Checking e-service consistency using Description Logics

Luigi Dragone
CM Sistemi S.p.A.
via Simone Martini, 126 – 00146 Roma – Italy
luigi.dragone@gruppocm.it

Riccardo Rosati
DIS, Univ. di Roma "La Sapienza"
via Salaria, 113 – 00198 Roma – Italy
rosati@dis.uniroma1.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 providing both a rich domain specification and a suitable operational semantics of the e-service contract, on which we define functional consistency properties. It allows for specifying complex e-services based on the* IOPE *paradigm, in which the static properties of the modeled system are specified using a* Description Logic *knowledge base, as assumed in* Semantic Web *applications. Moreover, it enforces a minimal-change semantics for the axiomatization of the update operator and also includes the ability to reason about update repairing w.r.t. the domain constraints, thus allowing for incomplete service specification. On this foundation, we formally devise several consistency and validity properties of services, providing decidable checking procedures.*

1 Introduction

The *service-oriented computing* paradigm [2] 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 *requestors*). Such a kind of network is the milieu on which new applications are built in terms of composition of available services (*service orchestration* and *synthesis*) and users look for services suitable to their needs (*service discovery* and *binding*). 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. On the other hand, there is a strong requirement on the well-foundedness of the service contract specification between involved actors, like in composition-oriented software development and *design-by-contract* frameworks ([14]). Generally, the adoption of a highly expressive language, as *First-Order Predicate Logic* (FOL), for modeling dynamic systems makes the verification of e-services formal properties very hard or even unsolvable. So, in order to devise a suitable expressive language for e-service functional modeling, preserving the computability of the associated reasoning problems, we turn our attention to the family of *Description Logics* (DL) [3]. Such a family of logics (which are mainly fragments of FOL) has been explictly defined with the main aim of constituting an optimal trade-off between representational abilites and computational properties of reasoning. We basically adopt expressive DL languages (i.e., $\mathcal{ALCQI}$ and $\mathcal{ALCQIO}$ [3]) 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. In particular, our proposal relies upon the decidable fragment of FOL $\mathcal{C}^2$ ([17]): function-free predicate logic with at most two variables and counting quantifiers. 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. More specifically, in this paper we present the following 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; (2) from the semantic side, the formalization of the dynamic model underlying a suitable semantics of e-services; (3) from the computational side, a first set of results on the decidability and complexity of the automatic verification of the above properties. The rest of the document is organized as follows: in Sec. 2 we introduce our reference scenario and some related works regarding the so-called *semantic web-services*. In Sec. 3 we provide the basic definitions of the various notions subse-

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE

quently employed to lay out the framework. In Sec. 4 and 5 we present a first axiomatization of a typical kind of e-services that we are able to model, and we analyze various consistency properties related to update, keeping also into account a feasible (in term of decidability) repair strategy that allows for incomplete effect specification. The full version of the paper is available in [8].

2 Scenario and Related Work

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* [2]. This is a general integration model that is applicable in many business scenarios like: *e-government*, EIS, B2B integration, etc., and that leverages on strong encapsulation to decouple (possibly autonomous) implementations. 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*, *preconditions* and *effects* (IOPE)[1]. Despite this large agreement, formal assumptions done by different authors can vary not negligibly on several modeling aspects, so the approaches are not easily directly comparable. More specifically, most of these approaches lack the formalization of some intuitive and, in our opinion, interesting notions, like, for example: consistency (are the given services consistently defined w.r.t. the domain knowledge/constraints?), functional similarity/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 equivalence 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 servicing different user communities, have potentially been specified independently, and do not expose the same interfaces?). 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. Despite many recent works deal with the problem of modeling and inventorying e-services, both from the technical perspective (e.g., [9]) and from the formal one (i.e., [18]), the approaches proposed so far are not entirely satisfactory. 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), but most of such kind of approaches (e.g.,[15]) essentially ignore dynamic features. Several planning-based approaches (e.g., [13]) share with our framework the emphasis on the operational nature, but while these ones aim at verifying if the available services are sufficient to achieve a specific goal, we are essentially interested in the characterization of the suitability of a set of available services for a class of abstract goals, not only ground ones. Some limitations of the applicability of the planning paradigm are analyzed in [6]: 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., software reuse). Such a problem is addressed in works like [5], which provides an algorithm suitable for automatic e-service synthesis. 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* ([7]). However, in such kind of works, a single complex service, that operates on a relational database, is analyzed in order to derive some correctness properties. Like in the present paper, an operational semantics based on the update of the relational theory is used, in [7] but constraints on the interaction are only based on state automata, services are generally accessible, and there is no way to enforce applicability. Other verification approaches, based upon formal tools like *Model Checking*, *Process Algebra* and Petri nets, are also proposed for e-service applications by many authors (e.g., [12]). 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. On the other hand, the adoption of highly expressive process languages (e.g., high level Petri nets [20]) allows for the definition of arbitrary behavior constraints, but requires high-order logical reasoning frameworks, in which many interesting problems are undecidable.

[1] The IOPE paradigm has been adopted by the semantic web-services community in the definition of modeling languages and related standards (i.e., OWL [18]) and has been generally assumed by most approaches in this area.

3 The Framework

In this section we introduce the definitions of the primitive constructs on which the proposed framework relies, assuming that the reader is familiar with Description Logics and FOL. 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) **A**, a finite set of binary predicate names (or role names) **P** and a finite set of constant names (or object names) **O**. We assume that object names are 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 bijec-

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE

tive function $\cdot^{\mathcal{I}} : \mathbf{O} \mapsto \mathfrak{D}$. We assume that the modeled system, or in other words, the application domain, can be described in terms of static features/constraints using an expressive knowledge base formalism, which allows both to define complex data structures and to easily include also extensional specification elements.

Definition 1 (Domain specification)**.** *A domain specification is a triple of finite mutually disjoint sets: (1) a concept alphabet (**A**); (2) a role alphabet (**P**); (3) an object alphabet (**O**).*

A system state (or world state) is described using an interpretation of the alphabet on the universe according to standard set-based semantics. Since not every interpretation can be assumed to be a legal system state specification, we introduce the ability to restrict the state space to the valid ones by means of a constraint set, expressed using a suitable language. A world state is *legal* if it is a suitable model for the given specification, while a specification is *consistent* if it admits at least a legal world state.

Definition 2 (World specification)**.** *A world specification $\mathcal{W}$ is a knowledge base $\langle \mathcal{T}, \mathcal{A} \rangle$ expressed on the alphabet $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$ using the expressive DL $\mathcal{ALCQI}$.*

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 Tab. 1 for any any concept name $A \in \mathbf{A}$, role name $P \in \mathbf{P}$ and object name $o \in \mathbf{O}$. 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 catched by the following definition: it will be extended in the following as we go along, in order to cope with various modeling refinements.

Table 1.

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

Definition 3 (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 $\hat{\omega} = \langle \mathfrak{U}, \cdot^{\hat{\omega}} \rangle$ 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: $\Delta^{\omega} = \mathsf{Top}^{\hat{\omega}}$, $N^{\omