Y}}(Y)$.

In the first case, we can conclude that $x \in [m(A)^+]^{\hat{\omega}}$, while in the second that $x \in Y^{\hat{\omega}}$. According to standard semantics, we can state that:

$$\hat{\omega} \models x : m(A)^+ \sqcup \bigsqcup_{\langle +,A,Y\rangle \in E} Y$$

and, hence, that:

$$\hat{\omega} \models m(A)^* \sqsubseteq m(A)^+ \sqcup \bigsqcup_{\langle +,A,Y\rangle \in E} Y$$

On the other hand, given an element $x \in [m(A)^+ \sqcup \bigsqcup_{\langle +,A,Y\rangle \in E} Y]^{\hat{\omega}}$, since the standard semantics we have that:

- $x \in [m(A)^+]^{\hat{\omega}}$
- or exists an effect $\langle +, A, Y\rangle \in E$ s.t. $x \in Y^{\hat{\omega}}$.

In the first case, we can conclude that $x \in A^+(\omega, \sigma_{\mathbf{X}})$, while in the second that $x = \sigma'_{\mathbf{Y}}(Y)$. Applying the definition of the insert set and the construction of the embedding function we can conclude that:

$$\hat{\omega} \models x : m(A)^*$$

and, consequently, that:

$$\hat{\omega} \models m(A)^+ \sqcup \bigsqcup_{\langle +,A,Y\rangle \in E} Y \sqsubseteq m(A)^*$$

We need, instead, explicitly show that update axioms hold.

We consider concept update axioms, the argumentation, despite the notation, can be also adapted to the role corresponding ones. Since ω' is a successor state of the state ω, according to the definition of the relation we have that:

$$A^{\omega'} \supseteq A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

Given the definition of the mapping function, we can establish that:

$$m(A)^{\hat{\omega}} \supseteq [m(A)^*]^{\hat{\omega}}$$

which means that $\hat{\omega} \models m(A)^* \sqsubseteq m(A)$.

Since the service is consistently defined, for each enactment we have that insert and delete sets for each concept set are mutually disjoint, given the construction we can conclude that:

$$\hat{\omega} \models m(A)^+ \sqcap m(A)^- \sqsubseteq \bot$$

Given the definition of the delete set for a concept A, we have that:

$$A^\omega \supseteq A^-(\omega, \sigma_\mathbf{X})$$

so applying the construction according to the definition of the embedding function we have that:

$$\hat{\omega} \models m(A)^- \sqsubseteq A$$

By contradiction, assuming that there exists an element x s.t. $\hat{\omega} \models x : m(A) \sqcap \neg m(A)^*$, but $\hat{\omega} \not\models x : A \sqcap \neg m(A)^-$. Such an element does not belong to the insert set, and, since $A^-(\omega, \sigma_\mathbf{X}) \subseteq A^\omega$ and $A^-(\omega, \sigma_\mathbf{X}) \cap A^{\omega'} \subseteq \emptyset$, we have also that $x \notin A^\omega$.

In other words, we are assuming that the resulting concept extension contains at least an element that neither belongs to the initial concept extension and neither has been inserted according to service effect definition. Intuitively, by inertia, we expecting that such an element must not exist.

Considering the distance between the initial and final state, we have that:

$$d(\omega, \omega') = k + \left\| A^\omega \nabla A^{\omega'} \right\| = k + \left\| A^{\omega'} \setminus A^\omega \right\| + \left\| A^\omega \setminus A^{\omega'} \right\|$$

Since $A^{\omega'} \supseteq A^+(\omega, \sigma_\mathbf{X}, \sigma'_\mathbf{Y})$ and $x \notin A^+(\omega, \sigma_\mathbf{X}, \sigma'_\mathbf{Y})$ we also that:

$$d(\omega, \omega') \geq k + 1 + \left\| A^+(\omega, \sigma_\mathbf{X}, \sigma'_\mathbf{Y}) \setminus A^\omega \right\| + \left\| A^\omega \setminus A^{\omega'} \right\|$$

But given ω' we can build a new structure ω'' that is equal to ω' except that the element x does not belong to $A^{\omega''}$. This new structure is a successor of the state ω, but we can observe also that:

$$d(\omega, \omega'') = k + \left\| A^+(\omega, \sigma_\mathbf{X}, \sigma'_\mathbf{Y}) \setminus A^\omega \right\| + \left\| A^\omega \setminus A^{\omega''} \right\|$$

Since $A^{\omega''} = A^{\omega'} \setminus \{x\}$ and $x \notin A^\omega$ we can conclude that $A^\omega \setminus A^{\omega'} = A^\omega \setminus A^{\omega''}$ and so:

$$d(\omega, \omega'') = k + \left\| A^+(\omega, \sigma_\mathbf{X}, \sigma'_\mathbf{Y}) \setminus A^\omega \right\| + \left\| A^\omega \setminus A^{\omega'} \right\| < d(\omega, \omega')$$

or, in other words, that ω' is not the nearest successor of the state ω, hence there is not enactment between them. So we can conclude that:

$$\hat{\omega} \models m(A) \sqcap \neg m(A)^* \sqsubseteq A \sqcap \neg m(A)^-$$

By contradiction, assuming that there exists an element x s.t. $\hat{\omega} \models x : A \sqcap \neg m(A)^-$, but $\hat{\omega} \not\models x : m(A) \sqcap \neg m(A)^*$. Such an element does not belong to the delete set, and, since $A^*(\omega, \sigma_\mathbf{X}, \sigma'_\mathbf{Y}) \subseteq A^{\omega'}$ and $A^+(\omega, \sigma_\mathbf{X}, \sigma'_\mathbf{Y}) \cap A^\omega \subseteq \emptyset$, we have also that $x \notin A^{\omega'}$.

Also in this case, we are assuming that the resulting concept extension does not contain an element that belongs to the initial concept extension and has not been deleted according to service effect definition. As in the previous case, by inertia, we are intuitively expecting that such an element must be included in the final concept extension.

Considering the distance between the initial and final state, we have that:

$$d(\omega,\omega') = k + \left\|A^{\omega} \nabla A^{\omega'}\right\| = k + \left\|A^{\omega'} \setminus A^{\omega}\right\| + \left\|A^{\omega} \setminus A^{\omega'}\right\|$$

Since $A^{\omega} \supseteq A^{-}(\omega,\sigma_{\mathbf{X}})$ and $x \notin A^{-}(\omega,\sigma_{\mathbf{X}})$ we also that:

$$d(\omega,\omega') \geq k + \left\|A^{\omega'} \setminus A^{\omega}\right\| + 1 + \left\|A^{-}(\omega,\sigma_{\mathbf{X}}) \setminus A^{\omega}\right\|$$

But given ω' we can build a new structure ω'' that is equal to ω' except that the element x belongs to $A^{\omega''}$. This new structure is a successor of the state ω, and as in the previous case we can observe also that:

$$d(\omega,\omega'') = k + \left\|A^{\omega''} \setminus A^{\omega}\right\| + \left\|A^{-}(\omega,\sigma_{\mathbf{X}}) \setminus A^{\omega}\right\|$$

Since $A^{\omega''} = A^{\omega'} \cup \{x\}$ and $x \in A^{\omega}$ we can conclude that $A^{\omega'} \setminus A^{\omega} = A^{\omega''} \setminus A^{\omega}$ and so:

$$d(\omega,\omega'') = k + \left\|A^{\omega'} \setminus A^{\omega}\right\| + \left\|A^{-}(\omega,\sigma_{\mathbf{X}}) \setminus A^{\omega}\right\| < d(\omega,\omega')$$

or, in other words, that ω' is not the nearest successor of the state ω, contradicting with the assumption that there is an enactment between them. Consequently we have that:

$$\hat{\omega} \models A \sqcap \neg m(A)^{-} \sqsubseteq m(A) \sqcap \neg m(A)^{*}$$

Combining this result with the previous we establish that:

$$\hat{\omega} \models m(A) \sqcap \neg m(A)^{*} \equiv A \sqcap \neg m(A)^{-}$$

The same argumentation can be adapted to prove the claim also in the case of role-related axioms, since definitions, also in terms of model distance, are completely equivalent. □

Theorem 22. *Given a model $\hat{\omega}$ of the knowledge base KB^{U}_{m}, then the quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle = \pi(\hat{\omega})$, is an enactment of a simple service S from the state ω to the state ω', having $\sigma_{\mathbf{X}}$ and $\sigma'_{\mathbf{Y}}$ as, resp., input and instantiation assignments.*

Proof. Given the definition of service enactment we need to show that:

1. that the assignment $\sigma'_{\mathbf{Y}}$ is actually a consistently defined instantiation assignment w.r.t. ω;

2. that there is a transition of the system from ω to ω' w.r.t the assignment $\sigma_{\mathbf{X}}$ and $\sigma'_{\mathbf{Y}}$.

Since $KB^{U}_{m} \supset KB^{P} \cup \tilde{KB}^{\mathbf{X},\mathbf{Y}}(\mathsf{Top}_{m})$, applying Theorem 16, we can prove that the instantiation assignment is well-formed and also that the final state active domain $\Delta^{\omega'}$ is consistently defined, given the constraint that forces its interpretation to be the same of the extended one defined as $\omega \lhd \langle \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$.

Regarding the point 2, we begin the proof observing that, since object names are always the same, there isn't any form of alias so they are constantly interpreted on the same universe elements in all structures (ω, ω' and $\hat{\omega}$).

Furthermore, we need to show that the insert set is included in the extension of name in the successor state structure:

$$A^{\omega'} \supseteq A^{+}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

but it can be easily verified since the model $\hat{\omega}$ satisfies the axiom $m(A)^{*} \sqsubseteq m(A)$, and, according to the definition of embedding function, we have that $A^{\omega'} = m(A)^{\hat{\omega}}$ and $A^{+}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}) = [m(A)^{*}]^{\hat{\omega}}$.

Given the mapping, in order to prove that:

$$A^{\omega'} \cap A^{-}(\omega, \sigma_{\mathbf{X}}) \subseteq \emptyset$$

we need to show that:

$$\hat{\omega} \models m(A) \sqcap m(A)^{-} \sqsubseteq \bot$$

Assuming that there exists an element x s.t.:

$$\hat{\omega} \models x : m(A) \sqcap m(A)^{-}$$

for some $A \in \mathbf{A}$. Since the axiom $m(A)^{-} \sqsubseteq A$ we can conclude that:

$$\hat{\omega} \models x : m(A) \sqcap A$$

On the other hand, we can conclude also that:

$$\hat{\omega} \not\models x : A \sqcap \neg m(A)^{-}$$

But, since the axiom $m(A) \sqcap \neg m(A)^{*} \equiv A \sqcap \neg m(A)^{-}$, we have also that:

$$\hat{\omega} \not\models x : m(A) \sqcap \neg m(A)^{*}$$

Since we have assumed that $x \in [m(A)]^{\hat{\omega}}$, in order to enforce the constraint we must conclude that $x \in [m(A)^{*}]^{\hat{\omega}}$. If $x \in [m(A)^{*}]^{\hat{\omega}}$, since the definition of the concept we can consider two cases:

1. $x \in Y^{\hat{\omega}}$ for some $Y \in \mathbf{Y}$;
2. $x \in [m(A)^{+}]^{\hat{\omega}}$.

The first case can never occur, since $x \in A^{\hat{\omega}}$ and knowledge base axioms impose that:

$$\hat{\omega} \models A \sqsubseteq \mathsf{Top}, Y \sqsubseteq \mathsf{Top}' \sqcap \neg\mathsf{Top}$$

Also in the latter there is a contradiction since we have concluded that:

$$\hat{\omega} \models x : m(A)^{-} \sqcap m(A)^{+}$$

while the interpretation $\hat{\omega}$ is assumed to satisfy the axioms $m(A)^{+} \sqcap m(A)^{-} \sqsubseteq \bot$.

We have concluded that ω' is a potential successor state of ω given the assignments. To completely prove the claim we must also prove that is the nearest structure having such properties, since we have assumed a minimal-change semantics that enables only system transitions that requires the minimal amount of updates.

By contradiction, we assume that ω' is not the state resulting from the enactment of the service S in ω given the variable assignments, because there exists another successor state $\omega'' \in \Omega(\sigma, \sigma_{\mathbf{X}}, \sigma_{\mathbf{Y}})$ s.t. $d(\omega, \omega'') < d(\omega, \omega')$.

According to the definition of the metric function there must exists at least a concept name $A \in \mathbf{A}$ (or at least a role name $P \in \mathbf{P}$) s.t. there exists at least an element $x \in \mathfrak{U}$ (or a pair $\langle x, y\rangle \in \mathfrak{U} \times \mathfrak{U}$) s.t. one of the following cases occurs:

1. $x \in A^{\omega}$ ($\langle x, y\rangle \in P^{\omega}$) and $x \in A^{\omega''}$ ($\langle x, y\rangle \in P^{\omega''}$) but $x \notin A^{\omega'}$ ($\langle x, y\rangle \notin P^{\omega'}$);
2. or $x \notin A^{\omega}$ ($\langle x, y\rangle \notin P^{\omega}$) and $x \notin A^{\omega''}$ ($\langle x, y\rangle \notin P^{\omega''}$) but $x \in A^{\omega'}$ ($\langle x, y\rangle \in P^{\omega'}$).

Assuming that the name is not affected by any update (i.e., $A^{\omega} = A^{\omega''}$). According to our formulation the knowledge base implies that:

$$KB_m^U \models m(A) \equiv A, m(A)^+ \sqsubseteq \bot, m(A)^- \sqsubseteq \bot, m(A)^* \sqsubseteq \bot$$

and, hence, that $A^{\omega} = A^{\omega'} = A^{\omega''}$.

Otherwise, we assume that the name is affected by some update effect. We consider the first case: $x \in A^{\omega}$ and $x \notin A^{\omega'}$. Since the axioms of the knowledge base we can conclude that:

$$KB_m^U \models A \sqcap \neg m(A) \sqsubseteq m(A)^-$$

which means that:

$$\hat{\omega} \models x : m(A)^-$$

and, according to the definition of the embedding relation, that:

$$x \in A^-(\omega, \sigma_{\mathbf{X}})$$

Since ω'' is a successor state of ω we have also that:

$$A^{\omega''} \cap A^-(\omega, \sigma_{\mathbf{X}}) \subseteq \emptyset$$

contradicting the hypothesis that $x \in A^{\omega''}$. The other case is analogous. Since the axioms of the knowledge base we can conclude that:

$$KB_m^U \models m(A) \sqcap \neg A \sqsubseteq m(A)^*$$

which means that:

$$\hat{\omega} \models x : m(A)^*$$

Applying the definition of the embedding relation, it follows that:

$$x \in A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

Since ω'' is a successor state of ω we have also that:

$$A^{\omega''} \supseteq \cap A^*(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

contradicting the hypothesis that $x \notin A^{\omega''}$. □

The following corollary generalizes the above result.

Corollary 4. *Let $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ be an enactment of a simple e-service S, then:*

- *for each concept name $A \in \boldsymbol{A}$*

$$A^{\omega'} = A^{+}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cup (A^{\omega} \setminus A^{-}(\omega, \sigma_{\boldsymbol{X}}))$$

- *for each role name $P \in \boldsymbol{P}$*

$$P^{\omega'} = P^{+}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cup (P^{\omega} \setminus P^{-}(\omega, \sigma_{\boldsymbol{X}}))$$

We now extend the result of Theorem 14 to service enactment, showing that the resulting states of a service enactment are isomorphic.

Remark 15. *This is a foundational result, since isomorphic structures are indistinguishable using $\mathcal{ALCQI}$ description logic language.*

Theorem 23. *Let ω be a world state, $\sigma_{\boldsymbol{X}}$ a consistent input variable assignment, S a simple e-service accessible in ω using $\sigma_{\boldsymbol{X}}$. If $\langle \omega'_1, \sigma'_1 \rangle$ and $\langle \omega'_2, \sigma'_2 \rangle$ are two enactments in $S(\omega, \sigma_{\boldsymbol{X}})$, then they are isomorphic.*

Proof. In this proof we use the same function h employed in Theorem 14, where we have shown that extended interpretation are also isomorphic w.r.t. the instantiation variable assignment.

Such a function $h : \Delta^{\omega'_1} \mapsto \Delta^{\omega'_2}$ can be defined as:

$$h(x) \triangleq \begin{cases} x & x \in \Delta^{\omega} \\ \sigma'_2(Y) & \sigma'_1(Y) = x \end{cases}$$

Since, according to Lemma 14, the interpretation domains of successor states are the same as extended interpretations that take into account also instantiation variables ($\omega \lhd \langle \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$), we can show that h is a total bijective function from $\Delta^{\omega'_1}$ to $\Delta^{\omega'_2}$ using the same argumentation, so we need to prove that it is also a homomorphism.

By contradiction: let x be an element of $\Delta^{\omega'_1}$ and let A be a concept name in $\mathbf{A}$, s.t. $x \in A^{\omega'_1}$, but $x \notin A^{\omega'_2}$. Since Corollary 4, we need to consider the following cases:

1. the element has been inserted into the concept extension ($x \in A^{+}(\omega, \sigma_{\mathbf{X}}, \sigma'_1)$), implying two other cases:
   (a) the element has been newly created, i.e., exists an instantiation variable Y s.t. $\sigma'_1(Y) = x$ and $x \in A^{+}(\omega, \sigma_{\mathbf{X}}, \sigma'_1) \setminus A^{+}(\omega, \sigma_{\mathbf{X}})$;
   (b) the element is already in the active domain and it belongs to the concept insert set ($x \in A^{+}(\omega, \sigma_{\mathbf{X}})$).
2. the element was already in the concept extension and it has not been deleted ($x \in A^{\omega} \setminus A^{-}(\omega, \sigma_{\mathbf{X}})$).

In case 1a, according to the definition of function h, there exists an element x' s.t. $\sigma'_2(Y) = x'$ and $h(x) = x'$, where Y is the name of instantiation variable s.t. $\sigma'_1(Y) = x$. Since the definition of insert set an insert effect of the form $\langle +, A, Y\rangle$ must exist and we can also conclude that:

$$x' \in A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_2) \setminus A^+(\omega, \sigma_{\mathbf{X}})$$

but, by definition of successor state, we have that:

$$A^{\omega'_2} \supseteq A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_2)$$

and, hence, that $h(x) = x' \in A^{\omega'_2}$.

In case 1b, we point out that insert set, excluding instantiation variables, is the same in both enactments since it only depends upon initial state and input variable assignment and, since we have that $h(x) = x$, we can conclude that:

$$h(x) = x \in A^+(\omega, \sigma_{\mathbf{X}})$$

Applying Corollary 4, we have that $h(x) = x \in A^{\omega'_2}$ too.

In the last case we can adopt the same argumentation of the previous one, since the initial extension of the concept A and the delete set do not depend on instantiation variables in any way, concluding that $h(x) = x \in A^{\omega'_2}$.

Also by contradiction: let x be an element of $\Delta^{\omega'_1}$ and let A be a concept name in $\mathbf{A}$, s.t. $x \notin A^{\omega'_1}$ despite $x \in A^{\omega'_2}$. Since Corollary 4, we need to consider the following cases:

1. the element neither initially belongs to the concept extension neither it has been inserted ($x \notin A^\omega \cup A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_1)$), implying two other cases:
   (a) the element has been newly created, i.e., exists an instantiation variable Y s.t. $\sigma'_1(Y) = x$;
   (b) the element is already in the active domain and it does belong neither to the concept insert set and initial concept extension ($x \notin A^\omega \cup A^+(\omega, \sigma_{\mathbf{X}})$).
2. the element was already in the concept extension and it has been deleted ($x \in A^-(\omega, \sigma_{\mathbf{X}})$).

In the first case, since $x \notin A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_1) \setminus A^+(\omega, \sigma_{\mathbf{X}})$, according to effect definition, we have that an effect of the form $\langle +, A, Y\rangle$ cannot exist. On the other hand, there exists an element x' s.t. $\sigma'_1(Y) = x'$ and $h(x) = x'$. and, according to insert set definition, otherwise such a kind of effect must be present in service specification leading to a contradiction:

$$x' \notin A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_2)$$

but, since $x' \in \Delta^{\omega'_2} \setminus \Delta^\omega$, it cannot belong to $A^\omega \cup A^+(\omega, \sigma_{\mathbf{X}})$, consequently we have that $h(x) = x' \notin A^{\omega'_2}$.

In case 1b, analogously to previous proofs, in order to prove that $x = h(x)$ is a member of the set $A^{\omega'_2}$ we need to prove that $x \in A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_2) \setminus A^+(\omega, \sigma_{\mathbf{X}})$, but it is contradicting the hypothesis that assumes $x \in \Delta^\omega$, since:

$$A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_2) \setminus A^+(\omega, \sigma_{\mathbf{X}}) \subseteq \Delta^{\omega'_2} \setminus \Delta^\omega$$

Consequently we can establish that $h(x) = x \notin A^{\omega'_2}$.

In case 2, since the initial extension of the concept A and the delete set do not depend on instantiation variables in any way, concluding that $h(x) = x \notin A^{\omega'_2}$.

The proof can be extended to role interpretations too, keeping into account various cases in the definition of role insert and delete sets.

Regarding object names $O \in \mathbf{O}$, we remark that their interpretation is always preserved by any effect and it is the same in any admissible state. Let o be an object name interpreted as $o^\omega \in \Delta^\omega$, since the definition of successor state, we have also that:

$$o^\omega = o^{\omega'_1} = o^{\omega'_2}$$

Since $o^\omega \in \Delta^\omega$, then $h(o^\omega) = o^\omega$. □

The devised definition of consistency for service effects is a necessary but not sufficient condition in order to ensure the correctness of an e-service acting in a world subject to a constraint set represented by the specification knowledge base $\mathcal{W}$. In fact, this is a kind of internal effect consistency, since it simply assures that the enactment effects are *per se* not contradictory.

On the other hand, we are also interested in the property of a service that always acts consistently with the specification of the system, or, in other words, s. t. the world state resulting from an enactment is always legal.

The service contract is defined presuming that the invocation preconditions are sufficient in order to achieve the realization of one of declared service effects by an enactment. In other terms, given a state where the service preconditions hold, the service invocation must result into a new state where the declared effects are realized. This assumption is fundamental for the verification of the consistency of service specifications, since it allows for providing a complete contract specification, excluding external world altering events.

We remark that the service contract imposes to the service provider that, whenever the client is conforming to the preconditions. it must not fail: we are essentially ignoring reliability implementation and communication issues and we are meaning failure in purely functional terms. Such an approach is, by the way, the same adopted in *design-by-contract* frameworks ([Mey88]) and it is also coherent with many agent-based solutions as shown in Section 2.2.7.

Definition 39 (Legal e-service)**.** *An e-service is legal iff it implements only transitions from a legal state to another legal state.*

In order to enforce the service contract, once an e-service has been consistently activated, it must update the system state consistently too. Thus, we formally combine the previous definitions providing the simplest form of consistency property.

Definition 40 (Valid simple e-service)**.** *Let E be the effect of a simple e-service S. We say that the service is valid w.r.t. a world specification $\mathcal{W}$ iff:*

- *the effect E is consistent;*
- *for each legal world state ω, for each consistent input assignment $\sigma_{\mathbf{X}}$, s.t. the service is accessible in ω using it, there exists at least a legal state ω' in the enactment.*

Example 5. *Given the specification $\mathcal{W}$ provided in Example 2, consider the following e-service that allows a customer to buy a new vehicle, or in other words to become the owner*[7]*:*

$$\begin{aligned}
\boldsymbol{X} &= \{x\}\\
\boldsymbol{Y} &= \{y\}\\
\mathcal{P} &= x_1 \sqcap \mathsf{Citizen}\\
E &= \{+\mathsf{Vehicle}(y), +\mathsf{owner}(y,x)\}
\end{aligned}$$

Despite the service is accessible and its effects are consistently defined, it is not legal, because it is not valid. In fact, its enactments violate several domain constraints: a vehicle is also a good and it must be recorded to the town authority. A valid version of the service effects is the following:

$$E = \{+\mathsf{Vehicle}(y), +\mathsf{owner}(y,x), +\mathsf{Good}(y), +\mathsf{registeredId}(y, \exists \mathsf{residentIn}.x)\}$$

We inductively define a generalized translation function τ_m, given the name mapping function m, over the concept expression of the description logic language $\mathcal{ALCQIO}$, from the alphabet $\langle \mathbf{A}, \mathbf{P}, \mathbf{O}\rangle$ to the new alphabet $\langle m(\mathbf{A}) \cup \{\mathsf{Top_m}\}, m(\mathbf{P}), \mathbf{O}\rangle$, as the follows:

$$\begin{aligned}
\tau_m(A) &\triangleq m(A)\\
\tau_m(C \sqcap C') &\triangleq \tau_m(C) \sqcap \tau_m(C')\\
\tau_m((\geq\ n\, R\, C)) &\triangleq (\geq\ n\, R\, \tau_m(C))\\
\tau_m(\{o_1, \ldots, o_n\}) &\triangleq \{o_1, \ldots, o_n\}\\
\tau_m(\neg C) &\triangleq \mathsf{Top_m} \sqcap \neg \tau_m(C)
\end{aligned}$$

As the translation function τ, the generalized one τ_m can be applied to each assertion in the knowledge base $KB = \langle \mathcal{T}, \mathcal{A}\rangle$, obtaining a new knowledge base $KB_m = \langle \tau_m(\mathcal{T}), \tau_m(\mathcal{A})\rangle$ in the mapped name space. Generalizing the results of Theorems 1 and 2 we can easily achieve the following corollaries.

Corollary 5. *Let ω and $\hat{\omega}$ be respectively a world state and an arbitrary interpretation s.t. the world state is embedded into the interpretation ($\omega \rightsquigarrow_m \hat{\omega}$) w.r.t. the name mapping function m, then:*

$$R^{\omega} = m(R)^{\hat{\omega}}$$

for any $\mathcal{ALCQIO}$ role expression R built over the domain specification alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$.

Corollary 6. *Let ω and $\hat{\omega}$ be respectively a world state and an arbitrary interpretation s.t. the world state is embedded into the interpretation ($\omega \rightsquigarrow_m \hat{\omega}$) w.r.t. the name mapping function m, then:*

$$C^{\omega} = [\tau_m(C)]^{\hat{\omega}}$$

for any $\mathcal{ALCQIO}$ concept expression C built over the domain specification alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$.

[7]Please consider that this is an abstraction of the real-world process: the service does not create the "thing" vehicle, but simply makes the system aware of its existence. Moreover, an e-government system is concerning only about things that have been accordingly recorded.

Theorem 24. *A consistent and accessible simple e-service S is valid w.r.t. a world specification $\mathcal{W}$ iff the following implication holds:*

$$KB_m^U \wedge \tau(\mathcal{W}) \models \tau_m(\mathcal{W})$$

where m is a name mapping function for the domain.

Proof. We assume that the service is valid: since it is also accessible and consistent we have at least an enactment $\langle \omega', \sigma'_{\mathbf{Y}} \rangle \in S(\omega, \sigma_{\mathbf{X}})$. Let $\tilde{\omega} = \mu(\omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$ the embedding structure: according to Theorem 21 it is a model of the knowledge based KB_m^U. W.l.o.g., we assume that there exists an axiom $\tau_m(C) \sqsubseteq \tau_m(D) \in \tau_m(\mathcal{W})$ s.t.:

$$\tilde{\omega} \not\models \tau_m(C) \sqsubseteq \tau_m(D)$$

Now we consider two cases:

1. if there is no instantiation variable ($\mathbf{Y} = \emptyset$);
2. otherwise if there is at least an instantiation variable ($\|\mathbf{Y}\| > 0$).

In the first case, there is exactly one successor state ω', so applying Lemma 15 and Corollary 6 we can conclude that:

$$\omega' \not\models C \sqsubseteq D$$

In the second case, since we have assumed that the implication does not hold, applying the same observations of the previous case we can also conclude that:

$$\omega' \not\models C \sqsubseteq D$$

but it is not enough to conclude the proof, since the definition of valid e-service requires that exists at least a valid instantiation assignment satisfying the constraints, not that any assignment has this property. However, according to Theorem 23, a successor state is isomorphic to any other in the enactment. If we assume that $\langle [\omega']^*, [\sigma'_{\mathbf{Y}}]^* \rangle \in S(\omega, \sigma_{\mathbf{X}})$ is the pair s.t.:

$$[\omega']^* \models \mathcal{W}$$

and, since the interpretations $[\omega']^*$ and ω' are isomorphic, they must satisfying the same DL axioms, so:

$$\omega' \models C \sqsubseteq D$$

Obtaining the contradiction that concludes the proof.

Now we assume that the implication holds, since the service is consistent and accessible there exists at least an enactment $\langle \omega', \sigma'_{\mathbf{Y}} \rangle \in S(\omega, \sigma_{\mathbf{X}})$, but, since we are assuming by contradiction, that the service is not valid, does not exist any ω' resulting from the enactment s.t.:

$$\omega' \models \mathcal{W}$$

W.l.o.g., we assume that there exists an axiom $C \sqsubseteq D \in \mathcal{W}$ s.t.:

$$\omega' \not\models C \sqsubseteq D$$

Applying Theorem 21 we obtain a structure $\tilde{\omega}$ that is also a model of the knowledge based KB_m^U. Since the assumption, it is also a model of $\tau_m(\mathcal{W})$:

$$\tilde{\omega} \models \tau_m(\mathcal{W})$$

that implies also that:

$$\tau_m(C)^{\tilde{\omega}} \subseteq \tau_m(D)^{\tilde{\omega}}$$

According to Lemma 15, we can establish that the final state ω' is embedded. w.r.t. the name mapping m into $\tilde{\omega}$, so we can apply Corollary 6, obtaining that $\tau_m(C)^{\tilde{\omega}} = C^{\omega'}$ and $\tau_m(D)^{\tilde{\omega}} = D^{\omega'}$, concluding that:

$$\omega' \models C \sqsubseteq D$$

contradicting the hypothesis that ω' is not a legal world state. □

Theorem 25. *Given a world specification $\mathcal{W}$ and an accessible and consistent service S, the problem of checking if S is also valid is in* coNEXP.

Proof. As done for other cases previously analyzed, we can solve the problem applying the property proved in Theorem 24, reducing it to an implication decision in $\mathcal{C}^2$ logics, hence we use the result of Proposition 1. Like other reductions it is also polynomial in the size of the input (number and length of axioms, preconditions and effects specifications). □

4.3 Updates and Repairs

Differently from other approaches, so far we have assumed that the effect specification of a service is completely defined: in other words, there is no left space for any kind of collateral effects. Despite this assumption is closer to traditional information system design methodologies, it is quite different from applications of knowledge-based techniques, which generally consider the specification as incomplete and exploit reasoning features in order to achieve a consistent behavior.

There are different examples of such a philosophy in different fields, from the query-answering over inconsistent databases [CLR03] to the revision of knowledge bases given a new piece of knowledge [EG92]. In the field of update theory, the notion of repair is old at least as the notion of update itself [Win90, AHV95a], also in applications based upon expressive Description Logics [DLPR06]. However, the problem of repairing even a simple update in the presence of a complex intensional knowledge base (or a complex constraint set, using the DB terminology) turns out to be very hard, since, given the complexity of the axiom language, non-local repair side-effects may arise. It means that, in order to enforce consistently an update, we are required to retract a relevant part of the previous knowledge base. Some authors have addressed the problem limiting the constraint language to a simpler form (e.g., acyclic or definitorial TBox), but in the general case the problem is undecidable both in Description Logics ([BLM+05a]) and in relational database schemas ([AV89]). Intuitively, the search for a repair must be carried out considering necessarily an infinite number of candidate repairs.

Now, since we have introduced some limitations on the expressive power of the constraint language, do we need to renounce to the ability of repairing a partially-specified update? Generally speaking, since we can reduce, using some adjustments, our framework to the proposal of [BLM+05a], the problem shows up as undecidable also in this case, but, if we renounce to the completeness of the repair search, limiting to a restricted and finite set of possible repairs, the answer can be positive, since, as we show in the follow, we regain the decidability.

We are not necessarily interested into repairs that impose a complete revision of the world state, so we can keep into account an incomplete strategy, but how can we bound the search obtaining a possibly useful result?

The devised approach, that is clearly an approximation of a minimal-change repair, relies on the syntactical generation of repairing additional effects starting from singleton values, like variables and constants, mentioned in the service definition. The intuition behind this is that any repair not only should be as small as possible in terms of affected elements, but also should act locally, involving, if possible, only elements "close" to elements affected by the service itself.

Example 6. *Given the following world specification:*

$$\mathcal{W} = \{\mathsf{Student} \sqsubseteq \mathsf{Person}, \mathsf{Worker} \sqsubseteq \mathsf{Person}\}$$

on the alphabet $\boldsymbol{A} = \{\mathsf{Person}, \mathsf{Student}, \mathsf{Worker}\}$ *and the following service specification:* $\langle \boldsymbol{X} = \emptyset, \boldsymbol{Y} = \{y\}, \mathcal{P} = \emptyset, E = \{+\mathsf{Student}(y)\}\rangle$*, it is trivial to observe that the service is not valid, since the insertion of a new element in the extension of the concept* Student *violates the first axiom.*

In order to enforce this constraint, a simple repair like $\{+\mathsf{Person}(y)\}$ *is enough.*

Example 7. *The following version of the service S presented in Example 2 is not valid:*

$$\begin{aligned}
&\boldsymbol{X} = \{x_1, x_2\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x_1 \sqcap \mathsf{Citizen} \text{ and } x_2 \sqcap \mathsf{Town} \text{ and not } x_2 \sqcap \exists \mathsf{residentIn}^{-}.x_1\\
&E = \left\{-\mathsf{residentIn}(x_1, \exists \mathsf{residentIn}^{-}.x_1)\right\}
\end{aligned}$$

because it violates the mandatory participation of concept Citizen *in relation* residentIn*:*

$$\mathsf{Citizen} \sqsubseteq \exists^{=1}\mathsf{residentIn}.\mathsf{Town}$$

However, the service effect can easily repaired considering an atomic update as:

$$+\mathsf{residentIn}(x_1, x_2)$$

Now we present the basic forms of update repair, in order to accordingly define the repair search space.

Definition 41 (Positive repair argument)**.** *A positive repair argument is any element in* $\boldsymbol{X}_S \cup \boldsymbol{Y}_S \cup \boldsymbol{O}$*, where* $\boldsymbol{X}_S$ *and* $\boldsymbol{Y}_S$ *are resp. the input and the instantiation variable sets of the service specification, while* $\boldsymbol{O}$ *is the object name set of the domain specification.*

Definition 42 (Negative repair argument)**.** *A positive repair argument is any element in* $\boldsymbol{X}_S \cup \boldsymbol{O}$*, where* $\boldsymbol{X}_S$ *is the input variable set of the service specification, while* $\boldsymbol{O}$ *is the object name set of the domain specification.*

Definition 43 (Atomic concept repair). *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, and let $\boldsymbol{Y}_S$ be the corresponding output variable names. An atomic concept repair is a triple $\langle s, A, a \rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the repair (insert or delete);*
- *$A \in \boldsymbol{A}$ is the target concept name;*
- *a is the argument of the repair (positive or negative) according to the sign s.*

Definition 44 (Atomic role repair). *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, and let $\boldsymbol{Y}_S$ be the corresponding output variable names. An atomic role repair is a quadruple $\langle s, P, l, r \rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the repair (insert or delete);*
- *$P \in \boldsymbol{P}$ is the target role name;*
- *l and r are the arguments of the repair (positive or negative) according to the sign s.*

Definition 45 (Conflicting atomic repairs). *A pair of concept or role atomic repairs is conflicting iff the forming repairs have the same target concept or role name and same argument(s) but different sign.*

Definition 46 (Simple repair). *A simple repair R for an e-service S is an arbitrary set of atomic concept and role repairs, possibly empty, s.t. it does not contain any pair of conflicting atomic repairs.*

Given an e-service S, the set of all simple repairs is denoted as R_S^*.

Definition 47 (Repair search family). *A repair family R_S for S is an arbitrary subset of R_S^*.*

Restricting our attention to simple repairs, we can assume as repair search space the set R_S^* or a specific repair family. It is worth noticing that usually such a set contains also the null-repair (represented as an empty set), since a repairable but non-valid service can sometime, but not always, exhibit a legal behavior.

Theorem 26. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$ and a simple e-service specification $S = \langle \boldsymbol{X}_S, \boldsymbol{Y}_S, \mathcal{P}_S, E_S \rangle$, there are at most $\mathcal{O}\left(2^{\|\boldsymbol{A}\| \cdot n + \|\boldsymbol{P}\| \cdot n^2}\right)$ distinct simple repairs, where $n = \|\boldsymbol{O}\| + \|\boldsymbol{X}_S\| + \|\boldsymbol{Y}_S\|$.*

Proof. Since the concept and role repairs are separately defined any simple repair R can be split into two components R^A and R^P, s.t. the first one contains only concept repairs while the latter only role repairs. In terms of cardinality, we have that:

$$\|R_S^*\| = \|R_S^A\| \cdot \|R_S^P\|$$

where R_S^A (resp. R_S^P) is the set of the set of atomic concept (resp. role) repairs. Since not all arbitrary atomic repair sets are allowed they are subset of corresponding power-sets. Generally speaking, we have $n^+ = \|\mathbf{O}\| + \|\mathbf{X}_S\| + \|\mathbf{Y}_S\|$ distinct positive arguments and $n^- = \|\mathbf{O}\| + \|\mathbf{X}_S\|$ distinct negative arguments.

We observe that $n^+ \geq n^-$ and, since we are looking for an upper bound of the cardinality, we ignore the distinction between positive and negative argument types. A simple concept repair can be considered as a function that maps any pair formed by a concept name and an argument to a value picked from the set $\{0, +1, -1\}$ according to such a pair is ignored from the repair (0), it is included as positive repair (+1) or as negative repair (−1). The number of admissible functions is $3^{\|\mathbf{A}\| \cdot n^+} \geq \left\|R_S^A\right\|$. Using analogous argumentation, we can also conclude that $3^{\|\mathbf{P}\| \cdot n^{+2}} \geq \left\|R_S^A\right\|$ and then that:

$$\|R_S^*\| \leq 3^{\|\mathbf{A}\| \cdot n^+} \cdot 3^{\|\mathbf{P}\| \cdot n^{+2}} = 3^{\|\mathbf{A}\| \cdot n + \|\mathbf{P}\| \cdot n^2}$$

where $n^+ = n$. □

The number of different repairs, or, in other words, the size of the search space, is finite and exponentially bounded by the number of alphabet elements (in terms of names), while they are substantially independent of the complexity of the world specification axioms and service effect statements.

Moreover, the size of the largest simple repair is polynomially bounded by the problem setting size.

Theorem 27. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$ and a simple e-service specification $S = \langle \boldsymbol{X}_S, \boldsymbol{Y}_S, \mathcal{P}_S, E_S \rangle$, the size of a simple repair is at most $\mathcal{O}\left(\|\boldsymbol{A}\| \cdot n + \|\boldsymbol{P}\| \cdot n^2\right)$, where $n = \|\boldsymbol{O}\| + \|\boldsymbol{X}_S\| + \|\boldsymbol{Y}_S\|$.*

Proof. This result follows from the observation that the largest simple repair can at most contains, for any possible positive argument and concept (resp. role) name, a positive atomic repair (the number of positive arguments is greater or equal than the number of negative ones). □

Now, we define the semantics of devised repair primitives.

Definition 48 (Repair concept insert set)**.** *Given a simple repair $R \in R_S^*$ and a concept name $A \in \boldsymbol{A}$, a world state ω, an input assignment $\sigma_{\boldsymbol{X}}$, an instantiation assignment $\sigma'_{\boldsymbol{Y}}$, the repair concept insert set is defined as:*

$$A_R^+(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) = \bigcup_{\langle +, A, X \rangle \in R} \{\sigma_{\boldsymbol{X}}(X)\} \cup \bigcup_{\langle +, A, Y \rangle \in R} \{\sigma'_{\boldsymbol{Y}}(Y)\} \cup \bigcup_{\langle +, A, O \rangle \in R} \{O^\omega\}$$

Definition 49 (Repair concept delete set)**.** *Given a simple repair $R \in R_S^*$ and a concept name $A \in \boldsymbol{A}$, a world state ω, an input assignment $\sigma_{\boldsymbol{X}}$, the repair concept delete set is defined as:*

$$A_R^-(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) = \bigcup_{\langle -, A, X \rangle \in R} \{\sigma_{\boldsymbol{X}}(X)\} \cup \bigcup_{\langle -, A, O \rangle \in R} \{O^\omega\}$$

Definition 50 (Repair role insert set)**.** *Given a simple repair $R \in R_S^*$ and a role name $P \in \boldsymbol{P}$, a world state ω, an input assignment $\sigma_{\boldsymbol{X}}$, an instantiation* *assignment $\sigma'_{\boldsymbol{Y}}$, the repair role insert set is defined as:*

$$\begin{aligned}P^{+}_{R}(\omega,\sigma_{\boldsymbol{X}},\sigma'_{\boldsymbol{Y}}) = & \bigcup_{\langle +,P,X,X'\rangle\in R}\{\sigma_{\boldsymbol{X}}(X)\}\times\{\sigma_{\boldsymbol{X}}(X')\}\cup\bigcup_{\langle +,P,X,Y\rangle\in R}\{\sigma_{\boldsymbol{X}}(X)\}\times\{\sigma'_{\boldsymbol{Y}}(Y)\}\\ \cup & \bigcup_{\langle +,P,X,O\rangle\in R}\{\sigma_{\boldsymbol{X}}(X)\}\times\{O^{\omega}\}\cup\bigcup_{\langle +,P,Y,X\rangle\in R}\{\sigma'_{\boldsymbol{Y}}(Y)\}\times\{\sigma_{\boldsymbol{X}}(X)\}\\ \cup & \bigcup_{\langle +,P,Y,Y'\rangle\in R}\{\sigma'_{\boldsymbol{Y}}(Y)\}\times\{\sigma'_{\boldsymbol{Y}}(Y')\}\cup\bigcup_{\langle +,P,Y,O\rangle\in R}\{\sigma_{\boldsymbol{X}}(X)\}\times\{O^{\omega}\}\\ \cup & \bigcup_{\langle +,P,O,X\rangle\in R}\{O^{\omega}\}\times\{\sigma_{\boldsymbol{X}}(X)\}\cup\bigcup_{\langle +,P,O,Y\rangle\in R}\{O^{\omega}\}\times\{\sigma'_{\boldsymbol{Y}}(Y)\}\\ \cup & \bigcup_{\langle +,P,O,O'\rangle\in R}\{O^{\omega}\}\times\{O'^{\omega}\}\end{aligned}$$

Definition 51 (Repair role delete set)**.** *Given a simple repair $R \in R^{*}_{S}$ and a role name $P \in \boldsymbol{P}$, a world state ω, an input assignment $\sigma_{\boldsymbol{X}}$, the repair role delete set is defined as:*

$$\begin{aligned}P^{-}_{R}(\omega,\sigma_{\boldsymbol{X}},\sigma'_{\boldsymbol{Y}}) = & \bigcup_{\langle -,P,X,X'\rangle\in R}\{\sigma_{\boldsymbol{X}}(X)\}\times\{\sigma_{\boldsymbol{X}}(X')\}\cup\bigcup_{\langle -,P,X,O\rangle\in R}\{\sigma_{\boldsymbol{X}}(X)\}\times\{O^{\omega}\}\\ \cup & \bigcup_{\langle -,P,O,X\rangle\in R}\{O^{\omega}\}\times\{\sigma_{\boldsymbol{X}}(X)\}\cup\bigcup_{\langle -,P,O,O'\rangle\in R}\{O^{\omega}\}\times\{O'^{\omega}\}\end{aligned}$$

Remark 16. *The provided semantics is well-founded also in the case some instantiation variable name $Y \in \boldsymbol{Y}$ is not assigned: in such a case it is simply interpreted as the empty set.*

Intuitively, the repair concept (resp. role) insert (resp. delete) set of a name contains all the elements (resp. element pairs) that are also inserted (resp. deleted) from the name interpretation in order to obtain a service enactment that realizes its own effects giving a new legal world state.

Given the definition of repair sets we can easily obtain the following result:

Lemma 18. *Given a service S and a repair $R \in R^{*}_{S}$ and a concept name $A \in \boldsymbol{A}$, let ω, $\sigma_{\boldsymbol{X}}$, and $\sigma'_{\boldsymbol{Y}}$ be resp. the initial state, input and instantiation variable assignment, then the following conditions hold:*

$$\begin{aligned}\left\|A^{+}_{R}(\omega,\sigma_{\boldsymbol{X}},\sigma'_{\boldsymbol{Y}})\right\| &\leq \|\{r|r=\langle +,A,a\rangle\in R\}\|\\ \left\|A^{-}_{R}(\omega,\sigma_{\boldsymbol{X}})\right\| &\leq \|\{r|r=\langle -,A,a\rangle\in R\}\|\end{aligned}$$

Proof. The claim follows from the observation that each atomic repair at most affect one distinct element (i.e., two different atomic repairs can affect the same element depending upon the variable assignment), so the number of affected elements of a concept name is less than the number of atomic repair having such a name as target. The same argumentation can be applied to role repair. □

Lemma 19. *Given a service S and a repair $R \in R^{*}_{S}$ and a role name $P \in \boldsymbol{P}$, let ω, $\sigma_{\boldsymbol{X}}$, and $\sigma'_{\boldsymbol{Y}}$ be resp. the initial state, input and instantiation variable assignment, then the following conditions hold:*

$$\begin{aligned}\left\|P^{+}_{R}(\omega,\sigma_{\boldsymbol{X}},\sigma'_{\boldsymbol{Y}})\right\| &\leq \|\{r|r=\langle +,P,l,r\rangle\in R\}\|\\ \left\|P^{-}_{R}(\omega,\sigma_{\boldsymbol{X}})\right\| &\leq \|\{r|r=\langle -,P,l,r\rangle\in R\}\|\end{aligned}$$

Given a service provider with the update repair capability previously defined, we need to refine the definitions related to system dynamics, in order to keep into account also the repairing step that, intuitively, follows the instantiation and updating steps.

Definition 52 (Candidate repaired successor state). *Given a world specification $\mathcal{W}$ and an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a service S, let $R \in R^*_S$ be a simple repair. The associated candidate repaired successor state ω'_R is an interpretation structure s.t.:*

- *its interpretation domain is the same as the successor state ($\Delta^{\omega'} = \Delta^{\omega'_R}$);*
- *the interpretation of each concept name $A \in \boldsymbol{A}$ is adjusted accordingly the repair:*
$$A^{\omega'_R} = (A^{\omega'} \setminus A^-_R(\omega, \sigma_{\boldsymbol{X}})) \cup A^+_R(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$$
- *the interpretation of each role name $P \in \boldsymbol{P}$ is adjusted accordingly the repair:*
$$P^{\omega'_R} = (P^{\omega'} \setminus P^-_R(\omega, \sigma_{\boldsymbol{X}})) \cup P^+_R(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$$
- *the interpretation of each object name $O \in \boldsymbol{O}$ is left unchanged ($O^{\omega'_R} = O^{\omega'}$).*

The provided definition is not complete: in fact, in order to be useful and safe, a repair R should be s.t.:

- it does not "undo" the effects of the service (e.g., deleting an element just inserted);
- it actually updates the world state (e.g., it should not insert an element already present into a set or just inserted by the service enactment).

In other words, since candidate repairs are essentially generated in a syntactical fashion, we need to enforce that their semantics is consistent, given the world state and the variable assignments.

Definition 53 (Consistent simple repair). *Given a repair $R \in R^*_S$ for a simple e-service S, a world state ω, an input and an instantiation variable assignments $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ consistently defined, the repair is consistent iff for each concept name $A \in \boldsymbol{A}$ the following conditions hold:*

$$\begin{aligned}
A^+_R(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cap (A^\omega \cup A^+(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})) &= \emptyset \\
A^+_R(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cap (A^-_R(\omega, \sigma_{\boldsymbol{X}}) \cup A^-(\omega, \sigma_{\boldsymbol{X}})) &= \emptyset \\
A^-_R(\omega, \sigma_{\boldsymbol{X}}) \cap (A^+(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cup A^-(\omega, \sigma_{\boldsymbol{X}})) &= \emptyset \\
A^-_R(\omega, \sigma_{\boldsymbol{X}}) &\subseteq A^\omega
\end{aligned}$$

and for each role name $P \in \boldsymbol{P}$ the following conditions hold:

$$\begin{aligned}
P^+_R(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cap (P^\omega \cup P^+(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})) &= \emptyset \\
P^+_R(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cap (P^-_R(\omega, \sigma_{\boldsymbol{X}}) \cup P^-(\omega, \sigma_{\boldsymbol{X}})) &= \emptyset \\
P^-_R(\omega, \sigma_{\boldsymbol{X}}) \cap (P^+(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cup P^-(\omega, \sigma_{\boldsymbol{X}})) &= \emptyset \\
P^-_R(\omega, \sigma_{\boldsymbol{X}}) &\subseteq P^\omega
\end{aligned}$$

Definition 54 (Repaired successor state)**.** *Given a world specification $\mathcal{W}$ and an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a service S, let $R \in R^*_S$ be a simple repair. The associated repaired successor state ω'_R is an interpretation structure s.t.:*

- *the state ω'_R is a candidate repair successor state of the enactment applying R;*
- *R is consistent given ω, $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$;*
- *ω'_R is a legal world state w.r.t. $\mathcal{W}$ even if ω' is not.*

We notice that, since also the null repair is allowed ($R_\emptyset = \emptyset$), in the case of consistent e-service each successor state is also a repaired successor state. Moreover, among multiple repaired successor states (assuming that at least one of them exists), the repairing strategy selects the one nearest to the base successor state ω', in terms of symmetric difference between interpretation structures, in order to enforce a kind of minimal-change repair.

Definition 55 (Repaired transition relation)**.** *Let ω and ω'_R be a pair of world states, satisfying the world specification $\mathcal{W}$, s.t. the latter is resulting from the execution of the effect E of an e-service S in the state defined from the former applying a repair $R \in R^*_S$. Given an input and an output variable assignments $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ consistently defined, there exists a system state transition from ω to ω'_R using the specified e-service effect iff:*

- *ω'_R is a repaired successor state of the enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$;*
- *there does not exist any other repaired successor state $\omega'_{R'}$ of the same enactment s.t. it is closer to ω' than ω'_R according to the symmetric difference distance (which means that $d(\omega'_R, \omega') \leq d(\omega', \omega'')$ for each $\omega'_{R'}$ s.t. $R' \in R^*_S$ and R is consistent).*

Theorem 28. *Given a world specification $\mathcal{W}$, an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a service S and a repair $R \in R^*_S$, let ω'_R be the repaired successor state, then:*

$$d(\omega'_R, \omega') \leq \|R\|$$

Proof. Applying the definition of symmetric difference we obtain that:

$$d(\omega'_R, \omega') = \sum_{A \in \mathbf{A}} \left\| A^{\omega'_R} \uplus A^{\omega'} \right\| + \sum_{P \in \mathbf{P}} \left\| P^{\omega'_R} \uplus P^{\omega'} \right\|$$

Let A be a concept name[8], its contribution to distance is:

$$d_A = \left\| A^{\omega'_R} \setminus A^{\omega'} \right\| + \left\| A^{\omega'} \setminus A^{\omega'_R} \right\|$$

Since the repair is consistent, applying Corollary 4 we can conclude that:

$$\begin{array}{rcl} A^-_R(\omega, \sigma_{\mathbf{X}}) & \subseteq & A^{\omega'} \\ A^+_R(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}) \cap A^{\omega'} & = & \emptyset \end{array}$$

[8]For the sake of simplicity we consider only the case of concept contribution to structure distance, since the same argumentation also holds for the case of role names.

Using the definition of repaired successor state we can hence infer that:

$$\begin{array}{rcl} A^{\omega'_R} \setminus A^{\omega'} & = & A^{+}_{R}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}) \\ A^{\omega'} \setminus A^{\omega'_R} & = & A^{-}_{R}(\omega, \sigma_{\mathbf{X}}) \end{array}$$

Applying the result of Lemma 18, we can establish that:

$$d_A \leq \|\{r|\langle +, A, a\rangle \in R\}\| + \|\{r|\langle -, A, a\rangle \in R\}\|$$

Summing up we have that:

$$\begin{array}{rcl} d(\omega'_R, \omega') & \leq & \sum_{A \in \mathbf{A}} (\|\{r|\langle +, A, a\rangle \in R\}\| + \|\{r|\langle -, A, a\rangle \in R\}\|) \\ & + & \sum_{P \in \mathbf{P}} (\|\{r|\langle +, P, l, r\rangle \in R\}\| + \|\{r|\langle -, P, l, r\rangle \in R\}\|) \\ & = & \|R\| \end{array}$$

since the sum is computed over a complete partition of the set R. □

Theorem 29. *Given a world specification $\mathcal{W}$, an enactment $\langle \omega', \sigma'_{\mathbf{Y}}\rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a service S and a repair R s.t. , let ω'_R be the repaired successor state, $d(\omega'_R, \omega') < \|R\|$, then there exists a repair R' subset of R, s.t. $d(\omega'_{R'}, \omega') = \|R'\|$.*

Proof. The size of a repair is strictly greater than the distance among world states if, given the variable assignment, two or more atomic repair are interpreted on the same extension. Removing the redundant atomic repair we obtain a new repair whose size is equal to the distance between interpretation structure □

We notice that such property does not hold for any restricted repair: in fact the singleton semantics associated to atomic repairs allows us to determine an upper bound to the impact of the repair itself. Moreover, in case of general repairs there can be cases in which it is impossible *a priori* to state the distance between original and repaired states or the number of possible repaired alternatives at a given distance as the following example shows.

Example 8. *Let ω be a world state of a very simple domain that specifies only a relation name r. The world specification contains the following constraints:*

$$\begin{array}{rcl} \exists r^{-}.\top & \equiv & \exists r.\top \\ \exists^{=1} r^{-}.\top & \sqsubseteq & \exists^{=1} r.\top \end{array}$$

The role r is globally functional and it forms infinite chains on infinite interpretation domains or loop on finite ones. W.l.o.g., assume that Δ^{ω} is finite and that the interpretation of r is the following:

$$r^{\omega} = \{\langle o_1, o_2\rangle, \ldots, \langle o_{n-1}, o_n\rangle, \langle o_n, o_1\rangle\}$$

where $\{o_1, o_2, \ldots, o_n\} \subseteq \Delta^{\omega}$.

Given a generic update statement of the form $-r(o_i, o_{i+1})$, that simply breaks the chain in some point, it is worth noticing that a possible repair that enforces *the update effects and leads to a legal world state is s.t. the final interpretation of r is empty ($r^{\omega''} = \emptyset$). Hence the distance between the original and repaired final state is:*

$$d(\omega', \omega'') = n - 1$$

and it depends only on extensional elements, so it cannot be bounded a priori *in any way. Alternatively, any other repair that split the chain into more pieces and close them as loop adding a new link for each one. In this case the closest repair requires at least the removal of a link, splitting the chain into two pieces and two new links to close them as loops, hence it is at distance 3 from the updated model, but there is a number of alternatives ($n - 1$) among them it is possible to select the link to remove.*

Given Theorem 28, we are able to find the semantically minimal-change repair, considering it syntactical size (in terms of cardinality of atomic repair). The corresponding search algorithm (see Figure 4.5) explores the repair search state R_S considering at each step only simple repairs of specific size from the smallest (possibly empty) to largest one.

In order to completely deal with repairs in terms of size of the affected model we need to introduce a supplementary constraint on the search space, that turns to be extremely natural and not particularly restrictive.

Definition 56 (Normal repair search family)**.** *A repair family R_S is a normal repair family iff:*

- *it includes the null repair ($\emptyset$);*
- *given any non-empty repair $R \in R_S$, each subset $R' \subset R$ s.t. $\|R\| = \|R'\| + 1$ is also included in R_S.*

Remark 17. *Unless differently stated, we assume as repair search space the whole R_S^* set, but this characterization enables to tune the repair algorithm according to application requirements.*

Lemma 20. *A normal repair family contains any arbitrary subset of its elements.*

Proof. Trivial. □

Lemma 21. *The simple repair search algorithm always returns the minimal-change repair, provided that the repair search space (family) is normal.*

Proof. The algorithm clearly return a suitable repair if it exists. By contradiction, we assume that the algorithm returns a repair R having size n, but there is another repair $R' \in R_S$ s.t. $d(\omega', \omega''_{R'}) < d(\omega', \omega''_R) \leq n$, that should be preferred. Since the algorithm has selected R instead of R' its size should be at least n, but according to Theorem 29, a repair R'' subset of R' must exist and its size must be less than n. But if such a repair exists, it must be included into R_S according to Lemma 20, since it is normal, and consequently it has been visited by the algorithm before R and R', given the search strategy. □

According to the repair approach, which assumes a repair step following the service update, we need to refine the embedding relation in order to keep into account also the intermediate transient state.

Definition 57 (Simple e-service repaired enactment embedding relation)**.** *Given three world states (ω, ω', and ω''), having as interpretation domain a subset of $\mathfrak{U}$ and s.t. $\Delta^{\omega'} = \Delta^{\omega''}$, an input assignment $\sigma_{\boldsymbol{X}}$ and an instantiation assignment $\sigma'_{\boldsymbol{Y}}$ both consistent w.r.t. ω, let m and n be two functions that map each concept (resp. role) name A (resp. P) into a new one $m(A)$ or $n(A)$ (resp. $m(P)$ or $n(P)$), and let Top, Top_m and Top_n be new concept names and let $\hat{\omega} = \langle \mathfrak{U}, \cdot^{\hat{\omega}} \rangle$ be an interpretation over the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{X} \cup \boldsymbol{Y} \cup m(\boldsymbol{A}) \cup n(\boldsymbol{A}) \cup \{\mathsf{Top}, \mathsf{Top}_m, \mathsf{Top}_n\}, \boldsymbol{P} \cup m(\boldsymbol{P}) \cup n(\boldsymbol{P}), \boldsymbol{O} \rangle$. We say that the tuple is embedded by the repair into the interpretation $\hat{\omega}$ iff the pair ω, ω' is embedded into $\hat{\omega}$ and the following auxiliary conditions hold:*

$$\Delta^{\omega''} = \mathsf{Top}_n^{\hat{\omega}}$$
$$N^{\omega''} = n(N)^{\hat{\omega}}$$

for each $N \in \boldsymbol{A} \cup \boldsymbol{P}$.

This is a generalization of the enactment embedding relation that keeps into account also the repair step from the state ω' to the state ω''. From the definition can be easily obtained the following claims:

Lemma 22. *Given a quintuple $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ and an interpretation $\hat{\omega}$, s.t. the quintuple is embedded into it, then:*

1. *the quadruple $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ is embedded into $\hat{\omega}$;*
2. *the repaired world state ω'' is embedded, according to the mapping n, into the structure $\hat{\omega}$:*
$$\omega'' \rightsquigarrow_n \hat{\omega}$$

As done for simple e-services, we can employ the mapping function to menage multiple repairs at the same time, since also repairs act in a non-deterministic fashion. So, given a repair $R \in R^*_S$, the default mapping function in the case of simple e-service is $n(x) = x_R$ and $\mathsf{Top}_n = \mathsf{Top}'$. Moreover, in the case of multiple effects (e.g., non-deterministic or conditional service), the mapping function is defined as $n(x) = x_{E,R}$ and $\mathsf{Top}_n = \mathsf{Top}_E$, where E is the effect.

Given a service S, an effect specification E for the same service and a repair R, we define two new axiom schemas denoted as $\Delta KB^R(m,n)$ and $\Delta KB^C(m,n)$, presented in Tables 4.4 and 4.5, where $\cdot^+$, $\cdot^-$, $\cdot^*$ denotes the names of new auxiliary concepts ands role introduced for every concept A or role P in the domain specification.

Given such axiom schema $\Delta KB^U(m,n) \triangleq \Delta KB^R(m,n) \cup \Delta KB^C(m,n)$, we define a new knowledge base $KB^U_{m,n}$ adding to the knowledge base KB^U_m, defined as shown at page 72, the instantiation of the schemas on the domain specification alphabet.

Remark 18. *In order to complete the encoding of the knowledge base into a $\mathcal{C}^2$ theory we need to cope accordingly with nominals, but they can be replaced with distinct singleton sets implemented using additional unary predicate $\tilde{O}$ and adding the following axiom:*

$$\exists^{=1} x.\tilde{O}(x)$$

for each object name and the following for each distinct pair of object names O and O':

$$\neg \exists x.\tilde{O}(x) \wedge \tilde{O}'(x)$$

Table 4.4: The axiom schema $\Delta KB^R(m,n)$

$$\begin{aligned}
\mathsf{Top}_m &\equiv \mathsf{Top}_n && (4.16)\\
n(A) &\equiv (m(A) \sqcap \neg n(A)^-) \sqcup n(A)^+ && (4.17)\\
n(A)^+ &\equiv \bigsqcup_{\langle +,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle +,A,Y\rangle \in R} Y \sqcup \bigsqcup_{\langle +,A,O\rangle \in R} \{O\} && (4.18)\\
n(A)^- &\equiv \bigsqcup_{\langle -,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle -,A,O\rangle \in R} \{O\} && (4.19)\\
\forall x,y.n(P)(x,y) &\leftrightarrow (m(P)(x,y) \wedge \neg n(P)^-(x,y)) \vee n(P)^+(x,y) && (4.20)\\
\forall x,y.n(P)^+(x,y) &\leftrightarrow \bigvee_{\langle +,P,X,X'\rangle \in R} X(x) \wedge X'(y) && (4.21)\\
&\vee \ \dots\\
\forall x,y.n(P)^-(x,y) &\leftrightarrow \bigvee_{\langle -,P,X,X'\rangle \in R} X(x) \wedge X'(y) && (4.22)\\
&\vee \bigvee_{\langle -,P,X,O\rangle \in R} X(x) \wedge O = y\\
&\vee \bigvee_{\langle -,P,O,X\rangle \in R} O = x \wedge X(y)\\
&\vee \bigvee_{\langle -,P,O,O'\rangle \in R} O = x \wedge O' = y
\end{aligned}$$

Table 4.5: The axiom schema $\Delta KB^C(m,n)$

$$\begin{aligned}
n(A)^+ \sqcap (A \sqcup m(A)^*) &\sqsubseteq \bot && (4.23)\\
n(A)^+ \sqcap (n(A)^- \sqcup m(A)^-) &\sqsubseteq \bot && (4.24)\\
n(A)^- \sqcap (m(A)^+ \sqcup m(A)^-) &\sqsubseteq \bot && (4.25)\\
n(A)^- &\sqsubseteq A && (4.26)\\
\forall x,y.n(P)^+(x,y) \wedge (P(x,y) \vee m(P)^*(x,y)) &\rightarrow \bot && (4.27)\\
\forall x,y.n(P)^+(x,y) \wedge (n(P)^-(x,y) \vee m(P)^-(x,y)) &\rightarrow \bot && (4.28)\\
\forall x,y.n(P)^-(x,y) \wedge (m(P)^+(x,y) \vee m(P)^-(x,y)) &\rightarrow \bot && (4.29)\\
\forall x,y.n(P)^-(x,y) &\rightarrow P(x,y) && (4.30)
\end{aligned}$$

A comparison expression like $x = O$ (using FOL syntax) or $\{O\}$ (using DL syntax) can be replaced with the formula $\tilde{O}(x)$, while an extensional assertion $o : C$ is equivalent to $\forall x.\tilde{O}(x) \to C(x)$. We remark that, given the description logic language employed to axiomatize the world structural constraints ($\mathcal{ALCQI}$) and the kind of additional axioms introduced in the dynamic problem encoding, as done devising the various knowledge base presented, there is not any other case in which we need to deal with nominals or arbitrary constants. Furthermore, the provided construction can be extended to freely deal with a finite number of constants in $\mathcal{C}^2$ language as shown by [PH05].

Starting from the previous definition of embedding function μ, provided at page 74, we extend it in order to deal with repaired enactments too: given a quintuple $\langle \omega, \omega', \omega'' \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$ we define an embedding function μ^R that maps such structures into another interpretation $\hat{\omega}$ s.t.:

- the interpretation domain is the whole universe ($\Delta^{\hat{\omega}} = \mathfrak{U}$);
- the interpretation of concepts, roles and objects in the starting state is preserved ($N^{\omega} = N^{\hat{\omega}}$);
- the interpretation of concepts, roles and objects in the update resulting state is preserved ($N^{\omega'} = m(N)^{\hat{\omega}}$);
- the interpretation of concepts, roles and objects in the repair resulting state is preserved ($N^{\omega''} = n(N)^{\hat{\omega}}$);
- the interpretation of Top is the active domain of ω ($\mathsf{Top}^{\hat{\omega}} = \Delta^{\omega}$);
- the interpretation of Top_m and Top_n is the active domain of ω' ($\mathsf{Top}_m^{\hat{\omega}} = \mathsf{Top}_n^{\hat{\omega}} = \Delta^{\omega'} = \Delta^{\omega''}$)[9];
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;
- the interpretation of variable auxiliary concepts is defined according to the assignment ($\sigma_{\mathbf{X}}(X) = X^{\hat{\omega}}$ and $\sigma'_{\mathbf{Y}}(Y) = Y^{\hat{\omega}}$);
- the update-defining concepts and roles are interpreted according to the corresponding concept (resp. role) insert or delete set ($[m(N)^+]^{\hat{\omega}} = N^+(\omega, \sigma_{\mathbf{X}})$, $[m(N)^-]^{\hat{\omega}} = N^-(\omega, \sigma_{\mathbf{X}})$, $[m(N)^*]^{\hat{\omega}} = N^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$);
- the repair related concepts and roles are interpreted according to the corresponding concept (resp. role) insert or delete set ($[n(N)^-]^{\hat{\omega}} = N_R^-(\omega, \sigma_{\mathbf{X}})$, $[n(N)^+]^{\hat{\omega}} = N_R^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$);
- the object spy is assigned to an element of $\mathsf{New}^{\hat{\omega}}$, if any;
- the role aux is interpreted as $\left\{\mathsf{spy}^{\hat{\omega}}\right\} \times \mathsf{New}^{\hat{\omega}}$.

The function π^R computes the inverse of μ^R, it projects out from an interpretation $\hat{\omega}$ a quintuple, representing possibly an enactment between the world states ω and ω' repaired into ω'', given the variable assignments, and is defined only for structures that are models of the knowledge base $KB^U_{m,n}$. We remark

[9]According to the definition we have that ω' and ω'' have always the same interpretation domain, because we are ignoring instantiation-capable repairs.

that the devised construction is a conservative extension of the one used for the enactment embedding relation.

Moreover, using analogous argumentation, we can state the following claims that generalize Lemmas 16 and 17 and related results:

Lemma 23. *Given a quintuple $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$, and structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu^R(\omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then the quintuple is embedded into $\hat{\omega}$ as repaired enactment.*

Lemma 24. *Given a model $\hat{\omega}$ of $KB^U_{m,n}$, let $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ be a tuple s.t. $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi^R(\hat{\omega})$, then it is embedded into $\hat{\omega}$ as repaired enactment.*

Lemma 25. *Given a quintuple $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$, and structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu^R(\omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then, let $\bar{\omega}$ be the structure s.t. $\bar{\omega} = \mu(\omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then it is equal to $\hat{\omega}$ restricted to the common alphabet.*

Proof. The claim follows from the observation that the second mapping function (μ^R), the one that keeps into account also the repair, is a conservative generalization of the first one (μ), so the interpretation of shared names is the same. □

Lemma 26. *Given a model $\hat{\omega}$ of $KB^U_m \wedge \Delta KB^R(m, n)$, let $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ be a tuple s.t. $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi^R(\hat{\omega})$, then $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi(\hat{\omega})$.*

Proof. The claim follows from the fact that the mentioned mapping functions agree upon the definition of common structures. □

Theorem 30. *Given an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a consistently defined service, and a consistent simple repair $R \in R^*_S$ and repaired successor state ω'' of ω' using R, let $\hat{\omega}$ be a structure s.t. $\hat{\omega} = \mu^R(\omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then:*

$$\hat{\omega} \models KB^U_{m,n}$$

Proof. According to Lemma 25 we can apply Theorem 21 to show that the structure $\bar{\omega}$ is a model of the knowledge base KB^U_m. Since the structure $\hat{\omega}$ is an extension of the structure $\bar{\omega}$ that simply adds new name interpretations, without altering the interpretation on which the satisfiability result relies, we can also conclude that:

$$\hat{\omega} \models KB^U_m$$

so, as in previous proofs, in order to prove the claim, we need only to show that additional axioms also hold. Given the definition of the embedding function, the interpretation of concept Top_m and Top_n is always the same so the constraints in Eq. 4.16 is satisfied.

In order to completely prove that $\hat{\omega} \models KB^U_{m,n}$, we need to show that the remain axioms hold too. These axioms encode the extension affected by the repair as it is specified by the mean of repair concept/repair insert/delete set. For the sake of succinctness, we will show the result using the axiom of Eq. 4.18, other ones can be obtained using the same argumentation.

According to the definition of mapping function, we have that:

$$[n(A)^+]^{\hat{\omega}} = A^+_R(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

On the other hand, according to the definition of repair concept insert set:

$$[n(A)^{+}]^{\hat{\omega}} = \bigcup_{\langle +,A,X\rangle \in R} \{\sigma_{\mathbf{X}}(X)\} \cup \bigcup_{\langle +,A,Y\rangle \in R} \{\sigma'_{\mathbf{Y}}(Y)\} \cup \bigcup_{\langle +,A,O\rangle \in R} \{O^{\omega}\}$$

Applying the standard DL semantics to the right side of the axioms, we have also that:

$$\left[\bigsqcup_{\langle +,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle +,A,Y\rangle \in R} Y \sqcup \bigsqcup_{\langle +,A,O\rangle \in R} \{O\}\right]^{\hat{\omega}} =$$
$$\bigcup_{\langle +,A,X\rangle \in R} X^{\hat{\omega}} \cup \bigcup_{\langle +,A,Y\rangle \in R} Y^{\hat{\omega}} \cup \bigcup_{\langle +,A,O\rangle \in R} \{O^{\hat{\omega}}\}$$

Applying the definition of embedding we obtain that:

$$\left[\bigsqcup_{\langle +,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle +,A,Y\rangle \in R} Y \sqcup \bigsqcup_{\langle +,A,O\rangle \in R} \{O\}\right]^{\hat{\omega}} =$$
$$\bigcup_{\langle +,A,X\rangle \in R} \{\sigma_{\mathbf{X}}(X)\} \cup \bigcup_{\langle +,A,Y\rangle \in R} \{\sigma'_{\mathbf{Y}}(Y)\} \cup \bigcup_{\langle +,A,O\rangle \in R} \{O^{\omega}\}$$

Concluding that:

$$\hat{\omega} \models n(A)^{+} \equiv \bigsqcup_{\langle +,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle +,A,Y\rangle \in R} Y \sqcup \bigsqcup_{\langle +,A,O\rangle \in R} \{O\}$$

The argumentation can be extended also to other axioms encoded by Equations 4.19, 4.21, and 4.22, completing the proof of the claim.

We prove that Eq. 4.20 holds by contradiction. Assuming that there exists a pair $\langle x^*, y^*\rangle$ s.t.:

$$\hat{\omega} \not\models n(P)(x^*, y^*)$$

in the spite that we have that:

$$\hat{\omega} \models (m(P)(x^*, y^*) \wedge \neg n(P)^{-}(x^*, y^*)) \vee n(P)^{+}(x^*, y^*)$$

We have two cases:

1. if $\langle x^*, y^*\rangle \in [n(P)^{+}]^{\hat{\omega}}$, so, according to the definition of embedding function, we have that $\langle x^*, y^*\rangle \in P^{+}_{R}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$; given the definition of repaired successor state we can conclude that $\langle x^*, y^*\rangle \in P^{\omega''}$ and consequently that $\langle x^*, y^*\rangle \in n(P)^{\hat{\omega}}$;

2. if $\langle x^*, y^*\rangle \in [m(P)]^{\hat{\omega}}$ and $\langle x^*, y^*\rangle \notin [n(P)^{-}]^{\hat{\omega}}$; applying the definition of embedding function we obtain that $\langle x^*, y^*\rangle \in P^{\omega'}$ and $\langle x^*, y^*\rangle \notin P^{-}_{R}(\omega, \sigma_{\mathbf{X}})$; given the definition of repaired successor state we can conclude that $\langle x^*, y^*\rangle \in P^{\omega''}$ and, applying the definition of mapping function, that $\langle x^*, y^*\rangle \in n(P)^{\hat{\omega}}$.

Now we assume that there exists a pair $\langle x^*, y^*\rangle$ s.t.:

$$\hat{\omega} \not\models (m(P)(x^*, y^*) \wedge \neg n(P)^{-}(x^*, y^*)) \vee n(P)^{+}(x^*, y^*)$$

despite we have that:

$$\hat{\omega} \models n(P)(x^*, y^*)$$

Applying the De Morgan's laws we have two cases:

1. if $\langle x^*, y^* \rangle \notin [n(P)^+]^{\hat{\omega}}$ and $\langle x^*, y^* \rangle \notin [m(P)]^{\hat{\omega}}$, so, according to the definition of embedding function we have that $\langle x^*, y^* \rangle \notin P^+_R(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$ and $\langle x^*, y^* \rangle \notin P^{\omega'}$; given the definition of repaired successor state we can infer that $\langle x^*, y^* \rangle \notin P^{\omega''}$ and consequently that $\langle x^*, y^* \rangle \notin n(P)^{\hat{\omega}}$;

2. if $\langle x^*, y^* \rangle \notin [n(P)^+]^{\hat{\omega}}$ and $\langle x^*, y^* \rangle \in [n(P)^-]^{\hat{\omega}}$; applying the definition of embedding function we obtain that $\langle x^*, y^* \rangle \notin P^+_R(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$ and $\langle x^*, y^* \rangle \in P^-_R(\omega, \sigma_{\mathbf{X}})$; given the definition of repaired successor state we can conclude that $\langle x^*, y^* \rangle \notin P^{\omega''}$ and, applying the definition of mapping function, that $\langle x^*, y^* \rangle \notin n(P)^{\hat{\omega}}$.

The same argumentation can be applied to prove that also other axioms obtained from the instantiation of other axiom types in the schema $\Delta KB^R(m, n)$ are satisfied by the structure $\hat{\omega}$.

W.l.o.g., we consider the axiom represented by Eq. 4.23 and, since the standard semantics, we can establish that:

$$[n(A)^+ \sqcap (A \sqcup m(A)^*)]^{\hat{\omega}} = [n(A)^+]^{\hat{\omega}} \cap (A^{\hat{\omega}} \cup [m(A)^*]^{\hat{\omega}})$$

So applying the definition of embedding function:

$$[n(A)^+ \sqcap (A \sqcup m(A)^*)]^{\hat{\omega}} = A^+_R(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}) \cap (A^{\omega} \cup A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}))$$

Since the repair is consistently defined, we can conclude that:

$$[n(A)^+ \sqcap (A \sqcup m(A)^*)]^{\hat{\omega}} = \emptyset$$

or, in other words, that:

$$\hat{\omega} \models n(A)^+ \sqcap (A \sqcup m(A)^*) \sqsubseteq \bot$$

In a similar way we can also prove other axioms of $\Delta KB^C(m, n)$. $\square$

Generalizing the above result, we obtain the following useful property.

Corollary 7. *Given an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a consistently defined service, and a simple repair $R \in R^*_S$ and a candidate repaired successor state ω'' of ω' using R, let $\hat{\omega}$ be a structure s.t. $\hat{\omega} = \mu^R(\omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then:*

$$\hat{\omega} \models KB^U_m \cup \Delta KB^R(m, n)$$

The previous theorem extends the result of Theorem 21 to keep into account the repair strategy. Also Theorem 22 can be generalized in the same way.

Theorem 31. *Given a model $\hat{\omega}$ of the knowledge base $KB^U_{m,n} \wedge \tau_n(\mathcal{W})$, then the quintuple $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi^R(\hat{\omega})$ is s.t. ω'' is a successor state of the enactment from the state ω to the state ω', using $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ as, resp., input and instantiation assignment, applying the repair $R \in R^*_S$.*

Proof. In order to show the claim we need to prove that:

1. there exists an enactment of the service S from ω to ω' using $\sigma_{\mathbf{X}}$ and $\sigma'_{\mathbf{Y}}$ as variable assignments;

2. the repair R is consistently defined w.r.t. the provided initial world state and variable assignments;

3. the state ω'' is resulting from the application of repair R;

4. the state ω'' is legal w.r.t. the world specification $\mathcal{W}$.

The first point of the proof directly follows from the application of Lemma 26 and Theorem 22.

Regarding the point 2 we need to prove that constraints imposed in the definition of consistent repair are satisfied in structures projected out from the model $\hat{\omega}$ applying the function π^R. W.l.o.g. we consider the constraint:

$$P_R^-(\omega, \sigma_{\mathbf{X}}) \cap (P^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}) \cup P^-(\omega, \sigma_{\mathbf{X}})) = \emptyset$$

By contradiction we assume that this constraint is not satisfied, i.e., that there exists a pair $\langle x^*, y^* \rangle$ s.t.:

$$\langle x^*, y^* \rangle \in P_R^-(\omega, \sigma_{\mathbf{X}}) \cap P^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

But, given the definition of the mapping, from this assumption immediately follows that:

$$\langle x^*, y^* \rangle \in [n(P)^-]^{\hat{\omega}} \cap [n(P)^+]^{\hat{\omega}}$$

violating the axiom in Eq. 4.29, that is assumed satisfied in $\hat{\omega}$, since it is a model of the whole knowledge base.

To prove the point 3, we need to show that also constraints imposed in the definition of repair successor state are satisfied in resulting out from the model $\hat{\omega}$ applying the function π^R. W.l.o.g. we consider the constraint:

$$A^{\omega''} = (A^{\omega'} \setminus A_R^-(\omega, \sigma_{\mathbf{X}})) \cup A_R^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

By contradiction we assume that this constraint is not satisfied, i.e., that there exists an element x^* s.t. $x^* \in A^{\omega''}$ despite $x^* \notin (A^{\omega'} \setminus A_R^-(\omega, \sigma_{\mathbf{X}})) \cup A_R^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$. But, given the definition of the mapping, from this assumption immediately follows that:

$$\begin{array}{rcl} x^* & \in & n(A)^{\hat{\omega}} \\ x^* & \notin & (m(A)^{\hat{\omega}} \setminus [n(A)^-]^{\hat{\omega}}) \cup [n(A)^+]^{\hat{\omega}} \end{array}$$

Applying the standard semantics, we obtain that:

$$\hat{\omega} \models x^* : n(A) \sqcap \neg((m(A) \sqcap \neg n(A)^-) \sqcup n(A)^+)$$

clashing with axiom in Eq. 4.17, that is assumed satisfied in $\hat{\omega}$.

About the active domain of states ω' and ω'', it is preserved since the axiom in Eq. 4.16 and the definition of the mapping that interprets $\Delta^{\omega'}$ and $\Delta^{\omega''}$ on $[\mathsf{Top}_m]^{\hat{\omega}} = [\mathsf{Top}_n]^{\hat{\omega}}$.

Regarding the interpretation of object names, we observe that since they are always the same, there is no form of alias, they are constantly interpreted on the same universe elements in all structures (ω, ω', ω'', and $\hat{\omega}$).

In order to complete the prove, we apply Point 2 of Lemma 22. Since ω'' is embedded into $\hat{\omega}$ according to the mapping n, using the Corollaries 6 and 5, we can show that two structures agree upon the renaming on the interpretation of set on which the evaluation of constraints in the world specification $\mathcal{W}$ relies. Since $\tau_n(\mathcal{W})$ is satisfied in $\hat{\omega}$, then $\mathcal{W}$ also holds in ω'', that turn to be a legal world state. □

The ability to cope with a, even limited, form of service effect repair allows to extend the class of usable e-services, since it relaxes the constraints about the validity of service w.r.t. the world specification. In other words, we are able to provide a new definition of accessible e-service, replacing the definition of valid e-service with a more sophisticated one, which includes also an effect execution repair.

Definition 58 (Repairable simple e-service)**.** *Let E be the effect of a simple e-service S, and let R^*_S be the set of repairs for the service S. S is repairable w.r.t. a world specification $\mathcal{W}$ iff:*

- *the effect E is consistent;*
- *for each legal world state ω, for each consistent input assignment $\sigma_{\mathbf{X}}$, s.t. the service is accessible in ω using it, there exists at least a state ω' in the enactment and a repair $R \in R^*_S$ s.t. the repaired state ω'_R is legal.*

Intuitively, we are now considering as applicable a service (even if it is not valid) if, through the considered repair, there exists a way to realize its effects in a legal world state.

Remark 19. *We remark that, in order to provide the definition of repairable service, the notion of minimality of the repair does not play any significant role, but it will turn useful in subsequent considerations.*

In order to reason about multiple possible repairs, we need to extend the definition related to the embedding of enactment into structures to the case of multiple "concurrent" execution flows. It can be accomplished using different name mapping functions: in such a way, the specification of different executions is mapped to different names, thus avoiding any possible conflict among them.

Given a quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$, a finite set of repair $\mathbf{R} = \{R_1, \ldots, R_r\}$ and a finite set $\Omega = \{\omega''_1, \ldots, \omega''_r\}$ of world states, we define a new mapping function $\mu^{\mathbf{R}}(\omega, \omega', \Omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$ that embeds all arguments into a structure $\hat{\omega}$ s.t.:

- the interpretation domain is the whole universe ($\Delta^{\hat{\omega}} = \mathfrak{U}$);
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;
- the object spy is assigned to an element of $\mathsf{New}^{\hat{\omega}}$, if any;
- the role aux is interpreted as $\{\mathsf{spy}^{\hat{\omega}}\} \times \mathsf{New}^{\hat{\omega}}$;
- for remaining names, the interpretation function is obtained by the union of interpretation functions of $\hat{\omega}_i = \mu^{R_i}(\omega, \omega', \omega''_i, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$, s.t. each one is defined using a different name mapping function $n_i \in N_m$, s.t. $\mathsf{Top}_{n_i} = \mathsf{Top}_i$ and $n_i(x) = x_i$, while the same name mapping function m is s.t. $\mathsf{Top}_m = \mathsf{Top}'$ and $m(x) = x'$.

The provided definition is well-founded since the various interpretation functions agree upon the interpretation of shared names.

Given the definition the following property holds:

Lemma 27. *Given and structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu^{\mathbf{R}}(\omega, \omega', \Omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$, a quintuple $\langle \omega, \omega', \omega''_i, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$, s.t. $\omega''_i \in \Omega$, is embedded into $\hat{\omega}$ as repaired enactment.*

Theorem 32. *A consistent and accessible simple e-service S is repairable w.r.t. a world specification $\mathcal{W}$ using a family of repair $R_S = \{R_1, \ldots, R_r\}$, iff the following implication holds:*

$$KB^U_m \wedge \tau(\mathcal{W}) \wedge \bigwedge_{i=1}^{r} \Delta KB^R(m, n_i) \models \bigvee_{i=1}^{r} \tau_{n_i}(\mathcal{W}) \wedge \Delta KB^C(m, n_i)$$

where m and n_i are the name mapping functions for the domain.

Proof. By contradiction, we assume that the service is repairable but the implication does not hold. It means that exists at least a model $\hat{\omega}$ of $KB^U_m \wedge \bigwedge_{i=1}^{r} \Delta KB^R(m, n_i)$ that does not satisfy constraints in $\tau_{n_i}(\mathcal{W}) \wedge \Delta KB^C(m, n_i)$ for any $i \in 1 \ldots r$.

Applying the projection function π^R for all repair $R_i \in R_S$ we obtain a set of quintuples of the form:

$$\langle \omega, \omega', \omega''_i, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$$

By definition they agree on 4 components out of 5, since they are built upon shared names (e.g., $\mathbf{A}$, $m(\mathbf{P})$, and so on).

Since $\hat{\omega} \models KB^U_m$, the quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$ represents a valid service enactment by Theorem 22, and, since the service is deterministic and repairable, a repair $R_{i^*} \in R_S$, s.t. the associated final state ω''_{i^*} is legal, must exist. So, applying Theorem 30, we obtain that the structure $\hat{\omega}$ is also a model of the knowledge base $\Delta KB^C(m, n_{i^*})$.

On the other hand, since the state ω''_{i^*} is legal and it is embedded into $\hat{\omega}$, by Theorem 3, we can conclude that also:

$$\hat{\omega} \models \tau_{n_{i^*}}(\mathcal{W})$$

So we have shown that $\hat{\omega}$ satisfies at least a constraint, and hence the contradiction.

Now we assume that the service is not repairable despite the implication holds. Consequently there must exist at least a world state ω and an input assignment $\sigma_{\mathbf{X}}$ s.t. the resulting state can not be "adjusted" with any available repair.

Let ω' and $\sigma'_{\mathbf{Y}}$ represent possible enactment, if the service is not repairable, for each candidate repair $R_i \in R_S$ at least one of the following condition must be satisfied:

- the repair is not consistent with the enactment;
- the repaired state is not legal.

Let $\hat{\omega}$ be the structure obtained by the application of function $\mu^{\mathbf{R}}$ to all candidate repaired states in $\Omega = \{\omega''_1, \ldots, \omega''_r\}$ obtained from ω' using repair in

R_S. Since the service is accessible and we are considering a valid enactment, according to Theorem 21, the constructed interpretation $\hat{\omega}$ is a model of KB^U_m.

Given the definition of repair insert and update set, and also of repaired extension of role and concept names, applying the Corollary 7, we can also conclude that constraints of $\Delta KB^R(m, n_i)$ are satisfied for all $i \in 1 \ldots r$.

Since the implication holds, there also must exist at least a term of the implied disjunction that is satisfied on the interpretation $\hat{\omega}$, or, in other words, that:

$$\hat{\omega} \models \tau_{n_{i^*}}(\mathcal{W}) \wedge \Delta KB^C(m, n_{i^*})$$

for some i^*. Now we can apply Theorem 31 obtaining that the state ω''_{i^*} is legal, given the world specification, and it is obtained using a repair R_{i^*} that is also consistent with the enactment. □

We point out that the size of the reasoning problem is now exponential in the size of the problem statement, since an exponential number of possible repairs must be accordingly encoded. It means that reasoning about even this limited form of update repair w.r.t. integrity constraints implies at most an exponential complexity blow-up.

Theorem 33. *Given a world specification $\mathcal{W}$ and an accessible and consistent service S, the problem of checking if S is also repairable is in* coNEEXP.

Proof. The decision problem can be reduced to an entailment checking, that, according to Corollary 1, is in coNEXP. Differently from previous cases, we assume that the problem input size is defined in terms of length of domain, world and service specifications (e.g., number of names or complexity of constraints/preconditions, etc.). So, given Theorem 26, the reduction builds an implication problem instance whose length is exponential in the input size, due the necessity of encoding of an exponential number of repair alternatives. In fact, considering the complementary problem $\overline{\mathsf{RepService}}$, since we have that:

$$\mathsf{SAT}\mathcal{C}^2 \in \mathsf{NEXP} = \bigcup_k \mathsf{NTIME}\left(2^{m^k}\right)$$

where $m = \mathcal{O}\left(2^{p(n)}\right)$, let n be the size of the instance of the problem, we have also that:

$$\overline{\mathsf{RepService}} \in \bigcup_k \mathsf{NTIME}\left(2^{2^{p(n)^k}}\right) \subseteq \bigcup_k \mathsf{NTIME}\left(2^{2^{n^k}}\right) = \mathsf{NEEXP}$$

and, hence, that $\mathsf{RepService} \in \mathsf{coNEEXP}$. □

Remark 20. *The ability to deal with service effect repairs can be also viewed as a form of allowing partially specified services: a repairable service intentionally states only its primary effects, while its indirect effects (the ones implied by the primary effects and the domain constraints) are not specified.*

4.4 Conclusions

In this chapter we have presented the basic approach to modeling simple e-service with a consistent operational semantics acting in a complex domain specification according to the open-world assumption and the minimal-change update.

These kind of e-services, described according to the IOPE paradigm, can be used to axiomatize typical deterministic OWL-S semantically-annotated web-services, even if they are only partially specified, given the devised repair approach.

We have also analyzed the complexity of property verification problems, stating that foundational correctness properties are always decidable in our framework providing also a concrete implementation approach relying upon reasoning in a decidable fragment of first-order logic and, hence, currently addressable using Automated Theorem Proving tools ([BG01, Häh01, DV01, NR01]).

Moreover, the repair strategy devised can be also extended so that we are able to keep into account other update repairing alternatives (e.g., using also access functions in the definition of atomic repair). Using these approaches it is generally more difficult to preserve the minimality of the model change and some more restrictive hypothesis must be considered, even leveraging on counting ability of the working logic $\mathcal{C}^2$.

```
Input: a world specification $\mathcal{W}$, an initial world state $\omega$, a resulting world
       state $\omega'$, an input assignment $\sigma_{\mathbf{X}}$, and a finite set of repairs $R_S$
Output: a repair $R \in R_S$
$s \leftarrow \max_{R \in R_S} \|R\|$ ;
for $k \leftarrow 0$ to $s$ do
    $R_k \leftarrow \{R \in R_S | \, \|R\| = k\}$ ;
    foreach $R \in R_k$ do
        if R is consistently defined w.r.t the transition $\omega \rightarrow \omega'$ then
            let $\omega''$ be obtained from $\omega'$ applying $R$ given $\sigma_{\mathbf{X}}$;
            if $\omega'' \models \mathcal{W}$ then
                return $R$;
            end
        end
    end
    return Exception: the enactment is not repairable ;
end
```

Figure 4.5: The repair search algorithm

CHAPTER 5

Enhanced e-services

In this chapter we extend the axiomatization model of e-services devised in Chapter 4, describing some enhanced types of e-services that allow for the specification of complex behavior.

We basically distinguish between two kinds of such e-services:

conditional e-services, which explicitly state how they select, among a set of declared behaviors, the enactment results;

non-deterministic e-services, which declare multiple possible alternative behaviors, while keeping the decision protocol hidden.

While the first kind of e-service is more suitable to describe interfaces incapsulating components of conventional EISs, which are implementing some kind of automated task (e.g., a database procedure), the latter is applicable to model (as an e-service) a complex business process that can possibly involve multiple autonomous actors (not necessarily information systems), interacting in order to select the enactment outcome: this is a typical scenario in e-government or e-business applications, where an authoritative process is published as e-service. In fact, either the decision procedure may turn to be too complex to be expressed using a decidable logic framework, or it can be left intentionally unspecified or unknown. This is a more restrictive form of black-box encapsulation, since not only implementation details are hidden, but also the process logic. Moreover, such a kind of problem has been generally addressed by process modeling frameworks, i.e., process algebras, using τ-transitions and alternative definitions of weak bisimulation.

5.1 Conditional e-services

In this section we introduce the definition of e-services that allow for conditional behavior, extending properties and results devised for the basic case.

5.1.1 Preliminary definitions

In the following, we introduce the formal definitions of meta-model elements required to specify a conditional e-service.

Generally speaking, we adapt the approach used to define service preconditions to the case of conditional effects introducing a query-based condition language. Using such a language, we express the auxiliary preconditions under which a given behavior is adopted by the service implementation[1].

For the sake of clarity, we now present the condition language syntax and semantics.

Definition 59 (Atomic condition term)**.** *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$. An atomic condition term, that is suitable for such a service, is a pair $\langle s, Q(\boldsymbol{X}) \rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the precondition (positive or negative);*
- *$Q(\boldsymbol{X})$ is a parameterized query over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

Definition 60 (Simple condition)**.** *Let S be a service having the set $\boldsymbol{X}_S$ as its input variable names. A simple condition C is any finite arbitrary set of atomic condition terms on these input variables.*

Definition 61 (Branching condition)**.** *Let S be a service having the set $\boldsymbol{X}_S$ as its input variable names. A branching condition B is any finite arbitrary set of simple conditions on these input variables.*

As shown for service preconditions, a simple condition is a conjunction of positive (resp. negative) atomic condition terms that are satisfied if the query result is not empty (resp. is empty) given an input variable assignment and a world sate. A set of simple conditions is interpreted as a disjunction of such constraints: the branching condition is satisfied if at least an element of the set is satisfied. In other words, effect branching conditions are also expressed in *disjunctive normal form.*

Definition 62 (Branching condition satisfaction)**.** *Let B be branching condition of some complex service specification, let ω be a world state and let $\sigma_{\boldsymbol{X}}$ be an input assignment. We say that the condition B is satisfied in ω w.r.t. $\sigma_{\boldsymbol{X}}$ iff there exists at least a condition $C \in B$ s.t. for each atomic condition term $\langle s, Q(\boldsymbol{X}) \rangle \in C$:*

- *if $s = +$, then the evaluation of the query Q in ω using σ is not empty;*
- *or, if $s = -$, then the valuation of the query Q in ω using σ is empty.*

Remark 21. *We point out that also instantiation effects can be conditionally specified; hence, we can define a service that instantiates a different number of new objects according to the input and the initial state. Formalizing such an ability requires some minor modifications in the semantic definition.*

[1]We can consider a conditional e-service as a sort of composition of mutually disjoint services w.r.t. their preconditions: such a service is activable when at least one of the composing simple services is.

Definition 63 (Instantiation set)**.** *Given a service S, let $\boldsymbol{Y}_S$ be the output variable set. An instantiation set is any arbitrary, possibly empty, subset of $\boldsymbol{Y}_S$.*

The instantiation set denotes the set of newly created objects resulting from a service enactment.

Definition 64 (Positive effect argument)**.** *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, and let $\boldsymbol{Y}_S$ be the corresponding output variable names and $\boldsymbol{Y} \subseteq \boldsymbol{Y}_S$ an instantiation set. A positive effect argument is any element $Y \in \boldsymbol{Y}$ or any parameterized query $Q(\boldsymbol{X})$ over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

Definition 65 (Conditional effect)**.** *Let S be a service. A conditional effect is a finite rooted labeled binary tree s.t. for each node n:*

- *if n is a leaf node then it is labeled with a pair $v(n) = \langle \boldsymbol{Y}_n, E_n \rangle$, where $\boldsymbol{Y}_n$ is an instantiation set and E_n is a simple effect specification defined using only the former as instantiation variables;*
- *otherwise, it is labeled with a branching condition $v(n) = B_n$.*

Intuitively, the labels corresponding to branching conditions B_n are evaluated starting from the root node of E and using a pre-order visit strategy. If the current node n is a leaf one, the associated effect specification is kept as the execution outcome. For a non-leaf node, if the condition specified by the node label is satisfied, then the right child node $r(n)$ is visited, otherwise the left one $l(n)$ is visited, until a leaf node is reached.

Definition 66 (Selected simple effect)**.** *Let E be a conditional effect specification, let ω be a world state and $\sigma_{\boldsymbol{X}}$ an input assignment for some service to which E belongs. The selected effect $\langle \boldsymbol{Y}_{n_m}, E_{n_m} \rangle$ is the label of a leaf node n_m of the tree E s.t. there exists a path $\langle n_1, \ldots, n_m \rangle$ from the tree root node $n_1 = root(E)$ s.t., for each non-leaf node n_i, if $B_{n_i} = v(n_i)$ holds in ω w.r.t. the given assignment $\sigma_{\boldsymbol{X}}$, then n_{i+1} is the right child of n_i, otherwise it is the left one ($n_{i+1} = l(n_i)$).*

Remark 22. *Since the employ the term* effect *both for denoting complex conditional tree structure and simple effect, for the sake of clarity, we use the term* tree effect *to denote the whole tree structure and the term* leaf effect *to denote the effect specified by a leaf node of the tree.*

The branching language has a tree-based structure that enforces that given a world state ω and an input variable assignment $\sigma_{\mathbf{X}}$ exactly one effect is selected or, in other words, conditional e-services are always unambiguous and completely defined.

In a binary tree is always possible to identity a node, in this case a leaf node, using a rooted path specified in terms of left or right descendant. In other words, given an effect $\langle \mathbf{Y}_{n_m}, E_{n_m} \rangle$ as label of the leaf n_m, the path $\langle n_1, \ldots, n_m \rangle$ can be expressed as a string p of length $m - 1$ over the alphabet $\{R, L\}$ s.t. $p_i = R$ iff n_{i+1} is the right child of n_i, otherwise it is the left one ($n_{i+1} = l(n_i)$ and $p_i = L$). Let E be a tree and n a node of the tree, the function $path(n)$ returns the rooted path in E to n and the function $path(n, i)$ returns the i-th node of the path ($path(n, 1) = root(E)$).

Given a conditional effect specification E, let $n \in \mathit{leaves}(E)$ be a leaf effect specification and let $p = \mathit{path}(n)$ be the corresponding path, we define the formula ϕ_n as:

$$\phi_n \triangleq \bigwedge_{i \in 1 \ldots \|p\|} \begin{cases} \delta(v(\mathit{path}(p,i)) & \text{if } p_i = R \\ \neg\delta(v(\mathit{path}(p,i)) & \text{otherwise} \end{cases}$$

where δ is a function defined over the condition language as the follows:

$$\delta(B) \triangleq \bigvee_{C \in B} \bigwedge_{s \in C} \gamma(s)$$

being γ the function defined in Eq. 4.1. We also refer to such a formula as *branching path* of a leaf effect.

Remark 23. *We notice that as for service preconditions we have introduced a notational variation of classical description logics knowledge base, since we allows for arbitrary boolean combination of axioms either intensional or extensional.*

Theorem 34. *Given a conditional effect specification E, a world state ω and an input assignment $\sigma_{\boldsymbol{X}}$, then there exists one and only one selected leaf effect e among the leaves of the effect tree.*

Proof. To prove the claim we need to show that:

1. given a world state and an input variable assignment at least a leaf in the tree is selected;
2. if more that a leaf are selected they are the same element.

Regarding the first point, we consider the visiting strategy of the tree. In a top-down way, starting form the root element, we always evaluate the branching condition given the interpretation structure and the assignment: such an evaluation is a model checking problem that can be always solved given the employed languages. Hence at each step we can always select the next element to analyze through the visit according to the evaluation result, the visit algorithm at most requires a number of step linear in the depth of the branching tree, halting on every input on a leaf node, that is the selected one.

Moreover, if the assume, by contradiction that, given a world state and an input assignment, there exist two o more selected elements, we observe that each one has its own evaluation path from the tree root. W.l.o.g., we assume that both n' and n'' are selected nodes on the same input. Let p' and p'' be their paths: since the tree is rooted, they must have a common prefix p that identify a node n in the tree s.t., while p' contains the right child of n, p'' contains the left one. But given the definition of visiting algorithm, we are requiring for a branching condition $B = v(n)$ s.t. it can be evaluated both true and false on the same interpretation. □

We can finally provide the complete definition of conditional e-service as follows:

Definition 67 (Conditional e-service)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a conditional e-service specification S is a quadruple formed by:*

- *a (possibly empty) finite set of input variable names* $\boldsymbol{X}_S$*;*
- *a (possibly empty) finite set of output or instantiation variable names* $\boldsymbol{Y}_S$*;*
- *a (possibly empty) finite set of invocation precondition constraint* $\mathcal{P}_S$
- *a conditional effect specification* E_S*.*

Remark 24. *A simple e-service is a conditional e-service having as conditional effect a tree containing only a leaf node.*

5.1.2 Formal properties

In this section, we present the analysis of formal properties of conditional e-services, both extending the argumentation about general features presented in the previous chapter both discussing on peculiar aspects of this kind of services.

In order to keep into account the conditional behavior of an e-service, while we can employ the same embedding functions μ and π employed in former cases, we need to accordingly enrich the specification of the update axioms.

Despite the devised language allows for covering any possible initial/input configuration, we need to keep into account invocation precondition of the e-service $\mathcal{P}$. In fact, some branches could be turn as redundant since they can never be selected because they clash with some pre-condition.

Definition 68 (Non-redundant simple effect)**.** *Given a conditional effect E of an e-service having precondition $\mathcal{P}$ and a world specification $\mathcal{W}$, a simple effect e specified as a leaf of the branching tree E is non-redundant iff there exists at least a legal world state ω and an input assignment $\sigma_{\mathbf{X}}$ s.t. the service is accessible and the given effect is selected.*

Given an e-service having preconditions $\mathcal{P}$, let $e = \langle \mathbf{Y}, E \rangle$ be a leaf conditional effect of the service attached to the node n, we define a new knowledge base KB^e adding to the knowledge base KB^P, built using $\mathbf{Y}$ as instantiation variable set, the formula ϕ_n.

Theorem 35. *Given a state ω and an assignment σ, an effect e of a service accessible from ω using σ is selected iff there exists an interpretation ω^S s.t. $\langle \omega, \sigma \rangle = \pi^{\boldsymbol{V}}(\omega^S)$ and $\omega^S \models KB^e$*

Proof. Given a selected effect e of an accessible service S in ω using σ, we use the embedding function $\mu^{\mathbf{X}}$ to build a new interpretation structure ω^S. Since Theorem 17, the structure is a model of KB^P, so, in order to complete the prove, we need also to show that the formula ϕ_n also holds in ω^S.

Supposing that this is the case, or, in other words, that there exists a branching condition $B = v(n')$, where $n' = path(n, i)$ for some $i \in 1 \dots \|path(n)\|$, s.t. $\omega^S \not\models \delta(B)$ despite the fact that $path(n)_i = R$ or $\omega^S \not\models \neg\delta(B)$ having $path(n)_i = L$. Considering the case $path(n)_i = R$ (the other is analogous), since the effect is selected and the branching condition is marked on a right descending path of such an effect, it must be evaluated at true in ω given σ, or in other words, given the definition of condition evaluation, that there exists a condition $C \in B$ s.t. for each atomic condition $c \in C$, if the condition is positive:

$$\omega \triangleleft \sigma \models \alpha : Q_c$$

where Q_c denotes the associated query, or, if the condition has the negative sign:

$$\omega \lhd \sigma \models Q_c \sqsubseteq \bot$$

So, applying Theorems 2 and 10 we can also conclude that that there exists a condition $C \in B$ s.t. for each atomic condition $c \in C$, if the condition is positive:

$$\omega^S \models \alpha : \tau(Q_c)$$

or, if the condition has the negative sign:

$$\omega^S \models \tau(Q_c) \sqsubseteq \bot$$

Given the definition of the function δ, we have that $\omega^S \models \delta(B)$, contradicting the hypothesis that the formula ϕ_n is not satisfied.

Considering the converse case, we assume that there exists a model ω^S of the knowledge base KB^e: according to Theorem 17 its projection through the function $\pi^{\mathbf{X}}$ is a state from with the service is accessible using the associated assignment. To complete the proof, we need also to show that the selected effect e is the same associated with the node n s.t. $\phi_n \in KB^e$.

The effect is completely identified by its rooted path, hence, if the selected effect e in $\langle \omega, \sigma \rangle = \pi^{\mathbf{X}}(\omega^S)$ is different from n, there must exists a node n' s.t. it is in the common prefix of both paths $n' = \mathit{path}(n, i) = \mathit{path}(n_e, i)$ and while, for example, $\mathit{path}(n)_i = L$, we have that $\mathit{path}(n_e)_i = R$.

According to the definition of the formula ϕ_n, we have that the structure ω^S is s.t.:

$$\omega^S \not\models \delta(B)$$

where $B = v(n')$ is the branching condition attached to the node. In other words, according to the standard first-order semantics, we have that for each condition $C \in B$ there exists at least a positive condition $c \in C$ s.t. $\omega^S \not\models \alpha : \tau(Q_c)$ or a negative condition $c' \in C$ s.t. $\omega^S \not\models \tau(Q_{c'}) \sqsubseteq \bot$. Applying Theorem 11 and 2 we obtain that for each condition $C \in B$ there exists at least a positive condition $c \in C$ or a negative condition $c' \in C$ s.t. resp. $Q_c^\omega(\sigma) = \emptyset$ and $Q_{c'}^\omega(\sigma) \neq \emptyset$.

According to the definition of branching condition evaluation we have that the condition B does not hold in ω given the assignment σ, so the selected effect, also according to its definition, must include the left child of the node n', or, in other words, that $\mathit{path}(n_e)_i = L$, contradicting the initial hypothesis. $\square$

From the previous result, given the definition of non-redundant branch, is easy to conclude the following claim too.

Corollary 8. *Given a conditional effect E of a service having preconditions $\mathcal{P}$ and a world specification $\mathcal{W}$, an effect $e \in \mathit{leaves}(E)$ is non-redundant iff the knowledge base $KB^e \cup \tau(\mathcal{W})$ is satisfiable.*

We can also generalize the claims obtaining the following useful result.

Corollary 9. *Given a conditional effect specification E and an arbitrary world state ω and a suitable input variable assignment $\sigma_{\mathbf{X}}$, let $\tilde{\omega}$ be a structure embedding them, then there exists exactly one leaf effect $n \in \mathit{leaves}(E)$ s.t. $\tilde{\omega} \models \phi_n$.*

Definition 69 (Well-defined conditional effect)**.** *A conditional effect E of a service is well-defined w.r.t. a world specification $\mathcal{W}$ and service preconditions, if all its branches are non-redundant.*

Theorem 36. *Given a world specification $\mathcal{W}$ and a conditional effect E of a service, the problem of checking if E is well-defined is in* NEXP*.*

Proof. The result follows from the observation that checking if a single leaf-effect is non-redundant can be solved as satisfiability problem in $\mathcal{C}^2$ in non-deterministic exponential time. Moreover, a problem instance has a number of leaf-effects that is at most linear in the size of the input, so it can be solved also in NEXP. □

The definition of successor relation can be generalized in order to keep into account also conditionally defined services.

Definition 70 (Conditional e-service successor relation)**.** *Given a pair of world states ω and ω', ω' is a (potential) successor state of ω, resulting from the execution of a conditional service S, given an input and an output variable assignments $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ consistently defined w.r.t. ω, iff:*

- *let $\langle \boldsymbol{Y}_n, E_n \rangle$ be the selected effect given the state ω and the input assignment $\sigma_{\boldsymbol{X}}$;*
- *the output assignment is defined over $\boldsymbol{Y}_n$ or, in other words, that $dom(\sigma'_{\boldsymbol{Y}}) = \boldsymbol{Y}_n$;*
- *ω' is a successor of ω w.r.t. E_n and $\boldsymbol{Y}_n$, given the variable assignments.*

In the case of conditional e-service, the definition of transition relation does not require any adjustment, except that it now relies upon the latter version of successor relation definition.

Now we need to extend to conditional case also the various properties devised for simple e-service and associated checking problems (consistency, validity, etc.).

Definition 71 (Consistent conditional effect)**.** *A conditional service effect E is consistently defined w.r.t. a world specification $\mathcal{W}$ and a service precondition $\mathcal{P}$ iff for each legal world state ω and for each consistent assignment, there is no element or element pair that belongs both to insert set and delete set of some concept or role.*

We extend the definition of update axioms schema ΔKB^E so that we can keep into account branching conditions as shown in Table 5.1, so the new knowledge base KB_c^E is defined adding to KB^P the instantiation of this schema.

Theorem 37. *Given a conditional service, let E be its effect specification, and a world specification $\mathcal{W}$, the service is consistently defined iff for each concept name $A \in \boldsymbol{A}$ we have that:*

$$KB_c^E \wedge \tau(\mathcal{W}) \models A_E^+ \sqcap A_E^- \sqsubseteq \bot$$

and for each role name $P \in \boldsymbol{P}$:

$$KB_c^E \wedge \tau(\mathcal{W}) \models \neg\exists x, y. P_E^+(x, y) \wedge P_E^-(x, y)$$

Table 5.1: The axiom schema ΔKB_c^E.

Where $\langle \mathbf{Y}_n, E_n \rangle = v(n)$ is the leaf effect specification in the tree E associated to the node n and ϕ_n is the corresponding branching path formula.

$$\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \forall x. A_E^+(x) \leftrightarrow \left(\left(\bigvee_{\langle +, A, Q(\mathbf{X})) \rangle \in E_n} \tau(Q)(x) \right) \wedge \neg A(x) \right) \tag{5.1}$$

$$\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \forall x. A_E^-(x) \leftrightarrow \left(\left(\bigvee_{\langle -, A, Q(\mathbf{X}) \rangle \in E_n} \tau(Q)(x) \right) \wedge A(x) \right) \tag{5.2}$$

$$\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \forall x, y. P_E^+(x, y) \leftrightarrow \neg P(x, y) \tag{5.3}$$

$$\wedge \left(\bigvee_{\langle +, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E_n} \tau(Q)(x) \wedge \tau(Q')(y) \right)$$

$$\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \forall x, y. P_E^-(x, y) \leftrightarrow P(x, y) \tag{5.4}$$

$$\wedge \left(\bigvee_{\langle -, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E_n} \tau(Q)(x) \wedge \tau(Q')(y) \right)$$

Proof. This claim is a generalization of Theorem 19: the proof relies on the fact, stated in Corollary 9, that in any enactment state (identified by the initial state and the input assignment) exactly one effect is selected, or, in other words, that one and only one phi_n formula is evaluated at true, while others are not satisfied.

As in the previous case we are assuming the service is accessible, otherwise the claim is easily proved, and that the effect is consistently defined, but, by contradiction, that the implication does not hold. In other words, w.l.o.g., we assume that there exists at least a model $\hat{\omega} \models KB_c^E \wedge \tau(\mathcal{W})$ s.t. there is an element $x \in \Delta^{\hat{\omega}}$ s.t. for some concept A we have that:

$$\hat{\omega} \models x : A_E^+ \sqcap A_E^-$$

Given the structure of involved axioms, the other case can be easily obtained applying the same argumentation.

Given Theorem 11, applying the projection function $\pi^{\mathbf{X}}$, we obtain a model ω and an input variable assignment σ s.t. their extended interpretation is embedded into $\hat{\omega}$. Since $\hat{\omega} \models x : A_E^+$, given the definition axiom of the concept A_E^+ we have that at least a formula ϕ_{n^*} must be evaluated at true, given Corollary 9, and that there exists at least an atomic effect $\langle +, A, Q\rangle$, among them specified in $v(n^*)$, s.t. $\hat{\omega} \models x : \tau(Q)$. Analogously we can prove that also another effect $\langle -, A, Q'\rangle$ s.t. $\hat{\omega} \models x : \tau(Q')$ must exist in $v(n^*)$. But, applying Theorem 2, we obtain also that:

$$\omega \lhd \sigma \models x : Q, x : Q'$$

in other words, that $x \in Q^{\omega}(\sigma)$ and $x \in Q'^{\omega}(\sigma)$. According to definition of insert and delete set (ignoring possibly instantiated objects), we can conclude that:

$$x \in A^+(\omega, \sigma) \cap A^-(\omega, \sigma)$$

which means that exists a pair ω, σ, consistent w.r.t. world specification and service invocation preconditions that violates the service consistency assumption.

Now we assume that the implication holds, but, by contradiction, that the service effect is not consistently defined. W.l.o.g. we assume that there exists at least a pair ω, σ s.t. they are consistent with world specification $\mathcal{W}$ and service invocation precondition $\mathcal{P}_S$ but that there exists a pair $\langle x, y\rangle$ s.t.:

$$\langle x, y\rangle \in P^+(\omega, \sigma) \cap P^-(\omega, \sigma)$$

Following the proof of Theorem 19, we built applying the mapping function $\mu^{\mathbf{X}}$ a new structure ω' that is a model for the axioms of the knowledge base KB^P enriched with the interpretation of new concept and role names, defined accordingly w.r.t. the conditional service semantics. For example:

$$\left[A_E^+\right]^{\hat{\omega}} = A_e^+(\omega, \sigma)$$

where e is the update specification of the selected effect of E in ω given the assignment σ. The obtained structure $\hat{\omega}$ is yet a model of KB^P and to complete the proof we need to show that also axioms from ΔKB_c^E hold.

For example, we consider the axiom defined by Eq. 5.1 written as the follows[2]:

$$\forall x. A_E^+(x) \leftrightarrow \left(\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \bigvee_{\langle +, A, Q(\mathbf{X}))\rangle \in E_n} \tau(Q)(x) \right) \wedge \neg A(x)$$

If the structure $\hat{\omega}$ is not a model, the formula must be not satisfied by such an interpretation. In other words, we must consider two cases:

1. there exists an element x^* s.t. $\hat{\omega} \models A_E^+(x^*)$ but:

$$\hat{\omega} \not\models \left(\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \bigvee_{\langle +, A, Q(\mathbf{X}))\rangle \in E_n} \tau(Q)(x^*) \right) \wedge \neg A(x^*)$$

2. there exists an element x^* s.t. $\hat{\omega} \not\models A_E^+(x^*)$ but:

$$\hat{\omega} \models \left(\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \bigvee_{\langle +, A, Q(\mathbf{X}))\rangle \in E_n} \tau(Q)(x^*) \right) \wedge \neg A(x^*)$$

Considering the first case, given the definition of the structure, since $x^* \in [A_E^+]^{\hat{\omega}}$, there must exists a leaf n^* of the effect tree E s.t., let $\langle \mathbf{Y}_{n^*}, E_{n^*} \rangle = v(n^*)$ be the associated label, $x^* \in A_{E_n}^+(\omega, \sigma)$ and $\hat{\omega} \models \phi_{n^*}$.

Given the definition of insert set we can also conclude that $x^* \notin A^{\hat{\omega}}$, so we have that:

$$\hat{\omega} \not\models \bigwedge_{n \in leaves(E)} \phi_n \rightarrow \bigvee_{\langle +, A, Q(\mathbf{X}))\rangle \in E_n} \tau(Q)(x^*)$$

In other words, we need that for at least a leaf n of the effect tree, the implication is not satisfied. Such an effect node n' must be s.t. $x^* \notin A_{E_{n'}}^+(\omega, \sigma)$ and $\hat{\omega} \models \phi_{n'}$, where $\langle \mathbf{Y}_{n'}, E_{n'} \rangle = v(n')$ is the associated label. But, accordingly to Corollary 9, exactly one an effect is selected at time, so $n^* = n'$, hence a contradiction. Consequently we conclude that:

$$\forall x. A_E^+(x) \rightarrow \left(\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \bigvee_{\langle +, A, Q(\mathbf{X}))\rangle \in E_n} \tau(Q)(x) \right) \wedge \neg A(x)$$

Now, we consider the converse case. Since the hypothesis, we have that:

- $x^* \notin A^{\omega}$, since $x^* \notin A^{\hat{\omega}}$;
- there exists a leaf node n^* s.t. $\hat{\omega} \models \phi_{n^*}$ and an element $\langle +, A, Q(\mathbf{X}))\rangle \in E_{n^*}$ s.t. $x^* \in [\tau(Q)]^{\hat{\omega}}$.

We can apply Theorem 35, concluding that, since $KB_c^E \cup \phi_{n^*} \supset KB^e$ for the same effect node, the effect associated with the leaf n^* is actually selected. So, applying the definition of insert set, we can also conclude that:

$$x^* \in A_e^+(\omega, \sigma)$$

[2]Please notice that formulas ϕ_n do not contains any free variable, so they can easily pulled out the scope of the quantifier.

and, according to the employed construction algorithm of the structure $\hat{\omega}$, also that:

$$x^* \in [A_E^+]^{\hat{\omega}}$$

contradicting the hypothesis that $\hat{\omega} \not\models A_E^+(x^*)$. Adopting the same approach we can conclude that also other kinds of axioms deriving from the schema ΔKB_c^E are satisfied in the interpretation $\hat{\omega}$.

Since the hypothesis, the built model is s.t., for each concept name $A \in \mathbf{A}$ we have that:

$$\hat{\omega} \models A_E^+ \sqcap A_E^- \sqsubseteq \bot$$

and for each role name $P \in \mathbf{P}$:

$$\hat{\omega} \models \neg\exists x, y.P_E^+(x, y) \wedge P_E^-(x, y)$$

But, as contradiction hypothesis, we have assumed that exists at least an element $x \in A^+(\omega, \sigma) \cap A^-(\omega, \sigma)$ for some A, but given the construction we have also that:

$$x \in [A_E^+]^{\hat{\omega}} \cap [A_E^-]^{\hat{\omega}}$$

which means that:

$$\hat{\omega} \not\models A_E^+ \sqcap A_E^- \sqsubseteq \bot$$

contradicting the initial hypothesis. □

As done for provided definitions of activable service and consistent service effect, also other properties can be adapted to conditional e-services with minor adjustment to keep into account the notion of selected effect.

Definition 72 (Conditional e-service enactment)**.** *Let ω be a world state and $\sigma_{\boldsymbol{X}}$ a consistent input variable assignment. Given a conditional e-service S, the set of possible enactments of S in ω w.r.t. $\sigma_{\boldsymbol{X}}$ contains all the pairs $\langle\omega', \sigma'_{\boldsymbol{Y}}\rangle$ s.t.:*

- *$\sigma'_{\boldsymbol{Y}}$ is a consistent instantiation assignment of the selected effect in E_S w.r.t. ω and $\sigma_{\boldsymbol{X}}$;*
- *ω and ω' are in transition relation w.r.t. the assignment and the selected service effect.*

The validity definition for conditional e-service is quite similar to one provided for simple e-services, relying on the corresponding enactment definition. However, the axiom schema employed to solve the checking problem need a more careful modification, in order to keep into account the conditional behavior as reported in Tables 5.2 and 5.3 that show the new axiom schemas.

We need to extend results obtained in previous sections. As previously done, we denote as $\tilde{KB}_c^{\mathbf{X},\mathbf{Y}}$ the knowledge base resulting from the instantiation of schema $\tilde{KB} \wedge \Delta KB^{\mathbf{X}} \wedge \Delta KB_c^I(\mathsf{Top}')$.

Theorem 38. *Given a world state ω having room for at least $\|\boldsymbol{Y}_n\|$ new elements, and a consistent variable assignment $\sigma_{\boldsymbol{X}}$, s.t. the effect n is selected, then for any consistent instantiation assignment $\sigma'_{\boldsymbol{Y}}$ restricted to $\boldsymbol{Y}_n$, s.t. $cod(\sigma'_{\boldsymbol{Y}}) \cap \Delta^{\omega} \subseteq \emptyset$ and $\boldsymbol{X} \cap \boldsymbol{Y}_n \subseteq \emptyset$, the interpretation $\mu^{\boldsymbol{X},\boldsymbol{Y}_n}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$ is a model of the knowledge $\tilde{KB}_c^{\boldsymbol{X},\boldsymbol{Y}}$.*

Table 5.2: The axiom schema $\Delta KB_c^I(\mathsf{Top}')$.

Where $\langle \mathbf{Y}_n, E_n \rangle = v(n)$ is the leaf effect specification in the tree E associated to the node n, ϕ_n is the corresponding branching path formula and $\mathbf{Y} = \bigcup_{n \in leaves(E)} \mathbf{Y}_n$.

$$\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \forall x.\mathsf{Top}'(x) \leftrightarrow \left(\mathsf{Top}(x) \vee \bigvee_{Y \in \mathbf{Y}_n} Y(x) \right) \tag{5.5}$$

$$\bigwedge_{Y \in \mathbf{Y}} Y \sqsubseteq \mathsf{Top}' \sqcap \neg \mathsf{Top} \tag{5.6}$$

$$\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \bigwedge_{Y \in \mathbf{Y}_n} \exists^{=1} x.Y(x) \wedge \bigwedge_{Y \in \mathbf{Y} \setminus \mathbf{Y}_n} \exists^{=0} x.Y(x) \tag{5.7}$$

$$\bigwedge_{Y \in \mathbf{Y}, Y' \in \mathbf{Y}, Y \neq Y'} Y \sqcap Y' \sqsubseteq \bot \tag{5.8}$$

Proof. This is an extension of Theorem 15, so the proof is based on similar argumentation, only extended to keep into account the branching formulas.

As in the previous case, we have that the instantiation assignment exists and that the structure obtained applying the mapping function is of course a model of the axioms in $\tilde{KB}$.

Now, we need to consider the axioms obtained instantiating axiom schemas $\Delta KB^{\mathbf{X}}$ and $\Delta KB^{\mathbf{X},\mathbf{Y}}(\mathsf{Top}')$. Considering the former, we observe that, since the assignment σ is consistent, each variable $X \in \mathbf{X}$ is mapped to an element of Δ^ω, but since $\mathsf{Top}^{\tilde{\omega}} = \Delta^\omega$, where $\tilde{\omega} = \mu^{\mathbf{X},\mathbf{Y}}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$, we have that:

$$\tilde{\omega} \models X \sqsubseteq \mathsf{Top}$$

Furthermore, since each variable is assigned to only one element, given the definition of the mapping function:

$$\tilde{\omega} \models \sharp(X) = 1$$

hence, we can conclude that $\tilde{\omega} \models \Delta KB^{\mathbf{X}}$. Applying the definition of mapping function and consistent variable assignment, we can also conclude that the following equation holds:

$$\mathsf{Top}'^{\tilde{\omega}} = \mathsf{Top}^{\tilde{\omega}} \cup \bigcup_{Y \in \mathbf{Y}_{n^*}} Y^{\tilde{\omega}}$$

where n^* is the node of the selected effect. According to Corollary 9 exactly only one there exists, hence:

$$\tilde{\omega} \models \phi_{n^*}$$

and for each $n \neq n^*$:

$$\tilde{\omega} \not\models \phi_n$$

So, considering the axiom in Eq. 5.1, we have that for the node n^* the implication holds since both the antecedent and the consequent are true in $\tilde{\omega}$, while for

Table 5.3: The axiom schema $\Delta KB_c^U(m)$.

Where $\langle \mathbf{Y}_n, E_n \rangle = v(n)$ is the leaf effect specification in the tree E associated to the node n and ϕ_n is the corresponding branching path formula.

$$\bigwedge_{n \in leaves(E)} \phi_n \to \forall x. m(A)^+(x) \leftrightarrow \left(\left(\bigvee_{\langle +, A, Q(\mathbf{X}) \rangle \in E_n} \tau(Q)(x) \right) \wedge \neg A(x) \right) \tag{5.9}$$

$$\bigwedge_{n \in leaves(E)} \phi_n \to \forall x. m(A)^-(x) \leftrightarrow \left(\left(\bigvee_{\langle -, A, Q(\mathbf{X}) \rangle \in E_n} \tau(Q)(x) \right) \wedge A(x) \right) \tag{5.10}$$

$$\bigwedge_{n \in leaves(E)} \phi_n \to \forall x, y. m(P)^+(x, y) \leftrightarrow \neg P(x, y) \wedge \left(\bigvee_{\langle +, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E_n} (\tau(Q)(x) \wedge \tau(Q')(y)) \right) \tag{5.11}$$

$$\bigwedge_{n \in leaves(E)} \phi_n \to \forall x, y. m(P)^-(x, y) \leftrightarrow P(x, y) \wedge \left(\bigvee_{\langle -, P, Q(\mathbf{X}), Q'(\mathbf{X}) \rangle \in E_n} (\tau(Q)(x) \wedge \tau(Q')(y)) \right) \tag{5.12}$$

$$\bigwedge_{n \in leaves(E)} \phi_n \to \forall x. m(A)^*(x) \leftrightarrow \left(m(A)^+(x) \vee \bigvee_{\langle +, A, Y \rangle \in E_n} Y(x) \right) \tag{5.13}$$

$$\begin{aligned} \bigwedge_{n \in leaves(E)} \phi_n \to \forall x, y. m(P)^*(x, y) \leftrightarrow\ & m(P)^+(x, y) \\ & \vee \bigvee_{\langle +, P, Y, Y' \rangle \in E_n} (Y(x) \wedge Y'(y)) \\ & \vee \bigvee_{\langle +, P, Y, Q(\mathbf{X}) \rangle \in E_n} (Y(x) \wedge \tau(Q)(y)) \\ & \vee \bigvee_{\langle +, P, Q(\mathbf{X}), Y \rangle \in E_n} (\tau(Q)(x) \wedge Y(y)) \end{aligned} \tag{5.14}$$

$$m(A)^* \sqsubseteq m(A) \tag{5.15}$$

$$m(A)^+ \sqcap m(A)^- \sqsubseteq \bot \tag{5.16}$$

$$m(A)^- \sqsubseteq A \tag{5.17}$$

$$m(A) \sqcap \neg m(A)^* \equiv A \sqcap \neg m(A)^- \tag{5.18}$$

$$\forall x, y. m(P)^*(x, y) \to m(P)(x, y) \tag{5.19}$$

$$\forall x, y. \bot \leftarrow m(P)^+(x, y) \wedge m(P)^-(x, y) \tag{5.20}$$

$$\forall x, y. m(P)^-(x, y) \to P(x, y) \tag{5.21}$$

$$\forall x, y. m(P) \wedge \neg m(P)^*(x, y) \leftrightarrow P(x, y) \wedge \neg m(P)^-(x, y) \tag{5.22}$$

other ones the antecedent is false. Hence, given the standard semantics of implication, the axiom is satisfied by the provided structure, since each implication in the conjunction over the tree leaves holds.

Moreover, since also the instantiation assignment is valid, each variable $Y \in \mathbf{Y}_n$ is assigned to a distinct element, so we have that:

$$\tilde{\omega} \models \sharp(Y) = 1$$

while other variable names are unassigned we have also that:

$$\tilde{\omega} \models \sharp(Y) = 0$$

for each $Y \in \mathbf{Y} \setminus \mathbf{Y}_n$. Then, applying the same argumentation of the axiom in Eq. 5.1, we can use this property to prove that axiom in Eq. 5.7 is satisfied in $\tilde{\omega}$. In a similar way, we can also prove that axiom in Eq. 5.5 holds on the given interpretation structure. Given the injectivity property of the assignment we have also that:

$$\tilde{\omega} \models Y \sqcap Y' \sqsubseteq \bot$$

for each pair $Y \in \mathbf{Y}, Y' \in \mathbf{Y}$ s.t. $Y \neq Y'$. □

Theorem 39. *Let $\tilde{\omega}$ be a model of the knowledge base $\tilde{KB}_c^{\boldsymbol{X},\boldsymbol{Y}}$, s.t. $\tilde{\omega} \models \phi_n$ for some $n \in leaves(E)$, where E is the effect tree of some conditional service, and let $\langle \omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ be a triple s.t. $\tilde{\omega} = \mu^{\boldsymbol{X},\boldsymbol{Y}}(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then:*

$$\omega \rightsquigarrow \tilde{\omega}$$

on the concept alphabet $\boldsymbol{A}$, and:

$$\omega \lhd \sigma_{\boldsymbol{X}} \rightsquigarrow \tilde{\omega}$$

on the concept alphabet $\boldsymbol{A} \cup \boldsymbol{X}$, and:

$$\omega \lhd \langle \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle \rightsquigarrow \tilde{\omega}$$

where the domain of instantiation assignment $\sigma'_{\boldsymbol{Y}}$ is restricted to $\boldsymbol{Y}_n$.

Proof. The claim is obtained applying Theorem 16, considering only variable names in the instantiation set of the selected effect, since embedding function is the same. □

Stated these preliminary results we need to complete the formalization of the update analysis in the conditional case, so, as done at page 72, we define, given a conditional service S and the corresponding tree effect specification E, a new knowledge base cKB_m^U adding to the knowledge base KB^P the axioms obtained instantiating the schemas $\Delta KB_c^I(\mathsf{Top}_m)$ and $\Delta KB_c^U(m)$ for each concept $A \in \mathbf{A}$, role name $P \in \mathbf{P}$ and instantiation variable in $Y \in \mathbf{Y}$.

While the embedding μ-function provided at page 74 is left untouched[3], the definition of the projection π-function must be accommodated, since the domain space is restricted to only interpretation structures that are model of the new axiom schemas: let π_c be such a function projecting out from an embedded structure, that is a model of the knowledge base cKB_m^U, a quadruple describing a possible enactment.

[3]The only difference in the axiom definitions w.r.t. the simple case, is the introduction of branching path formulas, that do not require any new symbol name.

Lemma 28. *Given a model $\hat{\omega}$ of cKB^U_m, let $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ be a tuple s.t. $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi_c(\hat{\omega})$, then it is embedded into $\hat{\omega}$.*

Proof. Since the structure $\hat{\omega}$ is a model of the knowledge base cKB^U_m, the cardinality axioms ensures that from the interpretation of variable related auxiliary concepts it is always possible to build consistent variable assignments as observed in the proof of Theorem 39. The embedding relation among the quadruple and the structure follows from the definition of the projection function π and the previous lemma. $\square$

Theorem 40. *Given an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a consistently defined conditional service, let $\hat{\omega}$ be a structure s.t. $\hat{\omega} = \mu(\omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then:*

$$\hat{\omega} \models cKB^U_m$$

Proof. This theorem extends the result of Theorem 21 in order to deal with new conditional axioms. Let $\hat{\omega}$ be the structure built applying the function μ to the enactment: as done in the proof of Theorem 17, of which the function is a proper extension, we can use the same argumentation we can prove that $\hat{\omega} \models KB^P$.

Applying Theorem 34 we can conclude that in the given enactment exists exactly one selected effect n^* among these specified by the service.

So, hence the enactment itself implies that there is enough room to instantiate new objects (if any), assuming $\mathsf{Top}_m = \mathsf{Top}'$, given Theorem 38, we can conclude that $\hat{\omega} \models \tilde{KB}^{\mathbf{X},\mathbf{Y}}_c$, hence that the instantiation of the axiom schema $\Delta KB^I_c(\mathsf{Top}')$ is satisfied in $\hat{\omega}$.

In order to complete the proof, we need to show that also other axioms deriving from the instantiation of schema $\Delta KB^U_c(m)$ hold.

The key fact in the proof is the fact, that given the structure of these axioms, since for a given model $\hat{\omega}$ there is exactly one valid antecedent formula ϕ_{n^*}, we need only to show that for this conjunct also the consequent is satisfied, while other conjuncts are satisfied in the provided interpretation since their antecedents are evaluated to false. In fact, considering first-order formulas without free variables or simple propositional variables, we have that:

$$\begin{aligned} \phi_n \wedge \psi_n &\models \phi_n \rightarrow \psi_n \\ \neg\phi_n &\models \phi_n \rightarrow \psi_n \end{aligned}$$

where ψ_n and ϕ_n are arbitrary sentences. Since branching path formula properties, if a structure $\hat{\omega}$ is s.t.:

$$\hat{\omega} \models \phi_{n^*} \wedge \psi_{n^*} \wedge \bigwedge_{n \in leaves(E) \wedge n^* \neq n} \neg\phi_n$$

where ϕ_n is a branching path formula, we have also that:

$$\begin{aligned} \hat{\omega} &\models \phi_{n^*} \rightarrow \psi_{n^*} \wedge \bigwedge_{n \in leaves(E) \wedge n^* \neq n} (\phi_n \rightarrow \psi_n) \\ &\models \bigwedge_{n \in leaves(E)} (\phi_n \rightarrow \psi_n) \end{aligned}$$

where ψ_n are arbitrary sentences.

As in the previous case, we observe that axioms defined by in Equations 5.9, 5.10, 5.11, and 5.12 are a syntactical variation of which used in the proof of Theorem 37, so we can adopt the same argumentation also here. They need some adjustment in order to keep into account also instantiation variables (Equations 5.13 and 5.14).

Given an element $x \in [m(A)^*]^{\hat{\omega}}$, since the construction we have that:

$$x \in A^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

that, according to the definition of the insert set, means that:

- $x \in A^+(\omega, \sigma_{\mathbf{X}})$
- or exists an effect $\langle +, A, Y\rangle \in E_{n^*}$ s.t. $x = \sigma'_{\mathbf{Y}}(Y)$ for some $Y \in \mathbf{Y}_{n^*}$.

In the first case, we can conclude that $x \in [m(A)^+]^{\hat{\omega}}$, while in the second that $x \in Y^{\hat{\omega}}$. According to standard semantics, we can state that:

$$\hat{\omega} \models x : m(A)^+ \sqcup \bigsqcup_{\langle +,A,Y\rangle \in E_{n^*}} Y$$

hence, since the arbitrary choice of the element x, we can generalize the result, obtaining that:

$$\hat{\omega} \models \forall x. m(A)^*(x) \to m(A)^+(x) \vee \bigvee_{\langle +,A,Y\rangle \in E_{n^*}} Y(x)$$

On the other hand, given an element $x \in [m(A)^+ \sqcup \bigsqcup_{\langle +,A,Y\rangle \in E_{n^*}} Y]^{\hat{\omega}}$, since the standard semantics we have that:

- $x \in [m(A)^+]^{\hat{\omega}}$
- or exists an effect $\langle +, A, Y\rangle \in E_{n^*}$ s.t. $x \in Y^{\hat{\omega}}$ for some $Y \in \mathbf{Y}_{n^*}$.

In the first case, we can conclude that $x \in A^+(\omega, \sigma_{\mathbf{X}})$, while in the second that $x = \sigma'_{\mathbf{Y}}(Y)$. Applying the definition of the insert set and the construction of the embedding function we can conclude that:

$$\hat{\omega} \models x : m(A)^*$$

hence, generalizing as the previous case, that:

$$\hat{\omega} \models \forall x. \left(m(A)^+(x) \vee \bigvee_{\langle +,A,Y\rangle \in E} Y(x) \right) \to m(A)^*(x)$$

For the proof of validity of other axioms please refer to result for the simple case, since they unaffected by the introduction of conditional behavior (i.e., the minimal-change semantics and the interpretation structure metric do not rely on these aspects.). □

Theorem 41. *Given a model $\hat{\omega}$ of the knowledge base cKB^U_m, then the quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}\rangle = \pi(\hat{\omega})$, is an enactment of a conditional service S from the state ω to the state ω', having $\sigma_{\mathbf{X}}$ and $\sigma'_{\mathbf{Y}}$ as, resp., input and instantiation assignments.*

Proof. The proof provided for Theorem 40 can be employed also to prove this claim, hence we have stated that, according to Corollary 9, there exists a node $n^* \in leaves(E)$ s.t. $\hat{\omega} \models \phi_{n^*}$ and consequently related update formulas are valid in the given structure. $\square$

Corollary 10. *Let $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ be an enactment of a conditional service S, then:*

- *for each concept name $A \in \boldsymbol{A}$*

$$A^{\omega'} = A^+_n(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cup (A^\omega \setminus A^-_n(\omega, \sigma_{\boldsymbol{X}}))$$

- *for each role name $P \in \boldsymbol{P}$*

$$P^{\omega'} = P^+_n(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) \cup (P^\omega \setminus P^-_n(\omega, \sigma_{\boldsymbol{X}}))$$

where n is the selected effect in ω given $\sigma_{\boldsymbol{X}}$, while $\cdot^+_n$ and $\cdot^-_n$ denotes resp. the insert and delete set of a name associated to the corresponding leaf effect specification.

Also the isomorphism theorem can be extended to conditional behavior service.

Theorem 42. *Let ω be a world state, $\sigma_{\boldsymbol{X}}$ a consistent input variable assignment, S a conditional service accessible in ω using $\sigma_{\boldsymbol{X}}$. If $\langle \omega'_1, \sigma'_1 \rangle$ and $\langle \omega'_2, \sigma'_2 \rangle$ are two enactments in $S(\omega, \sigma_{\boldsymbol{X}})$, then they are isomorphic.*

Proof. The proof is quite similar to Theorem 23, excepting the fact that only selected effect specification and instantiation set is kept into account, so it essentially relies on Corollary 10. But, since the initial state and the input assignment are given and fixed, the atomic effect is always exactly identified, so the conditional behavior does not play any significant role in this case. The complete proof is omitted for the sake of brevity. $\square$

Theorem 43. *A consistent and accessible conditional e-service S is valid w.r.t. a world specification $\mathcal{W}$ iff the following implication holds:*

$$cKB^U_m \models \tau_m(\mathcal{W})$$

where m is a name mapping function for the domain.

Proof. We assume that the service is valid: since it is also accessible and consistent we have at least an enactment $\langle \omega', \sigma'_{\mathbf{Y}} \rangle \in S(\omega, \sigma_{\mathbf{X}})$. Let $\tilde{\omega} = \mu(\omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$ the embedding structure: according to Theorem 40 it is a model of the knowledge based cKB^U_m. W.l.o.g., we assume that there exists an axiom $\tau_m(C) \sqsubseteq \tau_m(D) \in \tau_m(\mathcal{W})$ s.t.:

$$\tilde{\omega} \not\models \tau_m(C) \sqsubseteq \tau_m(D)$$

Now we consider two cases, given the select effect:

1. if there is no instantiation variable ($\mathbf{Y} = \emptyset$);
2. otherwise if there is at least an instantiation variable ($\|\mathbf{Y}\| > 0$).

In the first case, there is exactly one successor state ω', so applying Lemma 15 and Corollary 6 we can conclude that:

$$\omega' \not\models C \sqsubseteq D$$

In the second case, since we have assumed that the implication does not hold, applying the same observations of the previous case we can also conclude that:

$$\omega' \not\models C \sqsubseteq D$$

but it is not enough to conclude the proof, since the definition of valid e-service requires that exists at least a valid instantiation assignment satisfying the constraints, not that any assignment has this property. But, according to Theorem 42, as for simple e-service, a successor state is isomorphic to any other in the same enactment. If we assume that $\langle [\omega']^*, [\sigma'_{\mathbf{Y}}]^* \rangle \in S(\omega, \sigma_{\mathbf{X}})$ is the pair s.t.:

$$[\omega']^* \models \mathcal{W}$$

and since the interpretations $[\omega']^*$ and ω' are isomorphic they must satisfying the same description logic axioms, so:

$$\omega' \models C \sqsubseteq D$$

Obtaining the contradiction that concludes the proof.

Now we assume that the implication holds, since the service is consistent and accessible there exists at least an enactment $\langle \omega', \sigma'_{\mathbf{Y}} \rangle \in S(\omega, \sigma_{\mathbf{X}})$, but, since we are assuming by contradiction, that the service is not valid, does not exist any ω' resulting from the enactment s.t.:

$$\omega' \models \mathcal{W}$$

W.l.o.g., we assume that there exists an axiom $C \sqsubseteq D \in \mathcal{W}$ s.t.:

$$\omega' \not\models C \sqsubseteq D$$

Applying Theorem 40, we obtain a structure $\tilde{\omega}$ that is also a model of the knowledge based cKB^U_m. Since the assumption, it is also a model of $\tau_m(\mathcal{W})$:

$$\tilde{\omega} \models \tau_m(\mathcal{W})$$

that implies also that:

$$\tau_m(C)^{\tilde{\omega}} \subseteq \tau_m(D)^{\tilde{\omega}}$$

According to Lemma 15 we have that the final state ω' is embedded. w.r.t. the name mapping m into $\tilde{\omega}$, so we can apply Corollary 6, obtaining that $\tau_m(C)^{\tilde{\omega}} = C^{\omega'}$ and $\tau_m(D)^{\tilde{\omega}} = D^{\omega'}$, concluding that:

$$\omega' \models C \sqsubseteq D$$

contradicting the hypothesis that ω' is not a legal world state. □

Theorem 44. *Given a world specification $\mathcal{W}$ and an accessible and consistent conditional service S, the problem of checking if S is also valid is in* coNEXP.

Proof. As done for other cases previously analyzed, we can solve the problem applying the property proved in Theorem 43, reducing it to an implication decision in $\mathcal{C}^2$ logics, hence we use the result of Proposition 1. Like other reductions it is also linear in the size of the input (number and length of axioms, preconditions and effects specifications). $\square$

Remark 25. *The ability of model conditional behavior of e-service does not substantially impact the complexity of decision procedures w.r.t. the simple case, at least in the analysis of the complexity class membership, if we consider the number of alternative behaviors as unary encoded. Since for each one we need also to explicitly define the update/instantiation effects, this assumption is quite reasonable.*

5.2 Non-deterministic e-services

The non-deterministic behavior can now be introduced allowing the declaration of a service that specifies a set of possible, mutually exclusive, effects instead of a single one, even conditional, hiding the selection logic. This is consistent with the requirement of strong encapsulation, that prevents from explicitly knowing the service outcome selection process.

5.2.1 Preliminary definitions

In the following, we present the formal definitions of non-deterministic e-services and related concepts.

In fact, while the specification of a non-deterministic e-service is a straightforward generalization of the conditional case, the definitions related to the update semantics need to be accommodated in order to capture also the branching of system evolution paths.

Definition 73 (Non-deterministic e-service)**.** *A non-deterministic e-service specification S, given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, is a quadruple formed by:*

- *a (possibly empty) finite set of input variable names $\boldsymbol{X}_S$;*
- *a (possibly empty) finite set of output or instantiation variable names $\boldsymbol{Y}_S$;*
- *a (possibly empty) finite set of invocation precondition constraints $\mathcal{P}_S$;*
- *a finite non-empty set of conditional effect specifications $\mathcal{E}_S$.*

Example 9. *The ability to deal with non-deterministic behaviors turn extremely useful to model a large class of e-government services:* authoritative services. *In fact, several services provided by public administration are concerning with allowing or disallowing a citizen to perform an action. Given the world specification introduced in Example 2, the following e-service T allows owners of commercial activity (i.e., a shop) to ask for the authorization to start the busi-*

ness (i.e., such an activity it is regulated by some restrictive laws):

$$\boldsymbol{X} = \{x\}$$
$$\boldsymbol{Y} = \emptyset$$
$$\mathcal{P} = x \sqcap \mathsf{Shop} \sqcap \exists\mathsf{owner}.\top \texttt{ and not } (\exists\mathsf{owner}^{-}.x) \sqcap (\exists\mathsf{authorizedFor}.x)$$
$$\mathcal{E} = \left\{\left\{+\mathsf{authorizedFor}(\exists\mathsf{owner}^{-}.x, x)\right\}, \emptyset\right\}$$

Notice that the preconditions enforce the fact that the requestor has not been authorized yet.

Remark 26. *The above example points out that the simple service contract specification cannot cope with the fact that an authorization can be granted according to a complex business logic, which can possibly include some arbitrary decision steps. In fact, they cannot be modeled in any feasible way, even allowing for complex conditional execution paths. In other words, the only suitable way is to deal with the service as a black-box that can eventually grant or deny the requested authorization.*

The definition of the semantics of non-deterministic e-services can be devised considering that the executing agent (the service provider) simply non-deterministically chooses a (conditional) effect and then realizes it. In other words, the ramification degree is always finite and it is equal to the size of effect set $\mathcal{E}_S$, in fact the other non-deterministic step (the instantiation of new objects) is not relevant w.r.t. the ramification since all possible alternatives are isomorphic. Generally speaking, this is a restricted form of non-deterministic behavior, but it can be usefully employed to cope with several application scenarios.

Given the previously stated approach, in case of non-deterministic e-services the transition relation must be adjusted in order to keep into account possible state ramifications, as well as the definition of service enactment provided at page 69.

Definition 74 (Non-deterministic e-service transition relation)**.** *Let ω and ω' be a pair of world states, s.t. the latter is resulting from the enactment of an e-service S in the state defined from the former, given an input and an output variable assignments $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ consistently defined, there exists a system state transition from ω to ω' using the specified e-service effect iff there exists an effect $E \in \mathcal{E}_S$ s.t.:*

- *ω' is a (potential) successor state of ω w.r.t. the given assignments according to E;*
- *there does not exist any other potential successor state ω'' of ω, w.r.t. the same assignments and service effect, s.t. it is closer to ω than ω' according to the symmetric difference distance (that means that $d(\omega, \omega') \leq d(\omega, \omega'')$ for any $\omega'' \in \Omega_E(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$).*

Definition 75 (Non-deterministic e-service enactment)**.** *Let ω be a world state and $\sigma_{\boldsymbol{X}}$ a consistent input variable assignment. Given an e-service S, the set of possible enactments contains all the pairs $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle$ s.t. there exists a service effect $E \in \mathcal{E}_S$ s.t.:*

- *$\sigma'_{\boldsymbol{Y}}$ is a consistent instantiation assignment of the selected effect in E w.r.t. ω and $\sigma_{\boldsymbol{X}}$;*

- *ω and ω' are in transition relation w.r.t. the assignment and the service effect.*

Clearly these definitions capture the intended semantics of a service behaving non-deterministically, that at each activation selects an effect and update the system state accordingly.

5.2.2 Formal properties

Once the required additional definitions have been introduced, we now discuss about semantic properties of e-services in the non-deterministic case.

The definitions of activable service and consistent service effect provided for other e-service kinds can be also applied to non-deterministic e-services. Regarding non-deterministic e-services, we need also to introduce a stronger requirement about effect consistency. In particular, a non-deterministic e-service is well-defined iff all its conditional effects are well-defined and it is consistently defined iff all of them are consistent. Since the number of effects is linearly bounded by the size of the problem instance, the associated decision problems are also in NEXP complexity class.

On the other side, the validity property can be restated as following:

Definition 76 (Valid non-deterministic e-service)**.** *Let S be a non-deterministic e-service, it is valid w.r.t. a world specification $\mathcal{W}$ iff:*

- *each effect $E \in \mathcal{E}_S$ is consistent;*
- *for each legal world state ω, for each consistent input assignment $\sigma_{\mathbf{X}}$, s.t. the service is accessible in ω using it, there exists at least a legal state ω' in the enactment.*

While we are requiring that all service effects are always consistent, we are relaxing the constraint regarding the "correctness" of the final state: we are, in fact, allowing for potentially unsafe behaviors since, in every activation condition, we require only that al least a legal successor state exists. In other words, we are assuming that a legal e-service in the community always selects a legal final state among alternative paths obtained from the transition relation: the validity property only ensures that every time the service is correctly activated, it can fulfill its own declared contract. However, update repair can be easily extended also to conditional and non-deterministic services.

Moreover, despite the well-foundedness and consistency of each effect can be checked essentially taking into account only an element (effect) at time, reducing such problems to the corresponding ones for deterministic e-services, the analysis of validity of the whole e-service must be carried on considering all possible behaviors. In order to cope with this issue, we adopt a strategy similar to the one employed to analyze the repairability of a service: in other terms we embed the service transition into an interpretation structure s.t. each possible transition path is represented, employing suitable name mapping functions that enforce that each candidate effect is described using its own "namespace" without interfering with other ones.

Given a service S, let $\mathcal{E}_S = \{E_1, \ldots, E_l\}$ be the set off possible effects, we define l mapping functions m_i s.t. $m_i(x) = x_i$, where i ranges in the interval

$1 \dots l$, and we introduce distinct names Top_i for the active domain of each execution path. Moreover, while in the service repair the instantiation variables do not play any relevant role (the repair step does not instantiate any new object), in this case we need also consider that each execution path can instantiate a different set of objects. Consequently we need to introduce a distinct name Y_i for each instantiation variable name $Y \in \mathbf{Y}_S$ and for each non-deterministic effect in $\mathcal{E}_S$. While the axiom schema $\Delta KB_c^U(m)$, reported in Table 5.3, is left untouched, we need to adjust the axioms involved in the formalization of object instantiation as shown in Table 5.4.

Let E be a tree effect specification $E \in \mathcal{E}_S$ of the service S, we define a new knowledge base nKB_m^U adding to the knowledge base KB^P the axioms obtained instantiating the schemas $\Delta KB_n^I(\mathsf{Top}', m)$ and $\Delta KB_c^U(m)$ for each concept $A \in \mathbf{A}$, role name $P \in \mathbf{P}$ and instantiation variable in $Y \in \mathbf{Y}_S$, using the name mapping function m as previously shown.

Table 5.4: The axiom schema $\Delta KB_n^I(\mathsf{Top}', m)$.

Where $\langle \mathbf{Y}_n, E_n \rangle = v(n)$ is the leaf effect specification in the tree E associated to the node n, ϕ_n is the corresponding branching path formula and $\mathbf{Y} = \bigcup_{n \in leaves(E)} \mathbf{Y}_n$.

$$\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \forall x.\mathsf{Top}'(x) \leftrightarrow \left(\mathsf{Top}(x) \vee \bigvee_{Y \in \mathbf{Y}_n} m(Y)(x) \right) \tag{5.23}$$

$$\bigwedge_{Y \in \mathbf{Y}} m(Y) \sqsubseteq \mathsf{Top}' \sqcap \neg \mathsf{Top} \tag{5.24}$$

$$\bigwedge_{n \in leaves(E)} \phi_n \rightarrow \bigwedge_{Y \in \mathbf{Y}_n} \exists^{=1} x.m(Y)(x) \wedge \bigwedge_{Y \in \mathbf{Y} \setminus \mathbf{Y}_n} \exists^{=0} x.m(Y)(x) \tag{5.25}$$

$$\bigwedge_{Y \in \mathbf{Y}, Y' \in \mathbf{Y}, Y \neq Y'} m(Y) \sqcap m(Y') \sqsubseteq \bot \tag{5.26}$$

Given an injective renaming function m, we can easily extend claims of Theorems 38 and 39, once we have accordingly adjusted the embedding function in order to deal with it.

Also the definition of enactment embedding relation need to be accommodated so that also the instantiation variable renaming can be kept into account.

Definition 77 (Non-deterministic e-service enactment embedding relation)**.** *Given a pair of world states $\omega = \langle \Delta^\omega, \cdot^\omega \rangle$ and $\omega' = \langle \Delta^{\omega'}, \cdot^{\omega'} \rangle$, defined on an interpretation domain that is a subset of $\mathfrak{U}$, an input assignment $\sigma_{\boldsymbol{X}}$ and an instantiation assignment $\sigma'_{\mathbf{Y}}$ both consistent w.r.t. ω, let m be a function that maps each concept (resp. role or output variable) name A (resp. P or Y) into a new one $m(A)$ (resp. $m(P)$ or $m(Y)$), let Top and Top_m be new concept names and let $\hat{\omega} = \langle \mathfrak{U}, \cdot^{\hat{\omega}} \rangle$ be an interpretation over the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{X} \cup \boldsymbol{Y} \cup m(\boldsymbol{A}) \cup \{\mathsf{Top}, \mathsf{Top}_m\}, \boldsymbol{P} \cup m(\boldsymbol{P}), \boldsymbol{O} \rangle$. The quadruple $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\mathbf{Y}} \rangle$*

is embedded into the interpretation $\hat{\omega}$ iff the following conditions hold:

$$\begin{aligned}
\Delta^{\omega} &= \mathsf{Top}^{\hat{\omega}} \\
\Delta^{\omega'} &= \mathsf{Top}_m^{\hat{\omega}} \\
N^{\omega} &= N^{\hat{\omega}} \\
N^{\omega'} &= m(N)^{\hat{\omega}} \\
\sigma_{\boldsymbol{X}}(X) &= X^{\hat{\omega}} \\
\sigma'_{\boldsymbol{Y}}(Y) &= m(Y)^{\hat{\omega}} \\
\left\|X^{\hat{\omega}}\right\| &= 1 \\
\left\|m(Y)^{\hat{\omega}}\right\| &= 1 \\
o^{\omega} &= o^{\hat{\omega}}
\end{aligned}$$

for each $N \in \boldsymbol{A} \cup \boldsymbol{P}$, $o \in \boldsymbol{O}$, $X \in \boldsymbol{X}$ and $Y \in \boldsymbol{Y}$.

As in previous cases, from the definitions of world state embedding relation and generalized embedding relation, keeping into account the renaming function m, we can easily prove the following claim.

Lemma 29. *Given a quadruple $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ and an interpretation $\tilde{\omega}$, s.t. the quadruple is embedded into it according to a name mapping function m, then:*

$$\omega \rightsquigarrow \tilde{\omega}$$

on the concept alphabet $\boldsymbol{A}$, and:

$$\omega \lhd \sigma_{\boldsymbol{X}} \rightsquigarrow \tilde{\omega}$$

on the concept alphabet $\boldsymbol{A} \cup \boldsymbol{X}$, and:

$$\omega \lhd \langle \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle \rightsquigarrow_{m|_{\boldsymbol{Y}}} \tilde{\omega}$$

where the domain of instantiation assignment $\sigma'_{\boldsymbol{Y}}$ is restricted to $\boldsymbol{Y}_n$ and the name mapping function is restricted to the domain $\boldsymbol{Y}$, and:

$$\omega' \rightsquigarrow_m \tilde{\omega}$$

on the concept alphabet $\boldsymbol{A}$.

Now, we can introduce the adjusted embedding function: it is quite similar to which one defined at page 74, excepting that it applies the renaming function also to instantiation variable names.

Given a quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$, and name mapping function m consistently defined, we define an embedding function μ_n that maps such structures into another interpretation $\hat{\omega}$ s.t.:

- the interpretation domain is the whole universe ($\Delta^{\hat{\omega}} = \mathfrak{U}$);
- the interpretation of concepts, roles and objects in the starting state is preserved ($N^{\omega} = N^{\hat{\omega}}$);
- the interpretation of concepts, roles and objects in the final state is preserved ($N^{\omega'} = m(N)^{\hat{\omega}}$);

- the interpretation of Top is the active domain of ω ($\mathsf{Top}^{\hat{\omega}} = \Delta^{\omega}$);
- the interpretation of Top_m is the active domain of ω' ($\mathsf{Top}_m^{\hat{\omega}} = \Delta^{\omega'}$);
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;
- the interpretation of variable auxiliary concepts is defined according to the assignment ($\sigma_{\mathbf{X}}(X) = X^{\hat{\omega}}$ and $\sigma'_{\mathbf{Y}}(Y) = m(Y)^{\hat{\omega}}$ if Y is assigned in $\sigma'_{\mathbf{Y}}(Y)$, $\emptyset = m(Y)^{\hat{\omega}}$ otherwise);
- the update-defining concepts and roles are interpreted according to the corresponding concept (resp. role) insert or delete set ($[m(A)^{+}]^{\hat{\omega}} = A^{+}(\omega, \sigma_{\mathbf{X}})$, $[m(A)^{-}]^{\hat{\omega}} = A^{-}(\omega, \sigma_{\mathbf{X}})$, $[m(A)^{*}]^{\hat{\omega}} = A^{+}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$);
- the object spy is assigned to an element of $\mathsf{New}^{\hat{\omega}}$, if any;
- the role aux is interpreted as $\left\{\mathsf{spy}^{\hat{\omega}}\right\} \times \mathsf{New}^{\hat{\omega}}$.

This is a non deterministic function, since the choice of the spy-point element: alternatively it can be treated as a multi-function.

The function π_n computes the inverse of μ_n, projecting out from an interpretation $\hat{\omega}$ a quadruple, representing a possible enactment between the world states ω and ω' given the variable assignments, and is defined only for structures that are models of the knowledge base nKB_m^U.

We can also extend to this case previously proved claims:

Lemma 30. *Given a quadruple $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$, and structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu_n(\omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then the quadruple is embedded into $\hat{\omega}$.*

Lemma 31. *Given a model $\hat{\omega}$ of nKB_m^U, let $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ be a tuple s.t. $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi_n(\hat{\omega})$, then it is embedded into $\hat{\omega}$.*

Theorem 45. *Given an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a consistently defined non-deterministic service, then there exists an effect $E \in \mathcal{E}_S$ s.t. let m be a suitable name mapping function and let $\hat{\omega}$ be a structure s.t. $\hat{\omega} = \mu_n(\omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$:*

$$\hat{\omega} \models nKB_m^U$$

where nKB_m^U is the knowledge base describing the effect E.

Proof. Given the definition of enactment of a non-deterministic service, if there exists a transition between two states, a service effect have been selected among declared ones and enforced by the service provider. Such an effect E is, generally speaking, a conditional effect, thus, in order to prove the claim, we can employ Theorem 40.

By contradiction, we assume that the structure $\hat{\omega}$ is not a model of the knowledge base cKB_m^U. Given such a structure we employ a new transformation ν s.t. the interpretation of every name, excepting those related to instantiation variables $Y \in \mathbf{Y}$, is preserved, while each set $m(Y)$ is replaced by a set named as Y having the same extension:

$$m(Y)^{\hat{\omega}} = Y^{\tilde{\omega}}$$

obtaining a new structure $\tilde{\omega}$. It is worth noticing that is exactly the same interpretation structure that we obtain applying the basic embedding function μ to the given enactment:

$$\nu(\hat{\omega}) = \tilde{\omega} = \mu(\omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

Since it is a suitable enactment according to the service effect specification, applying Theorem 40 we can also conclude that:

$$\tilde{\omega} \models cKB^U_m$$

But this is a contradiction: in fact, since we have assumed that knowledge base nKB^U_m is not satisfied in $\hat{\omega}$, at least an axiom must not hold on such structure. If we consider the definition of both knowledge bases (nKB^U_m and cKB^U_m), we observe that there some axioms that are identical in both theories and others that are a simply a rewriting of one of the other theory applying the concept renaming (substitution) $Y/m(Y)$ for some $Y \in \mathbf{Y}$. Consequently, in we consider the transformation employed to build the structure $\tilde{\omega}$, we can conclude that also nKB^U_m must hold in $\hat{\omega}$. □

We can also generalize the result concerning the relation among interpretation structures induced by the definition of the function ν into this useful claim, that easily enables us to extend results provided for conditional services.

Corollary 11. *Given two interpretation structures $\hat{\omega}$ and $\tilde{\omega}$, a domain specification alphabet, a service alphabet and a suitable name mapping function m, involving also instantiation variables $\boldsymbol{Y}$, if they are s.t. $\nu(\hat{\omega}) = \tilde{\omega}$, then the evaluation of any first-order formula, expressed using provided alphabets, is preserved applying the renaming $Y/m(Y)$ for each instantiation variable name $Y \in \boldsymbol{Y}$.*

Proof. The provided structure are identical up to renaming of singleton related to instantiation variables, so actually indistinguishable using first-order logic formulas. □

Corollary 12. *Given a quadruple $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$, then:*

$$\nu(\mu_n(\omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})) = \mu(\omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$$

Theorem 46. *Given a model $\hat{\omega}$ of the knowledge base nKB^U_m, for some service effect specification $E \in \mathcal{E}_S$, then the quadruple $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi_n(\hat{\omega})$, is an enactment of the non-deterministic service S from the state ω to the state ω', having $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ as, resp., input and instantiation assignments, enforcing the effect E.*

Proof. The claim follows from the observation that the structure $\nu(\hat{\omega})$ is a model of the knowledge base cKB^U_m, and according to Theorem 41 a system transition resulting from the conditional effect enactment has taken place. The result follows from the definition of the enactment of non-deterministic service enactment. □

Given a world state ω, an input assignment $\sigma_{\mathbf{X}}$, and a finite set of pairs $\Lambda = \{\langle \omega'_1, (\sigma'_{\mathbf{Y}})_1 \rangle, \ldots, \langle \omega'_l, (\sigma'_{\mathbf{Y}})_l \rangle\}$, where ω_i is a world state and $(\sigma'_{\mathbf{Y}})_i$ is an instantiation assignment, suitable given ω, we define a new mapping function $\mu(\omega, \sigma_{\mathbf{X}}, \Lambda)$ that embeds all arguments into a structure $\hat{\omega}$ s.t.:

- the interpretation domain is the whole universe ($\Delta^{\hat{\omega}} = \mathfrak{U}$);
- the interpretation Top is the active domain;
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;
- the object spy is assigned to an element of $\mathsf{New}^{\hat{\omega}}$, if any;
- the role aux is interpreted as $\{\mathsf{spy}^{\hat{\omega}}\} \times \mathsf{New}^{\hat{\omega}}$;
- for remaining names, the interpretation function is obtained by the union of interpretation functions of $\mu_n(\omega, \omega'_i, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_i)$, s.t. each one is defined using a different name mapping function m_i, s.t. $\mathsf{Top}_{m_i} = \mathsf{Top}_i$ and $m_i(x) = x_i$.

The provided definition is well-founded since the various interpretation functions μ_n agree upon the interpretation of shared names.

Given the definition the following property holds:

Lemma 32. *Given and structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu(\omega, \sigma_{\boldsymbol{X}}, \Lambda)$, each quadruple $\langle \omega, \omega'_i, \sigma_{\boldsymbol{X}}, (\sigma'_{\boldsymbol{Y}})_i \rangle$, s.t. $\langle \omega'_i, (\sigma'_{\boldsymbol{Y}})_i \rangle \in \Lambda$, is embedded into $\hat{\omega}$ as enactment.*

Theorem 47. *A consistent and accessible non-deterministic e-service S is valid w.r.t. a world specification $\mathcal{W}$, iff the following implication holds:*

$$KB^P \wedge \tau(\mathcal{W}) \wedge \bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E) \right) \models \bigvee_{E \in \mathcal{E}} \tau_{m_E}(\mathcal{W}) \quad (5.27)$$

where m_E is the name mapping function for the domain and the instantiation variable names related to effect $E \in \mathcal{E}$.

Proof. We assume that such an implication holds, but the service is not valid. It means that al least exists a world state ω^* and an input assignment $\sigma^*_{\mathbf{X}}$ s.t., for any possible service enactment, the transition results into a non-valid world state.

W.l.o.g. we consider, for each possible service effect $E \in \mathcal{E}$, an output assignment $(\sigma'_{\mathbf{Y}})_E$ and the world state ω'_E resulting from the enactment of the service, realizing the considered effect, given the starting state and the variable assignments. Since the service is assumed as non-valid, we can actually select such a kind of world states. We also point out, that since, for a given pair of starting state and input variable assignment, all successor states are isomorphic (see Theorem 42), we can restrict the analysis to a single output assignment for each effect, since other ones result into successor states isomorphic to the considered one, and, so equivalent in terms of the evaluation of the first-order formulas.

Now, we employ the embedding function μ in order to build a new interpretation structure $\hat{\omega}$, having $\Lambda \ni \langle \omega'_E, (\sigma'_{\mathbf{Y}})_E \rangle$. Since properties of structure embedding we can conclude that for any E:

$$\hat{\omega} \not\models \tau_{m_E}(\mathcal{W})$$

denoting as m_E the name mapping functions, or, in other words, that:

$$\hat{\omega} \not\models \bigvee_{E \in \mathcal{E}} \tau_{m_E}(\mathcal{W})$$

On the other hand, applying Theorem 45, since enactments are correctly defined despite they are supposed lead to non-legal world states, we can conclude also that:

$$\hat{\omega} \models nKB_{m_E}$$

for each $E \in \mathcal{E}_S$, or, given the definition of the knowledge base, that:

$$\hat{\omega} \models \bigwedge_{E \in \mathcal{E}} \left(KB^P \wedge \Delta KB_n^I(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB_c^U(m_E) \right)$$

Factoring the formula w.r.t. the term KB^P, we can conclude that implication antecedent is satisfied in $\hat{\omega}$ and, since the implication is assumed to hold, it follows that:

$$\hat{\omega} \models \bigvee_{E \in \mathcal{E}} \tau_{m_E}(\mathcal{W})$$

Hence a contradiction.

Now, we suppose that the given service is valid, but that the implication does not hold. In other terms, that there exists a structure $\hat{\omega}^*$, s.t. despite implication antecedent holds, it does not satisfy the consequent.

We can apply the projection function π considering every service effect, obtaining for each one a quadruple $\langle \omega^*, \omega'_E, \sigma^*_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$ that, according to Theorem 46, is an enactment, since the implication antecedent includes and, hence, implies the knowledge base $nKB^U_{m_E}$ for each service effect.

Since the hypothesis about the service validity, we can also conclude that at least for a service effect E^*, the corresponding final world state ω'_{E^*} is legal and, since, given the Lemma 32, it is embedded into $\hat{\omega}^*$, also that:

$$\hat{\omega}^* \models \tau_{m_E}(\mathcal{W}) \models \bigvee_{E \in \mathcal{E}} \tau_{m_E}(\mathcal{W})$$

From this contradiction follows the claim. □

Theorem 48. *Given a world specification $\mathcal{W}$ and an accessible and consistent non-deterministic service S, the problem of checking if S is also valid is in* coNEXP.

Proof. Since we solve the decision problem applying the property proved in Theorem 47, reducing it to an implication decision problem in $\mathcal{C}^2$ logics. We point out that the size of reasoning problem is linearly bounded by the size of the input specification, hence applying the result of Proposition 1, we can prove the claim. □

Moreover, in order to ensure that the specification of a non-deterministic e-service is not redundant we need to check also that each non-deterministic effect can be actually selected: in fact, if an effect is inherently invalid it will never be selected in the decision step from the service provider (assuming a consistent behavior of the latter).

Definition 78 (Non-redundant conditional effect)**.** *Given a conditional effect $E \in \mathcal{E}$ of an e-service having precondition $\mathcal{P}$ and a world specification $\mathcal{W}$, it is non-redundant iff there exists at least a legal world state ω and an input assignment $\sigma_{\mathbf{X}}$ s.t. the service is accessible and the given effect can be enforced.*

Theorem 49. *Given a world specification $\mathcal{W}$ an effect E of an accessible non-deterministic e-service S, it is non-redundant iff knowledge base $\tau(\mathcal{W}) \wedge nKB^U_{m_e} \wedge \tau_{m_e}(\mathcal{W})$ is satisfiable, where $nKB^U_{m_e}$ is the knowledge base associated to the specification of the service effect.*

Proof. Given an accessible and non-redundant service effect E w.r.t. a world specification $\mathcal{W}$, there must exist at least an enactment $\rangle \omega', \sigma'_{\mathbf{Y}}\rangle \in S(\omega, \mathbf{X})$, where ω and ω' are two legal world states, and $\sigma_{\mathbf{X}}$ and $\sigma'_{\mathbf{Y}}$ are resp. a legal input ad instantiation assignments.

So according to Theorem 45, there exists a structure $\hat{\omega}$ s.t. the enactment is embedded into it and it is also a model of $nKB^U_{m_e}$. Moreover, given Lemma 29 both world states are embedded into such a structure, directly or applying the name mapping function m_E, so applying Theorems 1 and 2 and Corollaries 5 and 6 we can easily conclude that also $\tau(\mathcal{W})$ and $\tau_{m_e}(\mathcal{W})$ hold in $\hat{\omega}$.

Now we assume that such a knowledge base is satisfiable, so there exists a model $\hat{\omega}$, that according to Theorem 46 embeds an enactment of the given service effect from the world state ω to the world state ω' using the provided assignment. Since $nKB^U_{m_e} \supset KB^P \supseteq \tilde{KB}$, applying Corollary 3 we can conclude that the given service is also accessible and that the initial and final states are also legal, so the service effect can be actually enforced by the service provided and consequently it is not redundant. $\square$

In order to complete the analysis of non-deterministic e-service, we need to keep into account also repairability properties. As we show in the following, the results achieved for simple e-services can be easily extended to this more complex scenario.

Remark 27. *As done for the basic analysis of conditional and non-deterministic e-services, the most relevant adjustment in the framework is due to the fact that instantiation and update effects can substantially differ along various execution paths according to explicitly stated decision points (i.e., conditional effects) or implicit ones. For example, an object can be instantiated only if a given condition holds or only if a non-deterministic effect is selected. In fact, the repair enumeration approach devised in 4.3 relies essentially on a syntactical strategy. But as can be easily verified, such an approach is sound w.r.t. the fact that an instantiation variable name Y can be actually assigned to a domain element: in fact, whenever an instantiation variable name $Y \in \mathbf{Y}_S$ is not bound because the effect selected does not include it into its instantiation set, the variable auxiliary concept (e.g., $m(Y)$) is simply interpreted as an empty set. It is worth noticing the all previously provided definitions (and results) are still valid.*

We start adjusting Definition 57 so that we can deal also with instantiation variable name mapping function.

Definition 79 (Non-deterministic e-service repaired enactment embedding relation)**.** *Given three world states (ω, ω', and ω''), having as interpretation domain a subset of $\mathfrak{U}$ and s.t. $\Delta^{\omega'} = \Delta^{\omega''}$, an input assignment $\sigma_{\mathbf{X}}$ and an instantiation assignment $\sigma'_{\mathbf{Y}}$ both consistent w.r.t. ω, let m and n be two functions s.t. the former maps each concept (resp. role or output variable) name A (resp. P or Y) into a new one $m(A)$ (resp. $m(P)$ or $m(Y)$) and the latter maps each concept (resp. role) name A (resp. P) into a new one $n(A)$ (resp. $n(P)$), and let Top, Top_m and Top_n be new concept names and let $\hat{\omega} = \langle \mathfrak{U}, \cdot^{\hat{\omega}} \rangle$ be an interpretation* *over the alphabet* $\langle \boldsymbol{A} \cup \boldsymbol{X} \cup \boldsymbol{Y} \cup m(\boldsymbol{A}) \cup n(\boldsymbol{A}) \cup \{\mathsf{Top}, \mathsf{Top}_m, \mathsf{Top}_n\}, \boldsymbol{P} \cup m(\boldsymbol{P}) \cup n(\boldsymbol{P}), \boldsymbol{O}\rangle$*. We say that the tuple is embedded by the repair into the interpretation* $\hat{\omega}$ *iff the pair* ω*,* ω' *is embedded as non-deterministic e-service enactment into* $\hat{\omega}$ *and the following auxiliary conditions hold:*

$$\Delta^{\omega''} = \mathsf{Top}_n^{\hat{\omega}}$$
$$N^{\omega''} = n(N)^{\hat{\omega}}$$

We can extend Lemma 22 to this new case obtaining the following claim:

Lemma 33. *Given a quintuple* $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}\rangle$ *and an interpretation* $\hat{\omega}$*, s.t. the quintuple is embedded into it, then:*

1. *the quadruple* $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}\rangle$ *is embedded into* $\hat{\omega}$*;*
2. *the repaired world state* ω'' *is embedded, according to the mapping* n*, into the structure* $\hat{\omega}$*:*
$$\omega'' \rightsquigarrow_n \hat{\omega}$$

We need also to adjust axiom schemas provided for the simple case, introducing instantiation variable name mapping too. So given a service S, an effect specification E for the same service and a repair R, we define two new axiom schemas denoted as $\Delta KB_n^R(m,n)$ and $\Delta KB_n^C(m,n)$, presented in Tables 5.5 and 5.6, where $\cdot^+, \cdot^-, \cdot^*$ denotes new auxiliary concept and role names introduced for every concept A or role P in the domain specification.

As done at page 95, given the axiom schema $\Delta KB_n^U(m,n) = \Delta KB_n^R(m,n) \cup \Delta KB_n^C(m,n)$, we define a new knowledge base $nKB_{m,n}^U$ adding to the knowledge base nKB_m^U, defined as shown at page 128, the instantiation of such a schema on the domain specification alphabet.

We need also to extend the definition embedding function μ_n, provided at page 129, as previously done in order to deal also with repaired enactments: given a quintuple $\langle \omega, \omega', \omega'' \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}\rangle$ we define an embedding function μ_n^R that maps such structures into another interpretation $\hat{\omega}$ s.t.:

- the interpretation domain is the whole universe ($\Delta^{\hat{\omega}} = \mathfrak{U}$);
- the interpretation of concepts, roles and objects in the starting state is preserved ($N^{\omega} = N^{\hat{\omega}}$);
- the interpretation of concepts, roles and objects in the update resulting state is preserved ($N^{\omega'} = m(N)^{\hat{\omega}}$);
- the interpretation of concepts, roles and objects in the repair resulting state is preserved ($N^{\omega''} = n(N)^{\hat{\omega}}$);
- the interpretation of Top is the active domain of ω ($\mathsf{Top}^{\hat{\omega}} = \Delta^{\omega}$);
- the interpretation of Top_m and Top_n is the active domain of ω' ($\mathsf{Top}_m^{\hat{\omega}} = \mathsf{Top}_n^{\hat{\omega}} = \Delta^{\omega'} = \Delta^{\omega''}$)[4];
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;

[4]Please recall that by definition the interpretation domain of ω' and ω'' is the same.

Table 5.5: The axiom schema $\Delta KB_n^R(m,n)$

$$\begin{aligned}
\mathsf{Top}_m &\equiv \mathsf{Top}_n && (5.28)\\
n(A) &\equiv (m(A) \sqcap \neg n(A)^-) \sqcup n(A)^+ && (5.29)\\
n(A)^+ &\equiv \bigsqcup_{\langle +,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle +,A,Y\rangle \in R} m(Y) \sqcup \bigsqcup_{\langle +,A,O\rangle \in R} \{O\} && (5.30)\\
n(A)^- &\equiv \bigsqcup_{\langle -,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle -,A,O\rangle \in R} \{O\} && (5.31)\\
\forall x,y.n(P)(x,y) &\leftrightarrow (m(P)(x,y) \wedge \neg n(P)^-(x,y)) \vee n(P)^+(x,y) && (5.32)\\
\forall x,y.n(P)^+(x,y) &\leftrightarrow \bigvee_{\langle +,P,X,X'\rangle \in R} X(x) \wedge X'(y) && (5.33)\\
&\vee \ldots\\
\forall x,y.n(P)^-(x,y) &\leftrightarrow \bigvee_{\langle -,P,X,X'\rangle \in R} X(x) \wedge X'(y) && (5.34)\\
&\vee \bigvee_{\langle -,P,X,O\rangle \in R} X(x) \wedge O = y\\
&\vee \bigvee_{\langle -,P,O,X\rangle \in R} O = x \wedge X(y)\\
&\vee \bigvee_{\langle -,P,O,O'\rangle \in R} O = x \wedge O' = y
\end{aligned}$$

Table 5.6: The axiom schema $\Delta KB_n^C(m,n)$

$$\begin{aligned}
n(A)^+ \sqcap (A \sqcup m(A)^*) &\sqsubseteq \bot && (5.35)\\
n(A)^+ \sqcap (n(A)^- \sqcup m(A)^-) &\sqsubseteq \bot && (5.36)\\
n(A)^- \sqcap (m(A)^+ \sqcup m(A)^-) &\sqsubseteq \bot && (5.37)\\
n(A)^- &\sqsubseteq A && (5.38)\\
\forall x,y.n(P)^+(x,y) \wedge (P(x,y) \vee m(P)^*(x,y)) &\rightarrow \bot && (5.39)\\
\forall x,y.n(P)^+(x,y) \wedge (n(P)^-(x,y) \vee m(P)^-(x,y)) &\rightarrow \bot && (5.40)\\
\forall x,y.n(P)^-(x,y) \wedge (m(P)^+(x,y) \vee m(P)^-(x,y)) &\rightarrow \bot && (5.41)\\
\forall x,y.n(P)^-(x,y) &\rightarrow P(x,y) && (5.42)
\end{aligned}$$

- the interpretation of variable auxiliary concepts is defined according to the assignment ($\sigma_{\mathbf{X}}(X) = X^{\hat{\omega}}$ and $\sigma'_{\mathbf{Y}}(Y) = m(Y)^{\hat{\omega}}$ if Y is assigned in $\sigma'_{\mathbf{Y}}(Y)$, $\emptyset = m(Y)^{\hat{\omega}}$ otherwise);
- the update-defining concepts and roles are interpreted according to the corresponding concept (resp. role) insert or delete set ($[m(N)^{+}]^{\hat{\omega}} = N^{+}(\omega, \sigma_{\mathbf{X}})$, $[m(N)^{-}]^{\hat{\omega}} = N^{-}(\omega, \sigma_{\mathbf{X}})$, $[m(N)^{*}]^{\hat{\omega}} = N^{+}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$);
- the repair related concepts and roles are interpreted according to the corresponding concept (resp. role) insert or delete set ($[n(N)^{-}]^{\hat{\omega}} = N_R^{-}(\omega, \sigma_{\mathbf{X}})$, $[n(N)^{+}]^{\hat{\omega}} = N_R^{+}(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$);
- the object spy is assigned to an element of $\mathsf{New}^{\hat{\omega}}$, if any;
- the role aux is interpreted as $\left\{\mathsf{spy}^{\hat{\omega}}\right\} \times \mathsf{New}^{\hat{\omega}}$.

Also in this case we provide a projection function π_n^R computes the inverse of μ_n^R, extracting from an interpretation $\hat{\omega}$ a quintuple, representing possibly an enactment between the world states ω and ω' repaired into ω'', given the variable assignments, and is defined only for structures that are models of the knowledge base $nKB^U_{m,n}$. The devised construction is a conservative extension of the one used for the enactment embedding relation.

We can also extend results of Lemmas 23, 24 and 25, obtaining the following useful claims.

Lemma 34. *Given a quintuple $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$, and structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu_n^R(\omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then the quintuple is embedded into $\hat{\omega}$ as repaired enactment.*

Lemma 35. *Given a model $\hat{\omega}$ of $nKB^U_{m,n}$, let $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ be a tuple s.t. $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi_n^R(\hat{\omega})$, then it is embedded into $\hat{\omega}$ as repaired enactment.*

Lemma 36. *Given a quintuple $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$, and structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu_n^R(\omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then, let $\bar{\omega}$ be the structure s.t. $\bar{\omega} = \mu_n(\omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then it is equal to $\hat{\omega}$ restricted to the common alphabet.*

Lemma 37. *Given a model $\hat{\omega}$ of $nKB^U_m \wedge \Delta KB^R_n(m,n)$, let $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ be a tuple s.t. $\langle \omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi_n^R(\hat{\omega})$, then $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle = \pi_n(\hat{\omega})$.*

Theorem 50. *Given an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of a consistently defined service, and a consistent simple repair $R \in R_S$ and repaired successor state ω'' of ω' using R, let $\hat{\omega}$ be a structure s.t. $\hat{\omega} = \mu_n^R(\omega, \omega', \omega'', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, then:*

$$\hat{\omega} \models nKB^U_{m,n}$$

Proof. This result is a generalization of Theorem 30 and the proof is quite similar, once it has been adjusted to keep into account conditional effect specifications and related enhancements.

According to Lemma 36 we can apply Theorem 45 to show that the structure $\bar{\omega}$ is a model of the knowledge base nKB^U_m. Since the structure $\hat{\omega}$ is an extension of the structure $\bar{\omega}$ that simply adds new name interpretations, without altering the interpretation on which the satisfiability result relies, we can also conclude that:

$$\hat{\omega} \models nKB^U_m$$

so, as in previous proofs, we need only to show that additional axioms also hold in order to prove the claim. Given the definition of the embedding function, the interpretation of concept Top_m and Top_n is always the same so the constraints in Eq. 5.28 is satisfied.

In order to completely prove that $\hat{\omega} \models \Delta KB^R_n(m,n)$, we need to show that the remain axioms hold too. These axioms encode the extension affected by the repair as it is specified by the mean of repair concept/repair insert/delete set. For the sake of succinctness, we will show the result using the axiom of Eq. 5.30, other ones can be obtained using the same argumentation.

According to the definition of mapping function we have that:

$$[n(A)^+]^{\hat{\omega}} = A^+_R(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

On the other hand, according to the definition of repair concept insert set:

$$[n(A)^+]^{\hat{\omega}} = \bigcup_{\langle +,A,X\rangle \in R} \{\sigma_{\mathbf{X}}(X)\} \cup \bigcup_{\langle +,A,Y\rangle \in R} \{\sigma'_{\mathbf{Y}}(Y)\} \cup \bigcup_{\langle +,A,O\rangle \in R} \{O^\omega\}$$

Applying the standard DL semantics to the right side of the axioms we have also that:

$$\left[\bigsqcup_{\langle +,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle +,A,Y\rangle \in R} m(Y) \sqcup \bigsqcup_{\langle +,A,O\rangle \in R} \{O\}\right]^{\hat{\omega}} = \bigcup_{\langle +,A,X\rangle \in R} X^{\hat{\omega}} \cup \bigcup_{\langle +,A,Y\rangle \in R} m(Y)^{\hat{\omega}} \cup \bigcup_{\langle +,A,O\rangle \in R} \{O^{\hat{\omega}}\}$$

Applying the definition of embedding relation among strcutures, it follows that:

$$\left[\bigsqcup_{\langle +,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle +,A,Y\rangle \in R} m(Y) \sqcup \bigsqcup_{\langle +,A,O\rangle \in R} \{O\}\right]^{\hat{\omega}} = \bigcup_{\langle +,A,X\rangle \in R} \{\sigma_{\mathbf{X}}(X)\} \cup \bigcup_{\langle +,A,Y\rangle \in R} \{\sigma'_{\mathbf{Y}}(Y)\} \cup \bigcup_{\langle +,A,O\rangle \in R} \{O^\omega\}$$

Concluding that:

$$\hat{\omega} \models n(A)^+ \equiv \bigsqcup_{\langle +,A,X\rangle \in R} X \sqcup \bigsqcup_{\langle +,A,Y\rangle \in R} m(Y) \sqcup \bigsqcup_{\langle +,A,O\rangle \in R} \{O\}$$

The argumentation can be extended also to other axioms encoded by Equations 5.31, 5.33, and 5.34, completing the proof of the claim.

The rest of the proof ($\hat{\omega} \models \Delta KB^C_n(m,n)$) is identical to the second part of the proof of Theorem 30 and it is omitted for brevity.

□

As done for simple e-service repair, we can generalize the above result, obtaining the following claim.

Corollary 13. *Given an enactment $\langle \omega', \sigma'_{\mathbf{Y}}\rangle \in S(\omega, \sigma_{\mathbf{X}})$ of a consistently defined service, and a simple repair $R \in R_S$ and a candidate repaired successor state ω'' of ω' using R, let $\hat{\omega}$ be a structure s.t. $\hat{\omega} = \mu^R_n(\omega, \omega', \omega'', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$, then:*

$$\hat{\omega} \models nKB^U_{m,n} \cup \Delta KB^R_n(m,n)$$

Now we extend also Theorem 31:

Theorem 51. *Given a model $\hat{\omega}$ of the knowledge base $nKB^U_{m,n} \wedge \tau_n(\mathcal{W})$, then the quintuple $\langle \omega, \omega', \omega'', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle = \pi^R_n(\hat{\omega})$, is s.t. ω'' is a successor state of the enactment from the state ω to the state ω', using $\sigma_{\mathbf{X}}$ and $\sigma'_{\mathbf{Y}}$ as, resp., input and instantiation assignment, applying the repair $R \in R_S$.*

Proof. As the previous theorem, also this one can be proved using a simple extension of the proof of the corresponding result for simple e-services.

In fact, as in the latter case, we need to prove that:

1. there exists an enactment of the service S from ω to ω' using $\sigma_{\mathbf{X}}$ and $\sigma'_{\mathbf{Y}}$ as variable assignments;

2. the repair R is consistently defined w.r.t. the provided initial world state and variable assignments;

3. the state ω'' is resulting from the application of repair R;

4. the state ω'' is legal w.r.t. the world specification $\mathcal{W}$.

The first point of the proof directly follows from the application of Lemma 37 and Theorem 46.

Regarding the point 2 we need to prove that constraints imposed in the definition of consistent repair are satisfied in structures projected out from the model $\hat{\omega}$ applying the function π^R_n. W.l.o.g. we consider the constraint:

$$P^-_R(\omega, \sigma_{\mathbf{X}}) \cap (P^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}) \cup P^-(\omega, \sigma_{\mathbf{X}})) = \emptyset$$

By contradiction we assume that this constraint is not satisfied, i.e., that there exists a pair $\langle x^*, y^* \rangle$ s.t.:

$$\langle x^*, y^* \rangle \in P^-_R(\omega, \sigma_{\mathbf{X}}) \cap P^+(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

But, given the definition of the mapping, from this assumption immediately follows that:

$$\langle x^*, y^* \rangle \in [n(P)^-]^{\hat{\omega}} \cap [n(P)^+]^{\hat{\omega}}$$

violating the constraint defined in 5.41, that is assumed satisfied in $\hat{\omega}$, since it is a model of the whole knowledge base.

To prove the point 3, we need to show that also constraints imposed in the definition of repair successor state are satisfied in resulting out from the model $\hat{\omega}$ applying the function π^R_n. W.l.o.g. we consider the constraint:

$$A^{\omega''} = (A^{\omega'} \setminus A^-_R(\omega, \sigma_{\mathbf{X}})) \cup A^+_R(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$$

By contradiction we assume that this constraint is not satisfied, i.e., that there exists an element x^* s.t. $x^* \in A^{\omega''}$ despite $x^* \notin (A^{\omega'} \setminus A^-_R(\omega, \sigma_{\mathbf{X}})) \cup A^+_R(\omega, \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}})$. But, given the definition of the mapping, from this assumption immediately follows that:

$$\begin{array}{lcl} x^* & \in & n(A)^{\hat{\omega}} \\ x^* & \notin & (m(A)^{\hat{\omega}} \setminus [n(A)^-]^{\hat{\omega}}) \cup [n(A)^+]^{\hat{\omega}} \end{array}$$

Applying the standard semantics, it follows that:

$$\hat{\omega} \models x^* : n(A) \sqcap \neg((m(A) \sqcap \neg n(A)^-) \sqcup n(A)^+)$$

clashing with Eq. 5.29, that is assumed satisfied in $\hat{\omega}$.

About the active domain of states ω' and ω'', it is preserved since the Eq. 5.28 and the definition of the mapping that interprets $\Delta^{\omega'}$ and $\Delta^{\omega''}$ on $[\mathsf{Top}_m]^{\hat{\omega}} = [\mathsf{Top}_n]^{\hat{\omega}}$.

Regarding the interpretation of object names, we observe that also in this case since they are always the same, they are constantly interpreted on the same universe elements in all structures (ω, ω', ω'', and $\hat{\omega}$).

In order to complete the prove, we apply Point 1 of Lemma 33. Since ω'' is embedded into $\hat{\omega}$ according to the mapping n, using the Corollaries 6 and 5, we can show that two structures agree upon the renaming on the interpretation of set on which the evaluation of constraints in the world specification $\mathcal{W}$ relies. Since $\tau_n(\mathcal{W})$ is satisfied in $\hat{\omega}$, then $\mathcal{W}$ also holds in ω'', that turn to be a legal world state. □

Stated these foundational results we can now extend the definition of repairable e-service to the non-deterministic case.

Definition 80 (Repairable non-deterministic e-service)**.** *Let $\mathcal{E}_S$ be effects of a non-deterministic e-service S, and let R_S be the set of repairs for the service S. S is repairable w.r.t. a world specification $\mathcal{W}$ iff:*

- *effects $E \in \mathcal{E}_S$ are consistent;*
- *for each legal world state ω, for each consistent input assignment $\sigma_{\mathbf{X}}$, s.t. the service is accessible in ω using it, there exists at least a state ω' in the enactment and a repair $R \in R_S$ s.t. the repaired state ω'_R is legal.*

In order to reason about multiple possible repairs, we apply the same strategy devised for simple e-services, keeping into account multiple "concurrent" execution flows, using different name mapping functions with mutually disjoint codomains in order to avoid any possible conflict among them.

Remark 28. *We point out that in the case of non-deterministic e-service we have two sources of behavior branching:*

1. *the service non-determinism: we need to consider any possible choice of the service provider regarding the enactment outcome;*
2. *the repair selection: any possible simple repair must be considered.*

Generally speaking, we need to keep into account any possible repair for any possible service effect: while the latter is linear in the size of the input the former, according to Theorem 26, is exponential in the problem specification size.

Given a world state ω, an input assignment $\sigma_{\mathbf{X}}$, and a finite set of pairs $\Lambda = \{\langle \omega'_1, (\sigma'_{\mathbf{Y}})_1 \rangle, \ldots, \langle \omega'_l, (\sigma'_{\mathbf{Y}})_l \rangle\}$, where ω_i is a world state and $(\sigma'_{\mathbf{Y}})_i$ is an instantiation assignment, suitable given ω, a finite set of repair $\mathbf{R} = \{R_1, \ldots, R_r\}$ and a finite set $\Omega = \left\{\omega''_{1,1}, \ldots, \omega''_{1,r}, \ldots, \omega''_{l,1}, \ldots, \omega''_{l,r}, \right\}$ of world states, we define a new mapping function $\mu^{\mathbf{R}}(\omega, \sigma_{\mathbf{X}}, \Lambda, \Omega)$ that embeds all arguments into a structure $\hat{\omega}$ s.t.:

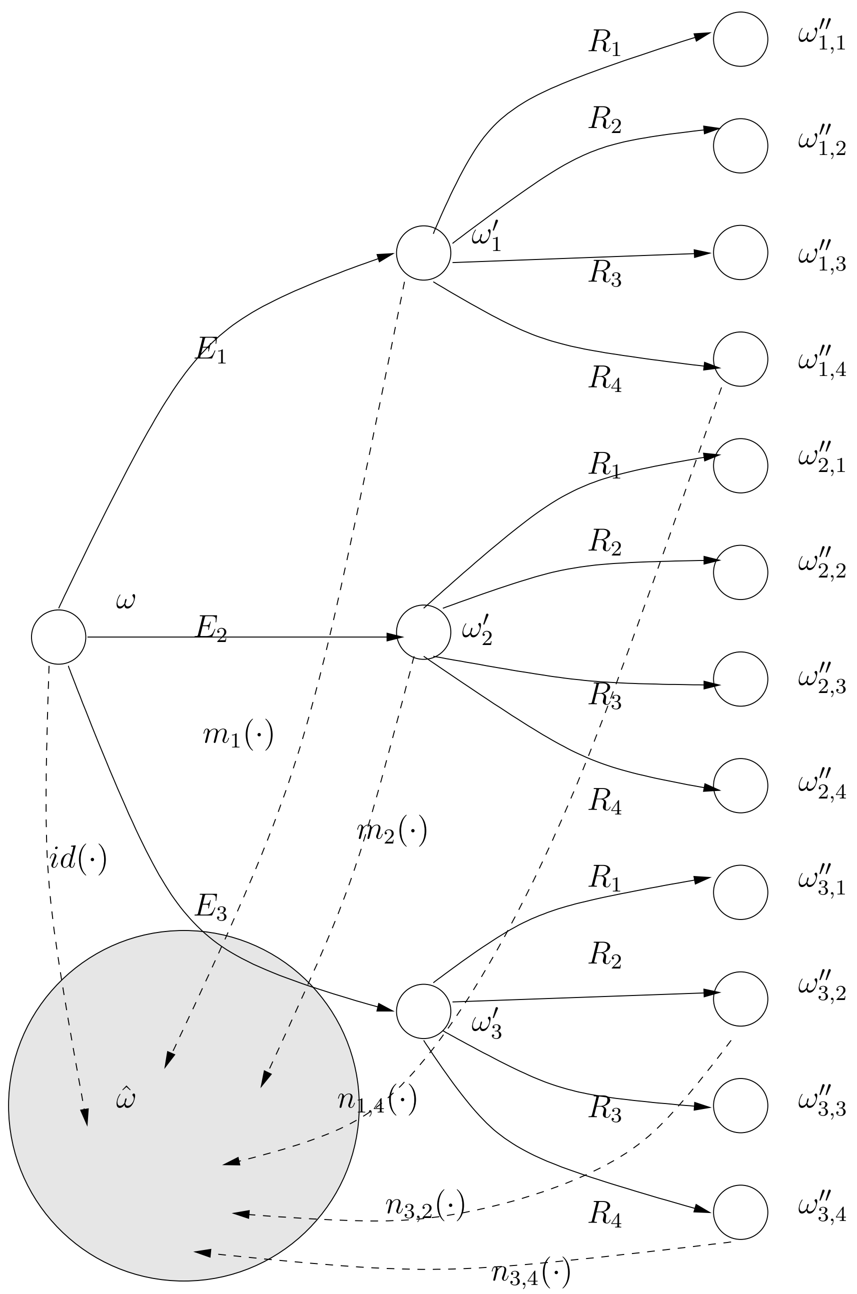

Figure 5.1: An example of possible execution paths of a non-deterministic e-service under repair.

This figure depicts an example of possible execution paths of a non-deterministic e-service having 3 effects, considering 4 different simple repairs: all the world states ω are embedded into an interpretation structure $\hat{\omega}$ encoded using suitable name mapping functions over mutually disjoint alphabets.

- the interpretation domain is the whole universe ($\Delta^{\hat{\omega}} = \mathfrak{U}$);
- the interpretation Top is the active domain;
- the interpretation of New is $\mathfrak{U} \setminus \Delta^{\omega}$;
- the object spy is assigned to an element of $\mathsf{New}^{\hat{\omega}}$, if any;
- the role aux is interpreted as $\left\{\mathsf{spy}^{\hat{\omega}}\right\} \times \mathsf{New}^{\hat{\omega}}$;
- for remaining names, the interpretation function is obtained by the union of interpretation functions of $\mu_{\mathbf{R}}(\omega, \omega'_i, \Omega_i, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_i)$, s.t.:
  - each one is defined using a different name mapping function m_i having $\mathsf{Top}_{m_i} = \mathsf{Top}_i$ and $m_i(x) = x_i$;
  - Ω_i is defined as $\left\{\omega''_{i,1}, \ldots, \omega''_{i,r}\right\} \subset \Omega$;
  - there is for each $i \in 1 \ldots l$ a different name mapping collection $N_{m_i} = \{n_{i,1}, \ldots, n_{i,l}\}$, s.t. $\mathsf{Top}_{n_{i,j}} = \mathsf{Top}_i$ and $n_{i,j}(x) = x_{i,j}$.

The provided definition is well-founded since the various interpretation functions agree upon the interpretation of shared names.

Lemma 38. *Given a structure $\hat{\omega}$ s.t. $\hat{\omega} = \mu^{\boldsymbol{R}}(\omega, \sigma_{\boldsymbol{X}}, \Lambda, \Omega)$, a quintuple:*

$$\langle \omega, \omega'_i, \omega''_{i,j}, \sigma_{\boldsymbol{X}}, (\sigma'_{\boldsymbol{Y}})_i \rangle$$

s.t. $\langle \omega'_i, (\sigma'_{\boldsymbol{Y}})_i \rangle \in \Lambda$ and $\omega''_{i,j} \in \Omega$, is embedded into $\hat{\omega}$ as repaired enactment.

Now we can finally provide a decision procedure to check whether a non-deterministic service is repairable using an approach quite similar to one devised for simple services.

Theorem 52. *A consistent and accessible non-deterministic e-service S is repairable w.r.t. a world specification $\mathcal{W}$ using a family of repair $R_S = \{R_1, \ldots, R_r\}$, iff the following implication holds:*

$$\begin{aligned} &\bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E) \wedge \bigwedge_{i=1}^{r} \left(\Delta KB^R_n(m_E, n_{E,i})\right) \right) \\ &\qquad \wedge KB^P \wedge \tau(\mathcal{W}) \models \bigvee_{E \in \mathcal{E}} \bigvee_{i=1}^{r} \left(\tau_{n_{E,i}}(\mathcal{W}) \wedge \Delta KB^C_n(m_E, n_{E,i})\right) \end{aligned} \tag{5.43}$$

where m_E and $n_{E,i}$ are the name mapping functions for the domain defined for each effect E and for each repair R_i.

Proof. By contradiction, we assume that the service is repairable but the implication does not hold. It means that exists at least a structure $\hat{\omega}$ s.t.:

$$\hat{\omega} \models KB^P \wedge \tau(\mathcal{W}) \wedge \bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E) \wedge \bigwedge_{i=1}^{r} \left(\Delta KB^R_n(m_E, n_{E,i})\right) \right)$$

but that, for any $E \in \mathcal{E}$ and for any $i \in 1 \ldots r$, it does not satisfy constraints in:

$$\bigwedge_{E \in \mathcal{E}} \bigwedge_{i=1}^{r} \left(\tau_{n_{E,i}}(\mathcal{W}) \wedge \Delta KB^C_n(m_E, n_{E,i})\right)$$

Applying the projection function π_n^R for every service effect $E \in \mathcal{E}$ and for every considered repair $R_i \in R_S$, we obtain a set of quintuples of the form:

$$\langle \omega, \omega'_E, \omega''_{E,i}, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$$

By definition they agree on 4 components out of 5, since they are built upon shared names (e.g., $\mathbf{A}$, $m(\mathbf{P})$, and so on).

Since for each effect $E \in \mathcal{E}$ the structure is s.t. $\hat{\omega} \models nKB^U_{m_E}$, any quadruple $\langle \omega, \omega'_E, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$ represents a valid service enactment according to Theorem 46. Moreover, since we have assumed that the service is also repairable, a repair $R_{E^*,i^*} \in R_S$, s.t. the associated final state ω''_{i^*} is legal, for some $E^* \in \mathcal{E}$ must exist. So, applying Theorem 50, it follows that the structure $\hat{\omega}$ is also a model of the knowledge base $\Delta KB^C_n(m_{E^*}, n_{E^*,i^*})$.

On the other hand, since the state ω''_{E^*,i^*} is legal, given the hypothesis of repairability of the service enactment, and it is embedded into $\hat{\omega}$, by Theorem 3, we can conclude that also:

$$\hat{\omega} \models \tau_{n_{E^*,i^*}}(\mathcal{W})$$

So we have found that $\hat{\omega}$ satisfies at least the constraint set for the pair E^*, i^*, hence the contradiction.

Now we assume that the service is not repairable despite the fact that the implication holds. Consequently there must exist at least a world state ω and an input assignment $\sigma_{\mathbf{X}}$ s.t. the resulting state can not be "adjusted" with any available repair, not matter which non-deterministic effect is selected.

Let ω'_E and $(\sigma'_{\mathbf{Y}})_E$ be a possible enactment, assuming that the effect $E \in \mathcal{E}$ has been non-deterministically chosen, if the service is not repairable, for each candidate repair $R_i \in R_S$ at least one of the following condition must be satisfied:

- the repair is not consistent with the enactment;
- the repaired state is not legal.

Let $\hat{\omega}$ be the structure obtained by the application of function $\mu^{\mathbf{R}}$ to all candidate repaired states in $\Omega_E = \left\{\omega''_{E,1}, \ldots, \omega''_{E,r}\right\}$ obtained from ω'_E using repair in R_S, considering any possible service effect $E \in \mathcal{E}$. Since the service is accessible and we are considering a valid enactment, according to Theorem 45, the constructed interpretation $\hat{\omega}$ is a model of $nKB^U_{m_E}$ for each $E \in \mathcal{E}$.

Now we consider the definition of repair insert and update set, and also of repaired extension of role and concept names: applying the Corollary 13, we can also conclude that constraints of $\Delta KB^R_n(m_E, n_{E,i})$ are satisfied for each $E \in \mathcal{E}$ and for each $i \in 1 \ldots r$.

We have verified that implication antecedent is satisfied in $\hat{\omega}$ and since, by the hypothesis, the implication holds, there also must exist at least a term of the implied disjunction that is satisfied on the same interpretation structure. This means that:

$$\hat{\omega} \models \tau_{n_{E^*,i^*}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,i^*})$$

for some E^*, i^*. Now we can apply Theorem 51 obtaining that the state ω'_{E^*,i^*} is legal according to the world specification, and it is obtained using a repair R_{i^*} that is also consistent with the enactment. □

Remark 29. *Is it worth noticing that reasoning about non-deterministic e-services is not more complex that reasoning about simple ones: in fact, the devised encoding is essentially linear in the number of service outcome, assumed as input size specification.*

Theorem 53. *Given a world specification $\mathcal{W}$ and an accessible and consistent non-deterministic service S, the problem of checking if S is also repairable is in* $\mathsf{coNEEXP}$.

Proof. This claim can be proved using the same argumentation of the proof of Theorem 33. In fact, the size of implication problem to solve in $\mathcal{C}^2$ is still exponential in the size of the input, since the number of service effects is linearly bounded by the length of service specification. □

5.3 Conclusions

In this chapter we have extended the specification framework, so that we can deal with more complex e-service behaviors. In particular, we have introduced the ability to encode a complex service specification that includes multiple possible behaviors according to two very general paradigms:

- the glass-box specification using the conditional effect specification primitive: despite the implementation details are always hidden to the service client[5], the provider is able to explicitly specify under which conditions an action is performed regardless the client is actually able to check this condition;
- the black-box specification using the non-deterministic effect specification primitive: in this case no details about the decision procedure is required.

Both approaches can be also mixed allowing to specify conditional effects that are non-deterministically selected by the service provider.

Interestingly, the introduction of these constructs minimally affects the computational complexity of related decision procedures: the working logic is enough powerful and expressive to cope with these aspects. Also implementation approaches devised for simple e-service (see Chapter 4) can be without difficulty extended to this scenario.

[5]It essentially stems from the abstraction degree induced by the adoption of a knowledge representation approach.

CHAPTER 6

Functional Properties

In the previous chapters we have devised a framework that, relying upon Description Logics as constraint/query language and upon a fragment of first-order logic in which reasoning is decidable, is able to model a community of e-services that act in an environment describing them in terms of their capabilities to manage the system state using a suitable update semantics. So far, we have been mainly interested in the definition of correctness properties: e.g., if a service is correctly defined w.r.t. the system specification or is actually activable from a client. We now go beyond the verification analysis and we present some complex functional properties that can be formalized using the same approach and that are interesting in the design of service-oriented solutions.

The analysis of functional properties is closely related to [GMP04, GM05, GMP06] and [d'A07]: while these ones are essentially static approaches, indeed very expressive, the present framework is oriented to the analysis of dynamic features coping explicitly with the world state update resulting from the service enactment[1]. On the other hand, the ability to reason locally according to the CWA enables these approaches to deal also with incomplete domain specification, while the present one assumes that such a specification is complete, and that the service can be partially specified assuming an update repair.

More specifically, we are interested in enriching the framework in order to support typical tasks such as:

- searching and discovering (matchmaking) an e-service given a parameterized user goal, that essentially models the client's side commitment and requirement expressed as properties of the world state;
- searching an e-service replacement in order to replace a faulty service provider considering, e.g., the state accessibility relation;
- comparing e-service capabilities abstracting w.r.t. user coverage, which we have pointed out as a foundational property of service-oriented appli-

[1]As pointed out in Section 2.2.13, DL applications predicating about enactments are considered as static, while other ones, based on the notion of knowledge base update/repair, are assumed as dynamic.

cations. In other words, this task aims to compare e-services considering only the intensional part of their specification ignoring the extensional one (that is indeed relevant in the replaceability analysis);

- the specification of e-service template to group different service instances that differ only w.r.t. extensional specification elements, in order to check common properties, inherited by the instances, directly on the template.

In the rest, unless differently stated, we always consider non-deterministic e-services.

6.1 Service Matchmaking

The analysis of available e-services can be performed in an absolute manner or considering the goal that the service client aims to achieve, as generally assumed in planning applications. Such a kind of analysis is the foundation for any discovery lookup method implemented in order to select a suitable, or preferably the most suitable, available service to achieve a user's goal (i.e., matchmaking).

6.1.1 Execution goals

Generally speaking, a goal is specified as a condition over the domain specification language stating the world desired properties as resulting from the execution of a service. Possibly a goal can be enriched with a condition describing the properties that hold in the world before the enactment, as a client commitment.

Definition 81 (Execution goal). *An execution goal G is defined as triple formed by:*

- *a (possibly empty) finite set $\boldsymbol{U}_G$ of goal instantiation parameter names;*
- *a (possibly empty) finite set of simple conditions, expressed using the parameter names as variables, stating parameter properties and client commitment $\mathcal{H}_G$;*
- *a non-empty finite set of simple conditions, expressed using the parameter names as variables, stating the properties, in terms of the state of the world, that the client wants to achieve, as requirement conditions $\mathcal{R}_G$.*

The syntax and semantics of condition expressions is the same employed for the definition of invocation preconditions and execution branching, so we omit them for the sake of brevity.

Definition 82 (Admissible goal specification). *An admissible execution goal specification must be:*

non-trivial, *which means that there exists at least a legal world state where the client requirement conditions do not hold for some parameter instantiation;*

achievable, *which means that there exists at least a legal world state where the client requirement conditions hold for some parameter instantiation;*

consistent, *which means that there exists at least a legal world state where the client commitment conditions hold for some parameter instantiation.*

Remark 30. *The language employed in the specification of goal constraints is the same employed in the specification of service conditions (e.g., preconditions or branching conditions), so we can easily extend results established so far to this scenario.*

Given a constraint expressed as $\mathbf{C} = \{C_1, \ldots, C_n\}$, where C_i is a simple condition defined over the environment $\mathbf{U}$, we define a knowledge base $\tilde{KB}^{\mathbf{C}}$ adding to axioms of $\tilde{KB}^{\mathbf{V}}$, instantiated using $\mathbf{V} = \mathbf{U}$, adding the following boolean axioms:

$$\bigvee_{C \in \mathbf{C}} \bigwedge_{c \in C} \gamma(c)$$

where γ is the function defined in Eq. 4.1.

Lemma 39. *Given a state ω and an assignment σ, a condition $\boldsymbol{C}$ is satisfied ω using σ iff there exists an interpretation ω^C s.t. $\langle \omega, \sigma \rangle = \pi^{\mathbf{V}}(\omega^C)$ and $\omega^C \models \tilde{KB}^{\boldsymbol{C}}$.*

Proof. This is a special case of Theorem 17, considering a simple service S_G s.t.:

- the input variables $\mathbf{X}$ are the environment of the condition;
- the output variable set is empty ($n = 0$);
- the condition is taken as invocation precondition constraint $\mathcal{P}$;
- the service effect is the identity transformation.

□

The previous result can be easily and slightly generalized into the following useful claims, stating that the proof argumentation essentially relies on embedding relation properties, enforced by mapping/projection:

Corollary 14. *Given a state ω and an assignment σ, a condition $\boldsymbol{C}$ is satisfied ω using σ iff there exists an interpretation ω^C s.t. $\langle \omega, \sigma \rangle$ is embedded into ω^C and $\omega^C \models \tilde{KB}^{\boldsymbol{C}}$.*

Corollary 15. *Given a state ω, an assignment σ, and a name mapping function m, a condition $\boldsymbol{C}$ is satisfied ω using σ iff there exists an interpretation ω^C s.t. $\langle \omega, \sigma \rangle$ is embedded into ω^C according to m and $\omega^C \models \tilde{KB}^{\boldsymbol{C}}(m)$.*

In the latter case, we compose the axiom schema with the name mapping function, or more specifically we adopt the following version of the function γ defined in Eq. 4.1:

$$\gamma_m(\langle s, Q(\mathbf{X}) \rangle) \triangleq \begin{cases} \alpha_p : \tau_m(Q) & \text{if } s = + \\ \tau_m(Q) \sqsubseteq \bot & \text{if } s = - \end{cases} \tag{6.1}$$

being α_p a new fresh constant name not appearing elsewhere.

For the sake of brevity, given a goal G we denote as KB^H and KB^R resp. the axiom schema $\tilde{KB}^{\mathbf{C}}$ instantiated over conditions in $\mathcal{H}_G$ and $\mathcal{R}_G$, using the environment $\mathbf{U}_G$.

Theorem 54. *Given a goal G and a world specification $\mathcal{W}$, the goal is consistent iff the knowledge base $KB^H \wedge \tau(\mathcal{W})$ is satisfiable.*

Proof. Assuming that the goal is consistent, than there must exist a legal world state ω and a parameter assignment σ s.t. both the world specification constraints and client commitment constraints hold in ω given σ. So, applying Lemma 39, we can build a structure ω^H, s.t. $\omega^H \models KB^H$, but since Theorem 3, using the function $\mu^{\mathbf{V}}$ we can embed the world model into a structure that is also a model for the $\tau(\mathcal{W})$.

Now we assume that the knowledge base is satisfiable: let ω^H be a model of such knowledge base. According to Lemma 39, the pair $\langle \omega, \sigma \rangle = \pi^{\mathbf{V}}(\omega^H)$ is s.t. the condition hold in ω using σ. Moreover, given Theorem 11, the structure ω is also embedded into ω^H, and since the latter is a model of $\tau(\mathcal{W})$, the former is a legal world state. $\square$

Theorem 55. *Given a goal G and a world specification $\mathcal{W}$, the goal is achievable iff the knowledge base $KB^R \wedge \tau(\mathcal{W})$ is satisfiable.*

Proof. The proof is analogous to Theorem 54. $\square$

Theorem 56. *Given a goal G and a world specification $\mathcal{W}$, the goal is non-trivial iff the following implication does not hold:*

$$\tau(\mathcal{W}) \wedge \tilde{KB}^{\mathbf{V}} \models KB^R$$

Proof. We assume that the goal is non-trivial, so there exists at least a pair $\langle \omega, \sigma \rangle$, s.t. ω is legal world state according to $\mathcal{W}$ and σ is consistent parameter instantiation and client requirement condition $\mathcal{R}$ are not satisfied in ω.

According to Theorem 10, we can employ the mapping function $\mu^{\mathbf{V}}$ to build an interpretation structure ω^R s.t. the given world state and assignment are embedded into it. Moreover the obtained structure is s.t. $\omega^R \models \tau(\mathcal{W}) \wedge \tilde{KB}^{\mathbf{V}}$, so, if the implication is assumed to be satisfied, we can also conclude that $\omega^R \models KB^R$, and according to Lemma 39, that requirement condition $\mathcal{R}$ also holds in ω, resulting in a contradiction.

Now we assume that the implication does not hold, but that the goal is trivial. Given the hypothesis, there must exist at least an interpretation structure ω^R s.t. the implication antecedents are satisfied, while the consequent does not hold. Applying the projection function $\pi^{\mathbf{V}}$ to the given structure, we obtain a pair $\langle \omega, \sigma \rangle$ s.t., according to Theorem 11 and related claims, ω is a legal world state given $\mathcal{W}$ and σ is a consistent assignment. Since we have assumed that the goal is trivial, we can also conclude that also requirement constraints are naively satisfied in ω given σ. Hence, according to Lemma 39, we conclude that $\omega^R \models KB^R$ too, obtaining a contradiction. $\square$

The previous claim can be restated as the following:

Corollary 16. *Given a goal G and a world specification $\mathcal{W}$, the goal is non-trivial iff the following formula is satisfiable:*

$$\tau(\mathcal{W}) \wedge \tilde{KB}^{\mathbf{V}} \wedge \neg \left(\bigvee_{C \in \mathcal{R}_G} \bigwedge_{c \in C} \gamma(c) \right)$$

Remark 31. *Is worth noticing that reasoning about goals is quite similar, given the present framework, to reasoning about services, also in terms of computational complexity.*

Theorem 57. *Given a world specification $\mathcal{W}$ and a goal G, the problem of checking if G is consistent or achievable or non-trivial is in* NEXP.

Proof. According to Theorems 54, 55 and Corollary 16 we can reduce the check of goal properties to the satisfiability of a $\mathcal{C}^2$ sentence having a length linear in the size of the input specification (world and goal). The result follows from the observation, already used to show previous complexity results, that satisfiability check problem for this language is solvable non-deterministic exponential time. □

Using the previous result, we can, without difficulty, obtain the following property.

Corollary 17. *Given a world specification $\mathcal{W}$ and a goal G, the problem of checking if G is admissible is in* NEXP.

6.1.2 Binding schemas

A parameter-less goal is also called *ground goal.* Generally speaking, in order to call a service, we need to accordingly assign its input parameters, so given a user goal (ground or not) we need to express a suitable way to bind service inputs. W.l.o.g. we can assume that a binding function is provided, which means that the adequacy of the service to the goal is analyzed w.r.t. this kind of binding.

This is of course a restriction, but removing it we can easily drop into a hardly non-decidable case since we need to reason over possible functions, which requires high order quantification primitives that are feasible only in restricted context (i.e., *constraints satisfaction programming*). On the other hand, if we devise a tool to reason about a given binding function, we can always extend it in order to search in a possible finite function space, as done, e.g., by *machine learning* algorithms [Mit97] that aim to select the function that fits observed data in the more correct way, statistically interpreted, possibly introducing a so-called *language bias.* In our framework, we restrict our attention to binding functions that are expressed by means of access function queries introduced at page 44. In particular, in the rest, we remove this limitation introducing a finite and decidable binding schema search strategy: while the decidability is preserved the computational complexity is dramatically affected.

Definition 83 (Binding schema)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a world specification $\mathcal{W}$, and two sets of variable names $\boldsymbol{X}$ and $\boldsymbol{Y}$, a binding schema B is a function that assigns to each $Y \in \boldsymbol{Y}$ an access function, w.r.t. $\mathcal{W}$, parameterized over $\boldsymbol{X}$.*

Definition 84 (Binding schema evaluation)**.** *Given a world state ω, an assignment $\sigma_{\boldsymbol{X}}$ and a binding schema B for $\boldsymbol{X}$ and $\boldsymbol{Y}$, the evaluation of B is an assignment $\sigma_{\boldsymbol{Y}}$ s.t. each name $Y \in \boldsymbol{Y}$ is assigned to the evaluation of the corresponding query:*

$$\sigma_{\boldsymbol{Y}}(Y) = B(Y)^{\omega}(\sigma_{\boldsymbol{X}})$$

The definition of access function is too weak since it allows for partial functions, so we need to introduce an additional constraint.

Definition 85 (Valid binding schema)**.** *Given a world specification $\mathcal{W}$, a binding schema B is valid iff, for any legal world state and for any input assignment, the evaluation of B assigns exactly one value to each output variable.*

Theorem 58. *Given a world specification $\mathcal{W}$, a binding schema B over $\boldsymbol{X}$ and $\boldsymbol{Y}$ is valid iff the following implication holds:*

$$\tilde{KB}^{\boldsymbol{V}} \wedge \tau(\mathcal{W}) \models \bigwedge_{Y \in \boldsymbol{Y}} \sharp(\tau(B(Y)(\boldsymbol{X}))) = 1$$

where the axiom schema $\tilde{KB}^{\boldsymbol{V}}$ is instantiated over $\boldsymbol{X}$.

Proof. Assuming that the implication holds for each assigned variable $\mathbf{Y}$, but by contradiction that there exists an a variable $Y \in \mathbf{Y}$ s.t. the query $B(Y)(\mathbf{X})$ evaluates to $B(Y)^{\omega}(\sigma_{\mathbf{X}})$ and $\|B(Y)^{\omega}(\sigma_{\mathbf{X}})\| \neq 1$ for some ω and $\sigma_{\mathbf{X}}$. So let ω be a legal world state and $\sigma_{\mathbf{X}}$ be a consistent assignment on it s.t., the evaluation of the query is empty or contains more than one element.

Since Theorem 3, using the function $\mu^{\mathbf{V}}$ we can embed the world model into a structure that is surely a model for the $\tau(\mathcal{W}) \wedge \tilde{KB}$ axioms except the newly introduced ones ($\tilde{KB}^{\mathbf{V}} \setminus \tilde{KB}$). In order to prove the claim, we need to show that also other axioms hold. Since the assignment $\sigma_{\mathbf{X}}$ is consistent, each variable X is mapped to an element of Δ^{ω}, but since $\mathsf{Top}^{\tilde{\omega}} = \Delta^{\omega}$, where $\tilde{\omega} = \mu^{\mathbf{V}}(\omega, \sigma_{\mathbf{X}})$, we have that:

$$\mu^{\mathbf{V}}(\omega, \sigma_{\mathbf{X}}) \models V \sqsubseteq \mathsf{Top}$$

Furthermore , since each variable is assigned to only one element, given the definition of the mapping function:

$$\mu^{\mathbf{V}}(\omega, \sigma_{\mathbf{X}}) \models \sharp(V) = 1$$

We have proved that $\mu^{\mathbf{V}}(\omega, \sigma_{\mathbf{X}})$ is a model of the knowledge base. By hypothesis the implication holds, so:

$$\mu^{\mathbf{V}}(\omega, \sigma_{\mathbf{X}}) \models \sharp(Q(\mathbf{V})) = 1$$

But according to Theorem 10 we have that $\omega \triangleleft \sigma_{\mathbf{X}} \rightsquigarrow \tilde{\omega}$ and applying Theorem 2 we can conclude that:

$$B(Y)(\mathbf{X})^{\omega \triangleleft \sigma_{\mathbf{X}}} = \tau(B(Y)(\mathbf{X}))^{\tilde{\omega}}$$

or, in other words, that:

$$\omega \triangleleft \sigma_{\mathbf{X}} \models \sharp(B(Y)(\mathbf{X})) = 1$$

Since the definition of query evaluation we have that:

$$\|B(Y)^{\omega}(\sigma_{\mathbf{X}})\| = 1$$

contradicting the hypothesis that binding schema is not valid.

Assuming that B is a valid binding schema, but by contradiction, that the implication does not hold for some $Y \in \mathbf{Y}$, let ω' be a model of the knowledge base s.t.:

$$\omega' \not\models \sharp(\tau(B(Y)(\mathbf{X}))) = 1$$

or, in other words, that:

$$\left\|\tau(B(Y)(\mathbf{X}))^{\omega'}\right\| \neq 1$$

Applying the $\pi^{\mathbf{V}}$ project function, we obtain a pair $\langle \omega, \sigma_{\mathbf{X}} \rangle$ s.t.

- σ is consistently defined, since $\omega' \models X \sqsubseteq \mathsf{Top}, \sharp(X) = 1$ for each $X \in \mathbf{X}$;
- according to Theorem 11 and to Theorem 4, since $\omega' \models \tau(\mathcal{W}) \wedge \tilde{KB}$ the world state ω is legal;
- applying result of Theorem 2 we can conclude also than:

$$B(Y)(\mathbf{X})^{\omega \triangleleft \sigma_{\mathbf{X}}} = \tau(B(Y)(\mathbf{X}))^{\omega'}$$

Since the binding schema is assumed valid, the evaluation of the query associated to the variable name Y, provided the pair $\langle \omega, \sigma \rangle$, is s.t.:

$$\|B(Y)^{\omega}(\sigma_{\mathbf{X}})\| = 1$$

Applying the definition of query evaluation we have that:

$$\left\|\tau(B(Y)(\mathbf{X}))^{\omega'}\right\| = 1$$

obtaining the contradiction that proves the claim. □

Theorem 59. *Given a world specification $\mathcal{W}$ and a binding schema B, the problem of checking if the binding schema is valid is in* coNEXP.

Proof. In order to solve the problem, we can apply the property shown in Theorem 58, reducing it to a reasoning task in $\mathcal{C}^2$ logics.

Also in this case the encoding is linear in the size of the input: in fact for each variable assigned in the schema there is a fragment in the implication consequent formula. So the size of resulting formula is linear in the number of binding schema queries. □

The absolute validity of a binding schema (which means considering every possible instantiation of input variables) is often a condition to hard to fulfill, but since we generally have to deal with assignments subject to various constraints, we can relax such a definition without any significant restriction.

Definition 86 (Valid binding schema w.r.t. a condition)**.** *Given a world specification $\mathcal{W}$, a binding schema B is valid w.r.t. a condition $\boldsymbol{C}$ (expressed on the same signature) iff, for any legal world state and for any input assignment consistent w.r.t. the condition, the evaluation of B assigns exactly one value to each output variable.*

We can state also the following claim generalizing Theorem 58 and Corollary 14:

Corollary 18. *Given a world specification $\mathcal{W}$, a binding schema B over $\boldsymbol{X}$ and $\boldsymbol{Y}$ is valid w.r.t. a condition $\boldsymbol{C}$ iff the following implication holds:*

$$\tilde{KB}^{\boldsymbol{V}} \wedge \tau(\mathcal{W}) \wedge \bigvee_{C \in \boldsymbol{C}} \bigwedge_{c \in C} \gamma(c) \models \bigwedge_{Y \in \boldsymbol{Y}} \sharp(\tau(B(Y)(\boldsymbol{X}))) = 1$$

where the axiom schema $\tilde{KB}^{\boldsymbol{V}}$ is instantiated over $\boldsymbol{X}$.

In the following we enrich such definitions so that we can apply them to a pair of service and goal.

Definition 87 (Weakly consistent binding schema)**.** *Let $\mathcal{W}$ be a consistent world specification, G a consistent user goal, S an accessible service. A valid binding schema B, w.r.t. user commitments, is weakly consistent for the pair $\langle G, S\rangle$ iff there exists a legal world state ω and goal instantiation $\sigma_{\boldsymbol{U}}$ s.t. the user commitments are satisfied in ω and the service is accessible using the binding $\sigma_{\boldsymbol{X}} = B(\omega, \sigma_{\boldsymbol{U}})$.*

Definition 88 (Consistent binding schema)**.** *Let $\mathcal{W}$ be a consistent world specification, G a consistent user goal, S an accessible service. A valid binding schema B, w.r.t. user commitments, is consistent for the pair $\langle G, S\rangle$ iff for each legal world state ω and goal instantiation $\sigma_{\boldsymbol{U}}$ s.t. the user commitments are satisfied in ω, the service is accessible using the binding $\sigma_{\boldsymbol{X}} = B(\omega, \sigma_{\boldsymbol{U}})$.*

Remark 32. *For the sake of clarity we always assume that distinct variable sets (belonging to service or goal specification) are mutually disjoint.*

We introduce the following new axiom schema ΔKB^B:

$$X \equiv B(X)(\mathbf{U})$$

that given, a binding schema B for a pair goal/service, is instantiated for each input variable name in $\mathbf{X}$, enforcing the constraints associated to the assignment of service input variables given the goal instantiation parameters.

Lemma 40. *Given a structure $\hat{\omega}$ and a structure ω and an assignment $\sigma_{\boldsymbol{V}}$ s.t. $\omega \lhd \sigma_{\boldsymbol{V}} \rightsquigarrow \hat{\omega}$, and a valid binding schema B form $\boldsymbol{Y}$ to $\boldsymbol{X}$ s.t. $\boldsymbol{V} = \boldsymbol{Y} \cup \boldsymbol{X}$ and $\boldsymbol{Y} \cap \boldsymbol{X} = \emptyset$. The assignment $\sigma_{\boldsymbol{X}}$ is obtained as evaluation of the binding schema B in ω given $\sigma_{\boldsymbol{Y}}$ iff the following condition holds:*

$$\hat{\omega} \models KB^{\boldsymbol{V}} \wedge \Delta KB^B$$

Proof. The claim follow from the definitions of embedding, query evaluation and binding schema semantics, since for each $X \in \mathbf{X}$ we have that:

$$\sigma(X) = X^{\omega \lhd \sigma_{\mathbf{V}}} = X^{\tilde{\omega}} = [B(X)(\mathbf{U}_G)]^{\tilde{\omega}} = [B(X)(\mathbf{U}_G)]^{\omega \lhd \sigma_{\mathbf{V}}} = B(X)^{\omega}(\sigma)$$

$\square$

Remark 33. *In the following we always consider the embedding relation (and associated mapping/projection functions $\mu^{\boldsymbol{V}}$ and $\pi^{\boldsymbol{V}}$) w.r.t. the environment $\boldsymbol{V} = \boldsymbol{X}_S \cup \boldsymbol{U}_G$. Moreover, unless it is explicitly denied, we also assume that all variable alphabets are always mutually disjoint.*

Theorem 60. *Given a consistent world specification $\mathcal{W}$, an admissible goal G, an accessible service S, and a valid binding schema B for $\boldsymbol{X}_S$ given $\boldsymbol{U}_G$, it is weakly consistent for the pair $\langle G, S\rangle$ iff the following formula is satisfiable:*

$$\tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge KB^P$$

Proof. We assume that the binding schema is weakly consistent w.r.t. the given service/goal pair, but that the formula is not consistent. In other words, there does not exist any structure $\tilde{\omega}$ s.t.:

$$\tilde{\omega}\tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \models KB^P$$

Let ω be a legal world state and $\sigma_{\mathbf{U}}$ be a goal instantiation s.t. the user commitments are satisfied in ω and S is accessible using the binding schema B: given the weak consistency assumption such a pair must exist.

Now, we can apply the mapping function $\mu^{\mathbf{V}}$ to the pair $\langle \omega, \sigma_{\mathbf{V}}\rangle$ obtaining a new interpretation structure $\tilde{\omega}$ that embeds such a pair and that, according to Theorems 10 and 54, is also s.t.:

$$\tilde{\omega} \models \tau(\mathcal{W}) \wedge KB^H$$

Moreover, given the definition of query evaluation, we can also conclude that for each $X \in \mathbf{X}$, we have that:

$$\tilde{\omega} \models X \equiv B(X)(\mathbf{U})$$

since we have assumed that $\sigma_{\mathbf{X}}(X) = B(X)^{\omega}(\sigma_{\mathbf{Y}})$. We can conclude that $\tilde{\omega} \models \Delta KB^B$ and applying Corollary 3, since we have assumed that the given service is accessible in ω using $\sigma_{\mathbf{X}}$, also that $\tilde{\omega} \models KB^P$. The structure $\tilde{\omega}$ is a model of the given formula that is clearly satisfiable.

We now assume that, despite the formula is consistent, the binding schema is not weakly consistent for the service/goal pair given the world specification. It means that there does not exist any legal world state ω and consistent instantiation $\sigma_{\mathbf{U}}$ of the goal s.t.:

- the goal user commitments are satisfied in ω;
- the evaluation of the binding schema (that is assumed to be valid) in this context provides a well-founded assignment $\sigma_{\mathbf{X}}$ for service input variables.

Given the hypothesis of non-consistency of the binding schema we assume that the service S is not accessible in ω using the assignment $\sigma_{\mathbf{X}}$.

Given a model $\tilde{\omega}$ of the satisfiable formula (such a model must exist), according to Theorems 11 and 54 we can build a pair $\langle \omega, \sigma_{\mathbf{V}}\rangle = \pi^{\mathbf{V}}(\tilde{\omega})$ s.t.:

- ω is legal world state w.r.t. the specification $\mathcal{W}$;
- the goal requester commitments are satisfied in ω given the assignment $\sigma_{\mathbf{U}}$ obtained from restricting $\sigma_{\mathbf{V}}$ to names is $\mathbf{U}$;
- the variables in $\mathbf{X}$ are consistently assigned to a domain element according to the evaluation of the binding schema (Lemma 40).

Moreover, applying Theorem 17, since we have assumed that $\tilde{\omega} \models KB^P$, we can conclude that such a service is also accessible in ω using the input assignment obtained from the evaluation of binding functions. In other words, at least the obtained pair is s.t. the service is accessible using the assignment obtained applying the binding schema to the goal instantiation and, hence, such a binding schema is at least weakly consistent, proving the claim. $\square$

Theorem 61. *Given a consistent world specification $\mathcal{W}$, an admissible goal G, an accessible service S, and a valid binding schema B for $\boldsymbol{X}_S$ given $\boldsymbol{U}_G$, it is consistent for the pair $\langle G, S\rangle$ iff the following implication hold:*

$$\tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \models KB^P$$

Proof. We assume that the binding schema is consistently defined w.r.t. the given service/goal pair, but that the implication does not hold. In other words, there must exist a structure $\tilde{\omega}$ s.t.:

$$\begin{aligned}&\tilde{\omega} \models \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B\\&\tilde{\omega} \not\models KB^P\end{aligned}$$

According to Theorems 11 and 54 we can build a pair $\omega, \sigma_{\mathbf{V}} = \pi^{\mathbf{V}}(\tilde{\omega})$ s.t.:

- ω is legal world state w.r.t. the specification $\mathcal{W}$;
- the goal requester commitments are satisfied in ω given the assignment $\sigma_{\mathbf{U}}$ obtained from restricting $\sigma_{\mathbf{V}}$ to names is $\mathbf{U}$;
- the variables in $\mathbf{X}$ are consistently assigned to a domain element according to the evaluation of the binding schema (Lemma 40).

So, given the assumption on the consistency of the binding schema we can conclude that such a service is also accessible in ω using the input assignment obtained from the evaluation of binding functions. Consequently, applying Theorem 17, we can conclude that $\tilde{\omega} \models KB^P$, resulting in a contradiction.

We now assume that, despite the implication holds, the binding schema is not consistently defined for the service/goal pair given the world specification. It means that there exists a legal world state ω and a consistent instantiation $\sigma_{\mathbf{U}}$ of the goal s.t.:

- the goal user commitments are satisfied in ω;
- the evaluation of the binding schema (that is assumed to be valid) in this context provides a well-founded assignment $\sigma_{\mathbf{X}}$ for service input variables.

Since the hypothesis of non-consistency of the binding schema we can also conclude that the service S is not accessible in ω using the assignment $\sigma_{\mathbf{X}}$.

We can apply the mapping function $\mu^{\mathbf{V}}$ to the pair $\langle \omega, \sigma_{\mathbf{V}}\rangle$, obtaining a new interpretation structure $\tilde{\omega}$ that embeds such a pair and that, according to Theorems 10 and 54, is also s.t.:

$$\tilde{\omega} \models \tau(\mathcal{W}) \wedge KB^H$$

Moreover, given the definition of query evaluation, we can also conclude that for each $X \in \mathbf{X}$, we have that:

$$\tilde{\omega} \models X \equiv B(X)(\mathbf{U})$$

since we have assumed that $\sigma_{\mathbf{X}}(X) = B(X)^{\omega}(\sigma_{\mathbf{Y}})$. We can conclude that $\tilde{\omega} \models \Delta KB^{B}$ and we can check that implication antecedents are satisfied and, since the implication is assumed to be verified, we can finally conclude that $\tilde{\omega} \models KB^{P}$. Applying Corollary 3, we can now infer also that the service is accessible in ω using $\sigma_{\mathbf{X}}$, obtaining a contradiction that proves the claim. □

6.1.3 Service/goal adequacy

In the previous sections we have introduced and discussed the basic elements required to formalize the matchmaking problem. In the following, we are interested in studying how a service is adequate to achieve a given goal.

Let G be an admissible execution goal, given a valid service S and a consistent binding schema B, we are interested in assessing the adequacy degree of the service to achieve a world state compatible with the goal requirement starting from a world state s.t. client commitments hold using the binding schema. According to the quantification on initial and resulting states from the service execution, we can distinguish various kinds of adequacy notions.

For example, if a service can always surely achieve a given goal, it can be declared as:

Definition 89 (Strong uniform adequacy). *A service S is strongly and uniformly adequate to a goal G iff, for each legal world state in which the service preconditions hold, and for each goal instantiation that is compliant to user commitments, every service enactment obtained applying the binding schema results in a state where the user requirement conditions hold.*

Other degrees of adequacy are the followings:

Definition 90 (Strong non-uniform adequacy). *A service S is strongly and non-uniformly adequate to a goal G iff there exist a legal world state in which preconditions hold and a goal instantiation that is compliant to user commitment s.t. every service enactment obtained applying the binding schema results in a state where user requirement conditions hold.*

Definition 91 (Weak uniform adequacy). *A service S is weakly and uniformly adequate to a goal G iff, for each legal world state in which preconditions hold and for each goal instantiation that is compliant to user commitment, there exists at least a service enactment obtained applying the binding schema resulting in a state where user requirement conditions hold.*

Definition 92 (Weak non-uniform adequacy). *A service S is weakly and non-uniformly adequate to a goal G iff there exist a legal world state in which preconditions hold and a goal instantiation that is compliant to user commitment s.t. there exists at least a service enactment obtained applying the binding schema resulting in a state where user requirement conditions hold.*

Roughly speaking, a strong adequacy degree implies that the service can surely achieve the execution goal, provided that service and goal preconditions are satisfied. On the other hand, a uniform adequacy degree implies that, every time service and goal preconditions are satisfied, there exists a suitable service computation that achieve the goal. The intuition behind this formalization is reported in Table 6.1.

Level	**Description**
SU	The service always surely achieves the goal
WU	The service can always possibly achieve the goal
SNU	The service sometime surely achieves the goal
WNU	The service can sometime possibly achieve the goal

Table 6.1: Service/goal adequacy levels

Lemma 41. *Given an accessible and valid service and an admissible goal, using a consistent binding schema, the strong goal adequacy implies the corresponding weak property.*

Proof. The claim follows from these observations:

- the world specification is assumed to be consistent, so at least a legal world state exists;
- the assumption on service accessibility and validity ensures that also a state suitable for the service activation exists and the service enactment result is always a legal state;
- the consistency assumption regarding the binding schema also ensures that always exists a goal instantiation s.t. both user commitments and activation preconditions hold.

$\square$

Example 10. *Given the services defined in Examples 2 and 3, we consider the following simple parametric goal G, denoting a citizen that is attempting to change its own residence to town* town_2*:*

$$\boldsymbol{U} = \{u\}$$
$$\mathcal{H} = u \sqcap \mathsf{Citizen}\ \mathtt{and}\ \mathtt{not}\ (\exists \mathsf{residentIn}^{-}.u) \sqcap \{\mathsf{town}_2\}$$
$$\mathcal{R} = \left\{-\mathsf{residentIn}(u, \exists \mathsf{residentIn}^{-}.u), +\mathsf{residentIn}(u, \{\mathsf{town}_2\})\right\}$$

The binding schema of input variable of the services is the same in both cases and it is defined as:

$$B = \{x_1(u) = u, x_2(u) = \mathsf{town}_2\}$$

It is worth noticing, that while the service S is strongly and uniformly adequate to accomplish the user goal, since its preconditions are always satisfied by user's commitment and its effects implies the user's requirements, the service S_1 is adequate but in a non-uniform way, since, if the requestor does not live in town_1*, (s)he cannot access the service, even-through, once enacted, the service always accomplished the required effects.*

Now we address the adequacy problem considering a valid non-deterministic e-service, but we initially ignore the effect repair, for the sake of simplicity. The case of partially specified services will be addressed in the next section.

Theorem 62. *A consistent, accessible and valid non-deterministic e-service S is strongly and uniformly adequate to an admissible goal G given a (weakly) consistent binding schema B w.r.t. a world specification $\mathcal{W}$, iff the following implication holds:*

$$KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \bigwedge_{E\in\mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E)\right) \models \bigwedge_{E\in\mathcal{E}} \tau_{m_E}(\mathcal{W}) \rightarrow KB^R(m_E)$$

where m_E is the name mapping function for the domain and the instantiation variable names related to effect $E \in \mathcal{E}$.

Proof. We assume that the implication is verified, but that the given service is not uniformly and strongly adequate to the goal. In other words, it means that exists at least a possible legal service enactment resulting into a legal world state, since the service validity assumption, s.t. the goal requirement constraints are not satisfied.

Let ω be the initial state and $\sigma_{\mathbf{X}}$ and $\sigma_{\mathbf{U}}$ be resp. the service input variable assignment and the goal parameter assignment. Since we are employing the binding schema B, the assignment to variables in $\mathbf{X}$ is functional depending upon the goal instantiation. The considered enactment is denoted as $\langle \omega', \sigma'_{\mathbf{Y}}\rangle$, while E^* is the selected effect. Considering also other possible enactments from the given initial condition, one for each possible service outcomes, employing the mapping function we can embed all them into a structure $\hat{\omega}$. It is worth noticing that implication antecedents are satisfied (Theorems 17, 45, Lemma 40, and Corollary 14 applied to goal user commitment constraints), so applying the implication we can conclude also that:

$$\hat{\omega} \models KB^R(m_{E^*})$$

since, given the validity assumption and the fact that ω' is embedded using m_{E^*} into $\hat{\omega}$, and the select effect is legal and hence that $\hat{\omega} \models \tau_{m_{E^*}}(\mathcal{W})$ (the assumption on service validity ensure that such an effect always exists). Now, applying Corollary 15 from $\hat{\omega}$ to ω' on $\mathcal{R}$ using m_{E^*}, we conclude that also goal requirement conditions hold in ω', resulting in a contradiction.

Now we assume that, despite the service is strongly and uniformly adequate to the goal, the implication does not hold. Let $\hat{\omega}$ be a structure s.t.:

$$\hat{\omega} \models KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \bigwedge_{E\in\mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E)\right)$$

but there exists at least an effect $E^* \in \mathcal{E}$ s.t.

$$\hat{\omega} \not\models \tau_{m_{E^*}}(\mathcal{W}) \rightarrow KB^R(m_{E^*})$$

or, in other words, that:

$$\hat{\omega} \models \tau_{m_{E^*}}(\mathcal{W}) \wedge \neg KB^R(m_{E^*})$$

Applying Theorem 46, we can conclude that the quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}}\rangle$, obtained from the structure $\hat{\omega}$ using the projection function π_n w.r.t. the name mapping m_{E^*}, is a valid enactment of the service form the state ω to the state ω' using the input assignment $\sigma_{\mathbf{X}}$. Since the service is assumed valid, applying Theorem 47 we can also conclude that:

- the initial and final state are legal w.r.t. $\mathcal{W}$, since they are embedded into a structure that satisfies both translated constraint sets;
- input assignment depends on the goal instantiation using the binding schema (Lemma 40);
- goal client commitment constraints are satisfied in the initial state (Corollary 14).

Given the hypothesis of strong and uniform service adequacy, we need also conclude that in ω' client requirement constraints are satisfied, and considering Lemma 32 and Corollary 15, it follows that:

$$\hat{\omega} \models KB^R(m_{E^*})$$

From this contradiction follows the claim. □

Theorem 63. *A consistent, accessible and valid non-deterministic e-service S is weakly and uniformly adequate to an admissible goal G given a (weakly) consistent binding schema B w.r.t. a world specification $\mathcal{W}$, iff the following implication holds:*

$$KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \bigwedge_{E\in\mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E)\right) \models \bigvee_{E\in\mathcal{E}} \tau_{m_E}(\mathcal{W}) \wedge KB^R(m_E)$$

where m_E is the name mapping function for the domain and the instantiation variable names related to effect $E \in \mathcal{E}$.

Proof. We assume that the implication is verified, but that the given service is not uniformly and weakly adequate to the goal. In other words, it means that exists at least a legal world state and a variable assignment s.t. the user commitments are satisfied, the service is accessible, and the goal requirement constraints are not satisfied for any enactment from the given initial invocation context.

Let ω be the initial state and $\sigma_{\mathbf{X}}$ and $\sigma_{\mathbf{U}}$ be resp. the service input variable assignment and the goal parameter assignment. Since we are employing the binding schema B, the assignment to variables in $\mathbf{X}$ is functional depending upon the goal instantiation. We consider all possible enactment, according to service outcome specification, and,employing the mapping function, we can embed all them into a structure $\hat{\omega}$. It is worth noticing that implication antecedents are satisfied (Theorems 17, 45, Lemma 40, and Corollary 14 applied to goal user commitment constraints), so applying the implication we can conclude also that at least an effect $E^* \in \mathcal{E}$ s.t.:

$$\hat{\omega} \models \tau_{m_{E^*}}(\mathcal{W}) \wedge KB^R(m_{E^*})$$

must exist. Now, applying Corollary 15 from $\hat{\omega}$ to ω' on $\mathcal{R}$ using m_{E^*}, we conclude that ω' is a legal world state and also goal requirement conditions hold in ω', obtaining a contradiction.

Now we assume that, despite the service is weakly and uniformly adequate to the goal, the implication does not hold. Let $\hat{\omega}$ be a structure s.t.:

$$\hat{\omega} \models KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E)\right)$$

but there does not exist any effect $E \in \mathcal{E}$ s.t.

$$\hat{\omega} \not\models \tau_{m_E}(\mathcal{W}) \wedge KB^R(m_E)$$

Applying Theorem 46, we can conclude that the quadruple $\langle \omega, \omega', \sigma_{\mathbf{X}}, \sigma'_{\mathbf{Y}} \rangle$, obtained from the structure $\hat{\omega}$ using the projection function π_n w.r.t. a name mapping m, is an enactment of the service form the state ω to the state ω' using the input assignment $\sigma_{\mathbf{X}}$. Since the service is assumed valid, applying Theorem 47 we can also conclude that:

- the initial state is legal w.r.t. $\mathcal{W}$;
- at least an effect E^* leading to a legal final state must exist;
- input assignment depends on the goal instantiation using the binding schema (Lemma 40);
- goal client commitment constraints are satisfied in the initial state (Corollary 14).

Given the hypothesis of weak and uniform service adequacy, we can also conclude that there exists at least an effect leading to a legal world state where also client requirements are satisfied. W.l.o.g., we can assume E^* as such an effect. Considering Lemma 32 and Corollary 15, it follows that

$$\hat{\omega} \models KB^R(m_{E^*})$$

or, in other words, given the validity of ω', that:

$$\hat{\omega} \models \tau_{m_{E^*}}(\mathcal{W}) \wedge KB^R(m_{E^*})$$

From this contradiction follows the claim. □

Theorem 64. *A consistent, accessible and valid non-deterministic e-service S is strongly and non-uniformly adequate to an admissible goal G given a (weakly) consistent binding schema B w.r.t. a world specification $\mathcal{W}$, iff the following formula is satisfiable:*

$$KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E)\right) \\ \wedge \bigwedge_{E \in \mathcal{E}} \left(\tau_{m_E}(\mathcal{W}) \rightarrow KB^R(m_E)\right)$$

where m_E is the name mapping function for the domain and the instantiation variable names related to effect $E \in \mathcal{E}$.

Proof. We assume that the formula is consistent, but that the given service is not non-uniformly and strongly adequate to the goal. In other words, it means that does not exist any legal world state and variable assignment s.t. the user commitments are satisfied, the service is accessible, and the goal requirement constraints are satisfied for any legal enactment from the given initial invocation context.

Let $\hat{\omega}$ be a model of the given knowledge base (it is assumed consistent, so such a structure must exist). Applying Lemma 31 to each distinct service effect, we obtain a set of tuples, each representing a possible enactment, possibly legal, from a state ω using the input assignment $\sigma_{\mathbf{X}}$, according to Theorem 46. Moreover, given Corollary 14, applied to goal user commitment constraints, and Lemma 40 we can conclude also that user commitments hold in ω and that such a service is activated using the binding schema B. Both service and binding schema are assumed to be valid, so there exists a non-empty set of legal service effects (which means service effect that given the invocation context lead the system into a legal world state). Given the hypothesis of non-adequacy of the pair goal/service, at least a legal effect $E^* \in \mathcal{E}$, s.t. in the final world state goal requirement constraints do not hold, must exist.

According to Theorem 47, since the implication antecedents are implied by the given knowledge base, we can also conclude that:

$$\hat{\omega} \models \tau_{m_{E^*}}(\mathcal{W})$$

and, since such knowledge base is satisfied in $om\hat{e}ga$, also that:

$$\hat{\omega} \models KB^R(m_{E^*})$$

Hence, applying Corollary 15 to goal requirement constraints using the m_{E^*} name mapping function, we can conclude that also effect E^* achieve them, contradicting the hypothesis of non-adequacy.

Now we assume that the pair goal/service is strongly non-uniformly adequate but the given formula is not satisfiable. Let ω be a legal world state and $\sigma_{\mathbf{U}}$ be a goal instantiation assignment s.t.:

- the service is activated using the provided binding schema applied to goal instantiation assignment;
- the service is accessible given the initial state and input assignment.

Applying the definition of service effect, we can compute a set possible enactment for any given non-deterministic outcome, from the initial state and, using the mapping function, embed all them into a structure $\hat{\omega}$. Applying Theorems 17, 45, 54, and Lemma 40, we can, without difficulty, conclude that the structure is s.t.:

$$\hat{\omega} \models KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \bigwedge_{E \in \mathcal{E}} \left(\Delta KB_n^I(\mathsf{Top_{m_E}}, m_E) \wedge \Delta KB_c^U(m_E) \right)$$

In order to complete the argumentation, we need to show that the remaining conjunct is also satisfied in $\hat{\omega}$. Considering a non-deterministic effect $E \in \mathcal{E}$, if $\hat{\omega} \not\models \tau_{m_E}(\mathcal{W})$, we can easily verify that the implication is valid, but according to Theorem 47 , at least an effect E^* s.t. $\hat{\omega} \models \tau_{m_{E^*}}(\mathcal{W})$ must exist, since preconditions hold in $\hat{\omega}$. Since we have assumed that such a pair goal/service is strongly adequate, given the enforcement of effect E^* by S lead to a legal world state where goal requirement constraint are satisfied, applying Corollary 15, we obtain that also $\hat{\omega} \models KB^R(m_{E^*})$ and, hence, that:

$$\hat{\omega} \models \bigwedge_{E \in \mathcal{E}} \left(\tau_{m_E}(\mathcal{W}) \rightarrow KB^R(m_E)\right)$$

proving the claim. □

Theorem 65. *A consistent, accessible and valid non-deterministic e-service S is weakly and non-uniformly adequate to an admissible goal G given a (weakly) consistent binding schema B w.r.t. a world specification $\mathcal{W}$, iff the following formula is satisfiable:*

$$KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E)\right)$$
$$\wedge \left(\bigvee_{E \in \mathcal{E}} \tau_{m_E}(\mathcal{W}) \wedge KB^R(m_E)\right)$$

where m_E is the name mapping function for the domain and the instantiation variable names related to effect $E \in \mathcal{E}$.

Proof. We assume that the formula is consistent, but that the given service is not non-uniformly and weakly adequate to the goal. In other words, it means that does not exist any legal world state and variable assignment s.t. the user commitments are satisfied, the service is accessible, and the goal requirement constraints are satisfied for at least an enactment from the given initial invocation context.

Let $\hat{\omega}$ be a model of the given knowledge base (it is assumed consistent, so such a structure must exist). Applying Lemma 31 to each distinct service effect, we obtain a set of tuples, each representing a possible enactment, possibly legal, from a state ω using the input assignment $\sigma_{\mathbf{X}}$, according to Theorem 46. Moreover, given Corollary 14, applied to goal user commitment constraints, and Lemma 40 we can conclude also that user commitments hold in ω and that such a service is activated using the binding schema B. Both service and binding schema are assumed to be valid, so there exists a non-empty set of legal service effects (which means service effect that given the invocation context lead the system into a legal world state). Given the hypothesis of non-adequacy of the pair goal/service, for each legal effect $E \in \mathcal{E}$, in the final world state goal requirement constraints must not hold. Since the service is valid at least a legal effect must exist.

According to Theorem 47, since the implication antecedents are implied by the given knowledge base, we can also conclude that:

$$\hat{\omega} \models \tau_{m_E}(\mathcal{W})$$

for any legal service outcome. Since such knowledge base is satisfied in $om\hat{e}ga$, there must exist at least a legal effect E^* s.t.:

$$\hat{\omega} \models KB^R(m_{E*})$$

Hence, applying Corollary 15 to goal requirement constraints using the m_{E^*} name mapping function, we can conclude that at least the effect E^* achieve the goal, contradicting the hypothesis of non-adequacy.

Now we assume that the pair goal/service is weakly non-uniformly adequate but the given formula is not satisfiable. Let ω be a legal world state and $\sigma_{\mathbf{U}}$ be a goal instantiation assignment s.t.:

- the service is activated using the provided binding schema applied to goal instantiation assignment;
- the service is accessible given the initial state and input assignment.

Applying the definition of service effect, we can compute a set possible enactment for each given non-deterministic outcome, from the initial state and, using the mapping function, embed all them into a structure $\hat{\omega}$. Applying Theorems 17, 45, 54, and Lemma 40, we can easily conclude that the structure is s.t.:

$$\hat{\omega} \models KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E)\right)$$

In order to complete the proof, we need to show that the remaining conjunct is also satisfied in $\hat{\omega}$. According to Theorem 47, at least a legal effect E^* s.t. $\hat{\omega} \models \tau_{m_{E^*}}(\mathcal{W})$ must exist, since preconditions hold in $\hat{\omega}$. Since we have also assumed that such a pair goal/service is weakly adequate, there must exist at least a legal effect whose enforcement by S lead to a legal world state where goal requirement constraint are satisfied. W.l.o.g., we can assume E^* as such an effect, so applying Corollary 15, we can conclude that also $\hat{\omega} \models KB^R(m_{E^*})$ and, hence, that:

$$\hat{\omega} \models \bigvee_{E \in \mathcal{E}} \tau_{m_E}(\mathcal{W}) \wedge KB^R(m_E)$$

proving the claim. □

Remark 34. *In case of strongly consistent binding schema the axioms KB^P are entailed according Theorem 61 from other axioms in the formula, so they can be omitted from the claim statement.*

Theorem 66. *Given a world specification $\mathcal{W}$, an accessible and consistent service S, a consistent goal G and a consistent binding schema B for the pair, the problem of checking if S is uniformly adequate to achieve G using B is in* coNEXP*, while the problem of if S is non-uniformly adequate to achieve G using B is in* NEXP*.*

Proof. As done for other cases analyzed in this work, we can solve the problem applying the logical properties proved in previous theorems reducing the verification of uniform adequacy to an implication decision in $\mathcal{C}^2$ logics, and the verification of non-uniform adequacy to a satisfiability decision $\mathcal{C}^2$. Like other reductions it is also linear in the size of the input (number and length of axioms, preconditions, effects specifications, goal constraints and binding queries). □

Remark 35. *While the verification of uniform adequacy seems, since we have not provided any lower bound in terms complexity class, to be more complex than the verification of the non-uniform property, the verification of strong and weak adequacy properties seem to belong to the same complexity class.*

6.1.4 Incompleteness and service/goal adequacy

In the previous section we have analyzed the problem of verifying the adequacy of an e-service to perform a user task described as a goal in the case of complete effect specification. Thus, in order to complete the discussion, we need to introduce also the repair step into the service effect evaluation.

As defined is Chapter 4, the repair strategy relies upon a minimal-change semantics in terms of model changes: in other words, we are assuming that always the "simplest" repair option is selected. While this aspect has not played a significant role devising previous results, it turns to be a critical point in the analysis of functional properties and, in particular, of the adequacy of a service to achieve a goal. Roughly speaking, in this scenario we do not only need to show that the service can accomplish the required task, but also that the repair does not interfere with its side-effects. In order to show that such a property hold, we need to simulate the repair selection procedure that means that generally a goal is achieved iff its requirement constraints hold in the world state obtained by the minimal-change effective repair.

In the rest, given a repair search space R_S we introduce the following notation for denoting some interesting subsets:

$$R_S^k \triangleq \{R \in R_S | \, \|R\| = k\}$$

$$\hat{R}_S^k \triangleq \bigcup_{j=0}^{j<k} R_S^j$$

while $s \triangleq \max_{R \in R_S} \|R\|$. We also define the formula $\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ to denote the following material implication finitely quantified over service effects and repairs grouped by their size:

$$\begin{aligned}\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S} \triangleq \bigwedge_{E \in \mathcal{E}} \bigwedge_{k=0}^{s} \bigwedge_{R \in R_S^k} & \tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R}) \\ & \wedge \bigwedge_{R' \in \hat{R}_S^k} \neg \Big(\tau_{n_{E,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R'})\Big) \rightarrow KB^R(n_{E,R})\end{aligned} \tag{6.2}$$

Finally, we also introduce an analogous formula $\eta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ to denote the following conjunction finitely quantified over service effects and repairs grouped by their size:

$$\begin{aligned}\eta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S} \triangleq \bigvee_{E \in \mathcal{E}} \bigvee_{k=0}^{s} \bigvee_{R \in R_S^k} & \tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R}) \\ & \wedge \bigwedge_{R' \in \hat{R}_S^k} \neg \Big(\tau_{n_{E,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R'})\Big) \wedge KB^R(n_{E,R})\end{aligned} \tag{6.3}$$

Remark 36. *We point out that the repair strategy has been devised in order to allow the employment of partially specified services, since it is able to adjust such a specification consistently w.r.t. the domain constraints according to a sort of* *minimal-change semantics. But, on the other hand, this approach is completely unaware of user's goal since it keeps into account only service and domain specification, which means that the repair strategy could potentially interfere with the adequacy of a (under-specified) service to achieve a goal.*

In order to solve the adequacy checking problem, as in the other cases, we devise a feasible encoding in terms of logical inference.

Theorem 67. *A consistent, accessible and repairable non-deterministic e-service S is strongly and uniformly adequate to an admissible goal G given a (weakly) consistent binding schema B w.r.t. a world specification $\mathcal{W}$ and a normal repair family R_S, iff the following implication holds:*

$$\bigwedge_{E\in\mathcal{E}} \left(\Delta KB_n^I(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB_c^U(m_E) \wedge \bigwedge_{R\in R_S} \Delta KB_n^R(m_E, n_{E,R}) \right) \wedge KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \models \theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S} \tag{6.4}$$

where m_E and $n_{E,R}$ are the name mapping functions for the domain and the instantiation variable names defined for each effect $E \in \mathcal{E}$ and for each repair $R \in R_S$.

Proof. We initially assume that the pair service/goal is strongly and uniformly adequate given the world specification $\mathcal{W}$, the binding schema B and the repair family R_S, but that the given implication does not hold. In other words, there exists a structure $\hat{\omega}$ s.t. the implication antecedents are satisfied in $\hat{\omega}$ but not the implication consequent, or, in other words, that:

$$\hat{\omega} \not\models \theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$$

Applying the projection function π_n^R to each service effect $E \in \mathcal{E}$ and to each considered repair $R \in R_S$, we obtain a set of quintuples of the form:

$$\langle \omega, \omega'_E, \omega''_{E,R}, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$$

By definition they agree on 4 components out of 5, since they are built upon shared names (e.g., $\mathbf{A}$, $m(\mathbf{P})$, and so on).

Since, for each effect $E \in \mathcal{E}$ the structure is s.t. $\hat{\omega} \models nKB^U_{m_E}$, any quadruple $\langle \omega, \omega'_E, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$ represents a valid service enactment by Theorem 46 and, since the service is assumed repairable, applying Theorem 52, we can also conclude that:

- the initial and final state are legal w.r.t. $\mathcal{W}$, since they are embedded into a structure that satisfies both translated constraint sets;
- input assignment depends on the goal instantiation using the binding schema (Lemma 40);
- goal client commitment constraints are satisfied in the initial state (Corollary 14);
- for every possible service invocation always exists at least pair effect/repair having the repair drawn from R_S, that leads the system to a new legal state w.r.t. $\mathcal{W}$.

According to the definition of repair strategy (see Figure 4.5), the selected repair is always the minimum w.r.t. size, so let E^* be the considered repairable effect, let R^* be the minimal-size repair that allows a consistent enactment.

So, applying Theorem 50, it follows that the structure $\hat{\omega}$ is also a model of the knowledge base $\Delta KB_n^C(m_{E^*}, n_{E^*,R^*})$. On the other hand, since the state ω''_{E^*,R^*} is legal, given the hypothesis of repairability of the service enactment, and it is embedded into $\hat{\omega}$, by Theorem 3, we can conclude that also:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W})$$

But, since R^* is the minimal-size repair having such a property given E^*, we can also conclude that:

$$\hat{\omega} \not\models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})$$

for every $R' \in \hat{R}_S^k$, where $k = \|R^*\|$.

Given the hypothesis of strong and uniform service adequacy, we need also conclude that in ω''_{E^*,R^*} client requirement constraints are satisfied, and, applying Lemma 32 and Corollary 15, we can conclude that:

$$\hat{\omega} \models KB^R(n_{E^*,R^*})$$

In other words, we have shown that:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R^*}) \\ \wedge \bigwedge_{R' \in \hat{R}_S^k} \neg \left(\tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})\right) \wedge KB^R(n_{E^*,R^*})$$

where $k = \|R^*\|$. For any other repair R, given the effect E^*, we have that at least one of the following conditions hold:

- the repair is not effective and the obtained state is not legal ($\hat{\omega} \not\models KB^R(n_{E^*,R})$;
- the repair is not consistent ($\hat{\omega} \not\models \Delta KB_n^C(m_{E^*}, n_{E^*,R})$);
- the repair is effective and consistent, but its size is greater than $\|R^*\|$.

It means that, for each repairable effect $E^* \in \mathcal{E}$, the corresponding material implication in $\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ holds. Moreover, if an effect E is not repairable, there does not exist any repair in the given family such that it is consistent and it leads the system to a legal state, hence the material implication antecedents are never satisfied, so the implication holds. We have shown that for every $E \in \mathcal{E}$ the structure $\hat{\omega}$ satisfies the material implication, or, in other words, that:

$$\hat{\omega} \models \bigwedge_{E \in \mathcal{E}} \bigwedge_{k=0}^{s} \bigwedge_{R \in R_S^k} \tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R}) \\ \wedge \bigwedge_{R' \in \hat{R}_S^k} \neg \left(\tau_{n_{E,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R'})\right) \rightarrow KB^R(n_{E,R})$$

From this contradiction we prove the first part of the theorem.

Now assume that the implication is satisfied, but the pair goal/service is not strongly and uniformly adequate given the binding schema, the repair family and world constraint. It means that there is at least an enactment where it is possible to select a repairable effect E^* and the corresponding repair R^* s.t. the final state ω^* is legal but does not enforce client requirements, despite the initial state was consistent with client commitments.

Let ω be the initial state and $\sigma_{\mathbf{X}}$ and $\sigma_{\mathbf{U}}$ be resp. the service input variable assignment and the goal parameter assignment. Since we are employing the binding schema B, the assignment to variables in $\mathbf{X}$ is functional depending upon the goal instantiation.

Considering also other possible enactments from the given initial condition, one for each possible service outcomes, and any repair $R \in R_S$, employing the mapping function $\mu^{\mathbf{R}}$ we can embed all them into a structure $\hat{\omega}$.

Since the service is accessible and we are considering a valid enactment, according to Theorem 45, the constructed interpretation $\hat{\omega}$ is a model of $nKB^U_{m_E}$ for every $E \in \mathcal{E}$. Now we consider the definition of repair insert and update set, and also of repaired extension of role and concept names: applying the Corollary 13, we can also conclude that constraints of $\Delta KB^R_n(m_E, n_{E,R})$ are satisfied for every $E \in \mathcal{E}$ and for every repair $R \in R_S$. Moreover, since we are assuming also that the binding schema is enforced and that user commitment constraints are satisfied in ω, so applying Corollary 14 and Lemma 40, we can, without difficulty, check that implication antecedents are satisfied.

Since we have assumed that the implication holds, we can conclude that:

$$\hat{\omega} \models \bigwedge_{E\in\mathcal{E}} \bigwedge_{k=0}^{s} \bigwedge_{R\in R^k_S} \tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB^C_n(m_E, n_{E,R})$$
$$\wedge \bigwedge_{R'\in\hat{R}^k_S} \neg\left(\tau_{n_{E,R'}}(\mathcal{W}) \wedge \Delta KB^C_n(m_E, n_{E,R'})\right) \rightarrow KB^R(n_{E,R})$$

Given an effect E^* and a repair R^*, s.t. they lead the system to legal state without enforcing user request, according to the definition of repairable service and Theorem 52, whose implication antecedents in Eq. 5.43 are subsumed by antecedents of the implication in Eq. 6.4, we can conclude that:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,R^*})$$

Since Lemma 21, the selected repair is also the minimal-size repair that is able to enforce domain constraint upon service effects without retracting them. In other words, for any other repair R' s.t. $\|R'\| < \|R^*\|$, we have that:

$$\hat{\omega} \not\models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,R'})$$

It means that antecedents of material implication in $\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ are satisfied for the effect E^* and the repair R^*, hence applying the implication we can also conclude that:

$$\hat{\omega} \models KB^R(n_{E^*,R^*})$$

Now, applying Corollary 15 from $\hat{\omega}$ to ω^* on $\mathcal{R}$ using n_{E^*,R^*}, we conclude that also goal requirement conditions hold in ω^*, obtaining a contradiction. □

Theorem 68. *A consistent, accessible and repairable non-deterministic e-service S is weakly and uniformly adequate to an admissible goal G given a (weakly) consistent binding schema B w.r.t. a world specification $\mathcal{W}$ and a normal repair family R_S, iff the following implication holds:*

$$\bigwedge_{E\in\mathcal{E}} \left(\Delta KB_n^I(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB_c^U(m_E) \wedge \bigwedge_{R\in R_S} \Delta KB_n^R(m_E, n_{E,R}) \right) \\ \wedge KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \models \eta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S} \tag{6.5}$$

where m_E and $n_{E,R}$ are the name mapping functions for the domain and the instantiation variable names defined for every effect $E \in \mathcal{E}$ and for every repair $R \in R_S$.

Proof. We initially assume that the pair service/goal is weakly and uniformly adequate given the world specification $\mathcal{W}$, the binding schema B and the repair family R_S, but that the given implication does not hold. In other words, there exists a structure $\hat{\omega}$ s.t. the implication antecedents are satisfied in $\hat{\omega}$ but not the implication consequent, or, in other words, that:

$$\hat{\omega} \not\models \eta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$$

As in the previous claim, we can apply the projection function π_n^R for every service effect $E \in \mathcal{E}$ and for every considered repair $R \in R_S$ in order to obtain a set of quintuples of the form:

$$\langle \omega, \omega'_E, \omega''_{E,R}, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$$

By definition they agree on 4 components out of 5, since they are built upon shared names (e.g., $\mathbf{A}$, $m(\mathbf{P})$, and so on).

Since, for every effect $E \in \mathcal{E}$ the structure is s.t. $\hat{\omega} \models nKB^U_{m_E}$, any quadruple $\langle \omega, \omega'_E, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$ represents a valid service enactment by Theorem 46 and since the service is assumed repairable, applying Theorem 52 we can also conclude that:

- the initial and final state are legal w.r.t. $\mathcal{W}$, since they are embedded into a structure that satisfies both translated constraint sets;
- input assignment depends on the goal instantiation using the binding schema (Lemma 40);
- goal client commitment constraints are satisfied in the initial state (Corollary 14);
- for every possible service invocation always exists at least pair effect/repair having the repair drawn from R_S, that leads the system to a new legal state w.r.t. $\mathcal{W}$.

Given the hypothesis of weak and uniform service adequacy (even keeping into account the repair) we can conclude that must exist a final state ω''_{E^*,R^*} where client requirement constraints are satisfied or, in other terms, considering Lemma 32 and Corollary 15, that:

$$\hat{\omega} \models KB^R(n_{E^*,R^*})$$

Moreover, according to the definition of repair strategy (see Figure 4.5), if the effect E^* is repairable using R^*, it must be the minimal-size repair that allows a consistent enactment using such an effect. So, applying Theorem 50, we obtain that the structure $\hat{\omega}$ is also a model of the knowledge base $\Delta KB^C_n(m_{E^*}, n_{E^*,R^*})$. On the other hand, since the state ω''_{E^*,R^*} is legal, given the hypothesis of repairability of the service enactment, and it is embedded into $\hat{\omega}$, by Theorem 3, we can conclude that also:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W})$$

But, since R^* is the minimal-size repair having such a property given E^*, we can also conclude that:

$$\hat{\omega} \not\models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,R'})$$

for every $R' \in \hat{R}^k_S$, where $k = \|R^*\|$.

In other words, we have shown that:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,R^*}) \\ \wedge \bigwedge_{R' \in \hat{R}^k_S} \neg\left(\tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,R'})\right) \wedge KB^R(n_{E^*,R^*})$$

where $k = \|R^*\|$. For any other repair R, given the effect E^*, we have that at least one of the following conditions hold:

- the repair is not effective and the obtained state is not legal ($\hat{\omega} \not\models KB^R(n_{E^*,R})$;
- the repair is not consistent ($\hat{\omega} \not\models \Delta KB^C_n(m_{E^*}, n_{E^*,R})$);
- the repair is effective and consistent, but its size is greater than $\|R^*\|$.

So we have proved that there exists at least a conjunct in $\eta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ that is satisfied in $\hat{\omega}$. From this contradiction we prove the first part of the theorem.

Now assume that the implication is satisfied, but the pair goal/service is not weakly and uniformly adequate given the binding schema, the repair family and world constraint. It means that there is at least an enactment where it is not possible to select a repairable effect E^* and an effective repair R^* s.t. the final state ω''_{E^*,R^*} is legal and enforces client requirements, despite the initial state was consistent with client commitments.

Let ω be the initial state and $\sigma_{\mathbf{X}}$ and $\sigma_{\mathbf{U}}$ be resp. the service input variable assignment and the goal parameter assignment. Since we are employing the binding schema B, the assignment to variables in $\mathbf{X}$ is functional depending upon the goal instantiation.

Considering also other possible enactments from the given initial condition, one for each possible service outcomes, and any repair $R \in R_S$, employing the mapping function $\mu^{\mathbf{R}}$ we can embed all them into a structure $\hat{\omega}$.

Since the service is accessible and we are considering a valid enactment, according to Theorem 45, the constructed interpretation $\hat{\omega}$ is a model of $nKB^U_{m_E}$ for every $E \in \mathcal{E}$. Now we consider the definition of repair insert and update set, and also of repaired extension of role and concept names: applying the Corollary 13, we can also conclude that constraints of $\Delta KB^R_n(m_E, n_{E,R})$ are satisfied for every $E \in \mathcal{E}$ and for every repair $R \in R_S$. Moreover, since we are assuming also that the binding schema is enforced and that user commitment constraints are satisfied in ω, so applying Corollary 14 and Lemma 40, we can easily check that implication antecedents are satisfied.

Since we have assumed that the implication holds, we can conclude that:

$$\hat{\omega} \models \bigvee_{E \in \mathcal{E}} \bigvee_{k=0}^{s} \bigvee_{R \in R_S^k} \tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R})$$
$$\wedge \bigwedge_{R' \in \hat{R}_S^k} \neg \left(\tau_{n_{E,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R'}) \right) \wedge KB^R(n_{E,R})$$

Let E^* and R^* be a pair s.t. the corresponding conjunct in the previous equation is satisfied in $\hat{\omega}$. According to Theorem 3, since:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R})$$

and the fact that the state ω''_{E^*,R^*} is embedded into $\hat{\omega}$, we can conclude that the latter is a legal world state, hence the enactment is admissible. Moreover, since for any other repair R' s.t. $R' \in \hat{R}_S^k$, where $K = \|R^*\|$, we have that:

$$\hat{\omega} \not\models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})$$

and we can also conclude that R^* must be one of minimal-size repair in R_S that can be actually applied to the effect E^*. Otherwise, the algorithm in Figure 4.5 had selected another consistent repair R' s.t. $\|R'\| < \|R^*\|$ and $\omega_{E^*,R'}$ is legal world state w.r.t. $\mathcal{W}$ instead of R^*, and according to Theorems 3 and 50, we have obtained that:

$$\hat{\omega} \models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})$$

We have, hence, shown that the effect E^* combined with the repair R^* can achieve a legal world state. But since the conjunct is satisfied in $\hat{\omega}$ we can also conclude that:

$$\hat{\omega} \models KB^R(m_{E^*,R^*})$$

Now, applying Corollary 15 from $\hat{\omega}$ to ω''_{E^*,R^*} on $\mathcal{R}$ using n_{E^*,R^*}, we conclude that also goal requirement conditions hold in ω''_{E^*,R^*}, obtaining a contradiction from which follows the claim. □

Theorem 69. *A consistent, accessible and repairable non-deterministic e-service S is strongly and non-uniformly adequate to an admissible goal G given a (weakly) consistent binding schema B w.r.t. a world specification $\mathcal{W}$ and a normal repair family R_S, iff the following formula is satisfiable:*

$$\bigwedge_{E \in \mathcal{E}} \left(\Delta KB_n^I(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB_c^U(m_E) \wedge \bigwedge_{R \in R_S} \Delta KB_n^R(m_E, n_{E,R}) \right)$$
$$\wedge KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S} \qquad (6.6)$$

where m_E and $n_{E,R}$ are the name mapping functions for the domain and the instantiation variable names defined for every effect $E \in \mathcal{E}$ and for every repair $R \in R_S$.

Proof. We assume that the formula is consistent, but that the given service is not non-uniformly and strongly adequate to the goal. In other words, it means that does not exist any legal world state and variable assignment s.t. the user commitments are satisfied, the service is accessible, and the goal requirement constraints are satisfied for every legal enactment from the given initial invocation context, applying the adequate (minimal) repair according to algorithm in Figure 4.5.

Let $\hat{\omega}$ be a model of the formula in Eq. 6.6 (it is assumed consistent, so such a structure must exist). Applying Lemma 35 for every distinct service effect $E \in \mathcal{E}$ and for every repair $R \in R_S$, we obtain a set of tuples, each representing a possible enactment, possibly legal, from a state ω using the input assignment $\sigma_{\mathbf{X}}$, according to Theorem 51. In fact, applying the projection function π_n^R for every service effect $E \in \mathcal{E}$ and for every considered repair $R \in R_S$ we obtain a set of quintuples of the form:

$$\langle \omega, \omega'_E, \omega''_{E,R}, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$$

By definition they agree on 4 components out of 5, since they are built upon shared names (e.g., $\mathbf{A}$, $m(\mathbf{P})$, and so on). Since, for every effect $E \in \mathcal{E}$ the structure is s.t. $\hat{\omega} \models nKB^U_{m_E}$, any quadruple $\langle \omega, \omega'_E, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$ represents a valid service enactment by Theorem 46 and since the service is assumed repairable, applying Theorem 52 we can also conclude that:

- the initial and final state are legal w.r.t. $\mathcal{W}$, since they are embedded into a structure that satisfies both translated constraint sets;
- input assignment depends on the goal instantiation using the binding schema (Lemma 40);
- goal client commitment constraints are satisfied in the initial state (Corollary 14);
- for every possible service invocation always exists at least pair effect/repair having the repair drawn from R_S, that leads the system to a new legal state w.r.t. $\mathcal{W}$.

Moreover, since such a service is activated using the binding schema B and binding schema is assumed to be valid, while the service is assumed to be repairable, there must exist a non-empty set of legal service outcomes (which means service effect that given the invocation context lead the system into a legal world state possibly applying a repair in R_S).

Given the hypothesis of non-adequacy of the pair goal/service, must exist at least an effect $E^* \in \mathcal{E}$ and a repair $R^* \in R_S$, s.t. R^* is possibly selected as minimal repair in order to enforce world constraint $\mathcal{W}$ in the state resulting from the application of the given effect and in such repaired final world state goal requirement constraints do not hold. According to the definition of repairable service and Theorem 52, whose implication antecedents in Eq. 5.43 are subsumed by formula in Eq. 6.6, we can conclude that:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,R^*})$$

Since Lemma 21, the selected repair is also the minimal-size repair that is able to enforce domain constraint upon service effects without retracting them. In other words, for any other repair R' s.t. $\|R'\| < \|R^*\|$, it follows that:

$$\hat{\omega} \not\models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})$$

It means that antecedents of material implication in $\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ are satisfied for the effect E^* and the repair R^*, hence applying the implication we can also conclude that:

$$\hat{\omega} \models KB^R(n_{E^*,R^*})$$

Now, applying Corollary 15 from $\hat{\omega}$ to ω^* on $\mathcal{R}$ using n_{E^*,R^*}, we conclude that also goal requirement conditions hold in ω^*, obtaining a contradiction.

Now we assume that the pair goal/service is strongly and non-uniformly adequate but the given formula is not satisfiable. Let ω be a legal world state and $\sigma_{\mathbf{U}}$ be a goal instantiation assignment s.t.:

- the service is activated using the provided binding schema applied to goal instantiation assignment;
- the service is accessible given the initial state and input assignment.

Applying the definition of service effect, we can compute a set possible enactment for every given non-deterministic outcome and repair in the provided family, from the initial state and using the mapping function embed all them into a structure $\hat{\omega}$.

Since the service is accessible and we are considering a valid enactment, according to Theorem 45, the constructed interpretation $\hat{\omega}$ is a model of $nKB_{m_E}^U$ for every $E \in \mathcal{E}$. Now we consider the definition of repair insert and update set, and also of repaired extension of role and concept names: applying the Corollary 13, we can also conclude that constraints of $\Delta KB_n^R(m_E, n_{E,R})$ are satisfied for every $E \in \mathcal{E}$ and for every repair $R \in R_S$.

Applying Theorems 17, 50, 54, and Lemma 40, we can, without difficulty, conclude that the structure is s.t.:

$$\hat{\omega} \models KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B$$
$$\wedge \bigwedge_{E\in\mathcal{E}} \left(\Delta KB_n^I(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB_c^U(m_E) \wedge \bigwedge_{R\in R_S} \Delta KB_n^R(m_E, n_{E,R}) \right)$$

In order to complete the argumentation, we need to show that the remaining conjunct is also satisfied in $\hat{\omega}$.

According to the definition of repair strategy (see Figure 4.5), the selected repair is always the minimum w.r.t. size, so let E^* be the considered repairable effect, let R^* be the minimal-size repair that allows a consistent enactment. So, applying Theorem 50, if follows that the structure $\hat{\omega}$ is also a model of the knowledge base $\Delta KB_n^C(m_{E^*}, n_{E^*,R^*})$. On the other hand, since the state ω''_{E^*,R^*} is legal, given the hypothesis of repairability of the service enactment, and it is embedded into $\hat{\omega}$, by Theorem 3, we can conclude that also:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W})$$

But, since R^* is the minimal-size repair having such a property given E^*, we can also conclude that:

$$\hat{\omega} \not\models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})$$

for every $R' \in \hat{R}_S^k$, where $k = \|R^*\|$.

Given the hypothesis of strong and non-uniform service adequacy we need also conclude that in ω''_{E^*,R^*} client requirement constraints are satisfied, and we obtain that:

$$\hat{\omega} \models KB^R(n_{E^*,R^*})$$

considering Lemma 32 and Corollary 15, In other words, we have shown that:

$$\begin{aligned} &\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R^*}) \\ \wedge \bigwedge_{R' \in \hat{R}_S^k} \neg &\left(\tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})\right) \wedge KB^R(n_{E^*,R^*}) \end{aligned}$$

where $k = \|R^*\|$. For any other repair R, given the effect E^*, we have that at least one of the following conditions hold:

- the repair is not effective and the obtained state is not legal ($\hat{\omega} \not\models KB^R(n_{E^*,R})$;
- the repair is not consistent ($\hat{\omega} \not\models \Delta KB_n^C(m_{E^*}, n_{E^*,R})$);
- the repair is effective and consistent, but its size is greater than $\|R^*\|$.

It means that for every repairable effect $E^* \in \mathcal{E}$, the corresponding material implication in $\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ holds. Moreover, if an effect E is not repairable, there does not exist any repair in the given family such that it is consistent and it leads the system to a legal state, hence the material implication antecedents are never satisfied, so the implication holds. We have shown that for every $E \in \mathcal{E}$ the structure $\hat{\omega}$ satisfies the material implication, or, in other words, that:

$$\begin{aligned} \hat{\omega} \models \bigwedge_{E \in \mathcal{E}} \bigwedge_{k=0}^{s} \bigwedge_{R \in R_S^k} &\tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R}) \\ \wedge \bigwedge_{R' \in \hat{R}_S^k} \neg \left(\tau_{n_{E,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R'})\right) &\rightarrow KB^R(n_{E,R}) \end{aligned}$$

From this contradiction we prove the second part of the theorem. □

Theorem 70. *A consistent, accessible and repairable non-deterministic e-service S is weakly and non-uniformly adequate to an admissible goal G given a (weakly) consistent binding schema B w.r.t. a world specification $\mathcal{W}$ and a normal repair family R_S, iff the following knowledge base is satisfiable:*

$$\begin{aligned} &\left(\Delta KB_n^I(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB_c^U(m_E) \wedge \bigwedge_{R \in R_S} \Delta KB_n^R(m_E, n_{E,R})\right) \\ &\qquad \wedge KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B \wedge \eta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S} \end{aligned} \tag{6.7}$$

where m_E and $n_{E,R}$ are the name mapping functions for the domain and the instantiation variable names defined for every effect $E \in \mathcal{E}$ and for every repair $R \in R_S$.

Proof. We assume that the formula is consistent, but that the given service is not non-uniformly and weakly adequate to the goal. In other words, it means that does not exist any legal world state and variable assignment s.t. the user commitments are satisfied, the service is accessible, and the goal requirement constraints are satisfied in at least legal enactment from the given initial invocation context, applying the adequate (minimal) repair according to the proposed repair strategy.

As in the previous proof, let $\hat{\omega}$ be a model of the formula in Eq. 6.7 (it is assumed consistent, so such a structure must exist). Applying Lemma 35 for every distinct service effect $E \in \mathcal{E}$ and for every repair $R \in R_S$, we obtain a set of tuples, each representing a possible enactment, possibly legal, from a state ω using the input assignment $\sigma_{\mathbf{X}}$, according to Theorem 51. In fact, applying the projection function π_n^R for every service effect $E \in \mathcal{E}$ and for every considered repair $R \in R_S$, we obtain a set of quintuples of the form:

$$\langle \omega, \omega'_E, \omega''_{E,R}, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$$

By definition they agree on 4 components out of 5, since they are built upon shared names (e.g., $\mathbf{A}$, $m(\mathbf{P})$, and so on). Since, for every effect $E \in \mathcal{E}$ the structure is s.t. $\hat{\omega} \models nKB^U_{m_E}$, any quadruple $\langle \omega, \omega'_E, \sigma_{\mathbf{X}}, (\sigma'_{\mathbf{Y}})_E \rangle$ represents a valid service enactment by Theorem 46 and since the service is assumed repairable, applying Theorem 52 we can also conclude that:

- the initial and final state are legal w.r.t. $\mathcal{W}$, since they are embedded into a structure that satisfies both translated constraint sets;
- input assignment depends on the goal instantiation using the binding schema (Lemma 40);
- goal client commitment constraints are satisfied in the initial state (Corollary 14);
- for every possible service invocation always exists at least pair effect/repair having the repair drawn from R_S, that leads the system to a new legal state w.r.t. $\mathcal{W}$.

Moreover, since such a service is activated using the binding schema B and binding schema is assumed to be valid, while the service is assumed to be repairable, there must exist a non-empty set of legal service outcomes (which means service effect that given the invocation context lead the system into a legal world state possibly applying a repair in R_S).

Since we have assumed that the formula holds in $\hat{\omega}$, we can conclude that:

$$\hat{\omega} \models \bigvee_{E \in \mathcal{E}} \bigvee_{k=0}^{s} \bigvee_{R \in R_S^k} \tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R})$$
$$\wedge \bigwedge_{R' \in \hat{R}_S^k} \neg \left(\tau_{n_{E,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R'}) \right) \wedge KB^R(n_{E,R})$$

Let E^* and R^* be a pair s.t. the corresponding conjunct in the previous equation is satisfied in $\hat{\omega}$. According to Theorem 3, since:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R})$$

and the fact that the state ω''_{E^*,R^*} is embedded into $\hat{\omega}$, we can conclude that the latter is a legal world state, hence the enactment is admissible. Moreover, since for any other repair R' s.t. $R' \in \hat{R}^k_S$, where $K = \|R^*\|$, we have that:

$$\hat{\omega} \not\models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,R'})$$

and, hence, we conclude that R^* must be one of minimal-size repair in R_S that can be actually applied to the effect E^*. Otherwise, the repair algorithm had selected another consistent repair R' s.t. $\|R'\| < \|R^*\|$ and $\omega_{E^*,R'}$ is legal world state w.r.t. $\mathcal{W}$ instead of R^*, and according to Theorems 3 and 50, we have obtained that:

$$\hat{\omega} \models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,R'})$$

We have, hence, shown that the effect E^* combined with the repair R^* can achieve a legal world state. But since the conjunct is satisfied in $\hat{\omega}$ we can also conclude that:

$$\hat{\omega} \models KB^R(m_{E^*,R^*})$$

Now, applying Corollary 15 from $\hat{\omega}$ to ω''_{E^*,R^*} on $\mathcal{R}$ using n_{E^*,R^*}, we conclude that also goal requirement conditions hold in ω''_{E^*,R^*}, contradicting the hypothesis of non-adequacy.

Now we assume that the pair goal/service is weakly and non-uniformly adequate but the given formula is not satisfiable. Let ω be a legal world state and $\sigma_{\mathbf{U}}$ be a goal instantiation assignment s.t.:

- the service is activated using the provided binding schema applied to goal instantiation assignment;
- the service is accessible given the initial state and input assignment.

Applying the definition of service effect, we can compute a set possible enactment for every given non-deterministic outcome and repair in the provided family, from the initial state and using the mapping function embed all them into a structure $\hat{\omega}$.

Since the service is accessible and we are considering a valid enactment, according to Theorem 45, the constructed interpretation $\hat{\omega}$ is a model of $nKB^U_{m_E}$ for every $E \in \mathcal{E}$. Now we consider the definition of repair insert and update set, and also of repaired extension of role and concept names: applying the Corollary 13, we can also conclude that constraints of $\Delta KB^R_n(m_E, n_{E,R})$ are satisfied for every $E \in \mathcal{E}$ and for every repair $R \in R_S$.

Applying Theorems 17, 50, 54, and Lemma 40, we can easily conclude that the structure is s.t.:

$$\hat{\omega} \models KB^P \wedge \tau(\mathcal{W}) \wedge KB^H \wedge \Delta KB^B$$
$$\wedge \bigwedge_{E\in\mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E) \wedge \bigwedge_{R\in R_S} \Delta KB^R_n(m_E, n_{E,R}) \right)$$

In order to complete the proof, we have to show that the remaining conjunct is also satisfied in $\hat{\omega}$.

According to the definition of repair strategy (see Figure 4.5), the selected repair is always the minimum w.r.t. size, so let E^* be the considered repairable effect, let R^* be the minimal-size repair that allows a consistent enactment. So, applying Theorem 50, we obtain that the structure $\hat{\omega}$ is also a model of the knowledge base $\Delta KB^C_n(m_{E^*}, n_{E^*,R^*})$. On the other hand, since the state

ω''_{E^*,R^*} is legal, given the hypothesis of repairability of the service enactment, and it is embedded into $\hat{\omega}$, by Theorem 3, we can conclude that also:

$$\hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W})$$

But, since R^* is the minimal-size repair having such a property given E^*, we can also conclude that:

$$\hat{\omega} \not\models \tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})$$

for every $R' \in \hat{R}_S^k$, where $k = \|R^*\|$.

In other words, we have shown that:

$$\begin{aligned} \hat{\omega} \models \tau_{n_{E^*,R^*}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R^*}) \\ \wedge \bigwedge_{R' \in \hat{R}_S^k} \neg \left(\tau_{n_{E^*,R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{E^*}, n_{E^*,R'})\right) \wedge KB^R(n_{E^*,R^*}) \end{aligned}$$

where $k = \|R^*\|$. For any other repair R, given the effect E^*, we have that at least one of the following conditions hold:

- the repair is not effective and the obtained state is not legal ($\hat{\omega} \not\models KB^R(n_{E^*,R})$;
- the repair is not consistent ($\hat{\omega} \not\models \Delta KB_n^C(m_{E^*}, n_{E^*,R})$);
- the repair is effective and consistent, but its size is greater than $\|R^*\|$.

So we have proved that there exists at least a conjunct in $\eta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ that is satisfied in $\hat{\omega}$. From this contradiction follows the second part of the claim. □

According to Theorems 26 and 27, the previous formula in Eq. 6.2 has a size that is double-exponential in the size of the problem settings. Despite in this form it can be easily employed to proof the following results, it can be rewritten in a more compact equivalent form, given the introduction of a suitable set of flag propositional variable names[2]:

$$\begin{aligned} \theta'_{\mathcal{E},\mathcal{R},\mathcal{W},R_S} \triangleq \bigwedge_{E \in \mathcal{E}} \bigwedge_{k=0}^{s} \left(\iota_{E,k} \leftrightarrow \bigwedge_{R \in R_S^k} \neg \left(\tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R})\right)\right) \\ \wedge \left(\bigwedge_{R \in R_S^k} \tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R}) \wedge \bigwedge_{j=0}^{k-1} \iota_{E,j} \rightarrow KB^R(n_{E,R})\right) \end{aligned}$$

Lemma 42. *The formulas $\theta'_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ and $\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ are equi-satisfiable*

Proof. Let ω' be a model of $\theta'_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$, it is clearly also a model of the formula $\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$. On the other side, given a model ω of the formula $\theta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ that does not interpret any name like $\iota_{E,R}$ (otherwise such a portion of the interpretation structure can be simply thrown out, since they does not play any role in the evaluation of the given formula), it can be extended to a structure ω', which interprets also the propositional variables ι, s.t.:

[2]This formula can be also rewritten in terms of DL, once we have replaced propositional variables names with new concept names interpreted as empty set (false assignment) or the whole interpretation domain (true assignment), but for the sake of simplicity we adopt the first-order compact form only to show complexity result in the following

- the interpretation domain is the same;
- the interpretation of every concept, role and object name is preserved;
- every variables $\iota_{E,k}$ is interpreted as $\top$ iff:

$$\omega \models \bigwedge_{R \in R_S^k} \neg \left(\tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R})\right)$$

  otherwise it is interpreted as $\perp$.

Such a structure is a model of the formula $\theta'_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$. □

We can apply a similar encoding also to the formula in Eq. 6.3, that, as in the previous case, is double exponentially long in the size of the problem setting, obtaining an exponentially compact equivalent form:

$$\eta'_{\mathcal{E},\mathcal{R},\mathcal{W},R_S} \triangleq \bigwedge_{E \in \mathcal{E}} \bigwedge_{k=0}^{s} \left(\iota_{E,k} \leftrightarrow \bigwedge_{R \in R_S^k} \neg \left(\tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R})\right)\right)$$
$$\wedge \bigvee_{E \in \mathcal{E}} \bigvee_{k=0}^{s} \bigvee_{R \in R_S^k} \left(\tau_{n_{E,R}}(\mathcal{W}) \wedge \Delta KB_n^C(m_E, n_{E,R}) \wedge \bigwedge_{j=0}^{k-1} \iota_{E,j} \wedge \Delta KB_n^C(m_E, n_{E,R})\right)$$

We can also obtain a claim analogous to Lemma 42.

Lemma 43. *The formulas $\eta'_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ and $\eta_{\mathcal{E},\mathcal{R},\mathcal{W},R_S}$ are equi-satisfiable*

Now, we can employ such claims to provide narrower complexity results for the goal/service adequacy checking problems devised in the previous.

Theorem 71. *Given a world specification $\mathcal{W}$, a consistent, accessible and repairable non-deterministic e-service S, an admissible goal G, a (weakly) consistent binding schema B, the problem of checking if the service is strongly (resp. weakly) and uniformly adequate to achieve the goal considering all possible simple repair is in* coNEEXP.

Proof. Given Theorem 67 (resp. Theorem 68), the decision problem can be reduced to an implication checking, that, according to Cor. 1, is in coNEXP. As done in previous cases regarding the effect repair, we are assuming that the input size is defined in terms of length of domain, world and service specifications (e.g., number of names or complexity of constraints/preconditions, etc.).

Given Lemma 42 (resp. Lemma 43) we can reduce the checking of the implication in Eq. 6.4 (resp. Eq. 6.5) to an equivalent, but more compact, one that use the formula $\theta'_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$ instead of $\theta_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$ (resp. $\eta'_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$ instead of $\eta_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$). Moreover, the length of the former formula is linearly bounded by the number of service effects, the maximum simple repair size (generally denoted ad s) and the size of the simple repair family (R_S). According to Theorem 26, there exists at most a number of distinct simple repair that is exponential in the size of the input, while given Theorem 27, the the maximum size of simple repair is linearly bounded by the input size.

In other words, the size of the formula $\theta'_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$ is exponential in the size of the input:

$$\left\|\theta'_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}\right\| \in \mathcal{O}\left(2^{p(n)}\right)$$

where n be the size of the instance of the problem, and $p(n)$ a suitable polynomial function in n.

In fact, considering the complementary problem $\overline{\mathsf{StrongUnifRepAdequacy}}$, since we have that:

$$\mathsf{SAT}\mathcal{C}^2 \in \mathsf{NEXP} = \bigcup_k \mathsf{NTIME}\left(2^{m^k}\right)$$

where $m = \mathcal{O}\left(2^{p(n)}\right)$. Consequently we have that:

$$\overline{\mathsf{StrongUnifRepAdequacy}} \in \bigcup_k \mathsf{NTIME}\left(2^{2^{p(n)^k}}\right) \subseteq \bigcup_k \mathsf{NTIME}\left(2^{2^{n^k}}\right) = \mathsf{NEEXP}$$

and, hence, that $\mathsf{StrongUnifRepAdequacy} \in \mathsf{coNEEXP}$.

The proof for the problem $\mathsf{WeakUnifRepAdequacy}$ is analogous. □

Theorem 72. *Given a world specification $\mathcal{W}$, a consistent, accessible and repairable non-deterministic e-service S, an admissible goal G, a (weakly) consistent binding schema B, the problem of checking if the service is strongly (resp. weakly) and non-uniformly adequate to achieve the goal considering all possible simple repair is in* NEEXP.

Proof. This claim essentially relies on the same reduction employed in the proof of Theorem 71, that allows to encode the decision problem resulting from Theorem 69 (resp. Theorem 70) into a $\mathcal{C}^2$ sentence that is exponentially long in the size of the problem instance, replacing $\theta_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$ with $\theta'_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$ in Eq. 6.6 (resp., $\eta_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$ with $\eta'_{\mathcal{E}_S,\mathcal{R}_G,\mathcal{W},R_S}$ in Eq. 6.7). The result follows from the observation, already used for the other complexity results, that satisfiability check problem for this language is solvable non-deterministic exponential time. □

The results about complexity of service/goal adequacy problems are summarized in Table 6.2, as for the correctness analysis, the introduction of a repair strategy, required to deal with partially-specified services, induces at most an exponential blow-up.

	Non-uniform	Uniform
Without repair	NEXP	coNEXP
With repair	NEEXP	coNEEXP

Table 6.2: Verification complexity upper-bounds for goal adequacy properties

6.2 Service Replaceability

Strictly related with the notion of service equivalence, the analysis of service replaceability aims at evaluating when a service can act as replacement of another one (i.e., a faulty provider).

Given two available services S_O and S_N, we are interested into the definition of an assessment criterion to check whether it is possible to employ the service S_N instead of S_O or, in other words, whether it is possible to replace the service S_O with S_N. In order to consistently achieve the service replacement, we need to impose some constraints:

- the replacing service S_N must be activable in every condition in which S_O is;
- in every condition in which S_N is activable instead of S_O, it must be able to achieve the "same" effects.

According to the adopted approach, the analysis of service replaceability is carried out considering the service behavior described in the contract specification. In other words, we require that a service can be replaced by another service behaving the same at least when it is invoked instead of the original one: the main issue is the definition of the behavior compatibility.

Definition 93 (Absolute service replaceability)**.** *Given two services S_N and S_O, we say that S_N can absolutely replace the service S_O iff for every valid activation of the latter there exists a valid activation of the former from the same starting state that leads to the same enactment, in terms of successor states.*

This condition is quite strong to fulfill since it requires that two services act exactly in the same way (they induce the same enactment relation) every time the service that has to be replaced is activable. Moreover the definition does not provide, as in the case of execution goal analysis, any tool to compute the invocation adapter (the component that is able to accordingly map input parameters). In the same way, we can also provide a strong notion of service equivalence.

Definition 94 (Service equivalence)**.** *Given two services S_N and S_O, we say they are strongly equivalent iff they are accessible in the same conditions and for every valid activation of the latter there exists a valid activation of the former from the same starting state that leads to the same enactment, in terms of successor states, and vice-versa.*

Proposition 2. *Given two mutually replaceable services, they are (strong) equivalent or equi-potent.*

6.2.1 Interface adapters

Since services are activated according to a parameter signature, even not intended as the usual operation signature[3], we need to analyze the replaceability w.r.t. parameter bindings. There are two main assumptions regarding the mismatching between considered services interfaces:

1. the mismatch is ignored, a service can replace another one even their interfaces are different and not matchable;

[3]Please see [MPC01] for an interface-only replacement approach, that allows stateful complex services.

2. the replaceability is considered only if an interface-adapter exists.

In the first case is easier find a service replacement, but the interface mismatching could make harder, even not tractable, the implementation of an automatic replacement mechanism. Conversely, in the second case, it deeply relies on the specification of the interface-adapter class and the replacement detection could be not complete (the required adapter could not be in the considered class), but this is the same situation of the service/goal binding, where also a feasible, but tractable approach has been proposed[4].

According to these observations, we now introduce a formalization attempt of an adapter that is able to mediate the activation of a service employed in lieu of the original one.

Definition 95 (Interface adapter)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a world specification $\mathcal{W}$, and two sets of variable names $\boldsymbol{X}_{S_N}$ and $\boldsymbol{X}_{S_O}$ an interface adapter is a binding schema A s.t. it assigns to each $X \in \boldsymbol{X}_{S_N}$ an access function w.r.t. $\mathcal{W}$, parameterized over $\boldsymbol{X}_{S_O}$.*

The definition of interface adapter is based on the same model adopted for access function and query in previous sections. Also the semantics of this constructs is similar.

Definition 96 (Interface adapter evaluation)**.** *Let S_O and S_N be two services. Given a world state ω, an assignment $\sigma_{\boldsymbol{X}_{S_O}}$ for the former and an interface adapter A, the evaluation of I is an assignment $\sigma_{\boldsymbol{X}_{S_N}}$ s.t. each name $X \in \boldsymbol{X}_{S_N}$ is assigned to the corresponding query evaluated in ω:*

$$\sigma_{\boldsymbol{X}_{S_N}}(X) = A(X)^{\omega}(\sigma_{\boldsymbol{X}_{S_O}})$$

The definition of access function is too weak since it allows for partial functions, so we need to introduce an additional constraint, as previously done for goal binding schema.

Definition 97 (Valid interface adapter)**.** *Given a world specification $\mathcal{W}$ and a service S, an interface adapter A is valid iff, for every legal world state and for every enactment of the service, and corresponding assignments, its evaluation assigns a value to each variable.*

Given the previous definitions, we can, without difficulty, extend claims provided in Section 6.1 regarding binding schemas, because the interface adapter is a special case of binding, defined over the input variable sets of two services.

In the following, we have to adequate formal tools provided so far: in particular, we need to extend the definition of structure embedding $\rightsquigarrow$ (and related concepts as mapping and projection functions μ and π) so that two service definitions at the same time are kept into account. It can be, without difficulty, achieved providing adequate name mapping functions that are always able to map constructs of distinct services to distinct names or, in other words, defining two mapping functions (unless differently stated) with disjoint domains.

[4]We remember that we are considering interface specification at conceptual level, not at implementation level. Two services are the same interface if they require and provide the same information, despite the actual component interfaces (e.g., IDL, WSDL, etc.) are slightly different, consequently we can assume that a large number of interface adapters is a kind of variable renaming.

Remark 37. *As done in the previous, w.l.o.g. we also assume that variable namespaces of different sets/services are mutually disjoint. Moreover, we assume that given services are always accessible and legal (which means valid or repairable, depending on the fact that we are keeping into account also repair strategies).*

As first step, in order to evaluate if a new service S_N can replace a service S_O using an adapter A, we need to check whether the accessibility is preserved, or in other words, if it is allowed to access the new service using the adapter in a state from which the old one is accessible.

Definition 98 (Accessibility preserving interface adapter)**.** *Given a world specification $\mathcal{W}$, an interface adapter A for a pair of accessible services $\langle S_N, S_O\rangle$ preserve the accessibility property iff, for every state and assignment s.t. the service S_O is accessible also the service S_N is accessible employing the adapter.*

Remark 38. *For the sake of clarity we are now ignoring the constraint related to the availability of enough room in the interpretation domain for the possible instantiation of new objects by the given services (since the actual number can depend on conditional/non-deterministic behavior). So we are assuming that in the instantiation of the axiom schema $\Delta KB_n(\mathsf{spy}, \mathsf{aux})$ the parameter n is assigned to the size of the largest instantiation set provided. This approximation will be removed in the following.*

Theorem 73. *Given a consistent world specification $\mathcal{W}$, two accessible services S_N and S_O, and a valid interface adapter A for $\boldsymbol{X}_{S_N}$ given $\boldsymbol{X}_{S_O}$, S_N is an accessibility preserving replacement for the service S_O iff the following implication hold:*

$$\tau(\mathcal{W}) \wedge KB^{P_O} \wedge \Delta KB^B \models KB^{P_N}$$

where KB^{P_O} and KB^{P_N} are the precondition knowledge bases instantiated on service S_O and S_N respectively and ΔKB^B is instantiated on the binding schema A.

Proof. We assume that the candidate replacing service S_N preserves the accessibility of S_O w.r.t. the world specification, but that the implication does not hold. In other words, there must exist a structure $\tilde{\omega}$ s.t.:

$$\tilde{\omega} \models \tau(\mathcal{W}) \wedge KB^{P_O} \wedge \Delta KB^B$$
$$\tilde{\omega} \not\models KB^{P_N}$$

According to Theorems 11 and 17 (applied to S_O) we can build a pair $\omega, \sigma_{\mathbf{V}} = \pi^{\mathbf{V}}(\tilde{\omega})$ s.t.:

- ω is legal world state w.r.t. the specification $\mathcal{W}$;
- the replacing service S_O preconditions are satisfied in ω given the assignment $\sigma_{\mathbf{X}_{S_O}}$ obtained from restricting $\sigma_{\mathbf{V}}$ to names is $\mathbf{X}_{S_O}$, so S_O is actually accessible;
- the variables in $\mathbf{X}_{S_N}$ are consistently assigned to a domain element according to the evaluation of the binding schema (Lemma 40).

So, given the assumption on the preservation of the accessibility we can conclude that the service S_N is also accessible in ω using the input assignment obtained from the evaluation of binding functions. Consequently, applying Theorem 17 to S_N, we can conclude that $\tilde{\omega} \models KB^{P_N}$, resulting in a contradiction.

We now assume that, despite the implication holds, the accessibility property is not preserved given the world specification. It means that there exists a legal world state ω and a consistent assignment $\sigma_{\mathbf{X}_{S_O}}$ of the service S_O s.t.:

- the service preconditions are satisfied in ω;
- the evaluation of the interface adapter A (that is assumed to be valid) in this context provides a well-founded assignment $\sigma_{\mathbf{X}_{S_N}}$ for S_N input variables.

Since the hypothesis of non-preservation of the accessibility property we can also assume that the service S_N is not accessible in ω using the assignment $\sigma_{\mathbf{X}_{S_N}}$.

We can apply the mapping function $\mu^{\mathbf{V}}$ to the pair $\langle \omega, \sigma_{\mathbf{V}} \rangle$, obtaining a new interpretation structure $\tilde{\omega}$ that embeds such a pair and that, according to Theorems 10 and 17 (applied to S_O), is also s.t.:

$$\tilde{\omega} \models \tau(\mathcal{W}) \wedge KB^{P_O}$$

Moreover, given the definition of query evaluation, we can also conclude that for each $X \in \mathbf{X}_{S_N}$, we have that:

$$\tilde{\omega} \models X \equiv A(X)(\mathbf{X}_{S_O})$$

since we have assumed that $\sigma_{\mathbf{X}_{S_N}}(X) = A(X)^{\omega}(\sigma_{\mathbf{X}_{S_O}})$. It follows that $\tilde{\omega} \models \Delta KB^B$ and we can check that implication antecedents are satisfied and, since the implication is assumed to be verified, we can finally conclude that $\tilde{\omega} \models KB^{P_N}$. Applying Corollary 3 to service S_N we can now infer also that this service is also accessible in ω using $\sigma_{\mathbf{X}_{S_N}}$, obtaining a contradiction that proves the claim. $\square$

Theorem 74. *Given a consistent world specification $\mathcal{W}$, two accessible services S_N and S_O, and a valid interface adapter A for $\boldsymbol{X}_{S_N}$ given $\boldsymbol{X}_{S_O}$, the problem of checking whether S_N is an accessibility preserving replacement for the service S_O using A is in* coNEXP.

Proof. As done for other cases previously analyzed, we can solve the problem applying the property proved in Theorem 73, reducing it to an implication decision in $\mathcal{C}^2$ logics, hence we use the result of Proposition 1. Like other reductions it is also linear in the size of the input (number and length of axioms, preconditions and effects specifications). $\square$

6.2.2 Reachability-preserving replaceability

In the previous section, we have generally discussed about the notions of service functional equivalence and replaceability, providing also a formal definition of interface adapter. Now, we address the problem of evaluating the possibility of replace a service with another one w.r.t. the state reachability.

Given the general equivalence definition, we can easily notice that it is in some way too hard to fulfill to be useful. Thus, a weaker equivalence condition can be stated as the following: it simply ensures that the new service is able, at least, to reach any state previously reachable (and so to instantiate at least the same objects).

Definition 99 (Reachability preserving service replaceability)**.** *Given two services S_N and S_O, we say that S_N can replace the service S_O preserving the reachability iff for every valid activation of the latter there exists a valid activation of the former from the same starting state ω that leads to an enactment containing at least every final state included in the enactment of the latter (modulo the isomorphism relation).*

Roughly speaking, in order to check whether is a new service can replace the old one, we have to verify if for every possible enactment of the latter the former can possibly simulate it:

- instantiating the same number of new objects[5];
- enforcing the same update effects on the extension of the concept/role name.

Example 11. *Given services S and S_1 described in Examples 2 and 3, we can easily observe that, despite service S is always activable when S_1 is activable (the preconditions of the latter simply imply the ones of the former), given a valid, and accessibility preserving, interface adapter A defined as:*

$$A = \{x_1(x_1, x_2) = x_1, x_2(x_1, x_2) = x_2\}$$

the service S cannot replace the service S_1 preserving the reachability of the enactments, since, it does not change the registration of vehicles eventually owned by the service requestor.

On the other hand, we can also prove that the service S_1 can never replace the service S since a suitable interface adapter cannot exists: in fact we have explicitly asserted the existence of at least two distinct towns (see Table 4.1), hence we can easily build a counterexample. Moreover, removing this constraint, we cannot ensure the accessibility of the service itself given the observation of Example 3.

Example 12. *Given services S and S_1 described in Examples 2 and 3, now we consider the service S_2 defined as:*

$$\begin{aligned}
&\boldsymbol{X} = \{x_1, x_2\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x_1 \sqcap \mathsf{Citizen}\ \mathsf{and}\ x_2 \sqcap \mathsf{Town}\ \mathsf{and\ not}\ x_2 \sqcap \exists \mathsf{residentIn}^-.x_1\\
&E = \left\{-\mathsf{residentIn}(x_1, \exists\mathsf{residentIn}^-.x_1), +\mathsf{residentIn}(x_1, x_2)\right\}\\
&\quad \cup \left\{-\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists\mathsf{owner}.x_1, \exists\mathsf{residentIn}^-.x_1)\right\}\\
&\quad \cup \{+\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists\mathsf{owner}.x_1, x_2)\}
\end{aligned}$$

[5]Given the definition of reachability the basic definition implies a stronger constraint: that the service is able to instantiate exactly the same new objects, but given the isomorphism relation among successor state this constraint can be satisfied once the condition on the instance set size is verified.

It clearly can replace the service S_1 preserving the reachability (it is a direct generalization), but it cannot replace the service S, while S cannot replace S_2.

Moreover, given the non-deterministic service S_3 that merges behaviors of S and S_2:

$$\begin{aligned}
&\boldsymbol{X} = \{x_1, x_2\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x_1 \sqcap \mathsf{Citizen}\ \mathtt{and}\ x_2 \sqcap \mathsf{Town}\ \mathtt{and\ not}\ x_2 \sqcap \exists \mathsf{residentIn}^{-}.x\\
&\mathcal{E} = \{E_1, E_2\}
\end{aligned}$$

where:

$$\begin{aligned}
E_1 &= \big\{-\mathsf{residentIn}(x_1, \exists \mathsf{residentIn}^{-}.x_1), +\mathsf{residentIn}(x_1, x_2)\big\}\\
E_2 &= E_1 \cup \big\{-\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, \exists \mathsf{residentIn}^{-}.x_1)\big\}\\
&\quad \cup \{+\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, x_2)\}
\end{aligned}$$

we can verify that while neither S nor S_2 can replace S_3 preserving the reachability property, S_3 can replace both of them.

We now analyze such a property in order to provide an effective procedure to evaluate it relying on the formalization strategy devised so far. In the rest we employ the following notations for conciseness sake:

$$\begin{aligned}
\kappa_{O,N,p} &\triangleq \bigwedge_{i \in 1 \ldots \|\mathbf{Y}_O\|} [\mathbf{Y}_O]_i \equiv [\mathbf{Y}_N]_{p_i} \wedge \Delta KB_n^I(\mathsf{Top}', m_{N,p}) \wedge \Delta KB_c^U(m_{N,p})\\
\epsilon_{N,p} &\triangleq \bigwedge_{A \in \mathbf{A}} m(A) \equiv m_{N,p}(A) \wedge \bigwedge_{P \in \mathbf{P}} \forall x, y. m(P)(x,y) \leftrightarrow m_{N,p}(P)(x,y)\\
\lambda_{\mathcal{E},n} &\triangleq \{N | E \in \mathcal{E}, N \in \mathit{leaves}(E), \langle \mathbf{Y}_N, E_N\rangle = v(N), \|\mathbf{Y}_N\| = n\}
\end{aligned}$$

where O and N denotes two leaf service effects (with related renaming functions), p is permutation of size n and $\mathcal{E}$ is a non-deterministic service effect specification.

Theorem 75. *Given two accessible and valid services S_O and S_N, without redundant effect specification, and a valid interface adapter A w.r.t. the world specification $\mathcal{W}$, s.t. A preserve the accessibility, the service S_N can replace the service S_O using A preserving the reachability iff for each $n \in 0 \ldots n_{max}$, where n_{max} is the size of the largest instantiation set for the service S_O, for each $E \in \mathcal{E}_O$, and for each $O \in \mathit{leaves}(E)$ s.t. $\langle \boldsymbol{Y}_O, E_O\rangle = v(O)$ and $\| \boldsymbol{Y}_O \| = n$ the following implication holds:*

$$\begin{aligned}
&\tau(\mathcal{W}) \wedge KB^{P_O} \wedge \Delta KB^B \wedge \phi_O \wedge \Delta KB_n^I(\mathsf{Top}', m) \wedge \Delta KB_c^U(m) \wedge \tau_m(\mathcal{W})\\
&\qquad \wedge \bigwedge_{N \in \lambda_{\mathcal{E}_N, n}} \phi_N \rightarrow \bigwedge_{p \in nPn} \kappa_{O,N,p} \models \bigvee_{N \in \lambda_{\mathcal{E}_N, n}} \phi_N \wedge \bigvee_{p \in nPn} \epsilon_{N,p}
\end{aligned} \tag{6.8}$$

where:

- *ϕ_O and ϕ_N are resp. the branching path formulas associated with the leaves O and N of corresponding effect specification trees;*

- *nPn denotes the set of all permutations of size n of $\{1 \dots n\}$;*
- Top' *is the concept denoting the active domain of resulting system state;*
- *m is the name mapping function for the service S_O while $m_{N,p}$ is the name mapping function for a leaf effect N of the service S_N, considering the permutation p.*

The axiom schemas are instantiated accordingly to the specification of various service effects.

Proof. We start assuming that the devised implications hold given a pair of services S_O and S_N, the interface adapter and the world specification, but that the reachability is not preserved by the replacement. In other words, we are assuming that there exists at least a valid enactment (i.e., leading to a legal world state) $\langle \omega', \sigma'_{\mathbf{Y}_O} \rangle \in S_O(\omega, \sigma_{\mathbf{X}_O})$ of S_O s.t. there does not exist any consistent instantiation assignment $\sigma'_{\mathbf{Y}_N}$ s.t. $\langle \omega', \sigma'_{\mathbf{Y}_N} \rangle \in S_N(\omega, A^{\omega}(\sigma_{\mathbf{X}_O}))$, where $A^{\omega}(\sigma)$ denotes the evaluation of interface adapter given a world state and an input assignment. We can restrict the analysis to the leaf effect of S_N that enforces the given enactment[6], let O^* be such an effect and let n be the size of its instantiation set. Considering every possible leaf effect N of S_N, without keeping into account its branching condition, having an instantiation set of size n and any possible bijective mapping between the instantiation set of N and O^*, we built the pair $\langle \omega'_{N,p}, \sigma'_{\mathbf{Y}_{N,p}} \rangle$, where:

- $\sigma'_{\mathbf{Y}_{N,p}}$ is the instantiation assignment for N over the active domain $\Delta^{\omega'} \setminus \Delta^{\omega}$ applying the permutation $p \in nPn$ to the instantiation assignment of O^*[7];
- $\omega'_{N,p}$ is the target state obtained enforcing the effect N given the instantiation assignment and the input assignment obtained by the evaluation of the interface adapter to $\sigma_{\mathbf{X}_O}$ in ω.

Now, we build a structure $\hat{\omega}$ s.t.:

- the enactment of S_O is embedded into it according to the function μ_n (please refer to page 129) using the name mapping function m;
- each enactment of each leaf effect N of S_N having an instantiation set of size n obtained in the previous step applying a permutation p is embedded according to the function μ_n using the name mapping function $m_{N,p}$;
- the name Top' is interpreted as $\Delta^{\omega'}$.

Since the agreement on the interpretation of the result state active domain such a construction is well-founded. Given the hypothesis on the validity of service S_O and the enactment, applying Theorems 34, 35, 38, and 45 we can easily verify that the provided structure $\hat{\omega}$ is s.t.:

$$\hat{\omega} \models \tau(\mathcal{W}) \wedge KB^{P_O} \wedge \phi_{O^*} \wedge \Delta KB^{I}_{n}(\mathsf{Top}', m) \wedge \Delta KB^{U}_{c}(m) \wedge \tau_m(\mathcal{W})$$

[6]Generally speaking, there can exist multiple leaf effects, belonging to different trees $E \in \mathcal{E}_O$, given the non-reducing hypothesis, that can enforce the same effect, but they are equivalent for the argumentation of the proof.

[7]Other instantiation alternatives are irrelevant, since can not lead to a compatible target state.

Moreover, since we have applied the interface adapter A, assumed to be valid, according to Lemma 40, we can also conclude that:

$$\hat{\omega} \models \Delta KB^B$$

Considering each possible leaf effect N of S_N we can have two possible alternatives according to the fact that the branching precondition ϕ_N does or not hold in ω using the instantiation obtained by the application of the interface adapter A to $\sigma_{\mathbf{X}_O}$. Applying Theorem 35 (to S_N) in the latter case, since:

$$\hat{\omega} \not\models \phi_N$$

the corresponding material implication also holds $\hat{\omega}$. In the former case, considering each possible permutation p over n elements, we can, without difficulty, verify that the construction of $\hat{\omega}$ is s.t.:

$$\hat{\omega} \models \bigwedge_{i \in 1 \ldots \|\mathbf{Y}_{O^*}\|} [\mathbf{Y}_{O^*}]_i \equiv [\mathbf{Y}_N]_{p_i}$$

since for each instantiation variable of N we have assigned the value of an instantiation variable of the effect O^*. Applying Theorem 38, extended in order to keep into account the renaming also of the instantiation variables, as described at page 5.2.2, and Theorem 45 to each enactment of S_N we can conclude also that:

$$\hat{\omega} \models \Delta KB_n^I(\mathsf{Top}', m_{N,p}) \wedge \Delta KB_c^U(m_{N,p})$$

In other words, we have shown that antecedents of implication in Eq. 6.8 are satisfied in $\hat{\omega}$. Since, by hypothesis, such an implication holds, we can also conclude that:

$$\hat{\omega} \models \bigvee_{N \in \lambda_{\mathcal{E}_N, n}} \phi_N \wedge \bigvee_{p \in nPn} \epsilon_{N,p}$$

In other words, there must exist at least an effect N^* of S_N and a permutation $p^* \in nPn$ s.t. the interpretation of world state names (concepts and roles) agree. For each name $N \in \mathbf{A} \cup \mathbf{P}$ we have that:

$$m(N)^{\hat{\omega}} = m_{N^*,p^*}(N)^{\hat{\omega}}$$

but, given the properties of embedding functions μ, we have also that:

$$m(N)^{\hat{\omega}} = N^{\omega'}, N^{\omega'_{N^*,p^*}} = m_{N^*,p^*}(N)^{\hat{\omega}}$$

Given the agreement on the active domain and named objects, that are constantly interpreted in any structures, we can now conclude that $\omega'_{N^*,p^*} = \omega'$, but that means that also S_N can achieve the state ω', since ω'_{N^*,p^*} belongs to its enactment set, contradicting the hypothesis and proving the first part of the claim.

Now, we conversely assume that the service S_N can replace the service S_O using the adapter A preserving the reachability, but the implication does not hold. In other words, there must exist a structure $\hat{\omega}$ s.t. it is a model of the antecedent formula for some leaf effect O^* of S_O having an instantiation set of size n:

$$\tau(\mathcal{W}) \wedge KB^{P_O} \wedge \Delta KB^B \wedge \phi_{O^*} \wedge \Delta KB_n^I(\mathsf{Top}', m) \wedge \Delta KB_c^U(m) \wedge \tau_m(\mathcal{W}) \\ \wedge \bigwedge_{N \in \lambda_{\mathcal{E}_N, n}} \phi_N \rightarrow \bigwedge_{p \in nPn} \kappa_{O^*,N,p}$$

but there does not exist any pair $\langle N, p\rangle$, s.t.:

- N is a leaf effect of S_N having an instantiation set of size n;
- p is a permutation in nPn;
- $\hat{\omega}$ is a model of ϕ_N;
- $\hat{\omega}$ is a model of $\epsilon_{N,p}$;

According to Theorems 34, 35, 39, and 46 an enactment $\langle \omega', \sigma'_{\mathbf{Y}_O}\rangle \in S_O(\omega, \sigma_{\mathbf{X}_O})$ of S_O is embedded using m into $\hat{\omega}$ enforcing the effect O^*. Because the resulting state is legal the enactment can actually take place.

Since $\hat{\omega} \models KB^B$ we can conclude, according to Lemma 40, that we are considering the activations of service S_N obtained applying the interface adapter A to the assignment $\sigma_{\mathbf{X}_O}$ in ω. Moreover, we can project out from the structure $\hat{\omega}$ various world interpretations or extended interpretations using the adequate projection function π. We now consider all possible leaf effects N of S_N s.t. the instantiation set has a size of n. Also in this case we need to cope with two alternatives, depending on whether ϕ_N is satisfied in $\hat{\omega}$ or not. If $\hat{\omega} \models \phi_N$, according to Theorem 35, we can conclude that the effect N can be selected and possibly realized in ω by S_N, since its branching condition is satisfied in ω. Given the accessibility preserving property at least a suitable effect N^* must exist. Moreover, since we have also that:

$$\hat{\omega} \models \bigwedge_{p \in nPn} \kappa_{O^*,N,p}$$

Applying Theorems 39 and 46 we also have that it possible to project out from $\hat{\omega}$ using the name mapping $m_{N,p}$ $n!$ enactments $\langle \omega'_{N,p}, \sigma'_{\mathbf{Y}_{N,p}}\rangle$ of the effect N s.t.:

- each one instantiates the same objects, since the resulting active domain is the same $(\mathsf{Top}'^{\hat{\omega}} = \Delta^{\omega'} = \Delta^{\omega'_{N,p}})$;
- each one permutes in a specific way the instantiation set $\Delta^{\omega'} \setminus \Delta^{\omega}$ of S_O.

Since we are keeping into account all possible permutations of the n instantiation variables, we can also conclude that there is any other effective enactment of a leaf effect of S_N s.t. it has the same target state of S_O. Considering other leaf effects of S_N, since their branching conditions are not satisfied in ω, we can, without difficulty, see that they can never be selected by S_N and, thus, they can be ignored.

So far, we have built a set of possible enactments of S_N resulting from the application of the interface adapter A, embedded into the structure $\hat{\omega}$, and we have also shown that they are all the enactments relevant w.r.t. the reachability of ω'. In other words, since we have assumed that the reachability property of S_O is preserved using S_N applying A, we have that there is an element N^*, p^* of this set s.t. $\omega'_{N^*,p^*} = \omega'$. More specifically, we have that for each name $N \in \mathbf{A} \cup \mathbf{P}$ we have that:

$$N^{\omega'} = N^{\omega'_{N,p}}$$

but, given the properties of embedding functions μ, it follows also that:

$$m(N)^{\hat{\omega}} = N^{\omega'}, N^{\omega'_{N^*,p^*}} = m_{N^*,p^*}(N)^{\hat{\omega}}$$

and, hence, we can conclude that:

$$m(N)^{\hat{\omega}} = m_{N^*,p^*}(N)^{\hat{\omega}}$$

In other words, we have shown that:

$$\hat{\omega} \models \epsilon_{N^*,p^*}$$

Since, we have also that $\hat{\omega} \models \phi_{N^*}$, we can conclude that the consequent of implication in Eq. 6.8 is satisfied contradicting the hypothesis and proving the claim. □

The previous result enables an effective procedure to check whether a service replacement preserves the state reachability. Moreover we can also provide an upper bound of the problem complexity.

Theorem 76. *Given a consistent world specification $\mathcal{W}$, two accessible services S_N and S_O, and a valid interface adapter A for $\boldsymbol{X}_{S_N}$ given $\boldsymbol{X}_{S_O}$, the problem of checking whether S_N is a reachability preserving replacement for the service S_O using A is in* coNEEXP.

Proof. According to Theorem 75, the decision problem can be reduced to a linear number of entailment checks in $\mathcal{C}^2$: one for each leaf effect of S_O. Given the Cor. 1, the complexity of the entailment checking is coNEXP in the size of the formula.

We consider the complementary problem of checking whether a specific effect e of S_O having an instantiation set of size n_e can be preserved by the replacing.

$$\overline{\mathsf{Reachability_e}} \in \bigcup_k \mathsf{NTIME}\left(2^{f(n)^k}\right)$$

Since the formula has a number of terms that is factorial in n_e ($f(n) \approx g(n_e) \in \mathcal{O}(n_e!)$), we have that:

$$\overline{\mathsf{Reachability_e}} \in \bigcup_k \mathsf{NTIME}\left(2^{(n_e!)^k}\right)$$

Applying the Stirling's approximation we have so:

$$\overline{\mathsf{Reachability_e}} \in \bigcup_k \mathsf{NTIME}\left(2^{2^{{n_e}^k}}\right) \subseteq \bigcup_k \mathsf{NTIME}\left(2^{2^{n^k}}\right) = \mathsf{NEEXP}$$

Since we are looking for a complexity upper bound, we take as parameter the maximum number n_{max} of instantiation variables, that is linear in the size of the input, concluding that the resulting complexity class is coNEEXP. □

Now, we extend the analysis of this property keeping also into account the repair strategy: in other words, we analyze the replacement among accessible and repairable (but not necessarily valid) services. For the sake of brevity, we use the following notation:

$$\epsilon_{N,p,R,R'} \triangleq \bigwedge_{A \in \mathbf{A}} n_R(A) \equiv n_{N,p,R'}(A) \wedge \bigwedge_{P \in \mathbf{P}} \forall x, y. n_R(P)(x,y) \leftrightarrow n_{N,p,R'}(P)(x,y)$$

We point out, that since repair set is generated starting from the domain and service specifications, each service has its own.

Theorem 77. *Given two accessible and repairable services S_O and S_N, w.r.t. the repair sets R_O and R_N resp., without redundant effect specification, and a valid interface adapter A w.r.t. the world specification $\mathcal{W}$, s.t. A preserve the accessibility, the service S_N can replace the service S_O using A preserving the reachability iff for each $n \in 0 \dots n_{max}$, where n_{max} is the size of the largest instantiation set for the service S_O, for each $E \in \mathcal{E}_O$, and for each $O \in leaves(E)$ s.t. $\langle \boldsymbol{Y}_O, E_O \rangle = v(O)$ and $\| \boldsymbol{Y}_O \| = n$ the following implication holds:*

$$
\begin{aligned}
&\tau(\mathcal{W}) \wedge KB^{P_O} \wedge \Delta KB^B \wedge \phi_O \wedge \Delta KB^I_n(\mathsf{Top}', m) \wedge \Delta KB^U_c(m) \\
&\wedge \bigwedge_{R \in R_O} \Delta KB^R_n(m, n_R) \wedge \bigvee_{R \in R_O} \left(\tau_{n_R}(\mathcal{W}) \wedge \Delta KB^C_n(m, n_R)\right) \\
&\wedge \bigwedge_{N \in \lambda_{\mathcal{E}_N, n}} \phi_N \rightarrow \bigwedge_{p \in nPn} \kappa_{O,N,p} \wedge \bigwedge_{R \in R_N} \Delta KB^R_n(m_{N,p}, n_{N,p,R}) \models \\
\bigwedge_{k=0}^{s_O} \bigwedge_{R \in R^k_O} &\tau_{n_R}(\mathcal{W}) \wedge \Delta KB^C_n(m, n_R) \wedge \bigwedge_{R' \in \hat{R}^k_O} \neg \left(\tau_{n_{R'}}(\mathcal{W}) \wedge \Delta KB^C_n(m, n_{R'})\right) \rightarrow \\
&\bigvee_{N \in \lambda_{\mathcal{E}_N, n}} \phi_N \wedge \bigvee_{p \in nPn} \bigvee_{h=0}^{s_N} \bigvee_{R' \in R^h_N} \Delta KB^C_n(m_{N,p}, n_{N,p,R'}) \wedge \epsilon_{N,p,R,R'} \\
&\wedge \bigwedge_{R'' \in \hat{R}^h_N} \neg \Big(\tau_{n_{N,p,R''}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{N,p}, n_{N,p,R''})\Big)
\end{aligned}
\tag{6.9}
$$

where:

- *ϕ_O and ϕ_N are resp. the branching path formulas associated with the leaves O and N of corresponding effect specification trees;*
- *nPn denotes the set of all permutations of size n of $\{1 \dots n\}$;*
- *Top' is the concept denoting the active domain of resulting system state;*
- *m is the name mapping function for the service S_O while $m_{N,p}$ is the name mapping function for a leaf effect N of the service S_N, considering the permutation p;*
- *n_R is the name mapping function for the service S_O applying the repair R;*
- *$n_{N,p,R}$ is the name mapping function for the service S_N, enforcing the effect N, considering the permutation p, applying the repair R;*
- *s_o and s_n are resp. the size of repair set for S_O and S_N;*
- *R^k_O, $\hat{R}^k_O$, R^k_N, and $\hat{R}^k_N$ are the sets of repairs for S_O and S_N resp. of size equals or less than k as introduced at page 163.*

The axiom schemas are instantiated accordingly to the specification of various service effects.

Proof. The proof of this claim follows the outline of Theorem 75, extending them in order to deal also with the employment of repair search algorithm (see Figure 4.5). We start assuming that the devised implications hold given a pair of services S_O and S_N, the interface adapter and the world specification, but that the reachability is not preserved by the replacement. In other words, we are assuming that there exists at least an enactment $\langle \omega', \sigma'_{\mathbf{Y}_O} \rangle \in S_O(\omega, \sigma_{\mathbf{X}_O})$ of and a repair R^* (at least the empty one) of S_O s.t.:

- the enactment is, possibly, not legal but it can be repaired by R^* leading to ω'';
- there does not exist any other repair of S_O smaller than R^* that can actually repair the enactment;
- there does not exist any consistent instantiation assignment $\sigma'_{\mathbf{Y}_N}$ s.t. $\langle \omega'', \sigma'_{\mathbf{Y}_N} \rangle$ can be actually obtained from $S_N(\omega, A^\omega(\sigma_{\mathbf{X}_O}))$ applying any repair of S_N.

As in the previous case, we can restrict the analysis to the leaf effect of S_N that enforces the given enactment, let O^* be such an effect and let n be the size of its instantiation set.

For this effect of S_O, we consider also other possible repair $R \in R_O$ and resulting states ω''_R, while for S_N we consider every possible leaf effect N of S_N, ignoring its branching condition, having an instantiation set of size n, every possible bijective mapping between the instantiation set of N and O^*, and every possible repair $R' \in R_N$ and we built the pair $\langle \omega''_{N,p,R'}, \sigma'_{\mathbf{Y}_{N,p}} \rangle$, where:

- $\sigma'_{\mathbf{Y}_{N,p}}$ is the instantiation assignment for N over the active domain $\Delta^{\omega'} \setminus \Delta^{\omega}$ applying the permutation $p \in nPn$ to the instantiation assignment of O^*;
- $\omega''_{N,p,R''}$ is the target state obtained enforcing the effect N given the instantiation assignment, the input assignment obtained by the evaluation of the interface adapter to $\sigma_{\mathbf{X}_O}$ in ω, and applying the repair R''.

Now, we build a structure $\hat{\omega}$ s.t.:

- the enactment of S_O and related repairs are embedded into it according to the function $\mu^{\mathbf{R}}$ using a name mapping function m for the base enactment and a family of mapping functions n_R for the repaired ones;
- each enactment of each leaf effect N of S_N having an instantiation set of size n obtained in the previous step applying a permutation p and a repair R'' is embedded according to the function $\mu^{\mathbf{R}}$ using the name mapping functions $m_{N,p}$ and $m_{N,p,R''}$;
- the name Top' is interpreted as $\Delta^{\omega'}$.

Since the agreement on the interpretation of the result state active domain and common names such a construction is well-founded.

Given the hypothesis on the accessibility of service S_O and the enactment, applying Theorems 34, 35, 38, and 45 we can easily verify that the provided structure $\hat{\omega}$ is s.t.:

$$\hat{\omega} \models \tau(\mathcal{W}) \land KB^{P_O} \land \phi_{O^*} \land \Delta KB^I_n(\mathsf{Top}', m) \land \Delta KB^U_c(m)$$

Now we consider the definition of repair insert and update set, and also of repaired extension of role and concept names: applying the Corollary 13, we can also conclude that constraints of $\Delta KB_n^R(m, n_R)$ are satisfied, for the given effect O, for every repair $R \in R_O$. Since the service is reparable, applying Theorem 50, it follows that the structure $\hat{\omega}$ is also a model of the knowledge base $\Delta KB_n^C(m, n_{R^*})$. Moreover, since the state ω''_{R^*} is legal, given the hypothesis of repairability of the service enactment, and it is embedded into $\hat{\omega}$, by Theorem 3, we can conclude that also:

$$\hat{\omega} \models \tau_{n_{R^*}}(\mathcal{W})$$

Since we have applied the interface adapter A, assumed to be valid, according to Lemma 40, we can also conclude that:

$$\hat{\omega} \models \Delta KB^B$$

Considering each possible leaf effect N of S_N we can have two possible alternatives according to the fact that the branching precondition ϕ_N does or not hold in ω using the instantiation obtained by the application of the interface adapter A to $\sigma_{\mathbf{X}_O}$. Applying Theorem 35 (to S_N) in the latter case, since:

$$\hat{\omega} \not\models \phi_N$$

the corresponding material implication also holds $\hat{\omega}$. In the former case, considering each possible permutation p over n elements, we can easily verify that the construction of $\hat{\omega}$ is s.t.:

$$\hat{\omega} \models \bigwedge_{i \in 1 \ldots \|\mathbf{Y}_{O^*}\|} [\mathbf{Y}_{O^*}]_i \equiv [\mathbf{Y}_N]_{p_i}$$

since for each instantiation variable of N we have assigned the value of an instantiation variable of the effect O^*. Applying Theorem 38, extended in order to cope with the renaming also of instantiation variables as described at page 5.2.2, and Theorem 45 to each enactment of S_N we can conclude also that:

$$\hat{\omega} \models \Delta KB_n^I(\mathsf{Top}', m_{N,p}) \wedge \Delta KB_c^U(m_{N,p})$$

Finally, we apply the definition of repair insert and update sets w.r.t. the service S_N: given the Corollary 13, we can conclude that constraints obtained instantiating the axiom schema $\Delta KB_n^R(m_{N,p}, n_{N,p,R})$ are satisfied for the every effect N, for every permutation $p \in nPn$, and for every repair $R \in R_N$.

In other words, we have shown that antecedents of implication in Eq. 6.9 are satisfied in $\hat{\omega}$. Since, by hypothesis, such an implication holds, we can also conclude that its consequence are satisfied in $\hat{\omega}$. Considering the enactment of S_O and its repair R^*, since it is actually enforced, according to the given repair search strategy, we can also conclude that:

$$\hat{\omega} \not\models \tau_{n_{R'}}(\mathcal{W}) \wedge \Delta KB_n^C(m, n_{R'})$$

for every $R' \in \hat{R}_O^k$, where $k = \|R^*\|$. Applying the material implication, we can also conclude that there exist a leaf effect N^* of S_N, having an instantiation set of size n, a permutation p^*, and a repair $\tilde{R}$ of size h, s.t. $\hat{\omega}$ is a model of:

$$\begin{aligned} &\Delta KB_n^C(m_{N^*,p^*}, n_{N^*,p^*,\tilde{R}}) \wedge \epsilon_{N^*,p^*,R^*,\tilde{R}} \\ \wedge &\bigwedge_{R'' \in \hat{R}_N^h} \neg \left(\tau_{n_{N^*,p^*,R''}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{N^*,p^*}, n_{N^*,p^*,R''})\right) \end{aligned}$$

For any other repair R', given the effect N^*, we have that at least one of the following conditions hold:

- the repair is not effective and the obtained state is not legal:

$$\hat{\omega} \not\models KB^R(n_{N^*,p^*,R})$$

- the repair is not consistent:

$$\hat{\omega} \not\models \Delta KB_n^C(m_{N^*}, n_{N^*,p^*,R})$$

- the repair is effective and consistent, but its size is greater than $\left\|\tilde{R}\right\|$.

It means that repair $\tilde{R}$ is effective and can be actually enforced by S_N. For each name $N \in \mathbf{A} \cup \mathbf{P}$ we have that:

$$n_{R^*}(N)^{\hat{\omega}} = n_{N^*,p^*,\tilde{R}}(N)^{\hat{\omega}}$$

but, according to properties of embedding functions μ, we have also that:

$$n_{R^*}(N)^{\hat{\omega}} = N^{\omega''_{R^*}}, N^{\omega''_{N^*,p^*,\tilde{R}}} = n_{N^*,p^*,\tilde{R}}(N)^{\hat{\omega}}$$

Given the agreement on the active domain and named objects, that are constantly interpreted in any structures, we can now conclude that $\omega''_{N^*,p^*,\tilde{R}} = \omega''_{R^*} = \omega''$, but that means that also S_N can achieve the state ω'', since $\omega'_{N^*,p^*,\tilde{R}}$ belongs to its repaired enactment set, contradicting the hypothesis and proving the first part of the claim.

Now, we conversely assume that the service S_N can replace the service S_O using the adapter A preserving the reachability, but the implication does not hold. In other words, there must exist a structure $\hat{\omega}$ s.t. it is a model of the antecedent formula for some leaf effect O^* of S_O having an instantiation set of size n s.t. the implication antecedents are satisfied in it and, let $R^* \in R_O$ be the smallest repair of S_O s.t. $\hat{\omega}$ is a model of:

$$\hat{\omega} \models \tau_{n_{R^*}}(\mathcal{W}) \wedge \Delta KB_n^C(m, n_{R^*})$$

that, given the repairability hypothesis, must always exist, there does not exist any triple $\langle N, p, R' \rangle$, s.t.:

- N is a leaf effect of S_N having an instantiation set of size n;
- p is a permutation in nPn;
- R' is a repair of S_N;

- let $h = \|R'\|$, $\hat{\omega}$ is a model of:

$$\begin{aligned}&\phi_N \wedge \Delta KB_n^C(m_{N,p}, n_{N,p,R'}) \wedge \epsilon_{N,p,R^*,R'}\\ \wedge \bigwedge_{R'' \in \hat{R}_N^h} &\neg\Big(\tau_{n_{N,p,R''}}(\mathcal{W}) \wedge \Delta KB_n^C(m_{N,p}, n_{N,p,R''})\Big)\end{aligned}$$

According to Theorems 34, 35, 39, 46, and 51, an enactment $\langle \omega', \sigma'_{\mathbf{Y}_O}\rangle \in S_O(\omega, \sigma_{\mathbf{X}_O})$ of S_O is embedded using m into $\hat{\omega}$ enforcing the effect O^*, while the repair, according to R^*, is also embedded into this structure using n_{R^*}. Given the Lemma 21, because the repair is a minimal-size one and the final state is legal the enactment can actually take place. In other words, we have shown that including repairs in R_O, the service S_O can actually reach the state $\omega'' = \omega''_{R^*}$.

Since $\hat{\omega} \models KB^B$ we can conclude, according to Lemma 40, that we are considering the activations of service S_N obtained applying the interface adapter A to the assignment $\sigma_{\mathbf{X}_O}$ in ω. Moreover, we can project out from the structure $\hat{\omega}$ various world interpretations or extended interpretations using the adequate projection function π, given the available repair set R_N. As in the previous case, we focus the analysis on all possible leaf effects N of S_N s.t. their instantiation set size is n. Also in this case, we need to cope with two alternatives depending on whether ϕ_N is satisfied in $\hat{\omega}$ or not. If $\hat{\omega} \models \phi_N$, according to Theorem 35, we can conclude that the effect N can be selected and eventually enforced in ω by S_N if its branching condition is satisfied in ω. Given the accessibility preserving property, at least a suitable effect N^* must exist. Moreover, since we have also that:

$$\hat{\omega} \models \bigwedge_{p \in nPn} \kappa_{O^*,N,p} \wedge \bigwedge_{R \in R_N} \Delta KB_n^R(m_{N,p}, n_{N,p,R})$$

So, applying Theorems 39 and 51 we have that we can project out from $\hat{\omega}$ using the name mapping functions $n_{N,p,R}$ a set containing $\mathcal{O}\left(2^n \cdot n!\right)$ possible repaired enactments $\langle \omega''_{N,p,R}, \sigma'_{\mathbf{Y}_{N,p}}\rangle$ of the effect N s.t.:

- each one instantiates the same objects, since the resulting active domain is the same ($\mathsf{Top}'^{\hat{\omega}} = \Delta^{\omega''} = \Delta^{\omega''_{N,p,R}}$);
- each one permutes in a specific way the instantiation set $\Delta^{\omega''} \setminus \Delta^{\omega}$ of S_O;
- each one describe an enactment has been repaired using a specific element $R \in R_N$.

Since we are keeping into account all possible permutations of the n instantiation variables, we can also conclude that there is any other effective enactment of a leaf effect of S_N s.t. it has the same target state of S_O. Other leaf effects of S_N do not play a significant role and can be ignored: in fact, they can never be selected in ω by S_N, no matter as the service has been activated, since their branching conditions are not satisfied.

So far, we have built a set of possible enactments of S_N resulting from the application of the interface adapter A, applying any possible repair in R_N, embedded into the structure $\hat{\omega}$, and we have also shown that they are all the enactments relevant w.r.t. the reachability of ω''. In other words, since we have assumed that the reachability property of S_O is preserved using S_N applying A, we have that there is an element $N^*, p^*, \tilde{R}$ of this set s.t. $\omega''_{N^*,p^*,\tilde{\omega}} = \omega''$.

According to previous argumentation, we have that for each name $N \in \mathbf{A} \cup \mathbf{P}$ we have that:

$$N^{\omega''} = N^{\omega''_{N^*,p^*,\tilde{R}}}$$

but, given the properties of embedding functions μ, it follows also that:

$$n_{R^*}(N)^{\hat{\omega}} = N^{\omega''}, N^{\omega''_{N^*,p^*,\tilde{R}}} = n_{N^*,p^*,\tilde{R}}(N)^{\hat{\omega}}$$

and, hence, we can conclude that:

$$n_{R^*}(N)^{\hat{\omega}} = n_{N^*,p^*,\tilde{R}}(N)^{\hat{\omega}}$$

In other words, we have that:

$$\hat{\omega} \models \epsilon_{N^*,p^*,R^*,\tilde{R}}$$

Moreover, given the effect N^* and the repair $\tilde{R}$, according to the definition of repairable service and Theorem 50, we can conclude also that:

$$\hat{\omega} \models \tau_{n_{N^*,p^*,\tilde{R}}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{N^*,p^*}, n_{N^*,p^*,\tilde{R}})$$

Since the repair $\tilde{R}$ is also the smallest one that is able to enforce domain constraint upon service effects without retracting them, for any other repair $R' \in R_N$ s.t. $\|R'\| < \left\|\tilde{R}\right\|$, we have that:

$$\hat{\omega} \models \tau_{n_{N^*,p^*,R'}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{N^*,p^*}, n_{N^*,p^*,R'})$$

Finally, since we have also assumed that $\hat{\omega} \models \phi_{N^*}$, we have shown that also the consequent of implication in Eq. 6.9 is satisfied contradicting the hypothesis and proving the claim. □

Now we can provide an upper bound of the complexity for the problem of checking the reachability property of a pair of services in case of replacement keeping into account also the update repair.

Theorem 78. *Given a consistent world specification $\mathcal{W}$, two accessible services S_N and S_O, and a valid interface adapter A for $\boldsymbol{X}_{S_N}$ given $\boldsymbol{X}_{S_O}$, the problem of checking whether S_N is a reachability preserving replacement for the service S_O using A considering all possible simple repairs is in* coNEEXP.

Proof. According to Theorem 77, the decision problem can be reduced to a linear number of entailment checks in $\mathcal{C}^2$: one for each leaf effect of S_O. Given the Cor. 1, the complexity of the entailment checking is coNEXP in the size of the formula.

We consider the complementary problem of checking whether a specific effect e of S_O having an instantiation set of size n_e can be preserved by the replacing.

$$\overline{\mathsf{RepReachability_e}} \in \bigcup_k \mathsf{NTIME}\left(2^{f(n)^k}\right)$$

In this case the size of the formula is s.t. it contains for every possible repair $R \in R_O$ a sub-formula for every possible pair of permutations of n_e and repairs $R' \in R_N$ having a length that is linear in the number of possible repairs. Given

Theorem 26, the length of the formula is approx. $f(n) \in \mathcal{O}\left(2^n \cdot n_e! \cdot 2^{p(n)}\right)$ and we have that:

$$\overline{\mathsf{RepReachability_e}} \in \bigcup_k \mathsf{NTIME}\left(2^{\left(2^n \cdot n_e! \cdot 2^{p(n)}\right)^k}\right)$$

Applying the Stirling's approximation we have so:

$$\overline{\mathsf{RepReachability_e}} \in \bigcup_k \mathsf{NTIME}\left(2^{\left(2^n \cdot 2^{n_e} \cdot 2^{p(n)}\right)^k}\right) \subseteq \bigcup_k \mathsf{NTIME}\left(2^{2^{n^k}}\right) = \mathsf{NEEXP}$$

Since we are looking for a complexity upper bound, we take as parameter the maximum number n_{max} of instantiation variables, that is linear in the size of the input, obtaining that the resulting complexity class is coNEEXP. □

Remark 39. *We notice that in this case we have to cope with two exponential blow-ups: one due to the exponential number of suitable repairs (for both services) and the other due to the number of possible binding of instantiation variables. But, since, they are orthogonal, it does not involve an increment of the problem asymptotic complexity.*

6.2.3 Adequacy-preserving replaceability

As pointed out in previous discussion, since the reachability-preserving properties are not completely satisfactory, we analyze the problem of service replacement keeping into the account the tacitly intended user's goal: a service replacement is valid w.r.t. a goal only if it preserves the adequacy properties of the original service.

In fact, in the case of non-deterministic services we can easily try out that a service with a higher degree non-determinism can generally replace a more deterministic one preserving the reachability property, but from the point of view of the service user it can exhibit unexpected behaviors, leading to some inconsistency in terms of service contract. On the other hands a more restrictive form of state-based replaceability can not be easily defined without requiring a near-complete service equivalence. The explicit introduction of the user goal enable to filter out irrelevant behaviors, focusing the analysis on more interesting ones (at least according to client perspective).

According to Lemma 41, and the definition of uniform/non-uniform adequacy properties, we can, without difficulty, conclude that is possible to induce on the adequacy level space $\mathcal{L} = \{\mathsf{WNU}, \mathsf{WU}, \mathsf{SNU}, \mathsf{SU}\}$ a partial order relation $\prec$ defined as depicted in Figure 6.1.

Lemma 44. *Let G be a goal and S be a service s.t. the adequacy level of the service w.r.t. G given a binding B is $l \in \mathcal{L}$, then S is l'-adequate for each $l' \prec l$.*

Proof. Trivial. □

Definition 100 (Goal adequacy preserving service replaceability)**.** *Given two services S_N and S_O, we say that S_N can replace the service S_O preserving the adequacy w.r.t. a goal G if for a valid activation binding of the latter there exists a valid activation binding of the former s.t. it has at least the same adequacy level w.r.t. the goal.*

$$\begin{tikzcd}
 & \mathsf{SU} \arrow[dl] \arrow[dr] & \\
\mathsf{SNU} \arrow[dr] & & \mathsf{WU} \arrow[dl] \\
 & \mathsf{WNU} &
\end{tikzcd}$$

Figure 6.1: The service/goal adequacy level order relation $\prec$

In other words, if the service that has to be replaced is able to achieve the goal, then the replacing service must also be able to do so. On the other hand, if the replacing service can possibly lead to a world state where the goal is not achieved, also the replaced service must admit such a kind of outcome. In this way it is possible to preserve adequacy properties across the replacement.

Example 13. *Given services S and S_1 defined in Examples 2 and 3, the interface adapter A defined in Example 11, and the user goal G defined in Example 10, we can conclude that, since $\mathsf{SNU} \prec \mathsf{SU}$, the service S can replace S_1 preserving the user goal adequacy, despite it does not preserve the reachability. Moreover, for both services S_2 and S_3 defined in Example 12 the same property holds: it is possible since the user goal simply express that the service user does not care about the registration (despite it could eventually matter for the service provider).*

Service S_1 cannot replace S w.r.t. the goal G, while, if consider the more specific, but even non-ground, goal G_1 of a resident of town_1 aiming to change its own residence to town_2, we can use the service S_1 in vece *of service S.*

Remark 40. *The ability of reasoning w.r.t. to user goals enable us to enforce a kind of selective analysis on service effects, keeping into account only ones that are explicitly considered as relevant, providing another tool to deal with incomplete specifications (the other one is the effect repair). Moreover, the definition of a non-ground (intensional) user goal allows us to move such kind of analysis/evaluation of these properties at system design/tuning-time from the execution-time required by an approach leveraging on concrete goal definition. Given the high computational complexity of involved problems, it is a very desirable feature.*

In the following, we analyze this problem considering the specification of goal binding schemas and interface adapters. We initially provide some other useful definitions. Given a binding schema B defined over the alphabets $\mathbf{X}$ and $\mathbf{Y}$, and a specification $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$, we introduce an unfolding function ρ that, given any concept expression in $\mathcal{ALCQIO}$, possibly including references to variables in $\mathbf{Y}$, returns a new expression $\mathcal{ALCQIO}$, referencing variables in $\mathbf{X}$. The function is defined as follows:

$$\begin{aligned}
\rho_B(Y) &\triangleq B(X)\\
\rho_B(A) &\triangleq A\\
\rho_B(C \sqcap C') &\triangleq \rho_B(C) \sqcap \rho_B(C')\\
\rho_B((\bowtie\ n\, R\, C)) &\triangleq (\bowtie\ n\, R\, \rho_B(C))\\
\rho_B(\{o\}) &\triangleq \{o\}\\
\rho_B(\neg C) &\triangleq \neg \rho_B(C)
\end{aligned}$$

Definition 101 (Binding schema composition)**.** *Given two binding schemas A and B defined resp. over $\langle \boldsymbol{X}_A, \boldsymbol{Y}_A\rangle$ and $\langle \boldsymbol{X}_B, \boldsymbol{Y}_B\rangle$, s.t. the input alphabet of A is equal to the output alphabet of B (at least considering a variable renaming), the composition $A \lhd B$ is a new binding schema defined over $\langle \boldsymbol{X}_B, \boldsymbol{Y}_A\rangle$ s.t. for each variable name $V \in \boldsymbol{Y}_A$, the access function $(A \lhd B)(V)$ is defined as:*

$$(A \lhd B)(V) \triangleq \rho_B(A(V))$$

We can now characterize the semantics of the binding schema composition using the following results.

Lemma 45. *Given a specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$, two binding schemas A and B defined resp. over $\langle \boldsymbol{X}_A, \boldsymbol{Y}_A\rangle$ and $\langle \boldsymbol{X}_B, \boldsymbol{Y}_B\rangle$, s.t. the input alphabet of A is equal to the output alphabet of B, a world state ω and an assignment $\sigma_{\boldsymbol{X}_B}$, s.t. both B and $A \lhd B$ are consistently evaluated, let $\sigma_{\boldsymbol{Y}_B}$ be the assignment obtained by the evaluation of binding schema B in ω given such an input, its evaluation is:*

$$A(Y)^{\omega}(\sigma_{\boldsymbol{Y}_B}) = (A \lhd B)(Y)^{\omega}(\sigma_{\boldsymbol{X}_B})$$

for each $Y \in \boldsymbol{Y}_A$.

Proof. In order to prove the claim, we need to show that given an expression $C = A(Y)$, for some $Y \in \mathbf{Y}_A$ involving some singleton variable names in $\mathbf{X}_A = \mathbf{Y}_B$, its evaluation in the extended interpretation $\omega \lhd \sigma_{\mathbf{X}_A}$ is equal to the evaluation of $\rho_B(C)$ in the extended interpretation $\omega \lhd \sigma_{\mathbf{X}_B}$, assuming that $\sigma_{\mathbf{X}_A}$ is obtained evaluating B in the latter structure. In other words, we need to show that:

$$C^{\omega\lhd\sigma_{\mathbf{Y}}} = [\rho_B(C)]^{\omega\lhd\sigma_{\mathbf{X}}}$$

given that $\sigma_{\mathbf{Y}} = B^{\omega}(\sigma_{\mathbf{X}})$.

In the following, we prove the claim by induction on the expression language.

- $Y^{\omega\lhd\sigma_{\mathbf{Y}}} = [\rho_B(Y)]^{\omega\lhd\sigma_{\mathbf{X}}}$ where $Y \in \mathbf{Y}$. By the definition of function ρ_B:

$$[\rho_B(Y)]^{\omega\lhd\sigma_{\mathbf{X}}} = [B(Y)]^{\omega\lhd\sigma_{\mathbf{X}}}$$

  and according to the hypothesis and properties of extended interpretation:

$$[B(Y)]^{\omega\lhd\sigma_{\mathbf{X}}} = Y^{\omega\lhd\sigma_{\mathbf{Y}}}$$

- $A^{\omega\lhd\sigma_{\mathbf{Y}}} = [\rho_B(A)]^{\omega\lhd\sigma_{\mathbf{X}}}$ where $A \in \mathbf{A}$. By the definition of translation function ρ_B:

$$[\rho_B(A)]^{\omega\lhd\sigma_{\mathbf{X}}} = A^{\omega\lhd\sigma_{\mathbf{X}}}$$

  By the definition of extended interpretation:

$$A^{\omega\lhd\sigma_{\mathbf{X}}} = A^{\omega} = A^{\omega\lhd\sigma_{\mathbf{Y}}}$$

- $[\{o_1,\ldots,o_n\}]^{\omega\lhd\sigma_{\mathbf{Y}}} = [\rho_B(\{o_1,\ldots,o_n\})]^{\omega\lhd\sigma_{\mathbf{X}}}$ where $\{o_1,\ldots,o_n\} \subseteq \mathbf{O}$. By the definition of translation function ρ_B:

  $$[\rho_B(\{o_1,\ldots,o_n\})]^{\omega\lhd\sigma_{\mathbf{X}}} = [\{o_1,\ldots,o_n\}]^{\omega\lhd\sigma_{\mathbf{X}}}$$

  According to the standard semantics:

  $$[\{o_1,\ldots,o_n\}]^{\omega\lhd\sigma_{\mathbf{X}}} = \bigcup_{i=1}^{n} o_i^{\omega\lhd\sigma_{\mathbf{X}}}$$

  By the definition of extended interpretation:

  $$\bigcup_{i=1}^{n} o_i^{\omega\lhd\sigma_{\mathbf{X}}} = \bigcup_{i=1}^{n} o_i^{\omega} = \bigcup_{i=1}^{n} o_i^{\omega\lhd\sigma_{\mathbf{Y}}}$$

  According to the standard semantics:

  $$\bigcup_{i=1}^{n} o_i^{\omega\lhd\sigma_{\mathbf{Y}}} = [\{o_1,\ldots,o_n\}]^{\omega\lhd\sigma_{\mathbf{Y}}}$$

- $[C \sqcap C']^{\omega\lhd\sigma_{\mathbf{Y}}} = [\rho_B(C \sqcap C')]^{\omega\lhd\sigma_{\mathbf{X}}}$. By the definition of translation function ρ_B:

  $$[\rho_B(C \sqcap C')]^{\omega\lhd\sigma_{\mathbf{X}}} = [\rho_B(C) \sqcap \rho_B(C')]^{\hat{\omega}}$$

  According to the standard semantics:

  $$[\rho_B(C) \sqcap \rho_B(C')]^{\omega\lhd\sigma_{\mathbf{X}}} = \rho_B(C)^{\omega\lhd\sigma_{\mathbf{X}}} \cap \rho_B(C')^{\omega\lhd\sigma_{\mathbf{X}}}$$

  By the inductive hypothesis:

  $$\rho_B(C)^{\omega\lhd\sigma_{\mathbf{X}}} \cap \rho_B(C')^{\omega\lhd\sigma_{\mathbf{X}}} = C^{\omega\lhd\sigma_{\mathbf{Y}}} \cap C'^{\omega\lhd\sigma_{\mathbf{Y}}}$$

  According to the standard semantics:

  $$C^{\omega\lhd\sigma_{\mathbf{Y}}} \cap C'^{\omega\lhd\sigma_{\mathbf{Y}}} = [C \sqcap C']^{\omega\lhd\sigma_{\mathbf{Y}}}$$

- $[\neg C]^{\omega\lhd\sigma_{\mathbf{Y}}} = [\rho_B(\neg C)]^{\omega\lhd\sigma_{\mathbf{X}}}$. By the definition of function ρ_B:

  $$[\rho_B(\neg C)]^{\omega\lhd\sigma_{\mathbf{X}}} = [\neg\rho_B(C)]^{\omega\lhd\sigma_{\mathbf{X}}}$$

  According to the standard semantics:

  $$[\neg\rho_B(C)]^{\omega\lhd\sigma_{\mathbf{X}}} = \Delta^{\omega\lhd\sigma_{\mathbf{X}}} \setminus [\rho_B(C)]^{\omega\lhd\sigma_{\mathbf{X}}}$$

  By the inductive hypothesis:

  $$\Delta^{\omega\lhd\sigma_{\mathbf{X}}} \setminus [\rho_B(C)]^{\omega\lhd\sigma_{\mathbf{X}}} = \Delta^{\omega\lhd\sigma_{\mathbf{X}}} \setminus C^{\omega\lhd\sigma_{\mathbf{Y}}}$$

  Since both extended interpretations agree on the active domain ($\Delta^{\omega\lhd\sigma_{\mathbf{X}}} = \Delta^{\omega} = \Delta^{\omega\lhd\sigma_{\mathbf{Y}}}$):

  $$\Delta^{\omega\lhd\sigma_{\mathbf{X}}} \setminus C^{\omega\lhd\sigma_{\mathbf{Y}}} = \Delta^{\omega\lhd\sigma_{\mathbf{Y}}} \setminus [C]^{\omega\lhd\sigma_{\mathbf{Y}}}$$

  According to the standard semantics:

  $$\Delta^{\omega\lhd\sigma_{\mathbf{Y}}} \setminus C^{\omega} = [\neg C]^{\omega\lhd\sigma_{\mathbf{Y}}}$$

- $[(\geq\ n\, R\, C)]^{\omega \lhd \sigma_{\mathbf{Y}}} = [\rho_B((\geq\ n\, R\, C))]^{\omega \lhd \sigma_{\mathbf{X}}}$, where R is an arbitrary role expression ($R \rightarrow P|P^-$, $P \in \mathbf{P}$). By the definition of function ρ_B:

  $$[\rho_B((\geq\ n\, R\, C))]^{\omega \lhd \sigma_{\mathbf{X}}} = [(\geq\ n\, R\, \rho_B(C))]^{\omega \lhd \sigma_{\mathbf{X}}}$$

  According to the standard semantics:

  $$[(\geq\ n\, R\, \rho_B(C))]^{\omega \lhd \sigma_{\mathbf{X}}} = \left\{\alpha | \left\| S^{\omega \lhd \sigma_{\mathbf{X}}}_{\rho_B(C),R}(\alpha) \right\| \geq n \right\} \tag{6.10}$$

  where $S^{\omega \lhd \sigma_{\mathbf{X}}}_{\rho_B(C),R}(\alpha)$ denotes the set of R-successors of element α belonging to $\rho_B(C)$ defined as:

  $$S^{\omega \lhd \sigma_{\mathbf{X}}}_{\rho_B(C),R}(\alpha) = \{\beta | \beta \in \rho_B(C)^{\omega \lhd \sigma_{\mathbf{X}}}, \langle \alpha, \beta \rangle \in R^{\omega \lhd \sigma_{\mathbf{X}}}\}$$

  But, since the inductive hypothesis and the agreement of both structures on the interpretation of role names ($P^{\omega \lhd \sigma_{\mathbf{X}}} = P^{\omega} = P^{\omega \lhd \sigma_{\mathbf{Y}}}$ for each $P \in \mathbf{P}$), we have that:

  $$S^{\omega \lhd \sigma_{\mathbf{X}}}_{\rho_B(C),R}(\alpha) = \{\beta | \beta \in C^{\omega \lhd \sigma_{\mathbf{Y}}}, \langle \alpha, \beta \rangle \in R^{\omega \lhd \sigma_{\mathbf{Y}}}\}$$

  In other words, we can conclude that:

  $$S^{\omega \lhd \sigma_{\mathbf{X}}}_{\rho_B(C),R}(\alpha) = S^{\omega \lhd \sigma_{\mathbf{Y}}}_{C,R}(\alpha)$$

  Applying such result to Eq. 6.10, we have that:

  $$[(\geq\ n\, R\, \rho_B(C))]^{\omega \lhd \sigma_{\mathbf{X}}} = \left\{\alpha | \left\| S^{\omega \lhd \sigma_{\mathbf{Y}}}_{C,R}(\alpha) \right\| \geq n \right\}$$

  Observing, according to standard semantics, that:

  $$[(\geq\ n\, R\, C)]^{\omega \lhd \sigma_{\mathbf{Y}}} = \left\{\alpha | \left\| S^{\omega \lhd \sigma_{\mathbf{Y}}}_{C,R}(\alpha) \right\| \geq n \right\}$$

  we can conclude the proof.

□

Given two binding schemas, we now consider the validity of the composed one. The following results will help to establish the validity of a composed binding, given the corresponding properties of composing ones.

Lemma 46. *Given a world specification $\mathcal{W}$ and two valid binding schemas A and B w.r.t. it, the binding schema $A \lhd B$ is also valid.*

Proof. It follows immediately from the definition of valid binding schema: in fact the former (B) is evaluated in a legal state ω w.r.t. $\mathcal{W}$ and an assignment and, since it is valid, the result is also an assignment on which the latter (A) can be evaluated resulting in a final consistently defined assignment. □

Lemma 47. *Given a world specification $\mathcal{W}$, a binding schema B valid w.r.t. it and a condition $\boldsymbol{C}$, and a binding schema A valid w.r.t. $\mathcal{W}$, the binding schema $A \lhd B$ is also valid w.r.t. such a condition.*

Proof. Given a world state ω and an assignment for B s.t. the condition is satisfied w.r.t. them, the evaluation of the binding former binding schema results into a new consistent assignment on which the latter can be evaluated. □

Given a binding schema B and a condition $\mathbf{C}$ s.t. it defined over the output signature of the schema ($\mathbf{U}$), we define the composed condition $\mathbf{C} \lhd B$ as the condition obtained simultaneously replacing each occurrence of condition variables $U \in \mathbf{U}$ with the corresponding function in the schema $B(U)$:

$$\mathbf{C} \lhd B \triangleq \{\rho_B(C) | C \in \mathbf{C}\}$$

Given such a definition, applying properties established so far, we can also conclude that:

Lemma 48. *Given a world state ω, a condition $\boldsymbol{C}$ and binding schema B, that can be composed with $\boldsymbol{C}$, an input assignment σ for B, s.t. it is correctly evaluated in ω into σ', then $\boldsymbol{C} \lhd B$ is satisfied in ω using σ iff $\boldsymbol{C}$ is satisfied in ω using σ'.*

Lemma 49. *Given a world specification $\mathcal{W}$, a binding schema B valid w.r.t. it, and a binding schema A valid w.r.t. a condition $\boldsymbol{C}$, the binding schema $A \lhd B$ is also valid w.r.t. $\boldsymbol{C} \lhd B$.*

Proof. Let ω and σ' be resp. a world state and an assignment s.t. the binding schema A is correctly evaluated, according to Lemma 48, since the binding schema B is assumed to be valid, there exists an assignment σ on the input signature of B, and hence of $A \lhd B$, s.t. $\mathbf{C} \lhd B$ holds in ω. Conversely, if such a condition is satisfied, we can also conclude that also $\mathbf{C}$ is satisfied and, consequently, that the binding schema A is consistently evaluated. □

Given two conditions $\mathbf{C} = \{C_1, \dots, C_n\}$ and $\mathbf{C'} = \{C'_1, \dots, C'_m\}$, having disjoint signatures, the conjunction of such constraints is defined as:

$$\mathbf{C} \wedge \mathbf{C'} \triangleq \{C \cup C' | C \in \mathbf{C}, C' \in \mathbf{C'}\}$$

Applying the distributive and commutative properties of boolean operators, from this definition we can easily obtain the following result:

Lemma 50. *Given a world state ω, two conditions $\boldsymbol{C}$ and $\boldsymbol{C'}$ and their assignments σ and σ', the condition $\boldsymbol{C} \wedge \boldsymbol{C'}$ is satisfied in ω using $\sigma \cup \sigma'$ iff both $\boldsymbol{C}$ and $\boldsymbol{C'}$ are satisfied in ω using resp. σ and σ'.*

Theorem 79. *Given a world specification $\mathcal{W}$, a binding schema B valid w.r.t. to a condition $\boldsymbol{C}$, and a binding schema A valid w.r.t. a condition $\boldsymbol{C'}$, the binding schema $A \lhd B$ is also valid w.r.t. $\boldsymbol{C} \wedge (\boldsymbol{C'} \lhd B)$.*

Proof. Let ω be a world state and σ an assignment for the input signature of B s.t. the condition $\mathbf{C} \wedge (\mathbf{C'} \lhd B)$ holds in ω given σ. Since the binding schema B is assumed to be valid w.r.t. C, we can conclude that is evaluated into a consistent assignment σ' and, applying Lemma 48 and 50, that also $\mathbf{C'}$ holds in ω given σ'. According to the hypothesis on the validity of A w.r.t. this condition, we can finally conclude that also this binding schema is correctly evaluated and, hence, the composed binding schema is valid. □

On this foundation we can provide a criterion to check whether a service can replace another one preserving the adequacy w.r.t. a given goal. Such a criterion relies on the availability of both goal/service and service/service bindings as assumed so far in the analysis of both adequacy and replaceability.

Theorem 80. *Let G be a goal, $\langle S_N, S_O\rangle$ be a pair of services, A be an interface adapter $\boldsymbol{X}_O$ to $\boldsymbol{X}_N$ and B be binding from $\boldsymbol{U}_G$ to $\boldsymbol{X}_O$, s.t.:*

- *the binding B is consistent w.r.t. the goal G and the service S_O;*
- *the service S_O is l_O-adequate to achieve the goal G using the binding B, where $l_O \in \mathcal{L}$;*
- *the interface adapter A is valid and preserves the accessibility.*

If the service S_N is l_N-adequate to achieve the goal G using the binding $A \triangleleft B$ and $l_O \preceq l_N$, then the S_N can replace the service S_O preserving the adequacy w.r.t the goal G.

Proof. To prove that S_N is an adequacy-preserving replacement of S_O w.r.t. the goal G, we need to show that its adequacy level l_N is at least the same of S_O. Since we have assumed that, using the binding $A \triangleleft B$, we can determine that its level $l_N \succeq l_O$, in order to complete the proof, we have also to show that this binding schema is well-founded w.r.t. the goal G and the service S_N.

In terms of binding schema validity, we have that B is valid under the condition $\mathcal{H}_G$, while A is valid under $\mathcal{P}_O$. Since the binding B is consistent, for each world state ω and for each goal instantiation $\sigma_{\mathbf{U}}$ s.t. the condition $\mathcal{H}_G$ holds ($\omega \triangleleft \sigma_{\mathbf{U}} \models \mathcal{H}_G$), we have that invocation preconditions of S_O also hold in ω ($\omega \triangleleft B(\omega, \sigma_{\mathbf{U}}) \models \mathcal{P}_O$). By hypothesis we have assumed that interface adapter A is valid and it is s.t. that also invocation preconditions of S_N hold in ω ($\omega \triangleleft A(\omega, B(\omega, \sigma_{\mathbf{U}})) \models \mathcal{P}_N$) using the activation assignment obtained by the evaluation of the binding schema. In other words, we have that the service S_N is activable using the binding schema $A \triangleleft B$, but this composition is also valid under the condition $\mathcal{H}_G$. In fact, according to Theorem 79, the binding schema $A \triangleleft B$ must be valid under the condition $\mathcal{H}_G \wedge (\mathcal{P}_O \triangleleft B)$, but, given the consistency assumption on B w.r.t. S_O, we have that the second conjunct is logically implied by $\mathcal{H}_G$. We have shown that the binding $A \triangleleft B$ is valid w.r.t. $\mathcal{H}_G$ and it is suitable to be employed on S_N as consistent goal binding schema. □

Employing this property is easy to set up a procedure to check the replaceability among services: in fact it turns out that is sufficient to determine if the replace candidate service can achieve the same adequacy level using the composed binding. Moreover, such a property provides us also a way to effectively compute such a binding. Complexity results can easily extended this case too. Essentially the problem complexity depends upon the minimum level of adequacy to preserve according to results presented in Table 6.2.

Remark 41. *The procedure can be applied both considering or ignoring the service effect repairs, since this aspect is completely orthogonal. However, we point out that repair sets R_O and R_N, as in other cases, are distinct, because they depend both on domain and service specifications.*

6.3 Functional Similarity

A more general and articulated notion than (possibly mutual) service replaceability is the functional similarity of services. Roughly speaking, we are interested in checking whether two given services S_1 and S_2 of a community perform a similar task, as well as if their applicability contexts (e.g., invocation scope or coverage) are different. This concept is very useful so that a set of distinct services can be clustered into functional homogeneous groups, which are not strongly equivalent according to the notion previously stated.

Considering some implemented frameworks (e.g., DCOM, CORBA or XML web services), we are interested in the semantic characterization of the notion of components implementing the same functional contract. We remark that, according to such a kind of system agreement, functional comparable components are generally denoted by the fact that they expose the same interface (e.g., IDL or WSDL port-type). But this notion is too weak, since it relies only upon a syntactical specification and the only logical inferable consequence is that different components implementing the same interface are only able to establish the same enactment protocol or, in other words, that they are able to "speak" in the same language.

The assumption that such kinds of software components are performing an equivalent functional task implicitly holds only according to a development agreement among the community members. In fact, generally interface-based frameworks prescribe that operation signatures should be, at least informally, annotated with pre-conditions and effects specification in order to characterize the service contract. Also so-called static approaches require a substantial agreement on that effect reification concepts. In a scenario where a central authority rules the service contracts and interfaces in a top-down manner, such a kind of agreement is convincingly enforced, but in the case where the service community is built bottom-up from the actual availability of independently implemented functionalities[8], a more powerful notion, which relies only on an agreement upon the specification language, is required. Notably, considering the evolution of e-government systems, despite a top-down approach should be more conceivable, the implemented service communities have been developed so far in a bottom-up manner, at least w.r.t. functional aspects.

The functional comparison can be performed upon different criteria: in particular, we now provide a notion of functional comparison that allows to compare services that are distinguishable at the object level. It may be useful, e.g., when service activation scope definitions involve some domain objects (e.g., geographical names). More sophisticated definitions that keep into account also intensional level mismatch can be devised in a similar way. W.l.o.g., we assume that input and output parameters of the comparing services are the same: in fact, as shown in previous section if it this condition does not hold we can introduce a suitable interface adapter.

The comparison of service modulo the object-level characterization can be carried out in different manners: e.g., retracting the unique names assumption and introducing some links among different interpretation structures. However, in this case we can easily notice that object-level knowledge play a role only in this specific fragment of problem specification: in particular, the extensional knowledge (or in other words, the knowledge explicitly involving objects) is relegated to the ABox $\mathcal{A}$ part of the KB $\mathcal{W} = \langle \mathcal{T}, \mathcal{A} \rangle$ and in the queries employed into various constructs of the service specification (e.g., preconditions, effect arguments). So, instead to renounce to the unique names assumption, we prefer to adopt a strategy based on the search of world/service specification space of

[8]That are distinct also in terms of operation signatures.

suitable instances accordingly altering/replacing object names.

Remark 42. *Intuitively, the devised solution aims at implementing an heuristic criterion that allows to consider as functional comparable (or similar) entities (i.e., service specifications) that should turn out to be quite similar into a slightly different context (i.e., world specification).*

More specifically, we consider as functional comparable services that under suitable conditions, possibly w.r.t. world specifications that differs only by the extensional characterization, can be taken as replacement. In this way, moreover, the service replaceability turns out to be a special case of the functional comparability, requiring the agreement on the extensional level too.

Since the extensional level knowledge can be expressed only w.r.t. to object names $\mathbf{O}$ we can adopt the following construct to denote suitable specification transformations:

Definition 102 (Object renaming)**.** *Given an object alphabet $\boldsymbol{O}$, an object renaming ξ is a total function $\boldsymbol{O} \mapsto \boldsymbol{O}$, that arbitrary maps each object name to another object name.*

Such a kind of function is generally expressed a substitution object and represented as:

$$\xi = \{a/\xi(a), b/\xi(b), \dots z/\xi(z)\}$$

where $a, b, \dots, z$ are object names in $\mathbf{O}$, and $\xi(\cdot)$ denotes the evaluation of the function on the given argument. Generally, unaltered names are omitted from this representation.

We can also easily extend such a definition to $\mathcal{ALCQIO}$ language as the following:

$$\begin{aligned}
\xi(A) &\triangleq A \\
\xi(C \sqcap C') &\triangleq \xi(C) \sqcap \xi(C') \\
\xi((\bowtie\ n\, R\, C)) &\triangleq (\bowtie\ n\, R\, \xi(C)) \\
\xi(\{o\}) &\triangleq \{\xi(o)\} \\
\xi(\neg C) &\triangleq \neg \xi(C) \\
\xi(o : C) &\triangleq \xi(o) : \xi(C) \\
\xi((o, o') : R) &\triangleq (\xi(o), \xi(o')) : R \\
\xi(\mathcal{A}) &\triangleq \bigcup_{o:C \in \mathcal{A}} \xi(o : C) \cup \bigcup_{(o,o'):R \in \mathcal{A}} \xi((o, o') : R)
\end{aligned}$$

Remark 43. *It is important to point out that since the cardinality of object alphabet is finite also the cardinality of suitable object renaming function is finite and that previous properties are decidable in this framework.*

Lemma 51. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, there are at most $\mathcal{O}\left(2^{\|\boldsymbol{O}\|^2}\right)$ distinct object renaming functions.*

Proof. The claim follows immediately from the observation that there exists n^n distinct total functions mapping a domain of n elements on itself, assuming $n = \|\mathbf{O}\|$ and that $n^n = 2^{n \cdot log_2 n} \leq 2^{n^2}$. $\square$

Definition 103 (Renamed world specification)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, an object renaming ξ over $\boldsymbol{O}$ and a world specification $\mathcal{W} = \langle \mathcal{T}, \mathcal{A} \rangle$ defined on this domain, we define as renamed world specification the knowledge base $\mathcal{W}^{\xi} = \langle \mathcal{T}, \xi(\mathcal{A}) \rangle$ obtained applying the object renaming function to every assertion in the ABox.*

Given an arbitrary world specification and an object renaming function the obtained renamed specification can turn out to be not consistent w.r.t. the TBox constraint, hence we need to refine the definition so that only a consistent renaming is allowed.

Definition 104 (Consistency preserving object renaming)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$ and a consistent world specification $\mathcal{W} = \langle \mathcal{T}, \mathcal{A} \rangle$ defined on this domain, an object renaming ξ over $\boldsymbol{O}$ is consistent w.r.t. them or it preserves the consistency if the renamed world specification $\mathcal{W}^{\xi}$ is also consistent.*

We now apply the object renaming also to service specification.

Definition 105 (Renamed e-service)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, an object renaming ξ over $\boldsymbol{O}$ and an e-service specification $S = \langle \boldsymbol{X}, \boldsymbol{Y}, \mathcal{P}, \mathcal{E} \rangle$, the renamed e-service specification S^{ξ} is an e-service specification on the same signature obtained from S applying the renaming function ξ to every precondition, branching and effect argument query expressed as $\mathcal{ALCQIO}$-concept expression.*

As done for world specification, we can also check whether the object renaming preserves the service semantic properties.

Definition 106 (Accessibility preserving object renaming)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$ and an accessible service S both defined on this domain, an object renaming ξ over $\boldsymbol{O}$ preserves the accessibility if the renamed service specification S^{ξ} is also accessible.*

Analogous definitions can be provided for other service properties (effect consistency, validity and repairability).

Remark 44. *We notice that, since the object renaming involve only the assertions of the world specification and the concept expressions in the service specification, it has no impact on the repair search strategy that, instead, depends upon the domain specification and service signatures.*

Definition 107 (Service functional comparability)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$ and a pair of valid (or repairable) services S_O and S_N, we say that S_N is functionally comparable with S_O if there exists a consistent object renaming ξ s.t. it preserves the validity (resp. repairability) of given services and S_N^{ξ} is a reachability preserving replacement of S_O^{ξ}.*

Definition 108 (Service functional similarity)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$ and a pair of valid (or repairable) services S_O and S_N, we say that S_N and S_O are functionally similar if they are mutually functional comparable.*

This approach seems to be quite interesting in e-government scenarios, where different administrative department authorities provide similar services to different scopes, generally denoted in some extensional way (i.e., a service provided by some local public administration is reserved only to subjects resident in its own jurisdiction). It turns also interesting in e-commerce applications, e.g., in the management of supply-chain, since service providers can define a restricted served scope, not necessarily on geographic basis.

Example 14. *Given the service S_1 of Example 2, a simple functional equivalent service S_4 (under the object renaming $\xi = \{\mathsf{town}_2/\mathsf{town}_1\}$) is the following:*

$$\begin{aligned}
&\boldsymbol{X} = \{x_1, x_2\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x_1 \sqcap \exists \mathsf{residentIn}.\,\{\mathsf{town}_2\} \ \mathsf{and} \ x_2 \sqcap \mathsf{Town} \ \mathsf{and} \ \mathtt{not} \ x_2 \sqcap \{\mathsf{town}_2\}\\
&E = \left\{-\mathsf{residentIn}(x_1, \exists \mathsf{residentIn}^{-}.x_1), +\mathsf{residentIn}(x_1, x_2)\right\}\\
&\quad \cup \{-\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, \{\mathsf{town}_2\})\}\\
&\quad \cup \{+\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, x_2)\}
\end{aligned}$$

In this case, once the object renaming has been applied, the resuling service is syntactically equal, module possible rewriting of expression into equivalent forms (e.g., applying algebraic properties of logical operators), to the comparing one, but in the general case logical also consequences of the specification must be kept into account accordingly. Moreover these services are functionally equivalent, despite they are not strongly equivalent.

Example 15. *We consider the world specification $\mathcal{W}$ of Example 9 enriched with the following axioms:*

$$\begin{aligned}
\mathsf{Grocery} &\sqsubseteq \mathsf{Shop}\\
\mathsf{ClosedShop} &\equiv \neg\exists \mathsf{authorizedFor}.\top
\end{aligned}$$

where Grocery *and* OpenShop *are obviously new concept names, and the following pair of services T_1:*

$$\begin{aligned}
&\boldsymbol{X} = \{x\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x \sqcap \mathsf{Grocery} \sqcap \mathsf{ClosedShop} \sqcap (\exists \mathsf{locatedIn}.\,\{\mathsf{town}_1\}) \sqcap (\exists \mathsf{owner}.\exists \mathsf{residentIn}.\,\{\mathsf{town}_1\})\\
&\quad \mathsf{and} \ \mathtt{not} \ (\exists \mathsf{owner}^{-}.x) \sqcap (\exists \mathsf{authorizedFor}.x)\\
&\mathcal{E} = \left\{\left\{+\mathsf{authorizedFor}(\exists \mathsf{owner}^{-}.x, x), -\mathsf{ClosedShop}(x)\right\}, \emptyset\right\}
\end{aligned}$$

and T_2:

$$\begin{aligned}
&\boldsymbol{X} = \{x\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x \sqcap \mathsf{Shop} \sqcap (\exists \mathsf{locatedIn}.\,\{\mathsf{town}_2\}) \ \mathsf{and} \ \mathtt{not} \ (\exists \mathsf{owner}^{-}.x) \sqcap (\exists \mathsf{authorizedFor}.x)\\
&\mathcal{E} = \left\{\left\{+\mathsf{authorizedFor}(\exists \mathsf{owner}^{-}.x, x)\right\}, \emptyset\right\}
\end{aligned}$$

Roughly speaking T_1 is a service (eventually) issuing authorizations to the owner for a special kind of shop in town town_1 *for its inhabitants, assuming that it is closed, since none has been previously authorized, while T_2 is a service issuing the same kind of authorization for the activity located in town* town_2*. These* *services are accomplishing a quite similar, but not exactly the same, kind of task, despite they are not directly replaceable. In fact, while T_1 is functionally comparable to T_2, but the converse is not true and, hence, they are not similar. Moreover the basic authorization service T can replace both of them.*

Given complexity results stated so far we are also able to provide a complexity upper bound for the functional comparability decision problem, given a candidate interface adapter[9].

Theorem 81. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a pair of valid (resp. repairable) services S_O and S_N, and a suitable interface adapter A, the problem of checking if the service S_N is functionally comparable with S_O is in $\mathsf{NP}^{\mathsf{NEEXP}}$.*

Proof. We prove the claim showing how querying a suitable oracle, we can design a non-deterministic machine solving the problem in polynomial time.

According to Theorem 76 (resp. 78), employing a $\mathsf{coNEEXP}$-problem oracle $\mathsf{Reachability}(\mathcal{W}, S_O, S_N, A)$ (resp. $\mathsf{RepReachability}(\mathcal{W}, S_O, S_N, A)$) it is possible to check whether the service S_N can replace the service S_O using the adapter A w.r.t. $\mathcal{W}$, while, according to Theorem 7, an EXP-problem oracle $\mathsf{Consistency}(\mathcal{W})$ can check whether the world specification is consistent, analogous oracles can be defined to check other service properties (accessibility, effect consistency, validity/repairability). We can, consequently, define a new non-deterministic automaton that initially guesses a possible renaming ξ from a finite search space (see Lemma 51), computes the renamed version of given specification (this step is clearly linear), then checks whether the renaming has preserved the interesting properties invoking the suitable oracle and, in case of successful verification, that the service replacement is correct using $\mathsf{Reachability}$ (resp. $\mathsf{RepReachability}$) on $\mathcal{W}^\xi, S_O^\xi, S_N^\xi, A^\xi$. (possibly considering the repair search spaces). The devised automaton clearly operates in non-deterministic polynomial time once a $\mathsf{coNEEXP}$-problem oracle is available[10], hence its complexity is at most $\mathsf{NP}^{\mathsf{NEEXP}}$. □

The previous results assume that the interface adapter w.r.t. checking the replacement has to provided as input to the problem, but since we are already paying for a search step for solve the decision problem, we can exploit the guessing step also to compute a candidate interface adapter if we can impose some syntactical restriction in order to bound the search to a finite space. Moreover, given the definition of query language, the following property ensures that the complexity class is untouched.

Proposition 3. *Given a concept alphabet $\boldsymbol{A}$, a role alphabet $\boldsymbol{P}$ and an object alphabet $\boldsymbol{O}$ and an integer value $h \in \mathbb{N}$, there exist at most $\mathcal{O}\left(n^{2^h}\right)$ distinct possible $\mathcal{ALCFIO}$ expression on such alphabets having a syntax tree of height less or equal h, where $n = \|\boldsymbol{O}\| + \|\boldsymbol{A}\| + \|\boldsymbol{P}\|$.*

[9]In the following we use relative complexity class notation according to the approach introduced in [LL76].

[10]We assume that also the generation of expanded formula, having a length exponentially bounded in the problem input, is delegated to the oracle itself. Generally speaking, the communication overhead between automata is ignored.

In other words, if we set the parameter h, the number of possible query expressions is polynomial bounded w.r.t. the size of the problem instance, even the polynomial degree can be very high. We can consequently adjust the comparability definition so that also this aspect is considered. We also limit the ability to express arbitrary quantified range restrictions to only functional ones, but it is not a so significant limitation in the query language as in the constraint one. Roughly speaking, $\mathcal{ALCFIO}$ a specialization of $\mathcal{ALCQIO}$ s.t. the concept expression language is:

$$C, C' \longrightarrow A \mid \neg C \mid C \sqcap C' \mid (\bowtie 1\, R\, C) \mid \{o\}$$

Obviously every $\mathcal{ALCFIO}$ expression is also in $\mathcal{ALCQIO}$, hence results shown so far are preserved.

Definition 109 (h-bounded functional comparability)**.** *Let h be an integer value, given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$ and a pair of valid (or repairable) services S_O and S_N, we say that S_N is h-functionally comparable with S_O if there exists a consistent object renaming ξ s.t. it preserves the validity (resp. repairability) of given services and S_N^ξ is a reachability preserving replacement of S_O^ξ using a suitable interface adapter A s.t. the height of each query $\mathcal{ALCFIO}$-expression tree is at most h.*

So, for a given h, the following results holds:

Corollary 19. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, an integer value $h \in \mathbb{N}$, and a pair of valid or repairable services S_O and S_N, the problem of checking if the service S_N is h-functionally comparable with S_O is in $\mathsf{NP}^{\mathsf{NEEXP}}$.*

Proof. The claim follows extending the automaton employed in the proof of Theorem 81 simply adding also the guessing of an interface adapter A whose size is, according to Proposition 3, polynomially bounded. □

A weaker form of such kind of property can be obtained combining the domain mapping with a service replaceability notion. In other words, two services are functionally similar w.r.t. a user goal G if, given an accordingly defined a mapping function, former is a suitable replacement for the latter, w.r.t. G. As in previous cases, the ability to reason considering the user goal enables us to implement more refined analysis, since we are able to filter out irrelevant service effects according to user's perspective.

Example 16. *Considering the scenario shown in Example 15 and the following service T_3 as variation of the service T_2:*

$\boldsymbol{X} = \{x\}$
$\boldsymbol{Y} = \emptyset$
$\mathcal{P} = x \sqcap \mathsf{Shop} \sqcap (\exists \mathsf{locatedIn}.\{\mathsf{town}_2\})$ `and not` $(\exists \mathsf{owner}^-.x) \sqcap (\exists \mathsf{authorizedFor}.x)$
$\mathcal{E} = \{E_1, E_2\}$

where E_1 and E_2 are two conditional effects defined as following:

$$E_1 = \begin{cases} \{+\mathsf{authorizedFor}(\exists\mathsf{owner}^-.x, x)\} & \text{if } x \sqcap \mathsf{Grocery} \\ \emptyset & \text{otherwise} \end{cases}$$

$$E_2 = \begin{cases} \{+\mathsf{authorizedFor}(\exists\mathsf{owner}^-.x, x)\} & \text{if } x \sqcap \mathsf{Grocery} \\ \{+\mathsf{authorizedFor}(\exists\mathsf{owner}^-.x, x)\} & \text{otherwise} \end{cases}$$

Roughly speaking, in case the shop x belongs to a particular class, the authorization is automatic (e.g., there exists a kind of market deregulation). Obviously this service cannot be considered as functional comparable with T_1: in fact, it does not preserve the reachability property, since it does never reject the client request. But, it is worth noticing that the requestor is not asking for a reject and its probably described by the following goal:

$$\begin{aligned} &\boldsymbol{U} = \{u\} \\ &\mathcal{H} = u \sqcap \mathsf{Grocery} \sqcap \mathsf{ClosedShop} \sqcap (\exists\mathsf{locatedIn}.\{\mathsf{town}_1\}) \\ &\quad \sqcap (\exists\mathsf{owner}.\exists\mathsf{residentIn}.\{\mathsf{town}_1\})\ \texttt{and not}\ (\exists\mathsf{owner}^-.u) \sqcap (\exists\mathsf{authorizedFor}.u) \\ &\mathcal{R} = \left\{+\mathsf{authorizedFor}(\exists\mathsf{owner}^-.u, u)\right\} \end{aligned}$$

In fact, w.r.t. such a goal the service adequacy turns to be strong uniform, so it is preserved considering the weak uniform adequacy of T_1. In other words, the service T_1 is functionally comparable to T_3 relative to the given goal.

In order to keep also user goals into account, we need to extend the object level transformation also to goal primitives.

Definition 110 (Renamed execution goal)**.** *Given a goal specification $G = \langle \boldsymbol{U}, \mathcal{H}, \mathcal{R} \rangle$, a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, and an object renaming ξ over $\boldsymbol{O}$, the renamed goal specification G^{ξ} is a goal specification on the same signature obtained from G applying the renaming function ξ to every condition query expressed as $\mathcal{ALCQIO}$-concept expression.*

A suitable object renaming must preserve the admissibility of the renamed goal:

Definition 111 (Admissibility preserving object renaming)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$ and an admissible goal G both defined on this domain, an object renaming ξ over $\boldsymbol{O}$ preserves the admissibility if the renamed goal specification G^{ξ} is also admissible.*

We can now provide a refined notion of functional comparison among service specification:

Definition 112 (Functional comparability w.r.t. a goal)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, an admissible goal G, and a pair of valid (or repairable) services S_O and S_N, we say that S_N is functionally comparable with S_O w.r.t. G if there exists a consistent object renaming ξ s.t. it preserves the validity (resp. repairability) of given services, the admissibility of G, and S_N^{ξ} is an adequacy preserving replacement of S_O^{ξ} w.r.t. G^{ξ}, possibly keeping into account the effect repair strategy.*

Analogous complexity results also hold for the verification of this kind of functional similarity:

Theorem 82. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a pair of valid services S_O and S_N, an admissible goal G s.t. S_O is adequate using a binding schema B, and a suitable interface adapter A, the problem of checking if the service S_N is functionally comparable with S_O w.r.t. G is in $\mathsf{NP}^{\mathsf{NEXP}}$.*

Proof. Assuming that an oracle for NEXP-problem was available, we prove the claim showing a non-deterministic automaton that solves this decision problem running in polynomial time, hence in $\mathsf{NP}^{\mathsf{NEXP}}$. In fact, given a problem instance, this automaton first guess a possible object renaming ξ, then it verifies whether such renaming results into consistent world specification, legal service specifications, and suitable interface adapters. Finally, the automaton check if the resulting service S_N^{ξ} can replace S_O^{ξ} using A^{ξ} and B^{ξ} w.r.t. $\mathcal{W}^{\xi}$ preserving the adequacy level: according to Theorems 66 and 80 this task can be accomplished using an oracle for NEXP or coNEXP language that first computes the original service's adequacy level l_o testing the various conditions, then it checks if the replacing service has the same level. Clearly this automaton solve the decision problem operating in linear time. □

Theorem 83. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a pair of repairable services S_O and S_N, an admissible goal G s.t. S_O is adequate a binding schema B, and a suitable interface adapter A, the problem of checking if the service S_N is functionally comparable with S_O w.r.t. G is in $\mathsf{NP}^{\mathsf{NEEXP}}$.*

Proof. The proof is quite similar to previous case, excepting it relies on results concerning goal adequacy under repairs (see Theorems 71, 72, and 80). □

Also the bounded comparability can be extended to this case, accordingly adjusting the definition. As in the case of bounded functional comparability, complexity upper-bounds are preserved.

Definition 113 (h-bounded goal relative functional comparability)**.** *Let h be an integer value, given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, an admissible goal G, and a pair of valid (or repairable) services S_O and S_N, we say that S_N is h-functionally comparable with S_O w.r.t. G if there exists a consistent object renaming ξ s.t. it preserves the validity (resp. repairability) of given services, the admissibility of G, and S_N^{ξ} is an adequacy preserving replacement of S_O^{ξ} w.r.t. G^{ξ}, possibly keeping into account the effect repair strategy, using a suitable interface adapter A s.t. the height of each query $\mathcal{ALCFIO}$-expression tree is at most h.*

Given the Proposition 3, we can add an interface adapter guessing step to automata previously designed, also in this case the following claims hold:

Corollary 20. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, an integer value $h \in \mathbb{N}$, a pair of valid services S_O and S_N, and an admissible goal G s.t. S_O is adequate using a binding schema B, the problem of checking if the service S_N is h-functionally comparable with S_O w.r.t. G is in $\mathsf{NP}^{\mathsf{NEXP}}$.*

Corollary 21. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, an integer value $h \in \mathbb{N}$, a pair of repairable services S_O and S_N, and an admissible goal G s.t. S_O is adequate a binding schema B, the problem of checking if the service S_N is h-functionally comparable with S_O w.r.t. G is in* $\mathsf{NP}^{\mathsf{NEEXP}}$.

Generally speaking, the service functional similarity can be assessed considering as distinct analysis dimensions the object renaming generation function and the replacement criterion. While the latter defines what is relevant in terms of service effects, the former specifies when different objects can be considered as similar, e.g., given a partition of $\mathbf{O}$ into equivalence classes we can generate only renaming s.t. an object o is replaced by another element of its class $[o]$. As observed for the repair analysis, the search space can be accordingly shaped w.r.t. application requirements.

Remark 45. *Mutual service replaceability and functional similarity are slightly interchangeable concepts*[11]*: but while in the former case the replacing service can be invoked in place of the original one for the same enactments, in the latter we invoke the services in different contexts (i.e., world specification) in order to achieve, possibly different, but homologous, goals. Generally speaking, the functional similarity concept is suitable to model dynamic binding patterns, while service replacement is more appropriate for dynamic addressing and service provider failure management.*

6.4 Service Template

A further generalization, stemming from the analysis of the functional similarity of e-services, is the introduction of a *service template* primitive, so that we are able to provide also a form of abstraction w.r.t. concrete deployments in a top-down manner, as the previous approach is essentially bottom-up. In terms of methodology, while we employ the bottom-up analysis in the assessment of a service community in order to cluster similar features, the top-down approach turns to be useful, e.g., in the construction of service directory or in the implementation of run-time resolution and binding mechanism, which according to each enactment properties selects the most suitable available service (if any).

So far we have pointed out that the extensional characterization is a typical aspect of service-oriented systems, in particular large scale ones (e.g., inter-enterprise B2B integration, e-government cooperation platform), while it is not extremely relevant into component-oriented systems (i.e., intra-enterprise integration), where a strong organization model is generally enforced and where only a limited degree of autonomy exists and the overlap of competences among different organization units is avoided. As a consequence, the formalization attempts generally ignore the extensional part[12], while in the present approach it play a central role. Moreover, we employ a definition of service template that essentially provide a form of abstraction w.r.t. the object level.

[11]The similarity/comparability relation actually generalizes the repairability one.

[12]In terms of DL knowledge engineering it is essentially represented by the object alphabet and the ABox, but other languages provide similar constructs (e.g., constant names, ground facts).

Now we introduce some primitive constructs in order to provide the definition of service template. Template parameters essentially range over object alphabet of a given domain specification and it can be employed instead of object names in query definitions.

Definition 114 (Template parameter assignment)**.** *Given a set of template parameter names $\boldsymbol{Z}$, a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, an assignment $\sigma_{\boldsymbol{Z}}$ is a function mapping a name in $\boldsymbol{Z}$ to an element of $\boldsymbol{O}$.*

Definition 115 (Parameterized query template)**.** *Given a domain specification, a set of template parameter names $\boldsymbol{Z}$, and a finite set of variable names $\boldsymbol{V}$, distinct from domain alphabets, a parameterized query template $Q_{\boldsymbol{Z}}(\boldsymbol{V})$ is an arbitrary $\mathcal{ALCQIO}$-concept expression built on the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{V}, \boldsymbol{P}, \boldsymbol{O} \cup \boldsymbol{Z} \rangle$.*

Generally we can define a fine grained approach to query manipulation, introducing:

an abstraction operator, that given a query or query template, an object name and a parameter name, returns a query template s.t. each occurrence of the object name has been replaced by the parameter name;

an application operator, that given a query template, a parameter name and an object name, return a query template or a query s.t. each occurrence of the parameter name has been replaced by the object name.

According to this approach, a query is a particular template having no parameter name occurrences in its definition.

Definition 116 (e-service template)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, an e-service template T is a quintuple formed by:*

- *a finite non-empty set of parameter names $\boldsymbol{Z}_T$;*
- *a (possibly empty) finite set of input variable names $\boldsymbol{X}_T$;*
- *a (possibly empty) finite set of output or instantiation variable names $\boldsymbol{Y}_T$;*
- *a (possibly empty) finite set of invocation precondition constraint templates $\mathcal{P}_T$*
- *a finite non-empty set of conditional effect templates $\mathcal{E}_T$.*

The definition of precondition constraint templates and conditional effect templates are similar to ones provided at pages 57, 58, and 109, excepting that they are expressed using query templates over $\mathbf{Z}_T$ instead of (ground) queries.

Definition 117 (Query template grounding)**.** *Given a query template $Q_{\boldsymbol{Z}}(\boldsymbol{V})$ and a template parameter assignment $\sigma_{\boldsymbol{Z}}$ we define the grounding or instantiation of the template over the assignment the query $Q(\boldsymbol{V})$ obtained replacing simultaneously each occurrence of template parameter $Z \in \boldsymbol{Z}$ in the expression with the assigned value $\sigma_{\boldsymbol{Z}}(Z)$.*

Definition 118 (e-service template grounding)**.** *Given an e-service template T and a template parameter assignment $\sigma_{\boldsymbol{Z}}$ over $\boldsymbol{Z}_T$, we define the grounding or instantiation of the template over the assignment the e-service specification $S = T(\sigma_{\boldsymbol{Z}})$ s.t.:*

- $\boldsymbol{X}_S = \boldsymbol{X}_T$ *and* $\boldsymbol{Y}_S = \boldsymbol{Y}_T$*;*
- $\mathcal{P}_S$ *and* $\mathcal{E}_S$ *are obtained from* $\mathcal{P}_T$ *and* $\mathcal{E}_T$ *by grounding every query template over* $\sigma_{\boldsymbol{Z}_T}$*.*

Since the object alphabet is finite, also the set of possible grounding of a template is finite, leading to the decidability of various semantic properties as we show in the following. This is essentially a consequence of the open-world assumption, since, while we are allowing for an infinite, even countable, interpretation universe $\mathfrak{U}$, the extensional part the specification (i.e., object names and the ABox) states only the incomplete and finite knowledge about the problem that we are interested to keep into account.

Proposition 4. *Given a set of object names* $\boldsymbol{O}$ *and a set of template parameter* $\boldsymbol{Z}$ *the number of possible distinct assignments is* $\|\boldsymbol{O}\|^{\|\boldsymbol{Z}\|}$*.*

Moreover, not every possible grounding assignment is interesting in this scenario, since we can provide a stronger selection criterion, based on the semantic characterization of functional similarity.

Definition 119 (Template service community)**.** *Given a domain specification, an e-service template* T*, and a finite set of template assignments* Σ *over* $\boldsymbol{Z}_T$*, we define as template service community* $\mathcal{S} = T(\Sigma)$ *the community of e-service obtained grounding* T *using every assignment in* Σ*.*

Among these communities we are interested into some specific ones:

Definition 120 (Normal template service community)**.** *A template service community* $T(\Sigma)$ *is normal iff, let* $\langle S_1, S_2\rangle \in T(\Sigma) \times T(\Sigma)$ *be a pair of community elements, they are functionally similar.*

In fact, despite two service specifications have been obtained instantiating the same template, they are not necessarily functionally comparable or similar, since not every parameter assignment pairs can be composed obtaining a suitable object renaming. Moreover, we can provide a simple sufficient condition to check this property.

Theorem 84. *Given a domain specification, an e-service template* T*, and a finite set of template assignments* Σ *over* $\boldsymbol{Z}_T$*, the template service community* $\mathcal{S} = T(\Sigma)$ *is normal if the following conditions hold:*

- *the assignments in* Σ *are injective;*
- *the assignment codomain is disjoint from the set of object names involved in query templates in* T*.*

Proof. This claim follows from these observations:

1. the injectivity of assignment ensures that these functions are invertible, so given an occurrence o of an object name into a query, it is at most associated with a variable name Z;
2. the disjoint property of assignment codomains and active alphabet involved in the query definition ensures also that, given an occurrence o of an object name into a query, or there exists a variable Z s.t. $\sigma_{\mathbf{Z}}(Z) = o$ or this occurrence was already defined in the template T.

Consequently, we observe that, let σ_1 and σ_2 be the assignments employed to instantiate S_1 and S_2 from T, we can use the object renamings:

$$\rho_1 = \{\sigma_1(Z)/\sigma_2(Z) | Z \in \mathbf{Z}\}$$
$$\rho_2 = \rho_1^{-1}$$

to prove that these services are functionally similar. □

As previously done for e-service specification we can now complete the analysis defining the corresponding semantic properties and providing an effective decision procedure. In particular, as we need to verify is a given service specification is accessible, consistent, valid/repairable, we need also to check whether a service template, given a world specification $\mathcal{W}$, can be instantiated or, in other words, there exists at least an admissible service obtained from it that is legal according to the previous definitions.

Definition 121 (Legal e-service template). *Given a domain specification, a service template T and a world specification $\mathcal{W}$, we say that the template is legal iff there exists at least a service S obtained from it s.t. it is legal w.r.t. $\mathcal{W}$.*

Such property can be refined also to keep into account a specific set of template parameter assignments Σ according to application requirements. Now, we can decompose this property into more foundational ones, providing the following definitions:

Definition 122 (Accessible e-service template). *Given a domain specification, a service template T and a world specification $\mathcal{W}$, we say that the template is accessible iff there exists at least a service S obtained from it s.t. it is accessible w.r.t. $\mathcal{W}$.*

Definition 123 (Non-uniform template consistency). *Given a domain specification, an accessible service template T and a world specification $\mathcal{W}$, we say that the template is non-uniformly consistent iff there exists at least a service S obtained from it s.t. it is accessible and the effect specifications are consistently defined w.r.t. $\mathcal{W}$.*

Definition 124 (Uniform template consistency). *Given a domain specification, an accessible service template T and a world specification $\mathcal{W}$, we say that the template is uniformly consistent iff for each accessible service S obtained it, the effect specifications are consistently defined w.r.t. $\mathcal{W}$.*

Definition 125 (Non-uniform template validity). *Given a domain specification, a non-uniformly consistent service template T and a world specification $\mathcal{W}$, we say that the template is non-uniformly valid iff there exists at least a service S obtained from it s.t. it is accessible, consistent and valid w.r.t. $\mathcal{W}$.*

Definition 126 (Uniform template validity). *Given a domain specification, a uniformly consistent service template T and a world specification $\mathcal{W}$, we say that the template is uniformly valid iff for each accessible service S obtained it, it is valid w.r.t. $\mathcal{W}$.*

Definition 127 (Non-uniform template repairability). *Given a domain specification, a non-uniformly consistent service template T and a world specification*

$\mathcal{W}$, we say that the template is non-uniformly repairable iff there exists at least a service S obtained from it s.t. it is accessible, consistent and repairable w.r.t. $\mathcal{W}$.

Definition 128 (Uniform template repairability)**.** *Given a domain specification, a uniformly consistent service template T and a world specification $\mathcal{W}$, we say that the template is uniformly repairable iff for each accessible service S obtained it, it is repairable w.r.t. $\mathcal{W}$.*

Example 17. *Considering the service S_1 introduced in Example 2 and subsequently modified, since in its definition is cited an object name (town_1), it is possible to employ the abstraction operator to generate the following service template:*

$$\begin{aligned}
&\boldsymbol{U} = \{u\}\\
&\boldsymbol{X} = \{x_1, x_2\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x_1 \sqcap \exists \mathsf{residentIn}.u \text{ and } x_2 \sqcap \mathsf{Town} \text{ and not } x_2 \sqcap u\\
&E = \big\{-\mathsf{residentIn}(x_1, \exists \mathsf{residentIn}^-.x_1), +\mathsf{residentIn}(x_1, x_2)\big\}\\
&\quad \cup \{-\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, u)\}\\
&\quad \cup \{+\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, x_2)\}
\end{aligned}$$

Instantiating the template using the assignment $\{u/\mathsf{town}_1\}$ will result into the service S_1 itself (or into an equivalent version), but applying the assignment $\{u/\mathsf{town}_2\}$ we obtain a similar version that has a different coverage $S_1^{\mathsf{town}_2}$:

$$\begin{aligned}
&\boldsymbol{X} = \{x_1, x_2\}\\
&\boldsymbol{Y} = \emptyset\\
&\mathcal{P} = x_1 \sqcap \exists \mathsf{residentIn}.\,\{\mathsf{town}_2\} \text{ and } x_2 \sqcap \mathsf{Town} \text{ and not } x_2 \sqcap \{\mathsf{town}_2\}\\
&E = \big\{-\mathsf{residentIn}(x_1, \exists \mathsf{residentIn}^-.x_1), +\mathsf{residentIn}(x_1, x_2)\big\}\\
&\quad \cup \{-\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, \{\mathsf{town}_2\})\}\\
&\quad \cup \{+\mathsf{registeredIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{owner}.x_1, x_2)\}
\end{aligned}$$

Extending the specification $\mathcal{W}$ adding more other Town *instances, we can generate a whole community of residence change services, one for each town. In this case, since object are intensionally indistinguishable (i.e., the interpretation substructures are isomorphic), all template instances share the formal properties, so it is possible to reduce the analysis of the template to the analysis of a single representative instance.*

Starting from previous definitions, considering results proved so far, we can easily design automata able to solve the related decision problems, providing an upper-bound of the computational complexity.

Theorem 85. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a service template T, the problem of checking if the template is accessible w.r.t. $\mathcal{W}$ is in $\mathsf{NP}^{\mathsf{NEXP}}$.*

Proof. In order to prove the claim, we need to show that a non-deterministic automaton using an oracle for a NEXP-language can solve a problem instance in a polynomial time. Such an automaton simply generate (guess) every possible instantiation assignment σ that, according to Proposition 4, are finite and then testing each one grounding the template $S = T(\sigma)$ and checking the service using the oracle for solve the satisfiability problem obtained applying Corollary 3, that according to Theorem 18 can be solve in NEXP: if the satisfiability has been verified for an assignment σ, the automaton answers true, otherwise false. Such an automaton clearly is able to check if the given template is accessible, since at least a computation branch has to returns true iff the template accessible, and also it operates in polynomial time. □

Theorem 86. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a service template T, the problem of checking if the template is non-uniformly consistent w.r.t. $\mathcal{W}$ is in* $\mathsf{NP}^{\mathsf{NEXP}}$.

Proof. As in the previous case, in order to prove the claim, we need to show that a non-deterministic automaton using oracles for a coNEXP-language can solve a problem instance in a polynomial time. Such an automaton simply generate (guess) every possible instantiation assignment σ that, according to Proposition 4, are finite and then testing each one grounding the template $S = T(\sigma)$ and checking the service using the oracle for solve the satisfiability problem obtained applying Corollary 3 (the service must be also accessible) and, hence, the entailment problem obtained from the Theorem 19. If both tests are successfully passed the automaton returns true, otherwise false. According to Theorem 20, a coNEXP-problem oracle can solve these reasoning tasks, hence the automaton operates non-deterministically in polynomial time. This automaton effectively solve the problem since the template is non-uniformly consistent iff there exists at least an assignment σ s.t. the corresponding branch answers true. □

Theorem 87. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a service template T, the problem of checking if the template is non-uniformly valid w.r.t. $\mathcal{W}$ is in* $\mathsf{NP}^{\mathsf{NEXP}}$.

Proof. The proof of this claim is quite similar to the previous one, excepting it relies on Theorems 47 and 48. □

Theorem 88. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a service template T, the problem of checking if the template is non-uniformly repairable w.r.t. $\mathcal{W}$ is in* $\mathsf{NP}^{\mathsf{NEEXP}}$.

Proof. The proof of this claim is quite similar to the previous one, excepting it relies on Theorems 52 and 53 and the automaton requires a more powerful oracle (coNEEXP-problem) to solve the instance. □

Theorem 89. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a service template T, the problem of checking if the template is uniformly consistent w.r.t. $\mathcal{W}$ is in* $\mathsf{coNP}^{\mathsf{NEXP}}$.

Proof. In this case we prove the claim showing how the complementary problem $\overline{\mathsf{UnifConsistency}}$ can be solved in $\mathsf{NP}^{\mathsf{NEXP}}$ and, consequently, $\mathsf{UnifConsistency} \in \mathsf{coNP}^{\mathsf{NEXP}}$. A template is not uniformly consistent iff there exists at least a grounding assignment s.t. the resulting service is accessible w.r.t. $\mathcal{W}$, but the effect specification is not consistent. Since the finite search space, we can adopt a guess-and-test strategy as done in the proof of Theorem 86, employing the same automaton simply negating the output value. The automaton for $\overline{\mathsf{UnifConsistency}}$ clearly operates in non-deterministic polynomial time once a suitable oracle is provided, thus the problem $\mathsf{UnifConsistency}$ is solvable in coNP given the same oracle. □

Theorem 90. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a service template T, the problem of checking if the template is uniformly valid w.r.t. $\mathcal{W}$ is in $\mathsf{coNP}^{\mathsf{NEXP}}$.*

Proof. The proof of this claim is quite similar to the previous one, excepting it relies on an automaton to solve the complementary problem $\overline{\mathsf{UnifValidity}}$. □

Theorem 91. *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, a consistent world specification $\mathcal{W}$, a service template T, the problem of checking if the template is uniformly repairable w.r.t. $\mathcal{W}$ is in $\mathsf{coNP}^{\mathsf{NEEXP}}$.*

Proof. Also in this case we adopt the same strategy, devising a $\mathsf{NP}^{\mathsf{NEEXP}}$ automaton to solve the complementary problem $\overline{\mathsf{UnifRepairability}}$. □

Complexity upper bound for the verification of template semantic properties are also summarized in Table 6.3. As shown in Example 17, in the case of service template generated abstracting a ground service (a seed), assuming a uniform (i.e., isomorphic) extensional specification (i.e., the ABox), it is possible to easily generalize the seed properties to the whole community.

	Non-uniform	Uniform
Consistency	$\mathsf{NP}^{\mathsf{NEXP}}$	$\mathsf{coNP}^{\mathsf{NEXP}}$
Validity	$\mathsf{NP}^{\mathsf{NEXP}}$	$\mathsf{coNP}^{\mathsf{NEXP}}$
Repairability	$\mathsf{NP}^{\mathsf{NEEXP}}$	$\mathsf{coNP}^{\mathsf{NEEXP}}$

Table 6.3: Verification complexity upper-bounds for template properties

6.5 Conclusions

In this chapter we have analyzed a variety of formal properties that are relevant in different phases of the life-cycle of a service-oriented system leveraging on the present framework.

According to the formal semantics that we have adopted, we are now able to show that reasoning about such a kind of properties is actually decidable, even though it can be very complex in highly expressive scenarios. The analysis of the replaceability has shown that the assessment performed w.r.t. the state space accessibility relation is not very satisfactory in client-driven applications (i.e., a service could not be able to reach a state that is not required by any client), so we have refined this property introducing a compatibility comparison w.r.t. the adequacy of services to achieve a given goal. The approach based on service templates has shown a complexity blow-up, but we remark that such a kind of constructs are essentially involved in the design phase, while pre-computed results can be reused at run-time, reducing the requirements for an on-line reasoner.

Moreover, we have analyzed these properties considering also the repairability of service effects. We point out that, as in the case of correctness analysis, the most considerable impact on the computational complexity of these problems is due to the necessity of keeping into account also possible repair strategies.

CHAPTER 7

Language Extensions

In this chapter we describe some extensions to the specification languages (i.e., constraints, queries) in order to provide a more expressive analysis framework. In particular, we are interested in language constructs related to the role specification and in the ability to deal with dynamic constraints (i.e., constraints enforced on the system evolution path rather than on the system state).

7.1 Extended Role Constructs

Expressive Description Logic language like $\mathcal{ALCQI}$ and $\mathcal{ALCQIO}$ are known to be able to model complex concept related properties, but they are weaker in the definition of properties related to roles. In fact, simply from the language syntax we can assess that while the concept expression language is very powerful, the role expression language is rather restricted (i.e., only the inverse role construct is provided).

However, there exist several language extensions (see [HSG04] for a survey) that aim to enrich the knowledge expression language with suitable primitives. In particular, such approaches extends the definition of the KB introducing:

- the ability to express complex role term (e.g., boolean composition, transitive closure, path composition) or, in other words, to enrich the expression language;
- the ability to express role based axioms (RBox) or to enrich the intensional part of the knowledge base.

Generally speaking, role-enriched languages, despite the expressive power, get at loosing some interesting and useful semantic and computational properties. Historically, first generation knowledge representation systems as KL-ONE allowed the user to define even more complex role-related axioms, but they are simple undecidable ([SS89]) or in other terms, the corresponding DL axiomatization resulted into an undecidable language. First families of expressive DLs as precisely $\mathcal{ALCQI}$ and $\mathcal{ALCQIO}$ allowing for general concept inclusion

(GCI) TBoxes, ABoxes, etc., are hence allowing for complex constructs only in concept definition axioms, but, nowadays, some promising attempts to safely extend these languages in terms of role constructs has been made achieving a new family of expressive DLs including languages as $\mathcal{SHOIN}(\mathbf{D})$ and $\mathcal{SROIQ}(\mathbf{D})$ ([HPS03], [HKS06]).

In particular, we are interested to exploit the *working* logic $\mathcal{C}^2$ capability to deal with some, despite not any possible, role-related terms, given its symmetry w.r.t. unary and binary predicates, generally missed by most of DLs. The resulting language is an extension of $\mathcal{ALCQIO}$ that enriches the role expression language as follows:

$$R, R' \longrightarrow P \mid R^- \mid \neg R \mid R \sqcap R' \mid R \upharpoonright C$$

Using these primitive constructs is also possible to define other operators on roles as: $\sqcup$, $\downharpoonright$, $\times$, $\triangledown$, and $\triangle$. The constraint language has been extended consequently: the TBox $\mathcal{T}$ allows also for *general role inclusion* (GRI) axioms in the form $R \sqsubseteq S$, where R and S are role expressions. In the spirit of [HSG04], we denote such language as $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$, listing the new primitive role operators.

Remark 46. *We point out that this language, since it allows also for nominals, is a proper extension of $\mathcal{ALB}$ (see [HS98]), that essentially enriches $\mathcal{ALC}$ with top role, role complement, role inverse, role intersection, and domain/range role restrictions.*

An interpretation $\mathcal{I}$ satisfies, or is a *model* of, the knowledge base iff, for each $R \sqsubseteq S \in \mathcal{T}$, $R^\mathcal{I} \subseteq S^\mathcal{I}$. The semantics of this language can be expressed in terms of first-order logic fragment $\mathcal{C}^2$ refining the definition provided in Section 3.2.2.

$$\begin{aligned}
\pi_x(A) &\triangleq A(x)\\
\pi_x(\neg C) &\triangleq \neg\pi_x(C)\\
\pi_x(C \sqcap C') &\triangleq \pi_x(C) \wedge \pi_x(C')\\
\pi_x(\bowtie n\, R\, C) &\triangleq \exists^{\bowtie n} y.\pi_x y(R) \wedge \pi_y(C))\\
\pi_y(C) &\triangleq \pi_x(C)[x/y, y/x]\\
\pi_{xy}(P) &\triangleq P(x, y)\\
\pi_{xy}(R^-) &\triangleq \pi_{yx}(R)\\
\pi_{xy}(\neg R) &\triangleq \neg\pi_{xy}(R)\\
\pi_{xy}(R \sqcap R') &\triangleq \pi_{xy}(R) \wedge \pi_{xy}(R')\\
\pi_{xy}(R \upharpoonright C) &\triangleq \pi_{xy}(R) \wedge \pi_x(C)\\
\pi_{yx}(R) &\triangleq \pi_{xy}(R)[x/y, y/x]\\
\pi(\sharp(C) \bowtie n) &\triangleq \exists^{\bowtie n} x.\pi_x(C)\\
\pi(C \sqsubseteq D) &\triangleq \forall x.\pi_x(C) \rightarrow \pi_x(D)\\
\pi(R \sqsubseteq S) &\triangleq \forall x, y.\pi_{xy}(R) \rightarrow \pi_{xy}(S)
\end{aligned}$$

As in the previous case, nominals has been replaced using singleton concepts. Also the identity relation *id* can be expressed introducing a role name id and adequate constraints ($\forall x.\mathsf{id}(x, x)$ and $\forall x.\exists^{=1} y.\mathsf{id}(x, y)$).

Example 18. *Using these new kind of languages, we can easily model some more complex constraints.*

Considering the simple domain introduced in the Example 2, we need, now, also to express the following requirements: a citizen can be authorized only for activities held by itself. As shown in the Example 15, we can model such a restrictions using service preconditions, but what about the possibility to specify it as world constraint, that must be always enforced no matter how services are defined?

Clearly, introducing role inclusion axioms, a simple role assertion as the following is able to capture such application requirement:

$$\mathsf{authorizedFor} \sqsubseteq \mathsf{owner} \downharpoonright \mathsf{Shop}$$

Remark 47. *Interestingly, the devised repair strategy can easily infer the repair* $+\mathsf{owner}(x_1, x_2)$ *for the citizen* x_1 *and the shop* x_2*, in the case the constraint should be violated by a service enactment. Despite this inference is technically correct it seems "semantically" puzzling, since the service make the requesting citizen also the owner of the shop. In other words, the repair algorithm need to be tuned in different situations, accordingly shaping the search solution space, allowing or denying the possibility to repair a given group of predicates.*

This aspect is addressed in [BLM+05b] by means of the occlusion *restrictions: a similar approach can easily be adapted to the present framework.*

Example 19. *We extend the specification presented in previous examples, introducing the new concept of* District *and the new properties* partOf*,* headTown*, to denote, resp., administrative district and the part-of relation and the head town of a district. We can add various additional axioms as, in example:*

$$\mathsf{District} \sqcap A \sqsubseteq \bot$$

for every $A \in \boldsymbol{A} \setminus \{\mathsf{District}\}$*, in order to assert that this concept extension is disjoint from other ones*[1]*. W.l.o.g., we can also assume that a district is composed by towns and that an empty district cannot be defined, as well as a town can be assigned only to a district:*

$$\begin{array}{rcl} \mathsf{District} & \sqsubseteq & \exists \mathsf{partOf}^{-}.\mathsf{Town} \\ \mathsf{District} & \sqsubseteq & \forall \mathsf{partOf}^{-}.\mathsf{Town} \\ \mathsf{Town} & \sqsubseteq & (= 1\ \mathsf{partOf}\ \mathsf{District}) \end{array}$$

Similar axioms also are introduced for the property headTown*:*

$$\begin{array}{rcl} \exists \mathsf{headTown}^{-}.\top & \sqsubseteq & \mathsf{District} \\ \exists \mathsf{headTown}.\top & \sqsubseteq & \mathsf{Town} \\ \mathsf{District} & \sqsubseteq & (= 1\ \mathsf{headTown}\ \top) \end{array}$$

We need, also, to implement the requirement that restrict the head-town of a district to be a town assigned to district itself. This kind of constraint requires the ability to express role-based axioms as:

$$\mathsf{homeTown} \sqsubseteq \mathsf{partOf}^{-} \downharpoonright \mathsf{Town} \upharpoonright \mathsf{District}$$

[1]Whether there exist some sub-concepts of District, this kind of meta-axiom must be accordingly adjusted.

In order to extend to this new language the results presented so far, we need essentially to adequately adjust the notion of embedding relation and related concepts that allows to translate the axioms provided in the problem specification. We observe that definition of embedding relation can be left untouched, while the translation function τ (and its generalizations) needs introduced at page 33, to be extended to deal with new role operators too. The extended translation function τ^* rewrites any arbitrary concept or role expression in $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ over the alphabet $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$ to a new concept or role expression in $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ over the alphabet $\langle \mathbf{A} \cup \{\mathsf{Top}\}, \mathbf{P}, \mathbf{O} \rangle$, and it defined as follows:

$$
\begin{aligned}
\tau^*(A) &\triangleq A \\
\tau^*(C \sqcap C') &\triangleq \tau(C) \sqcap \tau^*(C') \\
\tau^*((\bowtie\ n\, R\, C)) &\triangleq (\bowtie\ n\, \tau^*(R)\, \tau^*(C)) \\
\tau^*(\{o\}) &\triangleq \{o\} \\
\tau^*(\neg C) &\triangleq \mathsf{Top} \sqcap \neg \tau^*(C) \\
\tau^*(P) &\triangleq P \\
\tau^*(R^-) &\triangleq \tau^*(R)^- \\
\tau^*(R \sqcap R') &\triangleq \tau^*(R) \sqcap \tau^*(R') \\
\tau^*(\neg R) &\triangleq (\mathsf{Top} \times \mathsf{Top}) \sqcap \neg \tau^*(R) \\
\tau^*(R \upharpoonright C) &\triangleq \tau^*(R) \upharpoonright \tau^*(C)
\end{aligned}
$$

The foundational property of this function is that it preserves the interpretation of arbitrary expressions through the embedding relation among the structures. In Theorems 1 and 2, we have shown has this property holds for the basic $\mathcal{ALCQIO}$ role and concept expression languages, now we need to extend such result to a more general case.

Theorem 92. *Let be ω and $\hat{\omega}$ be respectively a world state and an arbitrary interpretation s.t. the world state is embedded into the interpretation ($\omega \rightsquigarrow \hat{\omega}$), then:*

$$C^\omega = [\tau^*(C)]^{\hat{\omega}}$$

for any $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ concept expression C built over the domain specification alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$ and:

$$R^\omega = [\tau^*(R)]^{\hat{\omega}}$$

for any $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ role expression R built over the domain specification alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$.

Proof. We prove the theorem by mutual induction over the concept and role expression languages.

1. $N^\omega = [\tau^*(N)]^{\hat{\omega}}$ where $N \in \mathbf{A} \cup \mathbf{P}$. By the definition of translation function τ^*:

   $$[\tau^*(N)]^{\hat{\omega}} = N^{\hat{\omega}}$$

   By the definition of the embedding relation between the structures:

   $$N^{\hat{\omega}} = N^\omega$$

2. $[\{o_1,\ldots,o_n\}]^{\omega} = [\tau^*(\{o_1,\ldots,o_n\})]^{\hat{\omega}}$ where $\{o_1,\ldots,o_n\} \subseteq \mathbf{O}$. By the definition of translation function τ^*:

$$[\tau^*(\{o_1,\ldots,o_n\})]^{\hat{\omega}} = [\{o_1,\ldots,o_n\}]^{\hat{\omega}}$$

   According to the standard semantics:

$$[\{o_1,\ldots,o_n\}]^{\hat{\omega}} = \bigcup_{i=1}^{n} o_i^{\hat{\omega}}$$

   By the definition of the embedding relation between the structures:

$$\bigcup_{i=1}^{n} o_i^{\hat{\omega}} = \bigcup_{i=1}^{n} o_i^{\omega}$$

   According to the standard semantics:

$$\bigcup_{i=1}^{n} o_i^{\omega} = [\{o_1,\ldots,o_n\}]^{\omega}$$

3. $[C \sqcap C']^{\omega} = [\tau^*(C \sqcap C')]^{\hat{\omega}}$. By the definition of translation function τ^*:

$$[\tau(C \sqcap C')]^{\hat{\omega}} = [\tau(C) \sqcap \tau(C')]^{\hat{\omega}}$$

   According to the standard semantics:

$$[\tau^*(C) \sqcap \tau^*(C')]^{\hat{\omega}} = \tau^*(C)^{\hat{\omega}} \cap \tau^*(C')^{\hat{\omega}}$$

   By the inductive hypothesis:

$$\tau^*(C)^{\hat{\omega}} \cap \tau^*(C')^{\hat{\omega}} = C^{\omega} \cap C'^{\omega}$$

   According to the standard semantics:

$$C^{\omega} \cap C'^{\omega} = [C \sqcap C']^{\omega}$$

4. $[R \sqcap R']^{\omega} = [\tau^*(R \sqcap R')]^{\hat{\omega}}$. This case can be proved using the same argumentation of the previous one since the symmetry.

5. $[R^-]^{\omega} = [\tau^*(R^-)]^{\hat{\omega}}$. By the definition of translation function τ^*:

$$[\tau^*(R^-)]^{\hat{\omega}} = [\tau^*(R)^-]^{\hat{\omega}}$$

   According to standard semantics:

$$[\tau^*(R)^-]^{\hat{\omega}} = \left\{\langle\beta,\alpha\rangle | \langle\alpha,\beta\rangle \in \tau^*(R)^{\hat{\omega}}\right\}$$

   By the inductive hypothesis:

$$\left\{\langle\beta,\alpha\rangle | \langle\alpha,\beta\rangle \in \tau^*(R)^{\hat{\omega}}\right\} = \{\langle\beta,\alpha\rangle | \langle\alpha,\beta\rangle \in R^{\omega}\}$$

   Applying too the standard semantics, we can conclude that:

$$\{\langle\beta,\alpha\rangle | \langle\alpha,\beta\rangle \in R^{\omega}\} = [R^-]^{\omega}$$

6. $[\neg C]^{\omega} = [\tau^*(\neg C)]^{\hat{\omega}}$. By the definition of translation function τ^*:

$$[\tau^*(\neg C)]^{\hat{\omega}} = [\mathsf{Top} \sqcap \neg\tau^*(C)]^{\hat{\omega}}$$

According to the standard semantics:

$$[\mathsf{Top} \sqcap \neg\tau^*(C)]^{\hat{\omega}} = \mathsf{Top}^{\hat{\omega}} \cap [\neg\tau^*(C)]^{\hat{\omega}}$$

$$\mathsf{Top}^{\hat{\omega}} \cap [\neg\tau^*(C)]^{\hat{\omega}} = \mathsf{Top}^{\hat{\omega}} \cap (\mathfrak{U} \setminus [\tau^*(C)]^{\hat{\omega}})$$

By the inductive hypothesis:

$$\mathsf{Top}^{\hat{\omega}} \cap (\mathfrak{U} \setminus [\tau^*(C)]^{\hat{\omega}}) = \mathsf{Top}^{\hat{\omega}} \cap (\mathfrak{U} \setminus C^{\omega})$$

By the definition of the embedding relation between the structures:

$$\mathsf{Top}^{\hat{\omega}} \cap (\mathfrak{U} \setminus C^{\omega}) = \Delta^{\omega} \cap (\mathfrak{U} \setminus C^{\omega})$$

Since $\Delta^{\omega} \subseteq \mathfrak{U}$ and $C^{\omega} \subseteq \mathfrak{U}$:

$$\Delta^{\omega} \cap (\mathfrak{U} \setminus C^{\omega}) = \Delta^{\omega} \setminus C^{\omega}$$

According to the standard semantics:

$$\Delta^{\omega} \setminus C^{\omega} = [\neg C]^{\omega}$$

7. $[\neg R]^{\omega} = [\tau^*(\neg R)]^{\hat{\omega}}$. By the definition of translation function τ^*:

$$[\tau^*(\neg R)]^{\hat{\omega}} = [(\mathsf{Top} \times \mathsf{Top}) \sqcap \neg\tau^*(R)]^{\hat{\omega}}$$

According to the standard semantics:

$$[(\mathsf{Top} \times \mathsf{Top}) \sqcap \neg\tau^*(R)]^{\hat{\omega}} = (\mathsf{Top}^{\hat{\omega}} \times \mathsf{Top}^{\hat{\omega}}) \cap [\neg\tau^*(R)]^{\hat{\omega}}$$

$$(\mathsf{Top}^{\hat{\omega}} \times \mathsf{Top}^{\hat{\omega}}) \cap [\neg\tau^*(R)]^{\hat{\omega}} = (\mathsf{Top}^{\hat{\omega}} \times \mathsf{Top}^{\hat{\omega}}) \cap ((\mathfrak{U} \times \mathfrak{U}) \setminus [\tau^*(R)]^{\hat{\omega}})$$

By the inductive hypothesis:

$$(\mathsf{Top}^{\hat{\omega}} \times \mathsf{Top}^{\hat{\omega}}) \cap ((\mathfrak{U} \times \mathfrak{U}) \setminus [\tau^*(R)]^{\hat{\omega}}) = (\mathsf{Top}^{\hat{\omega}} \times \mathsf{Top}^{\hat{\omega}}) \cap ((\mathfrak{U} \times \mathfrak{U}) \setminus R^{\omega})$$

By the definition of the embedding relation between the structures:

$$(\mathsf{Top}^{\hat{\omega}} \times \mathsf{Top}^{\hat{\omega}}) \cap ((\mathfrak{U} \times \mathfrak{U}) \setminus R^{\omega}) = (\Delta^{\omega} \times \Delta^{\omega}) \cap ((\mathfrak{U} \times \mathfrak{U}) \setminus R^{\omega})$$

Since $\Delta^{\omega} \subseteq \mathfrak{U}$ and $R^{\omega} \subseteq \mathfrak{U} \times \mathfrak{U}$:

$$(\Delta^{\omega} \times \Delta^{\omega}) \cap ((\mathfrak{U} \times \mathfrak{U}) \setminus R^{\omega}) = (\Delta^{\omega} \times \Delta^{\omega}) \setminus R^{\omega}$$

According to the standard semantics:

$$(\Delta^{\omega} \times \Delta^{\omega}) \setminus R^{\omega} = [\neg R]^{\omega}$$

8. $[(\geq\ n\, R\, C)]^{\omega} = [\tau^{*}((\geq\ n\, R\, C))]^{\hat{\omega}}$, where R is an arbitrary role expression. By the definition of translation function τ^{*}:

$$[\tau^{*}((\geq\ n\, R\, C))]^{\hat{\omega}} = [(\geq\ n\, \tau^{*}(R)\, \tau(C))]^{\hat{\omega}}$$

According to the standard semantics:

$$[(\geq\ n\, \tau^{*}(R)\, \tau^{*}(C))]^{\hat{\omega}} = \left\{\alpha |\ \left\| S^{\hat{\omega}}_{\tau^{*}(C),\tau^{*}(R)}(\alpha) \right\| \geq n \right\} \tag{7.1}$$

where $S^{\hat{\omega}}_{\tau^{*}(C),\tau^{*}(R)}(\alpha)$ is the set of $\tau^{*}(R)$-successors of element α belonging to $\tau^{*}(C)$ defined as:

$$S^{\hat{\omega}}_{\tau^{*}(C),\tau^{*}(R)}(\alpha) = \left\{\beta | \beta \in \tau(C)^{\hat{\omega}}, \langle \alpha, \beta \rangle \in \tau^{*}(R)^{\hat{\omega}} \right\}$$

But, since the inductive hypothesis, we have that:

$$S^{\hat{\omega}}_{\tau^{*}(C),\tau^{*}(R)}(\alpha) = \{\beta | \beta \in C^{\omega}, \langle \alpha, \beta \rangle \in R^{\omega}\}$$

In other words, we can conclude that:

$$S^{\hat{\omega}}_{\tau^{*}(C),\tau^{*}(R)}(\alpha) = S^{\omega}_{C,R}(\alpha)$$

Applying such result back to Equation 7.1, we have that:

$$[(\geq\ n\, \tau^{*}(R)\, \tau^{*}(C))]^{\hat{\omega}} = \left\{\alpha |\ \left\| S^{\omega}_{C,R}(\alpha) \right\| \geq n \right\}$$

Observing, according to standard semantics, that:

$$[(\geq\ n\, R\, C)]^{\omega} = \left\{\alpha |\ \left\| S^{\omega}_{C,R}(\alpha) \right\| \geq n \right\}$$

we have proved the claim.

9. $[R \upharpoonright C]^{\omega} = [\tau^{*}(R \upharpoonright C)]^{\hat{\omega}}$. By the definition of translation function τ^{*}:

$$[\tau^{*}(R \upharpoonright C)]^{\hat{\omega}} = [\tau^{*}(R) \upharpoonright \tau^{*}(C)]^{\hat{\omega}}$$

According to standard semantics:

$$[\tau^{*}(R) \upharpoonright \tau^{*}(C)]^{\hat{\omega}} = \left\{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in \tau^{*}(R)^{\hat{\omega}}, \alpha \in \tau^{*}(C)^{\hat{\omega}} \right\}$$

By the inductive hypothesis:

$$\left\{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in \tau^{*}(R)^{\hat{\omega}}, \alpha \in \tau^{*}(C)^{\hat{\omega}} \right\} = \{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in R^{\omega}, \alpha \in C^{\omega}\}$$

Applying too the standard semantics we can conclude that:

$$\{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in R^{\omega}, \alpha \in C^{\omega}\} = [R \upharpoonright C]^{\omega}$$

□

We extend also the definition of translation function to knowledge base having role inclusion assertion, as the following. Let $KB = \langle \mathcal{T}, \mathcal{A} \rangle$ be an arbitrary $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ knowledge base built over the domain specification (i.e., a world specification $\mathcal{W}$), we define a new knowledge base $\tau^*(KB)$ over the extended alphabet s.t. for each general inclusion assertion $C \sqsubseteq D$ in the TBox $\mathcal{T}$ it includes a new axiom of the form:

$$\tau^*(C) \sqsubseteq \tau^*(D)$$

for each role inclusion assertion $R \sqsubseteq S$ in the TBox $\mathcal{T}$ it includes a new axiom of the form:

$$\tau^*(R) \sqsubseteq \tau^*(S)$$

for each ABox assertion $o : C$ in $\mathcal{A}$ it includes a new axiom of the form:

$$o : \tau^*(C)$$

and, for each ABox assertion $(o, o') : R$ in $\mathcal{A}$ it includes a new axiom of the form:

$$(o, o') : \tau^*(R)$$

Given this definition and applying Theorem 92, we can also extends Theorems 3 and 4 to this case, obtaining the following claims:

Theorem 93. *If a world state ω is a model of the $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ knowledge base KB, then the structure $\hat{\omega} = \mu(\omega)$ is a model of the knowledge base $\tilde{KB} \wedge \tau^*(KB)$.*

Theorem 94. *If $\hat{\omega}$ is a model of the knowledge base $\tilde{KB} \wedge \tau^*(KB)$, then the interpretation $\omega = \pi(\hat{\omega})$ is a world state that satisfies the $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ knowledge base KB.*

Corollary 22. *The $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ knowledge base KB is satisfiable on an arbitrary interpretation domain $\Delta \subseteq \mathfrak{U}$ iff the knowledge base $\tilde{KB} \wedge \tau^*(KB)$ is satisfiable on $\mathfrak{U}$.*

According to [LS01], the reasoning in $\mathcal{ALB}$ is NEXP-complete, hence, given the reduction to reasoning in $\mathcal{C}^2$ we can also state the following results:

Theorem 95. *Given an $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$ world specification $\mathcal{W}$, the problem of checking if it is consistent is* NEXP*-complete.*

Moreover, as discussed in Chapter 3, we have pointed out that not for any world specification language the original knowledge base KB that is satisfiable on a restricted active domain $\Delta^\omega \subsetneq \mathfrak{U}$ is also satisfiable on the whole interpretation universe $\mathfrak{U}$. In the case of $\mathcal{ALCQIO}$ we have proved that since cardinality restrictions it is possible, for example, to force the knowledge to be satisfiable only on finite models. This is essentially due to the lack of the disjoint union model property for nominal-enriched languages. What is the situation in the case of arbitrary boolean role constructs? Is it sufficient to renounce to nominals to achieve such a property? The answer is in general negative as the following example shows.

Example 20. *Given a simple knowledge KB base having the following axiom in the TBox:*

$$\neg R \sqsubseteq S$$

where R and S are two role names, we consider two disjoint models ω and ω'. Let x and x' be, resp., an element of ω and an element of ω', the resulting disjoint union $\omega \cup \omega'$ is not a model of KB since the pair $\langle x, x' \rangle$ does not belong to $R^{\omega \cup \omega'}$, but it neither belongs to $S^{\omega \cup \omega'}$, violating the axiom.

We now consider a new language that removes nominal and role negation, but maintains other main role operators. This language is denoted as $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonright, \downharpoonright, \triangle)$ and is defined from $\mathcal{ALCQI}$ introducing the following production rule for role expressions:

$$R, R' \longrightarrow P \mid R^- \mid R \sqcup R' \mid R \sqcap R' \mid R \upharpoonright C \mid R \downharpoonright C \mid \triangle$$

It is clearly a sub-language of $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonright)$, since additional constructs can be obtained combining primitive ones. For this language, we can state the following property:

Lemma 52. *Given an arbitrary $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonright, \downharpoonright, \triangle)$ role or concept expression E and two interpretation structures $\omega' = \langle \Delta^{\omega'}, \cdot^{\omega'} \rangle$ and $\omega'' = \langle \Delta^{\omega''}, \cdot^{\omega''} \rangle$, s.t. the interpretation domains are disjoint ($\Delta^{\omega'} \cap \Delta^{\omega''} = \emptyset$), then, let $\omega = \langle \Delta^{\omega'} \cup \Delta^{\omega''}, \cdot^{\omega' \cup \omega''} \rangle$ be the interpretation structure obtained by the union of ω' and ω'', $E^{\omega} = E^{\omega'} \cup E^{\omega''}$.*

Proof. We prove the claim by induction on the expression language.

1. $N^{\omega} = N^{\omega'} \cup N^{\omega''}$ where $N \in \mathbf{A} \cup \mathbf{P}$. This case follows from the definition of disjoint union structure ω, since $A^{\omega} = A^{\omega'} \cup A^{\omega''}$ for any $A \in \mathbf{A}$ and $P^{\omega} = P^{\omega'} \cup P^{\omega''}$ for any $P \in \mathbf{P}$

2. $[C \sqcap C']^{\omega} = [C \sqcap C']^{\omega'} \cup [C \sqcap C']^{\omega''}$. According to standard semantics:

   $$[C \sqcap C']^{\omega} = C^{\omega} \cap C'^{\omega}$$

   By inductive hypothesis:

   $$C^{\omega} \cap C'^{\omega} = (C^{\omega'} \cup C^{\omega''}) \cap (C'^{\omega'} \cup C'^{\omega''})$$

   Given properties of set operators, we can rearrange the expression as following:

   $$(C^{\omega'} \cup C^{\omega''}) \cap (C'^{\omega'} \cup C'^{\omega''}) = (C^{\omega'} \cap C'^{\omega'}) \cup (C^{\omega''} \cap C'^{\omega'}) \cup (C^{\omega'} \cap C'^{\omega''}) \cup (C^{\omega''} \cap C'^{\omega''})$$

   But since $C^{\omega'} \subseteq \Delta^{\omega'}$, $C'^{\omega'} \subseteq \Delta^{\omega'}$, $C^{\omega''} \subseteq \Delta^{\omega'}$, $C'^{\omega''} \subseteq \Delta^{\omega'}$, and given the disjointness hypothesis we have that:

   $$(C^{\omega'} \cap C'^{\omega'}) \cup (C^{\omega''} \cap C'^{\omega'}) \cup (C^{\omega'} \cap C'^{\omega''}) \cup (C^{\omega''} \cap C'^{\omega''}) = (C^{\omega'} \cap C'^{\omega'}) \cup (C^{\omega''} \cap C'^{\omega''})$$

   Applying the standard semantics, we finally obtain that:

   $$(C^{\omega'} \cap C'^{\omega'}) \cup (C^{\omega''} \cap C'^{\omega''}) = [C \sqcap C']^{\omega'} \cup [C \sqcap C']^{\omega''}$$

3. $[R \sqcap R']^{\omega} = [R \sqcap R']^{\omega'} \cup [R \sqcap R']^{\omega''}$. This case can be proved using the same argumentation of the previous one, observing that $R^{\omega'} \subseteq \Delta^{\omega'} \times \Delta^{\omega'}$, $R'^{\omega'} \subseteq \Delta^{\omega'} \times \Delta^{\omega'}$, $R^{\omega''} \subseteq \Delta^{\omega'} \times \Delta^{\omega'}$, $R'^{\omega''} \subseteq \Delta^{\omega'} \times \Delta^{\omega'}$.

4. $[R \sqcup R']^{\omega} = [R \sqcup R']^{\omega'} \cup [R \sqcup R']^{\omega''}$. According to standard semantics:

   $$[R \sqcup R']^{\omega} = R^{\omega} \cup R'^{\omega}$$

   By inductive hypothesis:

   $$R^{\omega} \cup R'^{\omega} = (R^{\omega'} \cup R^{\omega''}) \cup (R'^{\omega'} \cup R'^{\omega''})$$

We can simply rearrange the expression as following:

$$(R^{\omega'} \cup R^{\omega''}) \cup (R'^{\omega'} \cup R'^{\omega''}) = (R^{\omega'} \cup R'^{\omega'}) \cup (R^{\omega''} \cup R'^{\omega''})$$

Applying the standard semantics, it follows that:

$$(R^{\omega'} \cup R'^{\omega'}) \cup (R^{\omega''} \cup R'^{\omega''}) = [R \sqcup R']^{\omega'} \cup [R \sqcup R']^{\omega''}$$

5. $[\neg C]^{\omega} = [\neg C]^{\omega'} \cup [\neg C]^{\omega''}$. According to the standard semantics of negation, we obtain that:

$$[\neg C]^{\omega} = \Delta^{\omega} \setminus C^{\omega}$$

Moreover, by the inductive hypothesis and the definition of model union:

$$\Delta^{\omega} \setminus C^{\omega} = (\Delta^{\omega'} \cup \Delta^{\omega'}) \setminus (C^{\omega'} \cup C^{\omega''})$$

We can rearrange the expression as the following:

$$(\Delta^{\omega'} \cup \Delta^{\omega'}) \setminus (C^{\omega'} \cup C^{\omega''}) = (\Delta^{\omega'} \setminus (C^{\omega'} \cup C^{\omega''})) \cup (\Delta^{\omega''} \setminus (C^{\omega'} \cup C^{\omega''}))$$

But, since $\Delta^{\omega''} \cap C^{\omega'} = \emptyset$ and $\Delta^{\omega'} \cap C^{\omega''} = \emptyset$, we can conclude that:

$$(\Delta^{\omega'} \cup \Delta^{\omega'}) \setminus (C^{\omega'} \cup C^{\omega''}) = (\Delta^{\omega'} \setminus (C^{\omega'}) \cup (\Delta^{\omega'}) \setminus C^{\omega''})$$

According to semantics, we obtain that:

$$(\Delta^{\omega'} \setminus C^{\omega'}) \cup (\Delta^{\omega'} \setminus C^{\omega''}) = [\neg C]^{\omega'} \cup [\neg C]^{\omega''}$$

6. $[R^{-}]^{\omega} = [R^{-}]^{\omega'} \cup [R^{-}]^{\omega''}$. According to the definition of inverse role operator, it follows that:

$$[R^{-}]^{\omega} = \{\langle \beta, \alpha \rangle | \langle \alpha, \beta \rangle \in R^{\omega}\}$$

By induction, we have that:

$$\{\langle \beta, \alpha \rangle | \langle \alpha, \beta \rangle \in R^{\omega}\} = \left\{\langle \beta, \alpha \rangle | \langle \alpha, \beta \rangle \in R^{\omega'} \cup R^{\omega''}\right\}$$

Since $R^{\omega'} \cap R^{\omega''} = \emptyset$, we can rewrite the previous expression as:

$$\left\{\langle \beta, \alpha \rangle | \langle \alpha, \beta \rangle \in R^{\omega'} \cup R^{\omega''}\right\} = \left\{\langle \beta, \alpha \rangle | \langle \alpha, \beta \rangle \in R^{\omega'}\right\} \cup \left\{\langle \beta, \alpha \rangle | \langle \alpha, \beta \rangle \in R^{\omega''}\right\}$$

According to operator semantics, we can finally conclude that:

$$\left\{\langle \beta, \alpha \rangle | \langle \alpha, \beta \rangle \in R^{\omega'}\right\} \cup \left\{\langle \beta, \alpha \rangle | \langle \alpha, \beta \rangle \in R^{\omega''}\right\} = [R^{-}]^{\omega'} \cup [R^{-}]^{\omega''}$$

7. $[(\geq\ n\, R\, C)]^{\omega} = [(\geq\ n\, R\, C)]^{\omega'} \cup [(\geq\ n\, R\, C)]^{\omega''}$. We start applying the definition of the operator semantics:

$$[(\geq\ n\, R\, C)]^{\omega} = \{\alpha | \alpha \in \delta^{\omega}, \|\{\beta | \beta \in C^{\omega}, \langle \alpha, \beta \rangle \in R^{\omega}\}\| \geq n\}$$

We split this set into two components defined as:

$$[(\geq\ n\, R\, C)]^{\omega} \cap \Delta^{\omega'} = \left\{\alpha | \alpha \in \Delta^{\omega'}, \|\{\beta | \beta \in C^{\omega}, \langle \alpha, \beta \rangle \in R^{\omega}\}\| \geq n\right\}$$

$$[(\geq\ n\, R\, C)]^{\omega} \cap \Delta^{\omega''} = \left\{\alpha | \alpha \in \Delta^{\omega''}, \|\{\beta | \beta \in C^{\omega}, \langle \alpha, \beta \rangle \in R^{\omega}\}\| \geq n\right\}$$

Since interpretation domains are disjoint, this is a partitioning of the original set. Since the symmetry, we consider only the first partition. By induction, it follows that:

$$\begin{aligned}[(\geq\ n\,R\,C)]^{\omega} \cap \Delta^{\omega'} &= \left\{\alpha | \alpha \in \Delta^{\omega'}, \left\| S^{\omega' \cup \omega''}_{C,R}(\alpha) \right\| \geq n \right\} \\ &= \left\{\alpha | \alpha \in \Delta^{\omega'}, \left\| S^{\omega'/\omega''}_{C,R}(\alpha) \cup S^{\omega''/\omega''}_{C,R}(\alpha) \right\| \geq n \right\}\end{aligned}$$

where $S^{\omega' \cup \omega''}_{C,R}(\alpha)$, $S^{\omega'/\omega''}_{C,R}(\alpha)$, and $S^{\omega''/\omega''}_{C,R}(\alpha)$ are defined as follows:

$$\begin{aligned}S^{\omega' \cup \omega''}_{C,R}(\alpha) &\triangleq \left\{\beta | \beta \in C^{\omega'} \cup C^{\omega''}, \langle \alpha, \beta \rangle \in R^{\omega'} \cup R^{\omega''} \right\} \\ S^{\omega'/\omega''}_{C,R}(\alpha) &\triangleq \left\{\beta | \beta \in C^{\omega'}, \langle \alpha, \beta \rangle \in R^{\omega'} \cup R^{\omega''} \right\} \\ S^{\omega''/\omega''}_{C,R}(\alpha) &\triangleq \left\{\beta | \beta \in C^{\omega''}, \langle \alpha, \beta \rangle \in R^{\omega'} \cup R^{\omega''} \right\}\end{aligned}$$

We now consider the set defined as $\left\{\beta | \beta \in C^{\omega'}, \langle \alpha, \beta \rangle \in R^{\omega'} \cup R^{\omega''} \right\}$, since $R^{\omega''} \cup \Delta^{\omega'} \times \Delta^{\omega'} = \emptyset$ and $\alpha \in \Delta^{\omega'}$, we can conclude that:

$$\left\{\beta | \beta \in C^{\omega'}, \langle \alpha, \beta \rangle \in R^{\omega'} \cup R^{\omega''} \right\} = \left\{\beta | \beta \in C^{\omega'}, \langle \alpha, \beta \rangle \in R^{\omega'} \right\}$$

While, regarding the other set, since we are assuming that $\alpha \in \Delta^{\omega'}$ and $\beta \in C^{\omega''} \subseteq \Delta^{\omega''}$ and that $\Delta^{\omega'} \cap \Delta^{\omega''} = \emptyset$, we can conclude that it is empty. Hence, we have that:

$$[(\geq\ n\,R\,C)]^{\omega} \cap \Delta^{\omega'} = \left\{\alpha | \alpha \in \Delta^{\omega'}, \left\| \left\{\beta | \beta \in C^{\omega'}, \langle \alpha, \beta \rangle \in R^{\omega'} \right\} \right\| \geq n \right\}$$

In other words, it follows that:

$$[(\geq\ n\,R\,C)]^{\omega} \cap \Delta^{\omega'} = [(\geq\ n\,R\,C)]^{\omega'}$$

By the symmetry, a similar result also holds for the other set. Combining them we finally prove the claim.

8. $[R \upharpoonright C]^{\omega} = [R \upharpoonright C]^{\omega'} \cup [R \upharpoonright C]^{\omega''}$. According to definition of domain restriction operator in the standard semantics, we have that:

$$[R \upharpoonright C]^{\omega} = \{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in R^{\omega}, \alpha \in C^{\omega}\}$$

By the inductive hypothesis, we can state also that:

$$\{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in R^{\omega}, \alpha \in C^{\omega}\} = \left\{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in R^{\omega'} \cup R^{\omega''}, \alpha \in C^{\omega'} \cup C^{\omega''} \right\}$$

We, now, split this set into two parts as:

$$\begin{aligned}T' &= \left\{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in R^{\omega'}, \alpha \in C^{\omega'} \cup C^{\omega''} \right\} \\ T'' &= \left\{\langle \alpha, \beta \rangle | \langle \alpha, \beta \rangle \in R^{\omega''}, \alpha \in C^{\omega'} \cup C^{\omega''} \right\}\end{aligned}$$

Given the set T' we can observe that, since $R^\omega \subseteq \Delta^{\omega'} \times \Delta^{\omega'}$, $\Delta^{\omega'} \cap \Delta^{\omega''} = \emptyset$, and $C^{\omega''} \subseteq \Delta^{\omega''}$, we can conclude that $\alpha \in C^{\omega'}$. In other words, we have that:

$$T' = \left\{ \langle\alpha, \beta\rangle | \langle\alpha, \beta\rangle \in R^{\omega'}, \alpha \in C^{\omega'} \right\}$$

But, according to standard semantics we have that:

$$T' = [R \upharpoonleft C]^{\omega'}$$

By the symmetry among ω' and ω'', a specular result also holds for T'', hence the hypothesis.

$$\left\{ \langle\alpha, \beta\rangle | \langle\alpha, \beta\rangle \in R^{\omega'} \cup R^{\omega''}, \alpha \in C^{\omega'} \cup C^{\omega''} \right\} = [R \upharpoonleft C]^{\omega'} \cup [R \upharpoonleft C]^{\omega''}$$

9. $[R \downharpoonleft C]^\omega = [R \downharpoonleft C]^{\omega'} \cup [R \downharpoonleft C]^{\omega''}$. This case can be proved using an argumentation similar to the previous one.

10. $\triangle^\omega = \triangle^{\omega'} \cup \triangle^{\omega''}$. Trivial.

□

Theorem 96. *The disjoint union model property holds for the $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$ concept and role inclusion axiom language.*

Proof. In order to prove the claim we need to show that given any arbitrary pair of models ω' and ω'' of given $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$-TBox $\mathcal{T}$, the structure ω obtained by the union of the provided ones is also a model of $\mathcal{T}$. By contradiction, we assume that there exists at least an axiom, e.g., $R \sqsubseteq S$, where R and S are $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$-role expression, s.t., despite it holds in ω' and ω'', it is violated in ω. It means that there exists at least a pair of elements $\langle x, y\rangle$ s.t.:

$$\langle x, y\rangle \in R^\omega \setminus S^\omega$$

According to Lemma 52, we have that $R^\omega = R^{\omega'} \cup R^{\omega''}$ and $S^\omega = S^{\omega'} \cup S^{\omega''}$. In other terms:

$$\langle x, y\rangle \in (R^{\omega'} \cup R^{\omega''}) \setminus (S^{\omega'} \cup S^{\omega''}) = R^{\omega'} \setminus (S^{\omega'} \cup S^{\omega''}) \cup R^{\omega''} \setminus (S^{\omega'} \cup S^{\omega''})$$

Since the interpretation domains of ω' and ω'' are disjoint, also the interpretation of expressions are. So we have that:

$$\langle x, y\rangle \in R^{\omega'} \setminus S^{\omega'} \cup R^{\omega''} \setminus S^{\omega''}$$

In other words, we must conclude that $\langle x, y\rangle \in R^{\omega'} \setminus S^{\omega'}$ or that $\langle x, y\rangle \in \cup R^{\omega''} \setminus S^{\omega''}$, which means that the axiom is violated in ω' or in ω'' contradicting the hypothesis. Using a quite similar argumentation, we can prove the claim also for $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$-concept expressions. □

Given the previous results, we can easily extends Lemmas 6 and 7 to this case, obtaining the following claim as a generalization of Theorem 6.

Theorem 97. *Given an $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$-knowledge base, if it admits a model over an arbitrary subset Δ^ω of the interpretation universe $\mathcal{U}$, then it admits also a model on the whole universe.*

Remark 48. *Please notice that the extensional specification does not play any role in such a context, since given the disjointness condition on interpretation domains and the standard name assumption, we can not provide any suitable interpretation function for object names (i.e., an object name $o \in \boldsymbol{O}$ is interpreted as $o^{\omega'} \neq o^{\omega''}$, so it can not have an unambiguous interpretation in ω). Moreover given the usage of the disjoint union model property in the mentioned proof, this limitation is not pertinent.*

We point out that all results state so far for $\mathcal{ALCQI}$ world specification and $\mathcal{ALCQIO}$ queries are preserved using $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$ and $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonleft$ $)$ resp. as constraint and query languages since, generally speaking:

- the properties of embedding relation are substantially unmodified;
- the "working" language $\mathcal{C}^2$ expressive power is enough to deal with new constructs, generally Description Logics with boolean enhanced role primitives are strictly contained into such FOL fragment, but also additional equipment required to axiomatize update properties can successfully represented using the devised encoding;
- the encoding is still linear in the size of the input, hence the reductions employed to state complexity results are preserved in this scenario.

For the sake of brevity we omit the proofs of these claims, but we observe that they can be easily obtained replacing the references to results devised in Chapter 3 (in particular, Theorems 1 and 2) with the corresponding ones provided in this section.

In the expanded framework we can so employ the following definitions:

Definition 129 (World specification)**.** *A world specification $\mathcal{W}$ is a knowledge base $\langle \mathcal{T}_{\mathcal{W}}, \mathcal{A}_{\mathcal{W}} \rangle$ expressed on the alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$ using the expressive description logic with role axioms $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$.*

Definition 130 (Parameterized (concept) query)**.** *Given a domain specification and a finite set of variable names $\boldsymbol{V}$, distinct from domain alphabets, a parameterized (concept) query $Q(\boldsymbol{V})$ is an arbitrary $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonleft)$-concept expression built on the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{V}, \boldsymbol{P}, \boldsymbol{O} \rangle$.*

Definition 131 (Parameterized (concept) query template)**.** *Given a domain specification, a set of template parameter names $\boldsymbol{Z}$, and a finite set of variable names $\boldsymbol{V}$, distinct from domain alphabets, a parameterized (concept) query template $Q_{\boldsymbol{Z}}(\boldsymbol{V})$ is an arbitrary $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonleft)$-concept expression built on the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{V}, \boldsymbol{P}, \boldsymbol{O} \cup \boldsymbol{Z} \rangle$.*

We can also allows to role expression queries defined as:

Definition 132 (Parameterized role query)**.** *Given a domain specification and a finite set of variable names $\boldsymbol{V}$, distinct from domain alphabets, a parameterized role query $Q(\boldsymbol{V})$ is an arbitrary $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonleft)$-role expression built on the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{V}, \boldsymbol{P}, \boldsymbol{O} \rangle$.*

Definition 133 (Parameterized role query template)**.** *Given a domain specification, a set of template parameter names $\boldsymbol{Z}$, and a finite set of variable names $\boldsymbol{V}$, distinct from domain alphabets, a parameterized role query template $Q_{\boldsymbol{Z}}(\boldsymbol{V})$ is an arbitrary $\mathcal{ALCQIO}(\neg, \sqcap, \upharpoonleft)$-role expression built on the alphabet $\langle \boldsymbol{A} \cup \boldsymbol{V}, \boldsymbol{P}, \boldsymbol{O} \cup \boldsymbol{Z} \rangle$.*

The properties of concept expression queries can be easily generalized to this case. Generally, we can employ such kind of queries as condition expression in the specification of the services. As definition of precondition we can now, e.g., assume the following:

Definition 134 (Extended atomic precondition term)**.** *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$. An atomic precondition, which is suitable for such a service, is a pair $\langle s, Q(\boldsymbol{X})\rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the precondition (positive or negative);*
- *$Q(\boldsymbol{X})$ is a parameterized concept or role query over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

Definition 135 (Extended atomic condition term)**.** *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$. An atomic condition term, that is suitable for such a service, is a pair $\langle s, Q(\boldsymbol{X})\rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the precondition (positive or negative);*
- *$Q(\boldsymbol{X})$ is a parameterized concept or role query over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

The semantics of these constructs is left untouched. Give these constructs we can also extend also the definition of role update primitives adding the followings:

Definition 136 (Positive role effect argument)**.** *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$, and let $\boldsymbol{Y}_S$ be the corresponding output variable names. A positive role effect argument is any element $Y \in \boldsymbol{Y}_S$ or any parameterized role query $Q(\boldsymbol{X})$ over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

Definition 137 (Negative role effect argument)**.** *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$. A negative role effect argument is any parameterized role query $Q(\boldsymbol{X})$ over the domain specification in the variables $\boldsymbol{X} \subseteq \boldsymbol{X}_S$.*

Definition 138 (Extended atomic role effect)**.** *Let $\boldsymbol{X}_S$ be the input variable names of a service S defined in the domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$, and let $\boldsymbol{Y}_S$ be the corresponding output variable names. An extended atomic role effect, that is suitable for such a service, is a triple $\langle s, P, q\rangle$ where:*

- *$s \in \{+, -\}$ is the sign of the effect (insert or delete);*
- *$P \in \boldsymbol{P}$ is the target role name;*
- *q is the arguments of the update (positive or negative) according to the sign of the effect.*

Definition 139 (Role insert set)**.** *Let E be a simple service effect specification, $P \in \boldsymbol{P}$ a concept name, ω a world state, $\sigma_{\boldsymbol{X}}$ an input variable assignment consistently defined w.r.t. ω. The link insert set for the role P is defined as:*

$$P^{+}(\omega, \sigma_{\boldsymbol{X}}) = \left(\bigcup_{\langle +, P, Q(\boldsymbol{X}), Q'(\boldsymbol{X})\rangle \in E} Q^{\omega}(\sigma_{\boldsymbol{X}}) \times Q'^{\omega}(\sigma_{\boldsymbol{X}}) \cup \bigcup_{\langle +, P, Q''(\boldsymbol{X})\rangle \in E} Q''^{\omega}(\sigma_{\boldsymbol{X}}) \right) \setminus P^{\omega}$$

Definition 140 (Role delete set)**.** *Let E be a simple service effect specification, $P \in \boldsymbol{P}$ a role name, ω a world state, $\sigma_{\boldsymbol{X}}$ an input variable assignment consistently defined w.r.t. ω, the link delete set for the role P is defined as:*

$$P^{-}(\omega, \sigma_{\boldsymbol{X}}) = \left(\bigcup_{\langle -, P, Q(\boldsymbol{X}), Q'(\boldsymbol{X}) \rangle \in E} Q^{\omega}(\sigma_{\boldsymbol{X}}) \times Q'^{\omega}(\sigma_{\boldsymbol{X}}) \cup \bigcup_{\langle -, P, Q''(\boldsymbol{X}) \rangle \in E} Q''^{\omega}(\sigma_{\boldsymbol{X}}) \right) \cap P^{\omega}$$

The corresponding adjustment of related axiom schemas is straightforward.

7.2 State Transition Constraints

So far the proposed framework is able to deal with static constraints or, in other words, with constraints that can be evaluated/enforced on a given world state ω. In fact, in the Chapter 3 we have provided a definition of legal world state (i.e., a world state satisfying the constraint set) and we have devised on this foundation the notion of legal service too (see page 83): generally speaking, any transition from a legal state to another legal one is allowed once a suitable service is provided.

In this section, we are interested in the analysis the possibility to express and enforce also constraints related to the transition relation, filtering out some particular behaviors as not allowed. Such a kind of constraints can be employed in different system modeling language, e.g., in the UML class modeling ([OMG07]) it is possible to labeling a class association-end (essentially a role in DL according to [CCDL02] and [SB05]) as `frozen` or `add-only` denoting the fact that the extensional level of such role is immutable or that new pair can be added but not removed[2]. In a similar way, in a RDBMS, it is possible acting on user privilege definitions to prevent from removing or updating a record in a table or enforcing more complex dynamic constraints using triggers[3]. Moreover, a typical field of application of temporal logics is the verification of dynamic properties of automata or, in other words, if the update state sequence is always consistent with a given set of constraints. Different applications of Model Checking techniques to e-services (see 2.2.4, 2.2.7, 2.2.8, and 2.2.11) are actually facing with this kind of issues in terms of the temporal integrity of the system: from their perspective we are addressing the consistency analysis of atomic steps.

As first step we need to extend the definition of world specification in order to allow the ability of express such a kind of constraints.

Definition 141 (Concept dynamic axiom)**.** *Given a concept expression language L and a suitable alphabet (i.e., $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$), a concept dynamic axiom is a pair $\langle s, C \rangle$ s.t.:*

- *$s \in \{=, +, -\}$ is the sign of the axiom;*
- *C is an L-concept expression over the alphabet.*

[2]In the UML meta-model, this feature is denoted as *changeability* property of association link.

[3]For a survey on this aspect of data modeling see [Tha00]. Generally this topic is addressed also in *temporal databases* [DD02] and it is a typical application of *temporal logics* using model checking techniques ([CS01]).

Definition 142 (Role dynamic axiom)**.** *Given a role expression language L and a suitable alphabet, a role dynamic axiom is a pair $\langle s, R\rangle$ s.t.:*

- *$s \in \{=, +, -\}$ is the sign of the axiom;*
- *R is an L-role expression over the alphabet.*

Roughly speaking, "positive" dynamic axioms denote set of elements (or set of element pairs) denoted by the means of an expression s.t. their extensions must not decrease during the system evolution, the "negative" ones these s.t. their extensions must not increase, while others sets that have to be immutable.

For the sake of brevity we can also express such axioms as C^s or R^s.

Definition 143 (Dynamic axiom box (DBox))**.** *Given a language L allowing for role and concept expressions and an alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$, a dynamic axiom box is a finite set of concept or role dynamic axioms expressed in L over the provided alphabet.*

While the definition of consistency of a "traditional" KB is given w.r.t. a single interpretation structure, in the case of state transition constraint we need to keep into account at least two structures, assuming that there exists an arbitrary kind of update operator that evolve the system from a state to the other one. W.l.o.g. we are facing the problem adopting the unique name assumption, which means that the interpretation of object names is always immutable.

Definition 144 (DBox model). *Given a pair of interpretations ω and ω', ranging over two domains Δ^{ω} and $\Delta^{\omega'}$ defined in the same universe $\mathfrak{U}$, and a DBox $\mathcal{D}$, we say that the transition $\omega \rightarrow \omega'$ is a model of the DBox ($\omega \rightarrow \omega' \models \mathcal{D}$) if for each dynamic axiom $\langle s, E\rangle \in \mathcal{D}$ we have that:*

- *$E^{\omega} \subseteq E^{\omega'}$ if $s = +$;*
- *$E^{\omega} \supseteq E^{\omega'}$ if $s = -$;*
- *$E^{\omega} = E^{\omega'}$ otherwise.*

Proposition 5. *The idempotent transition $\omega \rightarrow \omega$ is always a model for any given DBox $\mathcal{D}$.*

Proof. Trivial. □

Definition 145 (Allowed transition set)**.** *Given a DBox $\mathcal{D}$, the allowed transition set $T_{\mathcal{D}}$ contains any transition $\omega \rightarrow \omega'$ s.t. $\omega \rightarrow \omega' \models \mathcal{D}$.*

This definition simply extends the concept of consistency for TBox/ABox, but as we are not interested in empty models, we are also interested in dynamic specifications that don't completely freeze the system state.

Definition 146 (Consistent DBox)**.** *A DBox $\mathcal{D}$ is non-immutable if there exists at least a transition $\omega \rightarrow \omega'$, s.t. $\omega \neq \omega'$, that is a model of it.*

This definition can also extended keeping into account also the remaining knowledge base $KB = \langle \mathcal{T}, \mathcal{A}\rangle$:

Definition 147 (KB-consistent DBox)**.** *Given a consistent knowledge base $KB = \langle \mathcal{T}, \mathcal{A} \rangle$, a DBox $\mathcal{D}$ is consistent w.r.t. it if there exists at least a transition $\omega \to \omega'$, s.t. $\omega \neq \omega'$, $\omega \models KB$, $\omega' \models KB$, that is a model of it.*

Remark 49. *Assuming a general DL axiomatization of the UML language in the spite of [CCDL02] and [SB05], we can easily verify that using this kind of axioms we can model dynamic restrictions implied by UML changeability property on association links, e.g., a positive role DBox axiom is a generalization of the `add-only` flag.*

Example 21. *Given the Example 19, we consider the following additional constraint: the composition of an administrative district cannot be altered (at least from the application perspective), as the assigned head town.*

This kind of dynamic constraint, essentially specifying frozen *properties, can be obtained adding the following dynamic axioms to the DBox component:*

$$\langle =, \mathsf{headTown} \rangle$$
$$\langle =, \mathsf{partOf} \upharpoonleft \mathsf{Town} \downharpoonleft \mathsf{District} \rangle$$

Remark 50. *While the first constraint can be also simulated simply removing* headTown *from the set of predicate that can be updated or repaired, the second one, since is based on actual instances (generally the predicate* partOf *can be modified without restrictions) requires a more sophisticated approach.*

Example 22. *We assume that the representation has been extended to capture also the history of the system. In other words, that some additional predicates have been introduced to keep track of previous assignment of some logged properties.*

Given the domain analyzed in previous examples, we introduce the additional property pastResidence *to store in which towns the citizen previously lived. Generally also this predicate can be updated, but, if an element has been recorded it cannot be forgotten*[4]*, so the predicate extension can be evolve only monotonically non-decreasing. In other words, we need to add the following axiom to the DBox:*

$$\langle +, \mathsf{pastResidence} \rangle$$

Moreover, assuming that we can admit that for some "special" people this requirement about the historical memory must not be enforced, we need to restate the previous axiom as:

$$\langle +, \mathsf{pastResidence} \upharpoonleft \mathsf{Special} \rangle$$

In the following we generally consider the $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$-language as in the definition of world specification/KB, as in the definition of DBox state transition axioms. More specifically, we interested in the analysis of consistency of state transitions resulting from the enactment of a generic service, in order to provide a refined definition of service validity (and even repairability).

Definition 148 (Valid e-service (w.r.t. a DBox))**.** *Let S be a full e-service, it is valid w.r.t. a world specification $\mathcal{W}$ and a DBox $\mathcal{D}$, iff:*

[4]We are simply ignoring erroneous updates.

- *each effect $E \in \mathcal{E}_S$ is consistent;*
- *for each legal world state ω, for each consistent input assignment $\sigma_{\mathbf{X}}$, s.t. the service is accessible in ω using it, there exists at least a legal state ω' in the enactment and the transition $\omega \rightarrow \omega'$ is allowed.*

Definition 149 (Repairable e-service (w.r.t. a DBox))**.** *Let $\mathcal{E}_S$ be effects of a non-deterministic e-service S, and let R_S be the set of repairs for the service S. S is repairable w.r.t. a world specification $\mathcal{W}$ and a DBox $\mathcal{D}$ iff:*

- *effects $E \in \mathcal{E}_S$ are consistent;*
- *for each legal world state ω, for each consistent input assignment $\sigma_{\mathbf{X}}$, s.t. the service is accessible in ω using it, there exists at least a state ω' in the enactment and a repair $R \in R_S$ s.t. the repaired state ω'_R is legal and the transition $\omega \rightarrow \omega'_R$ is allowed*

Remark 51. *In the follows we refer to validity and repairability properties w.r.t. a DBox also as* dynamic*, while properties defined in previous chapters will be also denoted as* static*.*

It is trivial to conclude the following properties:

Proposition 6. *A service S valid w.r.t. a world specification $\mathcal{W}$ and a DBox $\mathcal{D}$ is also valid w.r.t. $\mathcal{W}$.*

Proposition 7. *A service S repairable w.r.t. a world specification $\mathcal{W}$ and a DBox $\mathcal{D}$ is also repairable w.r.t. $\mathcal{W}$.*

Let m and n be two suitable (i.e., $cod(m) \cap cod(n) = \emptyset$) name mapping functions on a domain specification $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$ employed to embed two arbitrary world states ω and ω' into a structure $\hat{\omega}$ according the ways devised so far, given a DBox $\mathcal{D}$ we define a new axiom schema $\Delta KB^D(m, n)$ as reported in Table 7.1.

Remark 52. *For the sake of brevity, if the first name mapping function is the identity function $id(x) = x$, we simply omit it, denoting this axiom schema as $\Delta KB^D(m)$.*

The following claims that the devised framework is powerful enough to capture also this modeling aspect without any significant impact on the problem computational complexity.

Theorem 98. *Given a DBox $\mathcal{D}$, a pair of world states ω and ω' and structure $\hat{\omega}$ s.t. they are embedded into it according two suitable name mapping functions m and n, then the transition $\omega \rightarrow \omega'$ is allowed according to $\mathcal{D}$ iff $\hat{\omega} \models \Delta KB^D(m, n)$.*

Proof. We first assume that given such a transition $\omega \rightarrow \omega'$ and the embedding structure $\hat{\omega}$, the knowledge base obtained by the instantiation of the axiom schema is satisfied in $\hat{\omega}$, but the transition is not a model of the DBox $\mathcal{D}$.

In other words, there exists at least a dynamic axiom (concept or role, does not matter) $\langle s, E \rangle \in \mathcal{D}$ that is not satisfied over the state transition $\omega \rightarrow \omega'$. W.l.o.g., we assume that such a dynamic axiom is a role positive one in the form $\langle +, R \rangle$, where R is a $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$-role expression over the domain

Table 7.1: The axiom schema $\Delta KB^D(m,n)$

$$\bigwedge_{C^{=}\in\mathcal{D}} \tau^*_m(C) \equiv \tau^*_n(C) \tag{7.2}$$

$$\bigwedge_{C^{+}\in\mathcal{D}} \tau^*_m(C) \sqsubseteq \tau^*_n(C) \tag{7.3}$$

$$\bigwedge_{C^{-}\in\mathcal{D}} \tau^*_m(C) \sqsupseteq \tau^*_n(C) \tag{7.4}$$

$$\bigwedge_{R^{=}\in\mathcal{D}} \forall x,y.\tau^*_m(R)(x,y) \leftrightarrow \tau^*_n(R)(x,y) \tag{7.5}$$

$$\bigwedge_{R^{+}\in\mathcal{D}} \forall x,y.\tau^*_m(R)(x,y) \rightarrow \tau^*_n(R)(x,y) \tag{7.6}$$

$$\bigwedge_{R^{-}\in\mathcal{D}} \forall x,y.\tau^*_m(R)(x,y) \leftarrow \tau^*_n(R)(x,y) \tag{7.7}$$

specification alphabet. Since the axiom is violated, there must exist at least a pair of elements $\langle x^*, y^*\rangle \in \mathfrak{U}\times\mathfrak{U}$, s.t. $\langle x^*, y^*\rangle \in R^{\omega}$ and $\langle x^*, y^*\rangle \notin R^{\omega'}$. Since both ω and ω' are embedded into $\hat{\omega}$ using the name mapping functions m and n resp., applying Theorem 92, we can conclude that:

$$R^{\omega} = [\tau^*_m(R)]^{\hat{\omega}}$$
$$R^{\omega'} = [\tau^*_n(R)]^{\hat{\omega}}$$

Applying such results, we can also state that:

$$\langle x^*, y^*\rangle \in [\tau^*_m(R)]^{\hat{\omega}} \setminus [\tau^*_n(R)]^{\hat{\omega}}$$

concluding that, according to standard semantics, at least the formula following from the instantiation of axiom template in Equation 7.6 w.r.t. $\langle +, R\rangle$ does not hold in $\hat{\omega}$, obtaining a contradiction that proves the first part of theorem. Other cases can be shown in a similar way, so we omit them for the sake of brevity.

We now assume that the given transition $\omega \longrightarrow \omega'$ is a model for the DBox $\mathcal{D}$, but, despite of it, the structure $\hat{\omega}$, s.t. ω and ω' are embedded into it, is not a model of the instantiation of the given axiom schema.

According to this hypothesis there must exist at least a formula, obtained accordingly instantiating an axiom $\langle s, E\rangle \in \mathcal{D}$, that is not satisfied in $\hat{\omega}$. W.l.o.g., we can assume that such a formula is obtained by the instantiation of axiom template in Equation 7.4 w.r.t. the dynamic axiom $\langle -, C\rangle \in \mathcal{D}$, where C is a suitable $\mathcal{ALCQI}(\sqcup, \sqcap, \upharpoonleft, \downharpoonleft, \triangle)$-concept expression over the domain specification alphabet. Since the formula is not satisfied in $\hat{\omega}$ there must exist at least an element $x \in \mathfrak{U}$ s.t., according to standard semantics, $x \in [\tau^*_n(C)]^{\hat{\omega}} \setminus [\tau^*_m(C)]^{\hat{\omega}}$. Since ω and ω' are embedded into $\hat{\omega}$ we can apply Theorem 92, obtaining that:

$$C^{\omega} = [\tau^*_m(C)]^{\hat{\omega}}$$
$$C^{\omega'} = [\tau^*_n(C)]^{\hat{\omega}}$$

We can now conclude that $C^{\omega'} \not\subseteq C^{\omega}$ and, hence, that the given transition is not a model of the DBox. From this contradiction the claim follows. The proof can be easily extended using the same argumentation to other cases. $\square$

Using this theorem, we can easily extends results provided so far simply adding the adequate instantiations of the axiom schema $\Delta KB^D(m,n)$ to the reasoning problem encoding. Moreover, the constraint set size is clearly linear in the length of the input, so the reduction is still polynomial and the problem complexity results are also valid in this new scenario.

Remark 53. *As discussed in Section 3.4, we are always assuming that active domain is monotonically nondecreasing, or, in other words, that the DBox $\mathcal{D}$ always contains the constraint $\langle +, \top \rangle$ (or any equivalent form). This restriction can be removed adjusting adequately the framework without altering significantly results achieved so far. Moreover, it is quite obvious that the dynamic constraint notion can be applied to general time-evolving knowledge representation systems and it is not necessarily related to the service-oriented architecture.*

Theorem 99. *A consistent and accessible non-deterministic e-service S is valid w.r.t. a world specification $\mathcal{W}$ and a DBox $\mathcal{D}$, iff the following implication holds:*

$$KB^P \wedge \tau^*(\mathcal{W}) \wedge \bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E)\right) \models \bigvee_{E \in \mathcal{E}} \tau^*_{m_E}(\mathcal{W}) \wedge \Delta KB^D(m_E)$$

where m_E is the name mapping function for the domain and the instantiation variable names related to effect $E \in \mathcal{E}$.

Proof. This claim is a refinement of Theorem 47, so we prove it only showing additional parts.

We first assume that the given implication holds. Since the implication is specialization of the implication in Equation 5.27, also the latter holds and, according to Theorem 47, we can conclude that the service is valid at least applying the former definition. By contradiction, we assume that the given service is not valid according the definition involving the DBox. In other words, that there exists a legal world state ω and an input assignment $\sigma_{\mathbf{X}}$ s.t. any arbitrary legal enactment $\langle \omega'_E, (\sigma'_{\mathbf{Y}})_E \rangle$, despite ω' is valid w.r.t. $\mathcal{W}$, the transition $\omega \rightarrow \omega'$ is not allowed according to constraints in $\mathcal{D}$. Given the isomorphism property (see Theorem 23), we consider an enactment for any possible effect $E \in \mathcal{E}$, and employing the function $\mu(\omega, \sigma_{\mathbf{X}}, \Lambda)$ and suitable name mapping functions, we build a new structure $\hat{\omega}$ that embeds world states and enactments. Applying Theorem 45 and related results we can easily show that the antecedent of implication is satisfied in $\hat{\omega}$, hence, since the implication is assumed to be valid there exists at least an effect $E^* \in \mathcal{E}$ s.t.:

$$\hat{\omega} \models \tau^*_{m_{E^*}}(\mathcal{W}) \wedge \Delta KB^D(m_{E^*})$$

On the other hand, applying Theorem 98 we must conclude that the transition realized by the effect E^* in ω is legal according to $\mathcal{D}$ (and also to $\mathcal{W}$) and that the given service is valid, obtaining a contradiction.

Now we assume that the given service is valid w.r.t. both the world specification $\mathcal{W}$ and the DBox $\mathcal{D}$, but that the implication does not hold. Let $\hat{\omega}$ be an interpretation structure s.t. the given implication is not verified: the consequent is not satisfied despite the antecedent is. Since the hypothesis on the validity of the service we can also apply Theorem 47 and according to Equation 5.27 we can conclude that there exists at least an effect $E^* \in \mathcal{E}$ s.t.:

$$\hat{\omega} \models \tau^*_{m_{E^*}}(\mathcal{W})$$

On the other hand, since we have assumed that the implication is not verified in $\hat{\omega}$ we need also that:

$$\hat{\omega} \not\models \Delta KB^D(m_{E^*})$$

But, given Theorem 46 and related results, let ω and ω' be the states embedded into $\hat{\omega}$ given the enactment of the effect E^*, we have that $\omega \to \omega'$ is a state transition that, given Theorem 98, is not allowed w.r.t. $\mathcal{D}$, contradicting the initial hypothesis on the validity of the service. $\square$

Theorem 100. *A consistent and accessible non-deterministic e-service S is repairable w.r.t. a world specification $\mathcal{W}$ and a DBox $\mathcal{D}$ using a family of repair $R_S = \{R_1, \dots, R_r\}$, iff the following implication holds:*

$$\begin{gathered}\bigwedge_{E \in \mathcal{E}} \left(\Delta KB^I_n(\mathsf{Top}_{\mathsf{m_E}}, m_E) \wedge \Delta KB^U_c(m_E) \wedge \bigwedge_{i=1}^{r} \left(\Delta KB^R_n(m_E, n_{E,i}) \right) \right) \\ \wedge KB^P \wedge \tau^*(\mathcal{W}) \models \bigvee_{E \in \mathcal{E}} \bigvee_{i=1}^{r} \left(\tau^*_{n_{E,i}}(\mathcal{W}) \wedge \Delta KB^C_n(m_E, n_{E,i}) \wedge \Delta KB^D(n_{E,i}) \right)\end{gathered} \tag{7.8}$$

where m_E and $n_{E,i}$ are the name mapping functions for the domain defined for any effect E and for any repair R_i.

Proof. The prove of this claim is quite similar to the previous one, considering that, since the whole enactment spans from the initial (ω') to the final repaired state (ω''), the DBox is evaluated w.r.t. these ones. In other words, the selected repair must be able to eventually obtain a final state that is consistent w.r.t. both static $\mathcal{W}$ and dynamic $\mathcal{D}$ constraint sets.

We first assume that the given implication holds. Since the implication is specialization of the implication in Equation 5.43, also the latter holds and, according to Theorem 52, we can conclude that the service is repairable at least applying the former (static) definition that ignores dynamic constraints. By contradiction, we now assume that the given service is not repairable according the definition taking also them into account. In other words, that there exists a legal world state ω and an input assignment $\sigma_{\mathbf{X}}$ s.t. any arbitrary legal enactment $\langle \omega', \sigma''_{\mathbf{Y}} \rangle$, despite ω'' is valid w.r.t. $\mathcal{W}$, the transition $\omega \to \omega''$ is not allowed according to constraints in $\mathcal{D}$. Also in this case we can apply the isomorphism property of successor states and, hence, consider a repaired enactment for any possible effect $E \in \mathcal{E}$ and suitable repair $R_i \in R_S$, as representative (witness) of a possibility infinite set. Let $\Lambda = \{\langle, \omega'_E, (\sigma'_{\mathbf{Y}})_E \rangle | E \in \mathcal{E}\}$ and $\Omega = \left\{\omega''_{E,i} | E \in \mathcal{E}, R_i \in R_S\right\}$ be resp. the set of candidate enactments and repaired successor states, we employ the function $\mu(\omega, \sigma_{\mathbf{X}}, \Lambda, \Omega)$ to obtain a new structure $\hat{\omega}$ that, according to Lemma 38, embeds these states and enactments. Applying Theorem 50 and related results we can conclude that $\hat{\omega}$ satisfies antecedent of implication and since we have assumed that this implication is valid there exists at least an effect $E^* \in \mathcal{E}$ and a repair $R_{i^*} \in R_S$ s.t.:

$$\hat{\omega} \models \tau^*_{n_{E^*,i^*}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,i^*}) \wedge \Delta KB^D(n_{E^*,i^*})$$

On the other hand, applying Theorem 98 we must conclude that the transition $\omega \rightarrow \omega''_{E^*,i^*}$ realized by the effect E^* and repair R_{i^*} a model of $\mathcal{D}$ (and also to $\mathcal{W}$) and that the given service is repairable, obtaining a contradiction, that proves the first part of the claim.

Now we assume that the given service is repairable taking into account both the static $\mathcal{W}$ and the dynamic $\mathcal{D}$ specifications, but that the implication does not hold. Let $\hat{\omega}$ be an interpretation structure s.t. the given implication is not verified: the consequent is not satisfied despite the antecedent is. Since the hypothesis on the repairability of the service, we can also apply Theorem 52 and according to Equation 5.43 we can conclude that there exists at least an effect $E^* \in \mathcal{E}$ a repair $R_{i^*} \in R_S$ s.t. the corresponding successor state is valid and the repair is consistent with the update. In other words, we have that:

$$\hat{\omega} \models \tau^*_{n_{E^*,i^*}}(\mathcal{W}) \wedge \Delta KB^C_n(m_{E^*}, n_{E^*,i^*})$$

On the other hand, since we have assumed that the implication is not verified in $\hat{\omega}$, we need also that:

$$\hat{\omega} \not\models \Delta KB^D(n_{E^*,i^*})$$

But, given Theorem 51, let ω and ω''_{E^*,i^*} be the states embedded into $\hat{\omega}$, given the enactment of the effect E^* and the repair R_{i^*}, we have, according to Theorem 98, that $\omega \rightarrow \omega''_{E^*,i^*}$ is a state transition not allowed w.r.t. $\mathcal{D}$, contradicting the initial hypothesis on the repairability of the service. □

Given the previous results, it is easy to conclude that the upper bound of problem complexity is not affected by this framework extension, since the encoding of the DBox is essentially linear in the size of the specification. Moreover also other properties can be adequately adjusted to keep into account this new class of domain constraints.

7.3 Conclusions

In this chapter we have devised two interesting extensions of the specification language in order to deal with complex role and dynamic constraints. In particular, the latter kind is useful to capture a relevant subset of concrete modeling language primitives (i.e., UML) providing a formal foundation and a decidable (and eventually feasible) checking procedure.

These extensions have been introduced to accordingly enrich the description logic and to show how, given a suitable semantics, they can be also encoded using the working logic $\mathcal{C}^2$: in this case we have observed that these new constructs do not affect the computational complexity upper-bound. Both extensions can be employed simultaneously, so we can build a rather powerful and expressive e-service analysis framework.

We point out that these extensions are not necessarily restricted to this framework, but they can be easily generalized to be integrated also in other approaches. More specifically, the devised DBox notion is rather general and it is suitable to be included in a dynamic setting for Description Logics as well as extending the approach to include also execution-path constraints in the fashion of temporal logic languages (i.e., linear and branching time).

CHAPTER 8

Conclusions

As shown in the previous chapters, a number of functional properties of e-services relevant in the design of a service-oriented integration solution can be traced to a logic characterization, once a suitable operational semantic model is provided, based upon the enforcing of the service contract.

The main contribution of this work is to show how it is possible to reason about action consequences w.r.t. a complex domain constraint specification, even adopting a semantics of minimal change. While the proposed approach is sound, in the sense that the repairs obtained are correct, it is also not complete, in the sense that the decidability of the problem can be obtained only by renouncing to the search in the whole solution space. However, we have shown the semantic well-foundedness of the repairing options considered by our approach.

We have exhibited, as many other authors, that formal tools available in various fields of computer science can be employed to cope with issues arising in the context of service-oriented computing, but we also claim that they need to be adequately tuned in order to keep into account some functional aspects that distinguish e-services from other distributed computation paradigms, due to their higher level of abstraction. On the other hand, XML web-services and related technologies *per se* are only another integration middleware, especially suited for being employed in Internet (or Internet-like) networks. However, to fully exploit the potential of such a kind of pervasive computing milieu, they need to be enriched with an adequate semantic specification.

We have provided the formal foundation of the devised framework, introducing several decision problems that turn to be useful in order to deal with classic tasks (e.g., validation, matching, discovery, fault-management) as well with more peculiar ones (i.e., coverage analysis) in the construction of service oriented solutions. We have formally analyzed such problems, encoding them into reasoning problem instances using a decidable logic language, allowing us to characterize their computational complexity properties.

Although implementation issues are outside the scope of the present work, it is also possible to sketch a suitable strategy to effectively employ the results provided so far. In particular, the problem reduction approach adopted in the construction of the framework enable to prototype a reasoning meta-service by means of:

concrete specification language, that is suitable to encode arbitrary world specifications, in terms of alphabets, static and dynamic constraints, service and template specification: this language can rely on the XML and Semantic Web stack, eventually leveraging upon a repository manager, that actually acts as a semantic e-service directory, and an end-user editor tool;

semantic e-service specification parser(s), eventually, a collection of parsers that are able to analyze concrete specifications, not necessarily with semantic annotations, provided using different standard languages as WSDL, OWL/OWL-S, WSML, or even UML, and to import them into the repository: the most critical aspect in the design of this component is the capability to ignore irrelevant fragments that usually enrich these artifacts, so a computer-aided reverse-engineering tool appears to be more feasible;

problem encoder, that is able to encode a reasoning problem instance given a domain and a service specification essentially implementing the axiom schemas presented in this work;

$\mathcal{C}^2$-reasoner, that provides the inferencing services: despite such a kind of specialized component is not currently available, we can of course employ one of the several first-order automated theorem provers so far implemented, as [RV01, Sch04, Kal01, HBVL97, RV02], even though they do not necessarily exhibit optimal performances (at least, without a specific search-algorithm tuning).

Finally we remark that, given the high complexity of the studied problems, a crucial issue, to arrive at an effective tool based on the presented framework, is to adequately tailor the specification languages, in order to isolate sub-languages for which the relevant reasoning problems are tractable. Besides, it is also interesting to investigate the approximation of properties studied so far, because also a non-complete approach may show up as useful in some application contexts, once a suitable semantic characterization has been provided.

8.1 Comparison with Related Work

In the following, we summarizing some observations comparing our framework to more interesting approaches resulting from the state-of-the-art analysis presented in Chapter 2.

8.1.1 Planning

Considering the application of the planning technology to e-service problems, we remark that the defined framework conceptually resembles many approaches to composition proposed by this research community. In fact, these approaches share an emphasis on the operational nature of actions described as e-services.

For example, in approaches based on situation calculus, like the one based on Golog and presented in [MS02], an abstract specification of service composition (process), expressed as a template of a conditional plan, is given as problem input, while the system aims at verifying if the available services are applicable and sufficient to instantiate the execution in order to achieve a specific goal. Our goals are slightly different: essentially we are interested in the characterization of the suitability of a set of available services for a class of abstract goals, not only ground ones. But we are also interested in the explicit definition of functional compatibility between services, on which the binding relies during the process instantiation.

Other approaches, like [McD02, Pee03b, APY$^+$02], are essentially based on the reduction of the problem of orchestration of available e-services finalized to achieve a user goal to a corresponding planning problem, suggesting various techniques (i.e., estimated-regression planning) in order to devise the operation scheduling or a more complex conditional plan.

These approaches are unable to cope with the service matchmaking problem, since they naively assume that the whole available service community is precompiled into the planning problem domain definition. Despite it could be technically feasible, these approaches are unable to support dynamic service lookup, which can be useful to discover, at execution time, a replacement for an unavailable provider, binding it on-the-fly.

Moreover, the high expressiveness of planning domain specification languages (required by the complexity of the application scenarios) usually prevents the possibility on a consistency analysis. Indeed, there is a general assumption of the correctness of the specification and instead of domain constraints (as generally intended in the design of information systems) we have the ability to encode using rules arbitrary action consequences, even indirect. In terms of a model update approach, there is the foundational assumption that action effects are always consistent and that a locally performed action can trigger a global model "repair" enforcing domain specification rules. Roughly speaking, while rules employed in the most common planning approaches to describe the domain in terms of action consequences are essentially a sort of forward chaining effect propagation, update repair is more similar to a backward chaining process that indirectly gets from domain constraints the required side-effects.

On the other side, more general approaches to action formalization, as situation calculus, have been also employed to reason about update and integrity constraint interactions ([Rei92]): in fact, the *property persistence problem*, that means how to formalize and keep into account world state properties that always hold during the evolution of the system, is deeply connected to the *frame problem*. The main issue is related with the expressiveness of the language required to axiomatize the constraints implied by the domain requirements.

From the planning perspective, these language features are essentially a tool to explicitly stating system properties in order to automatically deduce action side-effects: this approach is quite similar to the update-repair in data/knowledge-based systems, since a side-effect can be considered as a repair induced from the enforcing of a domain constraint. As in the present approach: an action can be performed if its effects are compatible with persistent system properties or, in other words, if there exists a resulting system state where both action effects and system properties hold. Under a typical planning setting, the presented approach can be considered as a kind of approximated tool to deal with incomplete specifications enforcing a minimal-change semantics. Since this problem is unsolvable, given the necessity to employ second-order axiomatizations, we bound the repair search space considering a finite family of syntactically generated repairs, showing how it is possible to implement a strategy that selects the nearest repair w.r.t. the model distance function. In other words, since the incompleteness of the approach, the following situations can eventually show up:

- a service is detected as not-repairable but a suitable repair exists outside the search space;
- a service is repaired using a syntactically generated repair, but there exists a closer repair that can not be taken into account by the provided algorithm.

Despite there exists also some interesting proposals (e.g., [BAF98, KU99]) to the automation of the property checking procedure[1], there does not exist any strong tractability result, provided that in the general case the problem is undecidable. This property ([KP07]) stems out essentially from the possibility to employ an unrestricted first-order language to express domain constraints (or *uniform formulas* according to [PR99]). An interesting point is the possibility to combine a decidable constraint language (e.g., a DL) and a decidable situation calculus: do they result into a decidable action framework (at least for properties discussed so far)?

8.1.2 Synthesis

Considering the problem of service synthesis, some other conceptual limitations of the feasibility of planning technologies are analyzed in [CST03]: generally speaking, while (conditional) planning algorithms are devised to compute an actual action scheduling for achieving goals, service aggregation requires the ability to derive application specifications combining available functionalities, i.e., devising a specification of a target complex mediator that essentially reuses the application logic of the provided services. Such a problem is addressed in works like [FBS04a] and [BCD04], unified into a common framework [BCD$^+$05], which provides an algorithm suitable for automatic e-service synthesis. Like in our approach, the concept of atomic/primitive basic service is defined by the specification in terms of its own effects on the state of the world intended as a relational theory, and the frame problem is addressed using *model update* operators. More complex services, that include stateful interaction and multiple operations, are also representable using an annotated finite-state guarded-transition automata: such a model is used to describe available and target services. However, there are some remarkable basic differences:

- there are no preconditions, excepting these deriving on automata state, services are generally accessible and activable, so the contract specification is weaker, despite automata transitions allow guard conditions on transitions that could be feasible for implementing some kind of restrictions;
- the integrity constraints play a quite different role, since they are not used to enforce model repair or service consistency check (a service is always well-defined);

[1]As the present framework, they relies upon a first-order logic theorem automated prover.

- in order to ensure the decidability of automatic synthesis problem, the closed-world assumption is required.

Generally speaking, two frameworks agree upon the foundational aspect that e-service automation problems are related to the exploitation of available capabilities into complex process, from which matchmaking and verification problems arise. In other words, the proposed approach can be integrated into a more general service synthesis solution in order to provide semantic-based consistency and capability analysis tools. In order to achieve such a kind of result, a further extension is required to deal with *stateful* complex e-services.

In fact, from a service composition algorithm perspective, the proposed approach to deal with properties and issues of system transition by means of embedding different system states and their relations in an adequately enriched structure on which employ reasoning techniques can be extended further to verification of correctness and functional properties as presented in this work. Moreover, until the branching degree of the computation paths is finite as well as the path length, the embedding approach can be extended to longer computation in order to check properties of a complex composition. We point out that the finiteness requirement is critical, since we need to extended the alphabet used in definition the working structure proportionally to the number of execution branches.

8.1.3 Verification

The problem of complex service verification and analysis w.r.t. the evolution of the world state is also addressed in similar works based upon the notion of relational transducers (e.g., [DSV04, Spi00]). However, in these works, a single complex service, that operates on a relational database, is analyzed in order to derive some correctness properties. Various equivalence/containment properties (e.g., w.r.t. the interaction protocol, the log/storage evolution, etc.) are also proposed. As discussed in Section 2.2.6, an operational semantics based on the update of the relational theory is used, but constraints on the interaction are only based on state automata, services are generally accessible, and there is no way to enforce applicability preconditions.

A lot of other verification approaches, based upon formal tools like model checking, logic programming, Process Algebra and Petri nets, are also proposed for e-service applications by many authors (e.g., [Mar03, GTL00, HB03, Nak02, PC03, KvB03, CRR02, FBS04a, Wal04, DKR04, PR04]), often coming from the business-process and work-flow modeling fields. Such approaches are mainly focused on the analysis of the interaction protocol among actors in various network configurations, but generally they do not enforce any assumption about the semantics of the performed operations. We generally assume that interaction and acting behavior verification can be done separately, at least for some properties. On the other hand, however, the adoption of highly expressive process languages, like PSL ([S$^+$99]) or colored Petri nets ([vdA98]), allows for the definition of arbitrary behavior constraints, but also requires high-order logical reasoning frameworks, in which all relevant reasoning problems are undecidable. In more sophisticated settings, that accordingly shape the specification language, semantic consistency properties can be verified also for some complex process specifications ([NM02]).

Notably, other approaches based on Description Logics have been devised essentially to deal with the inventory and discovery problem (service catalog and matchmaking), but also a remarkable proposal to include update operators in a rich knowledge description framework has been presented in [BLM+05a] and [WL06].

As previously stated, this approach shares with the present one many common aspects, despite they differ substantially on technical details. The most important point to remark is that in this framework we adopt a decidable but incomplete update repair strategy allowing for arbitrary knowledge base specification, while in [BLM+05a] and related works only restricted TBoxes (despite of expressive languages) are tractable in order to preserve the decidability. Moreover, while in the present approach we have adopted a knowledge representation paradigm as a tool to model an information system, so we are reasoning on all possible computation paths, but we are assuming that the system has only a model at time, in other one, the attention is focused essentially on the update of the knowledge base. In other words, the action formalism is devised in order to deal also with the problem of express using the same language the knowledge base resulting from the update, assuming that it is a concisely denote a family of models. Despite it can turn more useful in knowledge-based application, it also clashes with general assumption adopted in the design of cooperative information systems, where a system state is required to be completely defined, while the available specification can be only partial. In fact, in the respect of source of incompleteness, while we are dealing with incomplete service and state incomplete specifications at design-time, assuming that at run-time the system state is completely defined, in [BLM+05a] the system state considered to be generally incomplete, since the approach is oriented toward knowledge revision applications[2].

Also considerations presented in the analysis of knowledge-based planning approaches can be reported in this case. In fact, from the perspective of the KB update-repair, in the presence of incomplete specification, the devised solution is clearly an approximation that allows us to deal with a problem that is in general unsolvable. Despite the incomplete search strategy can miss an effectively repairable service, we point out that, on the other side, the syntax-driven repair generation approach should generally consider updates that are "closer" to the user target, since it acts locally w.r.t. the updated model, i.e., affecting instances involved in the user request. While it is possible to tune this approach, shaping down the search space introducing also a locality definition in terms of intensional specification (e.g., repairing only certain names given the updated ones), an open issue is whether other wider search strategies can be employed. A first possibility is to extend the generation procedure to keep into account also access function queries, since they preserve the fact that an atomic repair involves at most an element or pair.

However, while there are other approaches based on DLs explicitly focused on dynamic properties, to the best of our knowledge, the present framework is the only proposal that includes also the ability to deal with transition-related constraints (i.e., dynamic constraints). This aspect is, moreover, also interesting from the perspective of conventional approaches in automated verification,

[2]Interestingly in approaches like [GMP04], is the domain specification considered as incomplete (e.g., missing constraints). However, this approach is inherently static and hence it does not address any of issues related to update and repair.

since, despite they adopt some kind of temporal logic to define complex path-related system properties to check, these languages, generally, allow to easily deal essentially with system states formalized in a propositional language.

8.2 Further Work

Starting from the results achieved in the present work, in order to extend the proposed framework in terms of modeling and analysis tools, we are working on the following topics:

1. refining the analysis of the formal properties of the automated reasoning tasks defined in our framework, in particular the ones involving enhanced e-service specification and functional property analysis (e.g., establishing additional complexity lower-bound, analyzing the expressiveness boundary of the working language);

2. extending the domain expression language in order to allow for some kinds of role-based constructs, that are generally not satisfactorily addressed by Description Logics;

3. investigating further on the devised notions of replaceability and functional comparability: while the former could be eventually reduced to quite similar concepts in the field on temporal/action reasoning, the latter is more innovative, at least in the application to the e-service domain, and needs to be adequately assessed;

4. refining the analysis of language constructs in order to shape out some more easily tractable fragments w.r.t. their computational complexity that are also interesting in terms of applicability to real-world problems.

Moreover, in order to devise a more reliable toolkit to support system integration engineering, we are also interested in the assessment of the proposed framework w.r.t. an industrial validation protocol for e-service modeling proposals, in order to point out the most relevant topics when dealing with real-world problems (in particular, in e-government and virtual-enterprise application scenarios).

Bibliography

[AAF$^+$02] Assaf Arkin, Sid Askary, Scott Fordin, Wolfgang Jekeli, Kohsuke Kawaguchi, David Orchard, Stefano Pogliani, Karsten Riemer, Susan Struble, Pal Takacsi-Nagy, Ivana Trickovic, and Sinisa Zimek. Web service choreography interface (WSCI) 1.0. Tech. note, World Wide Web Consortium (W3C), Aug 2002. 2.2.7, A.1.2

[ABG$^+$07] C. Areces, P. Blackburn, V. Goranko, M. Marx, and J. Seligman. The hybrid logic book. In preparation., 2007. 2.2.13

[ACD$^+$03] Tony Andrews, Francisco Curbera, Hitesh Dholakia, Yaron Goland, Johannes Klein, Frank Leymann, Kevin Liu, Dieter Roller, Doug Smith, Satish Thatte, Ivana Trickovic, and Sanjiva Weerawarana. Business process execution language for web services version 1.1. Specification, IBM, BEA, SAP, Microsoft, Siebel Systems, May 2003. A.1.2

[ACG86] S. Ahuja, N. Carriero, and D. Gelernter. Linda and friends. *IEEE Computer*, 19:26–34, 1986. 2.2.2

[ACKM04] Gustavo Alonso, Fabio Casati, Harumi Kuno, and Vijay Machiraju. *Web Services: Concepts, Architectures and Applications*. Springer-Verlag, Berlin, Germany, 2004. 1.1, 2.1, 3.1

[AG97] Robert Allen and David Garlan. A formal basis for architectural connection. *ACM Transactions on Software Engineering and Methodology*, 6(3):213–249, 1997. A.2

[AGP03] Liliana Ardissono, Anna Goy, and Giovanna Petrone. Enabling conversations with web services. In *Proc. of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '03)*, pages 819–826, Melbourne, Australia, July 2003. ACM. 2.2.7

[AHV95a] Serge Abiteboul, Richard Hull, and Victor Vianu. *Dynamic Aspects*, chapter 22. In [AHV95b], 1995. 3.1, 4.3

[AHV95b] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995. 8.2

[AKTV07] Anupriya Ankolekar, Markus Krötzsch, Thanh Tran, and Denny Vrandecic. The two cultures: mashing up web 2.0 and the semantic web. In *Proc. of the WWW 2007 Conference*, pages 825–834, New York, NY, USA, 2007. ACM Press. 2

[APY$^+$02] Marco Aiello, Mike P. Papazoglou, Jian Yang, Mark J. Carman, Marco Pistore, Luciano Serafini, and Paolo Traverso. A request language for web-services based on planning and costraint satisfaction. In *Proc. of the Technologies for E-Services Third International Workshop (TES 2002)*, Hong Kong, China, August 2002. 2.2.12, 3.1, 8.1.1, A.3

[AV89] Serge Abiteboul and Victor Vianu. A transaction-based approach to relational database specification. *Journal of the ACM*, 36(4):758–789, 1989. 4.3

[AVFY98] Serge Abiteboul, Victor Vianu, Brad Fordham, and Yelena Yesha. Relational transducers for electronic commerce. In *Proc. of Symposium on Principles of Database Systems (PODS 1998)*, Seattle, WA, 1998. ACM. 2.2.6, 3.1

[BAF98] Leopoldo E. Bertossi, Marcelo Arenas, and Cristian Ferretti. SCDBR: An automated reasoner for specifications of database updates. *Journal of Intelligent Information Systems*, 10(3):253–280, 1998. 8.1.1

[BAMP81] Mordechai Ben-Ari, Zohar Manna, and Amir Pnueli. The temporal logic of branching time. In *Proc. of the Symposium on Principles of Programming Languages (POPL 1981)*, pages 164–176, New York, NY, USA, 1981. ACM Press. 2.2.6

[Bat03] Steve Battle. Boxes: black, white, grey and glass box views of web-servivces. Technical Report HPL-2003-30, Digital Media Systems Laboratory, HP Laboratories Bristol, February 2003. A.1.1, C.3

[BBC$^+$04] Siddharth Bajaj, Don Box, Dave Chappell, et al. Web services policy framework (WSPolicy). Specification, IBM, BEA, Microsoft, SAP AG, Sonic Software, VeriSign, 2004. A.1.2

[BBM$^+$01] Keith Ballinger, Peter Brittenham, Ashok Malhotra, William A. Nagy, and Stefan Pharies. Web services inspection language (WS-Inspection) 1.0. Specification, IBM, Microsoft, Nov 2001. A.1.2

[BCD$^+$03] Daniela Berardi, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Massimo Mecella. Automatic composition of e-Services that export their behavior. In *Proc. of the 1st International Conference on Service Oriented Computing (ICSOC 2003)*, volume 2910 of *LNCS*. Springer, 2003. 2.2.8, 3.1, A.3, A.4

[BCD04] Daniela Berardi, Diego Calvanese, and Giuseppe De Giacomo. Synthesis of underspecified composite e-Service based on automated reasoning. Technical Report 9, Dip. di Informatica e Sistemistica, Univ. di Roma "la Sapienza", 2004. 2.2.8, 8.1.2

[BCD$^+$05] Daniela Berardi, Diego Calvanese, Giuseppe De Giacomo, Richard Hull, and Massimo Mecella. Automatic composition of transition-based semantic web services with messaging. In *Proc. of the 31st VLDB Conference*, Trondheim, Norway, 2005. 2.2.8, 8.1.2

[BCM$^+$03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications.* Cambridge University Press, 2003. 2, 3.2.2, 8.2

[BCN92] Carlo Batini, Stefano Ceri, and Shamkant B. Navathe. *Conceptual Database Design: An Entity-Relationship Approach.* Benjamin/Cummings, 1992. 3.2.2

[BCvR03] Tom Bellwood, Luc Clément, and Claus von Riegen. UDDI Version 3.0.1. Technical committee specification, OASIS (Organization for the Advancement of Structured Information Standards), Oct 2003. A.1.2

[BFHS03] Tevfik Bultan, Xiang Fu, Richard Hull, and Jianwen Su. Conversation specification: A new approach to design and analysis of E-Service composition. In *Proc. of the WWW 2003 Conference*, Budapest, Hungary, 2003. 2.2.8, 3.1

[BG01] L. Bachmair and H. Ganzinger. *Resolution Theorem Proving*, chapter 2, pages 19–99. Volume I of Robinson and Voronkov [RV01], 2001. 4.4

[BHL$^+$02] Mark H. Burstein, Jerry R. Hobbs, Ora Lassila, David L. Martin, Drew V. McDermott, Sheila A. McIlraith, Srini Narayanan, Massimo Paolucci, Terry R. Payne, and Katia P. Sycara. DAML-S: Web service description for the semantic web. In *Proc. of the 1st International Semantic Web Conference (ISWC 2002)*, Chia, Sardegna, Italy, 2002. A.1.2, A.3

[BHM$^+$04] David Booth, Hugo Haas, Francis McCabe, Eric Newcomer, Michael Champion, Chris Ferris, and David Orchard. Web services architecture. Working group note, World Wide Web Consortium (W3C), Feb 2004. 1, 2.1, 2.2.7

[Biz] Microsoft BizTalk$^{\text{TM}}$Server. `http://www.microsoft.com/biztalk`. A.1.2

[BK93] Anthony J. Bonner and Michael Kifer. Transaction logic programming. In *International Conference on Logic Programming*, pages 257–279, 1993. 2.2.5

[BK02] Abraham Bernstein and Mark Klein. Towards high-precision service retrieval. In *Proc. of the 1st International Semantic Web Conference (ISWC 2002)*, Chia, Sardegna, Italy, 2002. 2.2.13

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):34–44, May 2001. 2.2.13

[BLM$^+$05a] Franz Baader, Carsten Lutz, Maja Milicic, Ulrike Sattler, and Frank Wolter. Integrating description logics and action formalisms: First results. In Manuela M. Veloso and Subbarao Kambhampati, editors, *Proc. of the 2005 National Conference on Artificial Intellingence (AAAI 2005)*, pages 572–577. AAAI Press / The MIT Press, 2005. 2.2.13, 4.2, 4.3, 8.1.3

[BLM$^+$05b] Franz Baader, Carsten Lutz, Maja Milicic, Ulrike Sattler, and Frank Wolter. Integrating description logics and action formalisms for reasoning about web services. Technical report, Institute for Theoretical Computer Science, Desden University of Technology, 2005. 2.2.13, 47

[Bor94] Alex Borgida. On the relationship between description logic and predicate logic queries. In *Proc. of the 3rd International Conference on Information and Knowledge Management (CIKM '94)*, pages 219–225, New York, NY, USA, 1994. ACM Press. 3.5

[BPSM$^+$06] Tim Bray, Jean Paol, C. M. Sperberg-McQueen, Eve Maler, and François Yergeau. Extensible markup language (XML) 1.0 (fourth edition). Recommendation, World Wide Web Consortium (W3C), Sep 2006. 1, 2.1

[BTP04] BTP Technical Committee. Business transaction protocol (BTP) version 1.1. Technical committee specification, OASIS (Organization for the Advancement of Structured Information Standards), Oct 2004. 5

[CCDL02] Andrea Calì, Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. A formal framework for reasoning on UML class diagrams. In *Proc. of the 13th International Symposium on Foundations of Intelligent Systems (ISMIS '02)*, pages 503–513, London, UK, 2002. Springer-Verlag. 7.2, 49

[CD99] James Clark and Steve DeRose. XML Path language (XPath) version 1.0. Recommendation, World Wide Web Consortium (W3C), Nov 1999. 2.2.2

[CD03] Diego Calvanese and Giuseppe De Giacomo. *Expressive Description Logics*, chapter 5. In Baader et al. [BCM$^+$03], 2003. 3, 3.3

[CDLN01] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Daniele Nardi. *Reasoning in Expressive Description Logics*, chapter 23, pages 1581–1634. Volume II of Robinson and Voronkov [RV01], 2001. 3.3

[CDV05] Diego Calvanese, Giuseppe De Giacomo, and Moshe Y. Vardi. Decidable containment of recursive queries. *Theor. Comput. Sci.*, 336(1):33–56, 2005. 3.5

[CGM$^+$04] Roberto Chinnici, Martin Gudgin, Jean-Jacques Moreau, Jeffrey Schlimmer, and Sanjiva Weerawarana. Web services description language (WSDL) version 2.0. Working draft, World Wide Web Consortium (W3C), Aug 2004. A.1.2

[CIJ+00] Fabio Casati, Ski Ilnicki, LiJie Jin, Vasudev Krishnamoorthy, and Ming-Chien Shan. Adaptive and dynamic service composition in eFlow. In *The 12th International Conference on Advanced Information Systems Engineering (CAiSE 2000)*, Stockholm, Sweden, June 2000. 3.1, 1

[CJ01] Dipanjan Chakraborty and Anupam Joshi. Dynamic service composition: State-of-the-art and research directions. Technical Report TR-CS-01-20, Dept. of Computer Science and Electrical Engineering, University of Maryland, Baltimora, Maryland, December 2001. A.4

[CLNS97] Marco Cadoli, Maurizio Lenzerini, Paolo Naggar, and Andrea Schaerf. *Fondamenti della progettazione dei programmi. Principi, tecniche e loro applicazioni in C++*. Città Studi Edizioni di UTET Libreria, Torino, January 1997. 2.1

[CLR03] Andrea Calì, Domenico Lembo, and Riccardo Rosati. On the decidability and complexity of query answering over inconsistent and incomplete databases. In *Proc. of Symposium on Principles of Database Systems (PODS 2003)*, pages 260–271, New York, NY, USA, 2003. ACM Press. 4.3

[CNI05a] CNIPA SPCoop Working Group. The Italian Public Cooperation System (SPCoop): Architecture. Technical report, Centro Nazionale per l'Informatica nella Pubblica Amministrazione (CNIPA), 2005. in Italian. 3.1, 4

[CNI05b] CNIPA SPCoop Working Group. The Italian Public Cooperation System (SPCoop): Organization. Technical report, Centro Nazionale per l'Informatica nella Pubblica Amministrazione (CNIPA), 2005. in Italian. 3.1, 4

[CNI05c] CNIPA SPCoop Working Group. The Italian Public Cooperation System (SPCoop): Technology. Technical report, Centro Nazionale per l'Informatica nella Pubblica Amministrazione (CNIPA), 2005. in Italian. 3.1, 4

[CRR02] Sagar Chaki, Sriram K. Rajamani, and Jakob Rehof. Types as models: Model checking message-passing programs. In *Proc. of the Symposium on Principles of Programming Languages (POPL 2002)*, pages 45–57, Portland, OR, Janurary 2002. ACM. 2.2.4, 8.1.3

[CS01] E. M. Clarke and H. Schlingloff. *Model Checking*, chapter 24, pages 1635–1790. Volume II of Robinson and Voronkov [RV01], 2001. 2.2.7, 3

[CST03] Mark Carman, Luciano Serafini, and Paolo Traverso. Web Service Composition as Planning. In *Proc. of ICAPS 2003 Workshop on Planning for Web Services*, Trento, Italy, June 2003. 2.2.12, 8.1.2

[cXM] Commerce XML (cXML). http://www.cxml.org/. A.1.2

[d'A07] Claudia d'Amato. Constraint hardness for modelling, matching and ranking semantic web services. In Michelangelo Ceci, Donato Malerba, and Letizia Tanca, editors, *Proc. of the 15th Italian Symposium on Advanced Database Systems (SEBD 2007)*, pages 373–380, 2007. 2.2.13, 6

[dBFK$^+$05] Jos de Bruijn, Dieter Fensel, Uwe Keller, Michael Kifer, Holger Lausen, Reto Krummenacher, Axel Polleres, and Livia Predoiu. Web service modeling language (WSML). Submission, World Wide Web Consortium (W3C), Jun 2005. 1.2, 2.2.13

[DCO] Distributed Component Object Model (DCOM). http://www.microsoft.com/com/tech/DCOM.asp. A.1.2

[DD02] Chris Date and Hugn Darwen. *Temporal Data and the Relational Model*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002. 3

[De 95] Giuseppe De Giacomo. *Decidability of Class-Based Knowledge Representation Formalisms*. PhD thesis, Dip. di Informatica e Sistemistica, Univ. di Roma "La Sapienza", 1995. 3

[DKP$^+$99] Hasan Davulcu, Michael Kifer, L. R. Pokorny, C. R. Ramakrishnan, I. V. Ramakrishnan, and Steven Dawson. Modeling and analysis of interactions in virtual enterprises. *Research Issues on Data Engineering*, 1999. 2.2.5

[DKR04] Hasan Davulcu, Michael Kifer, and I. V. Ramakrishnan. CTR-S: A logic for specifying contracts in semantic web services. In *Proc. of the WWW 2004 Conference*, New York, NY, 2004. 2.2.5, 3.1, 8.1.3

[DKRR98] Hasan Davulcu, Michael Kifer, C. R. Ramakrishnan, and I. V. Ramakrishnan. Logic based modeling and analysis of workflows. In *Proc. of Symposium on Principles of Database Systems (PODS 1998)*, pages 25–33, Seattle, WA, 1998. ACM. 2.2.5

[DLPR06] Giuseppe De Giacomo, Maurizio Lenzerini, Antonella Poggi, and Riccardo Rosati. On the update of description logic ontologies at the instance level. In *Proc. of the 2006 National Conference on Artificial Intellingence (AAAI 2006)*, 2006. 4.3

[DNDSDM03] Tommaso Di Noia, Eugenio Di Sciascio, Francesco M. Donini, and Marina Mongiello. A system for principled matchmaking in an electronic marketplace. In *Proc. of the WWW 2003 Conference*, Budapest, Hungary, 2003. 2.2.13, 3.1

[DSV04] Alin Deutsch, Liying Sui, and Victor Vianu. Specification and verification of data-driven web services. In *Proc. of Symposium on Principles of Database Systems (PODS 2004)*, Paris, France, 2004. ACM. 2.2.6, 3.1, 8.1.3

[DV01] A. Degtyarev and A. Voronkov. *The Inverse Method*, chapter 4, pages 179–272. Volume I of Robinson and Voronkov [RV01], 2001. 4.4

[ebX03] ebXML Technical Commitee. ebXML – electronic business using XML. Technical committee specification, OASIS (Organization for the Advancement of Structured Information Standards), Oct 2003. A.1.2

[EDI] UN/EDIFACT – United Nations Directories for Electronic Data Interchange for Administration, Commerce and Transport. `http://www.unece.org/trade/untdid/`. A.1.2

[EFL$^+$04] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. A logic programming approach to knowledge-state planning: Semantics and complexity. *ACM Transactions on Computational Logic*, 5(2):206–263, April 2004. 3.1

[EG92] Thomas Eiter and Georg Gottlob. On the complexity of propositional knowledge base revision, updates, and counterfactuals. In *Proc. of Symposium on Principles of Database Systems (PODS 1992)*, pages 261–273, New York, NY, USA, 1992. ACM Press. 4.3

[Esp96] Javier Esparza. Decidability and complexity of petri net problems - an introduction. In Wolfgang Reisig and Grzegorz Rozenberg, editors, *Petri Nets*, volume 1491 of *Lecture Notes in Computer Science*, pages 374–428. Springer, 1996. 2.2.1

[EW90] Uffe Engberg and Glynn Winskel. Petri nets as models of linear logic. In A. Arnold, editor, *Proc. of Colloquium on Trees in Algebra and Programming*, pages 147–161, Copenhagen, Denmark, 1990. Springer-Verlag LNCS 389. B.2

[EW93] Uffe Engberg and Glynn Winskel. Completeness results for linear logic on Petri nets. In A. Borzyszkowski and S. Sokolowski, editors, *Proc. of the Conference on Mathematical Foundations of Computer Science*, pages 442–452, Gdańsk, Poland, 1993. Springer-Verlag LNCS 711. B.2

[FBS04a] Xiang Fu, Tevfik Bultan, and Jianwen Su. Analysis of interacting BPEL web services. In *Proc. of the WWW 2004 Conference*, New York, NY, 2004. 2.2.8, 8.1.2, 8.1.3

[FBS04b] Xiang Fu, Tevfik Bultan, and Jianwen Su. Model checking XML manipulating software. In *Proc. of the International Symposium on Software Testing and Analysis (ISSTA 2004)*, pages 252–262. ACM Press, 2004. 2.2.8

[FGM$^+$99] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. Technical report, Internet Engineering Task Force, June 1999. 2.1

[FIP99] Foundation for Intelligent Physical Agents. *FIPA Specification Part 2 – Communication Language*, 1999. 2.2.7, 2.2.7

[Fis01] Bernd Fischer. *Deduction based component retrieval*. PhD thesis, Univesität Passau, Nov 2001. A.2

[FLMS99] Daniela Florescu, Alon Levy, Ioana Manolescu, and Dan Suciu. Query optimization in the presence of limited access patterns. In *Proc. of the 1999 ACM SIGMOD International Conference on Management of Data*, pages 311–322. ACM Press, 1999. B.1

[FLP$^+$03] Renato Fileto, Ling Liu, Calton Pu, Eduardo Delgado Assad, and Claudia B. Medeiros. POESIA: An ontological workflow approach for composing web services in agriculture. *The VLDB Journal*, 12:352–367, 2003. A.3, 1

[G4B04] Progetto G4B: Government for Business. `http://www.g4b.it/`, 2004. in Italian. 3.1, 4

[GHM$^+$03] Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, and Henrik Frystyk Nielsen. SOAP version 1.2. Recommendation, World Wide Web Consortium (W3C), June 2003. A.1.2

[GHS96] Gerd Große, Steffen Hölldobler, and Josef Schneeberger. Linear deductive planning. *Journal of Logic and Computation*, 6(2):233–262, 1996. B.2

[Gir87] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987. B.2

[GL93] Michael Gelfond and Vladimir Lifschitz. Representing action and change by logic programs. *Journal of Logic Programming*, 17(2/3):301–321, 1993. 2.2.10

[GM05] Stephan Grimm and Boris Motik. Closed World Reasoning in the Semantic Web through Epistemic Operators. In Bernardo Cuenca Grau, Ian Horrocks, Bijan Parsia, and Peter Patel-Schneider, editors, *Proc. of the Workshop on OWL: Experiences and Directions (OWLED 2005)*, Galway, Ireland, November 11–12 2005. 2.2.13, 6

[GMP04] Stephan Grimm, Boris Motik, and Chris Preist. Variance in e-Business Service Discovery. In David Martin, Rubén Lara, and Takahira Yamaguchi, editors, *Proc. of the ISWC 2004 Workshop on Semantic Web Services: Preparing to Meet the World of Business Applications*, volume 119 of *CEUR Workshop Proceedings*, Hiroshima, Japan, November 8 2004. 2.2.13, 6, 2

[GMP06] Stephan Grimm, Boris Motik, and Chris Preist. Matching Semantic Service Descriptions with Local Closed-World Reasoning. In York Sure and John Domingue, editors, *Proc. of the 3rd European Semantic Web Conf. (ESWC 2006)*, volume 4011 of *LNCS*, pages 575–589, Budva, Montenegro, June 11–14 2006. Springer. 2.2.13, 3.1, 6

[GOR97] Erich Grädel, Martin Otto, and Eric Rosen. Two-variable logic with counting is decidable. In *Proc. of 12th IEEE Symposium on Logic in Computer Science (LICS '97)*, 1997. 3.2.1

[Gro03] Benjamin N. Grosof. SweetDeal: Representing agent contracts with exceptions using semantic web rules, ontologies, and process descriptions. *International Journal of Electronic Commerce (IJEC)*, 2003. A.3

[GTL00] Chitta Baral Goce Trajcevski and Jorge Lobo. Formalizing (and reasoning about) the specifications of workflows. In *Proc. of the Fifth IFCIS International conference on Cooperative Information Systems (CoopIS'2000)*, 2000. 2.2.10, 8.1.3, A.3

[Häh01] R. Hähnle. *Tableaux and Related Methods*, chapter 3, pages 100–178. Volume I of Robinson and Voronkov [RV01], 2001. 4.4

[HB03] Rachid Hamadi and Boualem Benatallah. A Petri net-based model for web service composition. In *Proc. of 14th Australasian Database Conference (ADC 2003)*, Adelaide, Australia, 2003. 2.2.1, 8.1.3, A.1.1, A.3

[HBCS03] Richard Hull, Michael Benedikt, Vassilis Christophides, and Jianwen Su. E-Services: A look behind the curtain. In *Proc. of Symposium on Principles of Database Systems (PODS 2003)*, San Diego, CA, 2003. 4, A.4

[HBVL97] Thomas Hillenbrand, Arnim Buch, Roland Vogt, and Bernd Löchner. WALDMEISTER - High-Performance Equational Deduction. *J. Autom. Reason.*, 18(2):265–270, 1997. 8

[HG97] David Harel and Eran Gery. Executable object modeling with statechart. *IEEE Computer*, 30(7):31–42, July 1997. 2.2.2

[HHO04] Hao He, Hugo Haas, and David Orchard. Web services architecture usage scenarios. Working group note, World Wide Web Consortium (W3C), Feb 2004. A.5

[HKS06] Ian Horrocks, Oliver Kutz, and Ulrike Sattler. The even more irrestistible sroiq. In *Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006)*. AAAI Press, 2006. 7.1

[Hoa69] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 1969. 3.1

[Hoa85] C. A. R. Hoare. *Communicating sequential processes*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1985. 2.2.4

[Hof03] K. Douglas Hoffman. Marketing + MIS = e-service. *Communications of the ACM*, 46(6):53–55, 2003. 2.1

[Hol04] Gerard J. Holzmann. *The SPIN Model Checker*. Addison Wesley Professional, 2004. 2.2.7

[HPPSH05] Ian Horrocks, Bijan Parsia, Peter F. Patel-Schneider, and James Hendler. Semantic web architecture: Stack or two towers? In Francois Fages and Sylvain Soliman, editors, *Principles and Practice of Semantic Web Reasoning (PPSWR 2005)*, number 3703 in LNCS, pages 37–41. SV, 2005. 3.1

[HPS03] Ian Horrocks and Peter F. Patel-Schneider. Reducing OWL entailment to description logic satisfiability. In *Proc. of the 2nd International Semantic Web Conference (ISWC 2003)*, 2003. 2.2.13, 7.1

[HS98] Ullrich Hustadt and Renate A. Schmidt. Issues of decidability for description logics in the framework of resolution. In *FTP (LNCS Selection)*, pages 191–205, 1998. 46

[HS04] Richard Hull and Jianwen Su. Tools for design of composite web services. Talk at SIGMOD'04, June 2004. 2.1, A.4

[HSG04] U. Hustadt, R. A. Schmidt, and L. Georgieva. A survey of decidable first-order fragments and description logics. *Journal of Relational Methods in Computer Science*, 1:251–276, 2004. Invited overview paper. 3.2.1, 7.1

[HU79] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, 1979. 2.2.8

[IK01] Atsushi Igarashi and Naoki Kobayashi. A generic type system for the π-calculus. In *Proc. of the Symposium on Principles of Programming Languages (POPL 2001)*, pages 128–141. ACM Press, 2001. 2.2.4

[IL90] Neil Immerman and Eric Lander. Describing graphs: A first-order approach to graph canonization. In *Alan L. Selman, Editor, Complexity Theory Retrospective, In Honor of Juris Hartmanis on the Occasion of His Sixtieth Birthday, July 5, 1988*, volume 1. 1990. 3.5

[Jan95] Petr Jancar. Undecidability of bisimilarity for petri nets and some related problems. *Theoretical Computer Science*, 148(2):281–301, 1995. 2.2.1

[JEE] Java$^{\text{TM}}$Platform Enterprise Edition 5. `http://java.sun.com/javaee`. 8

[KAJR03] Dan Jong Kim, Manish Agrawal, Bharat Jayaraman, and H. Raghav Rao. A comparison of B2B E-Service solutions. *Communications of the ACM*, 46(12):317–324, 2003. 2.1

[Kal01] John Arnold Kalman. *Automated Reasoning with Otter*. Rinton Press, Incorporated, 2001. 8

[Kal03] Muhammad F. Kaleem. A classification framework for approaches and methodologies to make web services compositions reliable. In *Proc. First European Workshop on Object Orientation and Web Service (EOOWS)*, Darmstadt, Germany, 2003. A.2

[Kar03] Alan H. Karp. E-speak e-xplained. *Communications of the ACM*, 46(7):112–118, 2003. A.1.2

[Kay03] Doug Kaye. *Loosely Coupled: The Missing Pieces of Web Services*. RDS Press, 2003. 3.1

[KBR04] Nickolaos Kavantzas, David Burdett, and Greg Ritzinger. Web services choreography description language version 1.0. Working draft, World Wide Web Consortium (W3C), Apr 2004. A.1.2

[KM06a] Jeff Kramer and Jeff Magee. *Concurrency: State Models Java Programs, 2nd Edition*. John Wiley Sons, April 2006. 2.2.11

[KM06b] Peep Kungas and Mihhail Matskin. Symbolic negotiation in linear logic with coalition formation. In *Proc. of the International Conference on Intelligent Agent Technology (IAT '06)*, pages 298–305, Washington, DC, USA, 2006. IEEE Computer Society. B.2

[KP07] Ryan F. Kelly and Adrian R. Pearce. Property persistence in the situation calculus. In Manuela M. Veloso, editor, *Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, volume 2, pages 1948–1953, Hyderabad, India, 2007. IJCAI. 8.1.1

[KS03] Jana Koehler and Biplav Srivastava. Web service composition: Current solutions and open problems. In *Proc. of ICAPS 2003 Workshop on Planning for Web Services*, pages 28–35, 2003. A.2

[KU99] Emmanuel Kounalis and Pascal Urso. Mechanizing proofs of integrity constraints in the situation calculus. In *Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, pages 372–381, 1999. 8.1.1

[Kün02] Peep Küngas. Resource-conscious ai planning with conjunctions and disjunctions. *Acta Cybernetica*, 15(4):601–620, 2002. B.2

[KV01] Max Kanovich and Jacqueline Vauzeilles. The classical ai planning problems in the mirror of horn linear logic: semantics, expressibility, complexity. *Mathematical. Structures in Comp. Sci.*, 11(6):689–716, 2001. B.2

[KvB03] Mariya Koshkina and Franck van Breugel. Verification of business process for web services. Technical Report CS-2003-11, Dept. of Computer Science, University of York, 4700 Keele Street, Toronto, Ontario M3J 1P3, Canada, October 2003. 2.2.4, 8.1.3

[Len02] Maurizio Lenzerini. Data integration: A theoretical perspective. In Lucian Popa, editor, *Proc. of Symposium on Principles of Database Systems (PODS 2002)*, pages 233–246. ACM Press, 2002. B.1

[Ler04] Barbara Staudt Lerner. Verifying process models built using parameterized state machines. *SIGSOFT Softw. Eng. Notes*, 29(4):274–284, 2004. 2.2.11

[LH03] Lei Li and Ian Horrocks. A software framework for matchmaking based on semantic web technology. In *Proc. of the WWW 2003 Conference*, Budapest, Hungary, 2003. 3.1

[Lit03] Mark Little. Transactions and web services. *Communications of the ACM*, 46(10):49–54, October 2003. 2

[LL76] Richard E. Ladner and Nancy A. Lynch. Relativization of questions about log space computability. *Mathematical Systems Theory*, 10:19–32, 1976. 3.2.3, 9

[LLW02] David Liu, Kincho H. Law, and Gio Wiederhold. Analysis of integration models for service composition. In *Proc. of the 3rd International Workshop on Software and Performance (WOSP '02)*, pages 158–165, Rome, Italy, July 2002. ACM. A.4

[LM96] Jinxin Lin and Alberto O. Mendelzon. Merging databases under constraints. *International Journal of Cooperative Information Systems*, 7(1):55–76, 1996. 4.2

[LRL$^+$97] Hector J. Levesque, Raymond Reiter, Yves Lesperance, Fangzhen Lin, and Richard B. Scherl. GOLOG: A logic programming language for dynamic domains. *Journal of Logic Programming*, 31(1–3):59–84, April–June 1997. 2.2.9

[LS01] Carsten Lutz and Ulrike Sattler. The complexity of reasoning with boolean modal logics. In Frank Wolter, Heinrich Wansing, Maarten de Rijke, and Michael Zakharyaschev, editors, *Advances in Modal Logics Volume 3*. CSLI Publications, Stanford, 2001. 7.1

[LSAS00] Amaia Lazcano, Heiko Schuldt, Gustavo Alonso, and Hans-Jörg Schek. The WISE approach to electronic commerce. *International Journal of Computer Systems Science and Engineering*, 15(5):343–355, 2000. 2.1

[Mar03] Axel Martens. Usability of web services. In *Proc. of the 1st Web Service Quality Workshop*. IEEE Press, 2003. 2.2.1, 8.1.3, A.3

[MB03] L. G. Meredith and Steve Bjorg. Contract and types. *Communications of the ACM*, 46(10):41–47, October 2003. 2.2.4, 3.1, A.3

[MBE03] Brahim Medjahed, Athman Bouguettaya, and Ahmed K. Elmagarmid. Composing web services on the semantic web. *The VLDB Journal*, 12:333–351, 2003. 3.1

[MBM03] Zakaria Maamar, Boualem Benatallah, and Wathiq Mansoor. Service chart diagrams – description & application. In *Proc. of the WWW 2003 Conference*, Budapest, Hungary, May 2003. 2.2.2

[McD98] Drew McDermott. The planning domain definition language manual. Technical Report 1165, Dept. of Computer Science, Univ. of Yale, 1998. 2.2.12

[McD02] Drew McDermott. Estimated-regression planning for interactions with web services. In *Proc. of the 2002 AI Planning Systems Conference*, 2002. 2.2.12, 3.1, 8.1.1

[McM93] Kenneth L. McMillan. *Symbolic Model Checking*. Kluwer Academic Publishers, Norwell, MA, USA, 1993. 2.2.4, 2.2.5

[Mey88] Bertrand Meyer. *Object-Oriented Software Construction*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988. 4.2

[MH69] John McCarthy and Patrick J. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. Edinburgh University Press, 1969. reprinted in McC90. 2.2.9

[Mil89] Robin Milner. *Communication and Concurrency*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989. 2.2.4

[Mil93] Robin Milner. The polyadic π-calculus: a tutorial. In F. L. Bauer, W. Brauer, and H. Schwichtenberg, editors, *Logic and Algebra of Specification*, pages 203–246. Springer-Verlag, 1993. 2.2.4

[Mit97] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997. 6.1.2

[MKF02] Nirmal Mukhi, Rania Khalaf, and Paul Fremantle. Multi-protocol web services for enterprises and the grid. In *Proc. of EuroWeb 2002 Conference*, Oxford, UK, December 2002. 3

[MKY04] Zakaria Maamar, Soraya Kouadri, and Hamdi Yahyaoui. A web services composition approach based on software agents and context. In *Proc. of the 2004 ACM Symposium on Applied Computing (SAC '04)*, pages 1619–1623, Nicosia, Cyprus, March 2004. ACM. 2.2.7

[MM03] Daniel J. Mandell and Sheila A. McIlraith. Adapting BPEL4WS for the semantic web: The bottom-up approach to web service interoperation. In *The Proc. of the 2nd International Semantic Web Conference (ISWC 2003)*, Sanibel Island, Florida, 2003. 5

[MPC01] Massimo Mecella, Barbara Pernici, and Paolo Craca. Compatibility of e-Services in a cooperative multi-platform environment. *Lecture Notes in Computer Science*, 2193, 2001. 2.1, 2.2.3, 3, A.3

[MPM$^+$04] David L. Martin, Massimo Paolucci, Sheila A. McIlraith, Mark H. Burstein, Drew V. McDermott, Deborah L. McGuinness, Bijan Parsia, Terry R. Payne, Marta Sabou, Monika Solanki, Naveen Srinivasan, and Katia P. Sycara. OWL-S: Semantic markup for web services. Submission, World Wide Web Consortium (W3C), November 2004. 1.2, A.1.2

[MS02] Sheila A. McIlraith and Tran Cao Son. Adapting Golog for composition of semantic web services. In *Proc. of the 8th International Conference on Knowledge Representation and Reasoning (KR 2002)*, Toulouse, France, 2002. 2.2.9, 3.1, 8.1.1, A.3

[MW71] Zohar Manna and Richard J. Waldinger. Toward automatic program synthesis. *Communications of the ACM*, 14(3):151–165, 1971. A.2

[Nak02] Shin Nakajima. Model-checking verification for reliable web services. In *OOPSLA 2002 Workshop on Object-Oriented Web Services*, 2002. 6, 8.1.3

[NET] Microsoft .NET 3.0. `http://www.microsoft.com/net`. 8

[NM02] Srini Narayanan and Sheila A. McIlraith. Simulation, verification an automated composition of web services. In *Proc. of the WWW 2002 Conference*, Honolulu, HI, May 2002. ACM. 2.2.1, 8.1.3

[NMAC$^+$01] Dana S. Nau, Héctor Muñoz-Avila, Yue Cao, Amnon Lotem, and Steven Mitchell. Total-order planning with partially ordered subtasks. In *IJCAI 2001*, pages 425–430, 2001. 2.2.11

[NR01] R. Nieuwenhuis and A. Rubio. *Paramodulation-Based Theorem Proving*, chapter 7, pages 371–443. Volume I of Robinson and Voronkov [RV01], 2001. 4.4

[NRF06] Eric Newcomer, Ian Robinson, and Max Feingold. Web services coordination (WS-Coordination). Technical committee specification, OASIS (Organization for the Advancement of Structured Information Standards), Mar 2006. 5

[NRLH06] Eric Newcomer, Ian Robinson, Mark Little, and John Harby. Web services business activity framework (WS-BusinessActivity). Technical committee specification, OASIS (Organization for the Advancement of Structured Information Standards), Mar 2006. 5

[NRLW06] Eric Newcomer, Ian Robinson, Mark Little, and Andrew Wilkinson. Web services atomic transaction (WS-AtomicTransaction). Technical committee specification, OASIS (Organization for the Advancement of Structured Information Standards), Mar 2006. 5

[OMG04] OMG CORBA Task Force. CORBA^{TM} version 3.0. Specification, Object Management Group (OMG), 2004. 1, A.1.2

[OMG07] OMG UML Task Force. Unified Modeling Language (UML^{TM}) version 2.2. Specification, Object Management Group (OMG), 2007. 2.2.2, 7.2

[PA99] John Penix and Perry Alexander. Efficient specification-based component retrieval. *Automated Software Engineering*, 1999. A.2

[Pap94] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994. 3.2.3

[PC03] C. Phal and M. Casey. Ontology support for web services processes. In *Proc. of the 9th European Software Engineering Conference (ESEC/FSE '03)*, pages 208–216, Helsinki, Finland, September 2003. ACM. 2.2.13, 8.1.3, A.3

[Pee03a] J. Peer. Review of existing work on service matchmaking. Avail. at http://sws.mcm.unisg.ch/docs/matchmaking.pdf, March 2003. C.1

[Pee03b] J. Peer. Towards automatic web service composition using AI planning techniques. Avail. at http://sws.mcm.unisg.ch/docs/wsplanning.pdf, August 2003. 2.2.12, 8.1.1

[Peo05] Progetto PEOPLE: Enti On-line Portali Locali E-government. http://www.progettopeople.it/, 2005. in Italian. 3.1, 4

[Pet62] Carl Adam Petri. *Kommunikation mit Automaten*. PhD thesis, Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962. Second Edition:, New York: Griffiss Air Force Base, Technical Report RADC-TR-65–377, Vol.1, 1966, Pages: Suppl. 1, English translation. 2.2.1

[PEZS02] Giacomo Piccinelli, Wolfgang Emmerich, Christian Zirpins, and Kevin Schuett. Web service interfaces for inter-organisational business processes: An infrastructure for automated reconciliation. In *Proc. of the 6th International Enterprise Distributed Object Computing Conference (EDOC 2002)*, pages 285–292, 2002. A.3

[PF02] Shankar R. Ponnekanti and Armando Fox. SWORD: A development toolkit for web service compositon. In *Proc. of the WWW 2002 Conference*, Honolulu, HI, May 2002. ACM. B.1

[PG03] Mike P. Papazoglou and D. Georgakopoulos. Service-oriented computing. *Communications of the ACM*, 46(10):25–28, October 2003. 2.1, 3.1

[PH05] Ian Pratt-Hartmann. Complexity of the two-variable fragment with counting quantifiers. *J. of Logic, Lang. and Inf.*, 14(3):369–395, 2005. 3.2.1, 3.3, 3.5, 18

[PKPS02] Massimo Paolucci, Takahiro Kawamura, Terry R. Payne, and Katia P. Sycara. Semantic matching of web services capabilities. In *Proc. of the 1st International Semantic Web Conference (ISWC 2002)*, Chia, Sardegna, Italy, 2002. 2.2.13, 3.1

[PL00] Rachel Pottinger and Alon Y. Levy. A scalable algorithm for answering queries using views. In *Proc. of the 26th VLDB Conference*, 2000. B.1

[Pnu77] Amir Pnueli. The temporal logic of programs. In *Proc. of the 18th Symposium Foundations of Computer Science (FOCS 1977)*, pages 46–57, 1977. 2.2.4

[PR99] Fiora Pirri and Raymond Reiter. Some contributions to the metatheory of the situation calculus. *Journal of the ACM*, 46(3):325–361, 1999. 8.1.1

[PR04] L. R. Pokorny and C. R. Ramakrishnan. Modeling and verification of distributed autonomous agents using logic programming. In *Declarative Agent Languages and Technologies (DALT)*, New York, New York, July 2004. 2.2.7, 8.1.3

[Rei91] Raymond Reiter. The frame problem in situation the calculus: a simple solution (sometimes) and a completeness result for goal regression. In *Artificial intelligence and mathematical theory of computation: papers in honor of John McCarthy*, pages 359–380. Academic Press Professional, Inc., San Diego, CA, USA, 1991. 2.2.9

[Rei92] Raymond Reiter. Formalizing database evolution in the situation calculus. In *Proceedings of the International Conference on Fifth Generation Computer Systems*, pages 600–609, ICOT, Japan, 1992. Association for Computing Machinery. 8.1.1

[RK03] Roland T. Rust and P. K. Kannan. E-service: a new paradigm for business in the electronic environment. *Communications of the ACM*, 46(6):36–42, 2003. 2.1

[RK07] Dumitru Roman and Michael Kifer. Reasoning about the behavior of semantic web services with concurrent transaction logic. In *Proc. of the 33rd VLDB Conference*, 2007. 2.2.5

[RKL$^+$05] Dumitru Roman, Uwe Keller, Holger Lausen, Jos de Bruijn, Rubén Lara, Michael Stollberg, Axel Polleres, Cristina Feier, Cristoph Bussler, and Dieter Fensel. Web service modeling ontology. *Applied Ontology*, 1(1):77 – 106, 2005. 2.2.13

[RN95] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995. 2.2.12, 3.1

[Ros] RosettaNet: Lingua franca for E-Business. `http://www.rosettanet.org/`. A.1.2

[RS03] Jinghai Rao and Xiaomeng Su. Toward composition of semantic web services. In *Proc. of the 2nd International Workshop on Grid and Cooperative Computing*, LNCS, Shanghai, China, December 2003. Springer-Verlag. B.2

[RSU95] Anand Rajaraman, Yehoshua Sagiv, and Jeffrey D. Ullman. Answering queries using templates with binding patterns (extended abstract). In *Proc. of Symposium on Principles of Database Systems (PODS 1995)*, pages 105–112. ACM Press, 1995. B.1

[RV01] Alan Robinson and Andrei Voronkov, editors. *Handbook of Automated Reasoning*. Elsevier Science Publishers, 2001. 8, 8.2

[RV02] Alexandre Riazanov and Andrei Voronkov. The design and implementation of VAMPIRE. *AI Commun.*, 15(2,3):91–110, 2002. 8

[S$^+$99] C. Schlenoff et al. The essence of the process specification language. *Transactions of the Society of Computer Simulation*, 16(4):204–216, 1999. 2.2.9, 8.1.3

[SB05] Jocelyn Simmonds and M. Cecilia Bastarrica. A tool for automatic uml model consistency checking. In *Proc. of the 20th IEEE/ACM international Conference on Automated Software Engineering (ASE '05)*, pages 431–432, New York, NY, USA, 2005. ACM Press. 7.2, 49

[SBD03] Quan Z. Sheng, Boualem Benatallah, and Marlon Dumas. The SELF-SERV environment for web services composition. *IEEE Internet Computing*, pages 40–48, January–February 2003. 2.2.2

[SBM$^+$04] Quan Z. Sheng, Boualem Benatallah, Zakaria Maamar, Marlon Dumas, and Anne H. H. Ngu. Enabling personalized composition and adaptive provisioning of web services. In *The 16th International Conference on Advanced Information Systems Engineering (CAiSE 2004)*, pages 322–337, Riga, Latvia, June 2004. 2.2.2

[SBS04] Gwen Salaün, Lucas Bordeaux, and Marco Schaerf. Describing and reasoning on web services using process algebra. In *Proc. of International Conference on Web Services (ICWS 2004)*. IEEE Computer Society, 2004. 3.1

[Sch04] Stephan Schulz. System Description: E 0.81. In D. Basin and M. Rusinowitch, editors, *Proc. of the 2nd IJCAR, Cork, Ireland*, volume 3097 of *LNAI*, pages 223–228. Springer, 2004. 8

[SdF03] Mithun Sheshagiri, Marie des Jardins, and Tim Finin. A planner for composing services described in DAML-S. In *Proc. of ICAPS 2003 Workshop on Planning for Web Services*, 2003. 3.1

[SKWL99] Katia P. Sycara, Matthias Klusch, Seth Widoff, and Jianguo Lu. Dynamic service matchmaking among agents in open information environments. *SIGMOD Record*, 28(1):47–53, March 1999. 2.2.13

[Spi00] Marc Spielmann. Verification of relational transducers for electronic commerce. In *Proc. of Symposium on Principles of Database Systems (PODS 2000)*, Dallas, TX, 2000. ACM. 2.2.6, 8.1.3

[Sri02] Biplav Srivastava. Automatic web services composition using planning. In *Proc. of KBCS 2002*, Mumbai, India, 2002. 3.1

[SS89] Manfred Schmidt-Schaubß. Subsumption in KL-ONE is undecidable. In *Proc. of the 1st International Conference on Principles of Knowledge Representation and Reasoning*, pages 421–431, San Francisco, CA, USA, 1989. Morgan Kaufmann Publishers Inc. 7.1

[Sta02] Michael Stal. Web services: beyond component-based computing. *Communications of the ACM*, 45(10):71–76, 2002. 2.1

[Sta03] Thomas F. Stafford. Introduction. *Communications of the ACM*, 46(6):26–28, 2003. 2.1

[SWM04] Michael K. Smith, Chris Welty, and Deborah L. McGuinness. OWL web ontology language guide. Recommendation, World Wide Web Consortium (W3C), Feb 2004. A.1.2

[TBB03] Mark Turner, David Budgen, and Pearl Brereton. Turning software into a service. *IEEE Computer*, 36:38–44, 2003. 2.1

[TBMM01] Henry S. Thompson, David Beech, Murray Maloney, and Noah Mendelsohn. XML Schema. Recommendation, World Wide Web Consortium (W3C), May 2001. A.1.2

[Tha00] Bernhard Thalheim. *Entity-Relationship Modeling: Foundations of Database Technology*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2000. 3

[Tha01] Satish Thatte. XLANG web services for business process design. Technical report, Microsoft, 2001. A.1.2

[TKAS02] Snehal Thakkar, Craig A. Knoblock, José Luis Ambite, and Cyrus Shahabi. Dinamically composing web services from on-line sources. In *Proc. of the AAAI 2002 Workshop on Intelligent Service Integration*, Edmonton, Alberta, Canada, 2002. B.1

[Tob99] Stephan Tobies. A NExpTime-complete description logic strictly contained in C^2. In Jörg Flum and Mario Rodríguez-Artalejo, editors, *CSL*, volume 1683 of *Lecture Notes in Computer Science*, pages 292–306. Springer, 1999. 5

[Tob00] Stephan Tobies. The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. *J. Artif. Intell. Res. (JAIR)*, 12:199–217, 2000. 3.2.2, 5

[TRC86] H. Tardieu, A. Rochfeld, and R. Colleti. *La Méthode MERISE*. Les Editions d'Organisation, Paris, 1986. A.4

[Var96] Moshe Y. Vardi. Why is modal logic so robustly decidable? In Neil Immerman and Phokion G. Kolaitis, editors, *Descriptive Complexity and Finite Models*, volume 31 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 149–184. American Mathematical Society, 1996. 1.1

[vdA98] Wil M. P. van der Aalst. The application of petri nets to workflow management. *J. of Circuits, Systems and Computer*, 8(1), 1998. 2.2.1, 8.1.3

[vdA03] Wil M. P. van der Aalst. Inheritance of interorganizational workflows: How to agree to disagree without loosing control? *Inf. Tech. and Management*, 4(4):345–389, 2003. A.3

[vdABC$^+$07] Wil M. P. van der Aalst, Boualem Benatallah, Fabio Casati, Francisco Curbera, and Eric Verbeek. Business process management: Where business processes and web services meet. *Data Knowl. Eng.*, 61(1):1–5, 2007. C.2

[vdABry] Wil M. P. van der Aalst and Twan Basten. Inheritance of workflows: an approach to tackling problems related to change. *Theor. Comput. Sci.*, 270(1-2):125–203, uary. A.3

[vdADtH03] Wil M. P. van der Aalst, Marlon Dumas, and Arthur H. M. ter Hofstede. Web service composition languages: Old wine in new bottles?. In *EUROMICRO*, pages 298–307. IEEE Computer Society, 2003. C.2

[vdAtHKB04] Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, B. Kiepuszewski, and A. P. Barros. Workflow patterns. *Distributed and Parallel Databases*, 14(3):5–51, July 2004. 2.2.1

[vdAvH04] Wil M. P. van der Aalst and Kees van Hee. *Workflow Management: Models, Methods, and Systems*. MIT Press, March 2004. A.1.2

[Wal04] Christopher D. Walton. Model checking multi-agent web services. In *Proc. of the 2004 Spring Symposium on Semantic Web Services*, Stanford, CA, March 2004. 2.2.7, 8.1.3

[Win90] Marianne Winslett. *Updating logical databases*. Cambridge University Press, New York, NY, USA, 1990. 3.1, 4, 4.3

[WL06] Hai Wang and Zengzhi Li. A semantic matchmaking method of web services based on $\mathcal{SHOIN}^{+}(\mathbf{D})^{*}$. In *Proc. of the Asia-Pacific Services Computing Conference (APSCC 2006)*, volume 0, pages 26–33, Los Alamitos, CA, USA, 2006. IEEE Computer Society. 2.2.13, 8.1.3

[WS] Web services @ W3C. `http://www.w3.org/2002/ws/`. 1, 2.1

[WSH$^+$03] Dan Wu, Evren Sirin, James Hendler, Dana Nau, and Bijan Parsia. Automatic Web Services Composition Using SHOP2. In *Proc. of ICAPS 2003 Workshop on Planning for Web Services*, Trento, Italy, June 2003. 2.2.11, A.3

[WW97] Dirk Wodtke and Gerhard Weikum. A formal foundation for distributed workflow execution based on state charts. In *Proc. of 6th International Conference on Database Theory (ICDT 97)*, 1997. 2.2.2

[WWWD96] Dirk Wodtke, Jeanine Weissenfels, Gerhard Weikum, and Angelika Kotz Dittrich. The Mentor project: Steps toward enterprise-wide workflow management. In *Proc. of the 12th International Conference on Data Engineering (ICDE '96)*, pages 556–565, Washington, DC, USA, 1996. IEEE Computer Society. 2.2.2

[xCB] XML Common Business Library Version 4.0 (xCBL). `http://www.xcbl.org/`. A.1.2

[XS03] Jie Xing and Munindar P. Singh. Engineering commitment-based multiagent systems: A temporal logic approach. In *Proc. of the 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '03)*, pages 1–8, Melbourne, Australia, July 2003. ACM. 2.2.7

[Yan03] Jian Yang. Web service componentization. *Communications of the ACM*, 46(10):35–40, 2003. 2.2.3

[YP00] Jian Yang and Mike P. Papazoglou. Interoperation support for electronic business. *Communications of the ACM*, 43(6):39–47, 2000. A

[YPvdH02] Jian Yang, Mike P. Papazoglou, and Willem-Jan van den Heuvel. Tackling the challenges of service composition in e-marketplaces. In *Proc. of the 12th International Workshop on Research Issues in Data Engineering: Engineering E-Commerce/E-Business Systems (RIDE'02)*, page 125. IEEE Computer Society, 2002. 2.2.3

[ZBD$^+$03] Liangzhao Zeng, Boualem Benatallah, Marlon Dumas, Jayant Kalagnanam, and Quan Z. Sheng. Quality driven web services composition. In *Proc. of the WWW 2003 Conference*, Budapest, Hungary, May 2003. 2.2.2

[Zim88] H. Zimmermann. *Innovations in Internetworking*, chapter OSI reference model: the ISO model of architecture for open systems interconnection, pages 2–9. Artech House, Inc., Norwood, MA, USA, 1988. A.1.2

APPENDIX A

Service-oriented applications

In the following we briefly analyze the structure of a service-oriented application and the related development and maintenance process, in order to provide a general introduction to the subject and to point out most relevant formalization aspects.

In the service-oriented application development framework three distinct roles (or actor types) has been identified:

service providers that implement, publish and provide a service to other members of the community;

service directories that manage *directory* of available services provided by some members in the community;

service requestors that consult directories, discover relevant services and access them to achieved their own goals.

The same agent in the community can play different roles at the same time: it could provide a service, while it consume other ones published by another agents.

In such a context, the development of a service can be achieved in various ways:

- by encapsulation and publishing of a pre-existing functionalities available in a software system (i.e., an EIS);
- by composition and integration of already available services in order to provide a more complex and *value-added service.*

Generally speaking, moreover, the service composition tools, employed during various phases of application life-cycle (e.g., analysis, design, implementation) and at different levels of abstraction, play a critical role and are a peculiarity of this computation paradigm. For this reason, in the rest, we pay a special attention to these respects, since many of its characteristics are interesting in a general discussion about e-services and their applications.

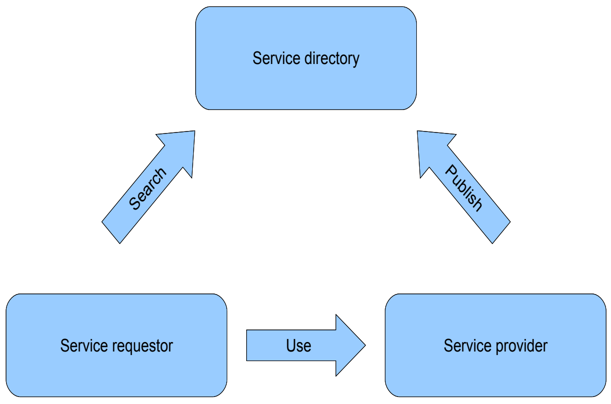

Figure A.1: The reference model for a service-oriented system

We remark that, in terms of economic feasibility, an actor that provides services by composition of other ones, in order to motivate its own offer, must be able to provide additional value to potential customer/client ([YP00]). Moreover, the composition strategy itself is the main value, since it save the clients the trouble of locating, discovering and integrating atomic available services[1].

A.1 Service modeling

In the following we briefly analyze the approach to the e-service modeling problem, in terms of paradigms (how it has to be modeled) and arguments of the representation (what has to be modeled).

A.1.1 Modeling paradigms

There are different approaches on how an e-service can be formally represented ([Bat03]): the selection of the right one depends upon many aspects, in particular the goal of the modeling activity and its context.

black-box This approaches requires that only accessible service interfaces have to be specified and that their properties are stated without any dependencies with the actual implementation. It is, of course, the most powerful approach to service modeling, because it ensures the highest degree of encapsulation but, at the same time, it is also the most complex to correctly

[1]We generally employ the term of *meta-service* to denote a service that act on other services. E.g., a directory of services exposes itself as a service that allows the client to locate and matching a suitable service. Also composition tools can be treated as a category of meta-services.

adopt and employ, since it is not always possible (or convenient) ignoring any implementation details, as it is difficult to capture and define exactly the service semantics in a completely abstract way. More specifically, the ability to deal with (nearly) black-box specification is a high desirable feature of any e-service framework.

white-box This approach is the opposite of the latter, since it is requires that any relevant implementation detail has to be exposed: in other terms, the service is specified by means of its own implementation. Despite the problem that such an approach does not ensure a significant form of encapsulation or information hiding, compromising the autonomy of the service provider, generally many implementation details are not very relevant, but since they cannot be filtered out, they overweight the specification leading to intractable problem instance (the degree of abstraction is rather low).

grey-box This other approach mixes the previous one, using a white-box to expose the high level specification elements and a black-box model to describe simple properties. The classical approach of modeling a complex service as a composition specification of other services is a typical case of grey-box: the composed services are the black-box part, while the orchestration specification is the white-box part.

glass-box This approach is, instead, based on the idea to specify a service in terms of a process that is equivalent w.r.t. the observed behavior, despite the implementation can be different. This process can be assimilated to a white-box representation that actually omits irrelevant implementation details w.r.t. the correct service behavioral semantics specification ([HB03]).

A.1.2 Modeling aspects

The description of an e-service can involve different aspects that turn out to be relevant in the design and development (and also in the delivery and management) of an application based on the SOC paradigm.

In the development of web services and cooperative information systems (e.g., EAI, BPI, e-government) many standards have been proposed and adopted, leading also to a babelization that is the main obstacle to an effective widespread of this kind of solutions. Moreover, some proposals cover different aspects, hence they are not easily directly comparable.

Interface The service interface is, essentially, a specification of the set of operations exposed by the service and the corresponding accessing directive (as a generic software components), or, in other terms, the specification of the message language that is possible to employ to build a conversation among the service and various involved actors (as a software agent). The more relevant (standard) languages employed to define a service interface are: WSDL ([CGM+04]), WSCI ([AAF+02]), XML Schema ([TBMM01]), CORBA IDL ([OMG04]), DCOM IDL ([DCO]). This aspect also covers the problem of the representation and the structure of exchanged information during an enactment, and the processing strategies. In the case of XML web service, given original aims of access protocol design (i.e., SOAP [GHM+03]) as state-less protocol, it is also interesting the problem of modeling the link (i.e, *correlation*) among different messages exchanged in a specific conversation or enactment instance.

Access This aspect essentially covers how the communication path from and to a service has been implemented. As other networking scenarios, protocols build actually a stack and can be distinguished on different levels:

Message transport protocol Network protocols that can be employed to deliver a message from an actor to another one: e.g., HTTP, SMTP, RPC, RMI/IIOP, MQSeries, MSMQ, DCOM.

Message format protocol Languages that specifies how messages and information items must be formatted and represented: e.g., SOAP, RMI/IIOP, XML, XML Schema.

Content format protocol Languages and set of constraints (e.g., of a specific application domain) that define how the information payload must be described and instantiated, often specifying at least an informal shared semantics: e.g, RosettaNet ([Ros]), EDIFACT ([EDI]), ebXML ([ebX03]).

We remark that, generally, there is no distinction between the message transport and format levels (i.e., DCOM, CORBA, RMI). This distinction, that has been introduced in the ISO/OSI stack ([Zim88]) uncoupling the application data representation from the network transportation mechanism, has been re-proposed in this field by XML web service through the concept of *mediated protocol*[2]. Other standards, such as RosettaNet, impose some restrictions on message formats and transport protocols that are allowed (and eventually specifying how to employ them in a coherent manner).

Composition This aspect is related to the specification of the integration and cooperation modalities of services aiming at exposing a kind of value-added functionality. Reference models elaborated to address this specification aspect can be classified into two main classes:

work-flow based composition, that models the service composition as a complex process orchestrated by a central coordinator or broker, that is, generally, the actor providing the composed service. This model class essentially derives from application of Work-flow Management Systems (WfMS) concepts and languages ([vdAvH04]) and includes standards as XLANG ([Tha01]) and BPEL ([ACD+03]), that are actually the more influencing in the industry and research community;

choreography based composition, that aims at modeling the composition as a complex conversation among various actors (expressed in terms of exchanged messages) that does not necessarily requires a specialized element delegated to act as a central coordinator. In

[2]Interesting, XML web services can be assumed as an implementation of the *presentation layer* of the ISO/OSI stack, since they provide a platform-independent representation of the exchanged data.

this case a *peer-to-peer* architecture is more suitable (e.g., WSCDL [KBR04]).

As previously discussed, in the modeling of a service composition, we need not only a formal tool to represent *tout-court* the structure of the composed service, but it is important also the ability to deal with the following characteristics:

Target The goal of the composed service.

Constraints The conditions, possibly deriving from some general/domain-level rules, that the composition must satisfy in order to be admissible.

Quality criteria The rules that allow to rank admissible compositions w.r.t. the level of quality of the provided service and eventually select an optimal one.

Architecture The architectural model of the system that provide the composed service.

Partner management criteria The specification of the nature of partner links (static/dynamic, short/long term) and the modalities employed in discovering, selecting and contracting the partner, both at technical and business level.

It is worth noticing that many of these characteristics can be also employed to better define also atomic e-service: in fact, such a kind of service can be also treated as a special case involving a single provider.

Offer This aspect concerns the representation of aims that a service purposes (according to the user perspective), in terms of resources, quality level, access rules and constraints. This element of service modeling is generally addressed by directory registration languages, e.g., DAML-S ([BHL+02]), OWL ([SWM04]), OWL-S ([MPM+04]), UDDI ([BCvR03]), WS-Policy ([BBC+04]), since it is the core of the service publishing task. Moreover, the definition of offered service, should also include business elements as fees, payment methods, SLAs, etc. In order to deal with high level specification issues, we need to employ a suitable semantic model, once a more expressive languages as DAML-S or OWL-S has been introduced[3].

Demand This is the dual aspect of the latter, mainly concerning the representation of user commitments, goals and constraints. The representation of service demand can generally be treated as a kind of complex query on the service directory aims at discovering one or more service suitable for the user needs. While there are different tools for the management of service offer, there only some initial proposals of approach to demand management as the WS-Inspection ([BBM+01]) standard that allows to browse (and querying in a limited way) an UDDI catalog. Also the UDDI specification provides a specialized API (exposed as XML web service, of

[3]The main problem dealing with service semantics is not the lack of adequate expressive power, but the agreement on a semantics model. Not surprisingly, as will turn out in the rest, many different, and often incompatible, approaches exist, and none is clearly better than others, since each one exhibits some interesting properties to address problems arising in specific scenarios.

course) to manage and browse the contents of a catalog. As in the latter case, DAML-S and OWL-S can be employed to deal with this aspect. It is worth of noticing, that the representation of both demand and offer in a service marketplace must agree on a foundational semantic model.

Context It is related to the specification of characteristics of different application domains (i.e., vertical market application) or, in other words, to the specification of standard/agreements to employ in a specific context expressed in terms of:

1. information representation (e.g., message structure and format);
2. domain objects and available operations;
3. conversation patterns.

Many B2B standardization proposals can be employed to deal with these modeling aspects as, e.g., xCBL ([xCB]), BizTalk ([Biz]), E-Speak ([Kar03]), cXML ([cXM]), ebXML, Commerce XML [cXM], RosettaNet, EDIFACT, as many e-government integration frameworks as well[4]. Moreover, the specification level is generally strongly not homogeneous, since these frameworks often introduce constraints on different stack levels and on some implementation elements (e.g., limiting the transport level options or imposing a specific document physical format). However, a wider adoption of these models should help the integration of different operators and the development of composed value-added services, since the standardization process leads to uniform the information representation model and to share at least a primitive operation set. The main problem in the adoption of such a kind of framework relies in the high complexity of specification and related implementation that are not economically feasible for small or medium organizations, while the rigidity imposed by existing technologies often conflict with the need of dealing with extemporaneous and accidental business cases.

A.2 e-service development and delivery

The development of a service-oriented application, that is based on the framework depicted in previous sections or simply provides some e-services to a cooperative community, requires some specific tasks during both the design/implementation and the delivery life-cycle phases.

Indeed, given the quite dynamic nature of such a kind of software system, the border between typical design and run-time tasks and features can not be sharply outlined, but it can differ in various scenarios. Moreover, actually available middleware solutions, even based on service-oriented paradigm as XML web-services, adopt an almost static approach, using these stacks only as enhanced encapsulation and remote access tools: in other words, most of applications use XML web services essentially as an Internet-ready RPC mechanism.

[4]Considering the Italian scenario we have a main standardization initiative concerning the cooperative environment and middleware [CNI05b, CNI05a, CNI05c], and several projects focused on the modeling and standardization of service functional aspects, mainly oriented w.r.t. the application domain ([G4B04, Peo05]).

Typical features required in the development and delivery of service-oriented applications are classified as the followings:

Discovering and Matchmaking It is the ability to locate an available service having specified features/characteristics or, in other terms, to evaluate the adequacy of a given service w.r.t. a set of requirements/constraints. This element is rather relevant both in the design/implementation phase of an application. In fact during the design phase, analogously to component catalog/directory ([AG97, PA99, Fis01]), we need to provide some tools to help the service lookup activity (allowing for a suitable query language): such an element is generally defined in an incomplete/astract way stemming from application requirements. Moreover, in the delivery phase the ability to dynamically select a service instance to perform a specific task (or achieve a given goal) is fundamental[5] in order:

1. to ensure a higher fault tolerance degree of the integrated system: in case of unavailability of a service instance during an enactment, it is possible to employ a meta-service that locates a suitable backup provider ([CIJ+00]), reducing the dependability;
2. to allow for the adaptation of the service delivery (*context awareness*) to actual execution environment: in case of mobile user, it is possible for a *location-based service* to select the nearest provider or the provider that better fits the user preferences;
3. to select among many service providers that are "equivalent" in terms of exposed functionalities the one that ensures the better quality level according to given criteria/constraints (e.g., quality/price ratio);
4. to specify composed/integrated service abstracting from service instances actually available (e.g., using class/template).

Synthesis It is the ability to obtain in an assisted or completely automatic manner the specification of a complex/composed service to achieve a given goal, using a community of pre-existing services. This kind of task is similar to the *automatic program synthesis* ([MW71]) and the *action plan generation* ([KS03]) problems, considering some additional constraints such as:

1. available information are incomplete, in most cases the information needed in order to complete the enactment is available only at run-time;
2. managed information are often described using very complex data structures;
3. the typical encapsulation level provided (aimed) by service-oriented approaches leads to hide the system state data that are generally required during the planning step;
4. the kind of synthesis goal can be more complex than the simple system configuration reaching;

[5]The inability to specify in a flexible, that is to say dynamic, manner the link to service providers in composition specification languages as BPEL and its evolutions is in the facts one of main limitations of actually available integration technologies ([MM03]).

5. there is a major emphasis on the reuse of software elements that implement fragment of the business logic.

Orchestration This feature is intended as the ability to monitor and control the execution of complex/composed service enactments. Despite this kind of task is under some perspectives assimilable to traditional work-flow and distributed transaction management, in case of composed service orchestration there exist some specific problems ([Kal03]) that are generally ascribable to the high degree of independence of involved parts:

1. the impossibility to provide a completely detailed specification of the whole system behavior (including the private part of various agents), that exposes anyway some non-deterministic properties and it is also impossible to prevent the raising of exceptions or fault conditions, as in any highly distributed environment;
2. the unsuitability of traditional transactional models ([Lit03]), both for technical and business causes, and the necessity to employ weaker definitions of concepts as atomicity, isolation, consistency, and other transaction-related properties;
3. the high criticality level due to the (indirect) management of private resources by external and autonomous parts;
4. the possibly of employ incomplete specifications (abstract or partially specified) that are adapted to the execution context at run-time;
5. the adoption of highly distributed architectural model that does not require a central coordination node (e.g., a peer-to-peer architecture).

Analysis and Verification It is the ability to analyze a service specification expressed at a specific detail degree and verify if it is consistent with a given set of constraints or whether some properties hold. This is a foundational aspects that is relevant in many contexts and it is very useful to cope with different service-related issues:

1. the high difficulty level of the testing and debugging of service-oriented applications that involve external partners, e.g., service providers included in a composition, that cannot be available at time of implementation;
2. the mismatching between business and technical requirements and specification and, generally, their ambiguity and incompleteness;
3. the potential inconsistency among (private) processes that implements services provided to the community since they can generally be realized by different actors in different periods (e.g., a legacy mainframe terminal transaction can be exposed using e-service as well as a new reasoning-enhanced network-accessible facility);
4. the need to ensure the consistency w.r.t. externally defined constraints, e.g., laws and regulations, that impose additional restrictions, independently from application constraints;
5. the requirement of negotiate the service contract both with providers and clients and the need to ensure the respect of contract clauses;

6. the requirement of stating the compatibility level of behavior exposed by involved actors (at least, service client and provider);
7. the necessity of analyze and simulate the behavior of a composed service in order to obtain elements to employ during the system capacity planning and the tuning of computing infrastructure.

A.3 e-service equivalence

Among various issues related to service-oriented application development it is of general interest the problem of defining and analyzing the *equivalence*, and different kinds of compatibility too, between e-services.

In fact, the service equivalence is a notion that turns out to be rather relevant in many scenarios: to address both the synthesis and the orchestration problems, since it enables to deal with incomplete service specifications that will be completely instantiated only during the enactment execution, to implement a dynamic service provider lookup tool (e.g., fault-tolerance, context-awareness), to minimize the complexity of a service (i.e., to compute the minimal equivalent service specification), etc.. It can be articulated w.r.t. different aspects:

Specification Considering the specification, the service equivalence can be assessed according different and heterogeneous criteria ([BCD+03]), in fact, generally speaking, we can compare services w.r.t.: service schemas, service implementations (i.e., the abstract components that implement the exposed behavior), services instances (i.e., the concrete software components, like XML web service, that are actually available).

Interface/Protocol Considering the component public interface and the message exchanging protocol with other engaged actors (i.e., client, provider, partner), that means that service are analyzed w.r.t. the signature of published operations or, instead, the structure of exchanged message and the language induced. Basically ([MB03]), services are considered compatible if they are compatible in terms of operation signature, message structure and conversation trace.

Function This aspect is related to an *intensional semantics*, that means that two services can be assessed as compatible if they perform "similar" tasks in different contexts, even exposing different cooperation interfaces, or in other words, we are asking if two services performs that same business function ignoring any context aspects. This kind of compatibility definition is quite analogous to the compatibility among software components/libraries that, despite they are accessible using different protocols, can be interchanged using a suitable adaptation layer[6].

Context This aspect is, instead, related to an *extensional semantics*, that means that two services can be treated as equivalent according this criterion if, even abstracting from the effective activation interface/protocol, they can be employed to obtain a common family of concrete effects. In

[6]As we will show in the followings, the service composition problem can be treated as a generalization of the interface adapter problem, i.e., given two service interfaces verify if a suitable bridge exists.

other words, two extensional compatible services can be replaced into preserving the extension of possible enactment set. Such a property does not necessarily hold for intensional equivalence, since we can define services that perform similar tasks in different contexts and hence have different enactment family extensions[7]. This is a typical aspect of service-oriented applications, since while software components are generally considered only as technical building block, an e-service is generally a software component related the management to some non-computational resources, and for this reason it is necessary to cope with the denotation of the applicability scope of the service itself that is generally related to its organizational deployment[8]. We remark also, that despite some initial attempts (i.e., [FLP+03]), this problem has not been adequately addressed, and available approaches are too specific to some applications or they are lacking a suitable semantic foundation. It is also worth noticing that extensional semantic characterization of a composed service derives from the extensional semantic of involved services, and that while the composition can be feasible w.r.t. the intensional semantics it cannot be also feasible once the extensional contexts have been specified.

QoS/SLA Two services are considered as compatible if they ensure the same quality levels assessed using a given criterion (e.g., the quality/price ratio).

A natural generalization of the service equivalence concept is the *service containment*: a service "contains" another service it is a generalization according to a given analysis dimension. In this case, from the order relation induced by the containment is possible to derive equivalence classes using the antisymmetric property.

Regarding this topic, the most relevant approaches are the ones based upon the definition of service ontologies using languages as DAML-S and OWL-S and their variations ([BHL+02, PC03, Gro03]): such an approaches use different strategies to characterize the equivalence, but generally they assume equivalent services that can implement the same conversations (roughly speaking, the effects of a service are the set of exchanged messages) or that have similar process implementations ([MPC01, HB03]).

The syntactic approach relies on the definition of an adequate type system to model service operation signature parameters, organized using complex data structures and hierarchical domains. This method is limited by the fact that in many situations the type system is only definable in highly abstract terms, since it can be instantiated only at run-time. In fact, many planning-based approaches ([MS02, WSH+03, GTL00, APY+02]) introduce explicitly information-gathering activities to dynamically get information about currently employed

[7]In terms of knowledge representation these aspects are mainly related to the extensional part of the domain specification. Please notice that in high expressing knowledge representation languages, there is no strong separation of schema and data elements as in traditional data representation approaches.

[8]For example, considering the technical perspective, in case of XML web services, an e-service described using the WDSL generally specified also it deployment (i.e., end-point URL). In other words, while other component-oriented architectures (e.g., DCOM, CORBA, Java Enterprise Edition [JEE], .NET [NET]) are using the *interface-description language* to denote a class of components, the corresponding construct in case of XML web service generally denotes a deployed instance that probably manages its own context. Moreover WDSL allows for a kind of abstraction by the mean of the `port-type` construct.

service instances (i.e., reflection).

A quite similar solution, since it relies mainly on syntactic features, even having different purposes, is presented in [Mar03]: the notion of *service usability* is defined in order to denoted the property that a service can be included in the definition of a process ensuring the correctness of the latter. Both the services and the process are modeled as work-flow fragments.

In [PEZS02] the equivalence analysis problem is reduced to the verification of compatibility of underlying business processes implemented by the organization units aiming at implementing the cooperation: in fact, also in the case of standard conversation protocols (e.g., EDIFACT, ebXML, RosettaNet) defining in unambiguous manner syntax and semantics of exchanged messages, actually implemented business processes can result to be incompatible requiring, hence, a preliminary re-engineering (i.e., interconnected internal processes can easily result into a non-safe global process). In order to encapsulate the internal business process, hiding private implementation details, in [vdA03] is proposed an approach relying on the notions *branching bisimulation* and *work-flow inheritance* ([vdABry]): roughly speaking two process models are assumed as to be "compatible" if they are indistinguishable by an external agent employing a given message exchange protocol.

It is worth noticing, that also the matchmaking problem can be reduced to the service equivalence analysis if the requirement is expressed as a service example to which available service description should be compared (i.e., *query-by-example* or *case-based reasoning*).

A.4 e-service composition

The *service composition* is another very interesting and articulated aspect that is addressed in the field of service-oriented applications.

The composition or, in other words, the ability of integrating functionalities available by means of more services into another one that provided a more sophisticated behavior to the user, can be analyzed w.r.t. different attributes/properties:

Proactive/Reactive Attribute introduced in [CJ01], a proactive composition is built in an off-line manner by a composed-service provided (*service aggregator*), while a reactive composition is built at runtime given a specific user request.

Mandatory/Optional engagement Also this property of service composition has been introduced in [CJ01], a service can need the participation of an involved composing service as a necessary condition for a correct execution. In the case of optional engagement, the service allows for the unavailability or the withdraw of a partner without necessarily compromising the successful outcome of the running execution, despite a performance degradation can arise.

Abstract/Concrete specification A service composition can be specified at different levels of details and abstraction: a concrete specification requires the exact denotation of execution steps and involved actors (i.e., service providers) at the design phase. On the other hand, an abstract specification can leave some aspects not completely defined, requiring at runtime some additional tasks, in order to locate and bind a service provider or to instantiate some execution steps or their (optimal) activation sequence. The ability to cope with an abstract composition is required since often at the service design phase, there is no way to obtain all the required information to exactly specify the execution plan[9]. This problem is also deeply analyzed in the field of Management Sciences and Business Process Analysis and Reengineering as, e.g., in the methodology MERISE ([TRC86]), where three different levels of specification are distinguished: conceptual, logical, and implementation (physical/organizational).

Target complexity A composed service can be treated as an execution plan designed in order to achieve a specific goal, expressed in terms of system state, employing the involved composing services, or as the specification of an orchestration strategy of these services that is consistent with their constraints and allows the user to interact with a new service that combines abilities of composing ones[10]. In other words, a composed service with a simple target can be seen as a component that exposes at most a single atomic operation, while a service with a complex target is characterized by a more articulated operation set (i.e., interface) defining many operations available to the client, but requiring also a more sophisticated language to correctly specifying the execution constraints (e.g., a finite state automaton as proposed in [BCD$^+$03]).

Autonomy level A service can allow for different levels of autonomy of involved partner in the execution of respective portion of an enactment without compromising the availability of the service itself. This characteristic is highly related to the notion of encapsulation and generally, since the most level of autonomy should be ensured to actors, in order to reduce the dependability among them, suitable approaches to fault-tolerance and recovery must be implemented.

Centralized/Distributed coordination In a service composition we can distinguish between a flow of control messages exchanged among actors (*control flow*) and a flow of information-carrying messages (*data flow*), despite this distinction can be often not explicitly stated. On this schema, in [LLW02] 4 different patterns has been defined and analyzed: the key aspect is the existence of a central agent acting as broker/mediator of some flows. In [HBCS03], moreover, a simplified classification in terms of *hub-and-spoke* and *peer-to-peer* architectures has been also discussed.

Composition complexity The degree of composition complexity, in terms of formal language constructs available during the design of composed services, has been also introduced as classification attribute in [HS04].

[9]A composition is more abstract than another if it has a higher degree of parametricity w.r.t. the information that is available only during the service enactment resulting from a user request. In other words, some decisions are delegated to the service executor from the designer.

[10]The degree of parametricity is expressed w.r.t. the set of options that are offered to service client, e.g., in a composed service with a simple target, the requestor can only decide to engage the service or not, while in a complex target service, it can control the enactment evolution.

Specification complexity This aspect, also introduced in [HS04], refers to the expressiveness of specification language available to defining characteristics and abilities of composing services.

We point out that these dimensions of classification are not completely mutual orthogonal: e.g., a reactive service synthesis is generally associated to a simple goal (i.e., a specific request about the system state has triggered the enactment).

A.5 Reference scenarios

In the development of service-oriented applications it is interesting to classify some reference scenarios that drive the research activity to elicit constraints and requirements. A similar analysis is also available in [HHO04].

A summary of the analysis of main characteristics of proposed scenarios is reported in Table A.1. We now briefly describe and discuss these cases.

	WSC	**SOE**	**WSB**
Business standard	Open	Proprietary	Proprietary
Technological standard	Open	Proprietary	Open
Environment	Public	Private	Mixed
Agreement	Extempore	Long term	Medium/long term
Encapsulation	High	Medium/low	Medium/high
Wrapping	On-availability	On-demand	
Negotiation	Run time	Never	Design time
Management autonomy	High	Low	Medium
Required features	Search and matchmaking, simple goal composition	Validation, complex goal composition	Dynamic binding, validation, complex goal composition

Table A.1: A comparison of various application scenarios

A.5.1 Web Service Customization (WSC)

Considering the e-commerce application context, the end-user is aiming to perform a specific task on the Internet using a new-generation web consisting both of pages and services.

The user has defined a concrete goal, even it is only partially specified (e.g., (s)he will to organized a trip, book a hotel in a given period, but (s)he does not explicitly state that (s)he will to pay for it), hence some necessary operations can be induced from domain constraints. Generally, the user goal is formalized in terms of desired system final state property with some quality criteria (e.g., minimal cost solution) and the solution can be achieved selecting among a community of "equivalent" services that are extensionally suitable for the user goal. In this scenario there is no explicit "procedural" specification available, despite some relevant implementation patterns can be available.

According to this description, the foundational abilities required in this scenario are: service directory search and selection (and generally catalog management), matching, consistency checking and synthesis (even w.r.t. simple goals). The agreement among parts is extempore, while the service quality level negotiation is quite relevant (e.g., price selection). The conversation is implemented by open standard protocol (i.e., the XML web service stack) and every actor has a nearly complete autonomy.

An example of business-to-business version of this scenario, can be the on-the-fly research of a shipping agent given the shipping data, the selection w.r.t. ensured service levels, the condition negotiation the agreement definition, the activation and the monitoring of the enactment.

A.5.2 Service-Oriented Engineering (SOE)

In this scenario, the end-user is the designer and developer of software applications in a SOA that aims at synthesizing a new service given available ones generally in a private/restricted-access network environment.

In this case, the user goal can be expressed in a very detailed vary, but it is essentially an abstract specification, since the required result is another software component that exhibits a complex behavior once it is deployed. The execution goal becomes concrete only at run-time, given a specific enactment, and hence such a component must be also able to correctly instantiate abstract fragments using available information. In a delimited and controller environment (i.e., in a EIS belonging to a single organization) accessed service can be treated as "transparent", that means that also implementation details are available and that the matching can be performed also w.r.t. the private part.

Differently from a public network (i.e., the Internet), in a single EIS the availability of a service can problematic since the wrapping activity is very cost intensive and services are low generalizable, hence a legacy transaction is exposed has a service only when it is required from at least another application (on-demand). As consequence, catalog management, search and matchmaking functions are rather important, despite they require a lower automation degree.

Also the consistency verification is an important but not critical feature, while the availability of automatic synthesis and orchestration tools is essential. The agreement is generally long-term and predefined with a limited requirement of condition negotiation, moreover employed integration protocols can be proprietary and there is a low autonomy degree, while the computing environment is assumed as reliable and fault-tolerant.

This scenario is fairly similar to classical integration of EISs, where service-related technologies aim at providing a higher degree of abstraction and flexibility, in order to reduce long-term maintenance costs.

A.5.3 Web Service Brokering (WSB)

Also in this scenario the end-user is designer/developer, but its task is aimed at integrating services provided by other organizations (external services) in order to implement a business partner integration or service marketplace (i.e., an environment where actors can trade their services and requests).

It is clearly a special version of previous scenario, characterized by the following properties:

1. given the privacy concern of involved partner, there no details about service implementations;

2. similar services with different extensional coverage, as in the former case, can be available.

The task goal is expressed in terms of complex service to synthesize, while as in the case of customization, we need to take into account several domain constraints to enforce. The designed composed services can eventually include also lookup operations to select and bind provider instances during the run-time (i.e., routing messages) while the SLA negotiation is done at design-time. The partner agreement is defined at medium/long-term, while the degree of autonomy is limited, since a minimal reliability degree of the integrated system must be assured. In case of different service providing organization units having a higher degree of autonomy, the SOE scenario can be reduced to this one.

A special case of service brokering is represented by the integration of e-government services. In fact, in this kind of applications, it is typical the case of many intensional equivalent services with different extensional applicability scope, since many public administrations provide service to citizen on territorial basis (e.g., city authorities, counties, administrative departments).

APPENDIX B

Other approaches to Service-Oriented Computing

In this appendix we discuss about other interesting approaches to formally cope with issues arising in the design of a service-oriented system that are not directly related to the subject of this Thesis.

B.1 Data Integration

The data integration framework ([Len02]) is a very general formal tool that can be successfully employed in service-oriented application. In fact, *information gathering* services (i.e., services that merely retrieve a specific information) can be easily considered as a typical data source in a data integration schema. They can be mapped to a reference data model and a user query can be mediated among them, implementing the service integration in a very natural way.

However, service accessible data sources are generally not queriable in an arbitrary way as a traditional DBMS: they have some *access limitations* and, hence, some special techniques ([RSU95, FLMS99]) should be employed to deal with them. In fact, e-services are generally employed to hide details about implementation of information systems and also about the design of private databases, thus it is very difficult (it is more assimilable to a business issue than to a technical one) to obtain a free-query access to data. So, an informative e-service is only able to return a result only providing a suitable input assignment executing a sort of fixed parametric query.

In [PF02, TKAS02] two approaches based on data integration techniques have been proposed to address the composition of information gathering services: in the former a *local-as-view* framework using the MiniCon algorithm ([PL00]) is employed, while in the latter a *global-as-view* model is extended with a forward-chaining mechanism to cope with source access limitations. Also the usage of a *discrimination matrix* to select only service providers that are potentially relevant in terms of their applicability extension has been proposed in [TKAS02].

Most notably, in spite of the fact that, as formal tools, data integration frameworks can not satisfactory address issues related to dynamic features in the design of service-oriented applications, we point out that, since, a preliminary step in the construction of a service-enriched web is the information sharing, they will eventually play a relevant role.

B.2 Linear Logics

The class of *linear logics* ([Gir87]) is a family of propositional logical languages particularly suited for addressing problems related to planning and program synthesis, since it is able to explicitly model (limited) resource usage.

In fact, several applications (e.g., [Kün02, KV01, GHS96]) of these languages have been proposed in order to deal with domain-independent action plan synthesis problems. As foundational tools, they can be generally, among other applications, employed in many formal approaches to e-services based on planning techniques.

In [RS03] an approach to service composition is devised: it is essentially based on a planning framework implemented using a theorem prover in linear logic. More specifically, given a set of available services annotated using DAML-S profiles, a DAML-S service process that implements a target specification (also expressed as a DAML-S service profile) must be synthesized using them. The problem is addressed translating the available service specifications in axioms of a linear logic theory and the target service in a formula. The derivation of the formula from the theory axioms can be translated back into the specification of the required service process, in fact, differing from other inference systems, in the case of linear logic, the structure of the derivation is fundamental: theorem proofs are first-class object of this language. The inference procedure is based on the check of type compatibility w.r.t. preconditions and postconditions expressed in service profiles, moreover, differing from other similar approach, in this case it is possible to deal also with non-functional requirements (i.e., service quality levels) and imposing constraints on admissible processes. In particular, the devised approach is also able, given some ad-hoc inference rules, to deal with functional attribute data domains. In fact, despite the model expressiveness is restricted since propositional terms are only bound to data types (values are ignored), it is also possible to model the application schema using a simplified ontology expressing equivalence/containment relations at the extensional level[1]

A further generalization to distributed problem solution is presented in [KM06b]: the reasoning framework is employed as a tool to implement a negotiation protocol among cooperating agents. The application is not limited to e-service analysis (in particular to e-service composition), but it is also relevant in such a context.

The more interesting aspect of such a kind of logics is that the proof itself of a theory (rather than the interpretation structure, as in many other cases) can be interpreted in a dynamic fashion.

On the other side, as it has been shown in [EW90] and [EW93], there exists also a link with linear logics and explicitly dynamic formal structures as Petri nets. It that can be also exploited to build linear logics theorem prover using

[1]The model is a generalization of the one discussed in [FLP+03].

tools devised in order to analyze these kind of structures.

APPENDIX C

Technologies for Service-Oriented Computing

In the following we present some observations about the some technologies, currently employed in industrial solutions, that are related to issues addressed in this work.

Given the large number of different platforms, architectures, communication protocols, an exhaustive analysis is far beyond the scope of the present work. Moreover, since we are mainly interested in formalization of properties to support the design and specification of this kind of applications w.r.t. dynamic concerns, we are ignoring some very relevant topics as data and transaction and management, security, fault tolerance, etc..

In other words, we are aiming to analyze the relation with some technologies currently adopted in order to show how (and why) they are not completely satisfactory in the implementation of a service-oriented application.

C.1 Component-based Software Development

The service-oriented computing paradigm in the field of Enterprise Application Systems development, management and integration is a natural evolution of distributed component architectures (e.g., CORBA, Java EE). In particular, while on a side several vendor-independent and possibly standard protocols (i.e., XML web service stack) have been introduced to reduce middleware integration costs, on the other hand, business process oriented design approaches have been devised levering on mega-programming technologies and orchestration/messaging tools.

According to the component-oriented perspective, as a generalization of object-oriented approaches, while software components are a way to implement e-service specializations, the latter can be considered as a tool to encapsulate the former. Several proposed e-service matchmaking approaches, according to [Pee03a], can be, in fact, reduced to similar proposal for component directory and discovery

But, from a wider perspective, the service-oriented paradigm is a strong generalization and it cannot simply be assessed as a new middleware protocol stack. In particular, we remark that:

- an e-service, differently from a software component (e.g., a software library, a reusable module), can have a life-cycle completely independent from the information system in which it is employed (i.e., extempore agreement). In fact, the e-service can be managed by a totally autonomous organization entity, that is in charge not only of the implementation of the module (i.e., as a reusable software piece), but also of the on-line providing. For example, in the B2B integration scenario, a business partner is responsible for the availability of shared services: in other words there is a dependability relation among autonomous information systems.
- an e-service is inherently characterized by the extension of its own application field (i.e., the served client/user community): in other words, it cannot be merely denoted in intensional terms (i.e., provided functions), but it need to specify also to which objects these capabilities can be applied.
- in the case of a service-oriented architecture there is a stronger emphasis on the dynamic nature of integration binding, since it can be completely defined only during a specific enactment. Given also the high concernment of encapsulation and autonomy issues, the semantic of transactional operations must be accordingly refined introducing several alternatives w.r.t. different participant commitments (i.e., atomic transactions and business transactions).

Generally, considering component-oriented architectures and solutions, in the case of e-service applications there are more relevant issues related to the management of the conversation among different elements than those related to the reuse, in particular, in the case of very flexible and dynamic integration and incomplete specifications.

C.2 Work-flow Management Systems (WfMS)

The work-flow management paradigm has been widely adopted in many software development contexts, in particular as base tool for the flexible implementation of business process in many software platforms, but it is also the reference model in the field of semantic e-services as foundational model for complex service specification.

However, in the latter scenario, many issues can be pointed out:

- the reference architectural model adopted by a WfMS generally imposes that a specialized (and privileged) node in the cooperative network acts as orchestrator. While this assumption can be easily satisfied in scenarios where the work-flow management technology has been more successfully applied (i.e., the management of internal business process of an organizational unit, possibly involved in a macro-process), it cannot generally hold in the case of automation of multi-organizational process fragments. Moreover, in the case of open network communities a peer-to-peer paradigm is preferable.

- the very dynamic design processes and execution environments, induced by the requirements of very flexible connection and integration of service-oriented applications, is not adequately addressed by work-flow specification languages and models (e.g., the dynamic discovery and the selection of a partner, the operation rescheduling), but it requires some ad-hoc extensions.

- the highly distributed computational environment makes rather hard, generally, to unambiguously denote the owner" of the process instance that implements a service enactment. In fact, each involved actor perceives only it own projection of the process (i.e., the macro-process fragment). This is a quite critical issue in the case of e-government applications, since its legal implications (i.e., who is the liable for a response?).

- B2B-like integration scenarios, in particular cases of e-marketplaces and virtual-enterprises, introduce other issues related to the security enforcing and management (e.g., authentications, identity management), the privacy protection, the agreement negotiation, the service fees and payment, generally not addressed in the process management models.

Moreover, despite approaches derived from work-flow management ([vdADtH03, vdABC$^+$07]) can deal with many inter-organizational issues, some peculiar aspects of e-service applications, in particular the denotation of extensional context, have not been adequately addressed so far.

C.3 Intelligent Software Agents

The remarks noticed regarding the relationship between service-oriented computing and component-based software construction can be easily extended by analogy to the case of software agents. In fact, an e-service can be considered as the specification of the capabilities that a cooperating agent provides to the computing environment, as the multi-agent technology can be a source of useful design, integration and communication tools in the development of a service-oriented application.

Moreover, in this case, w.r.t. the component-oriented platforms, multi-agent technologies can deal with some typical characteristics of e-services (e.g., applicability context, non-cooperative behavior, ephemeral agreement) in a more natural way, since many of these aspects are also relevant in the modeling of (intelligent) software agents.

On the other hand, we need also to point out that several of more interesting solutions in this field are not enough mature (i.e., in terms of reliability and scalability in presence of large-databases) to be actually employed in the development of EAI or EIS solutions. Moreover, the high expressive power of modeling languages developed in this area[1], although it enable to cope with involved problem specification, leads to high complexity/intractable computational problems, despite, in many cases, there are empirical tractability outcomes (e.g., using Description Logic reasoners or AI planners). At the same

[1]The Semantic Web and related technologies essentially stem out from these one. The DAML language has been, in fact, initially devised to model agent knowledge bases adopting a frame-based paradigm.

time, several issues concerning the data modeling (i.e., in the case of exchanged information) are ignored ([Bat03]): in particular, regarding the representation heterogeneity and the specification of constraints. Actually, there is any clear evidence showing that these aspects are orthogonal/separable or, in other words, it is possible to deal with behavioral and structural concerns (almost) independently.

Index

Colophon

This document has been typeset in LaTeX 2_{ε} using the report class and several useful additional packages such as: titling, titlesec, caption, tocloft, epigraph, float, booktabs, tabularx, fancyhdr, algorithm2e, complexity, and hyperref.

Most of figures have been constructed using Xfig and OpenOffice drawing programs and the XY-pic package. The bibliography and index have been prepared using resp. BibTeX and MakeIndex tools.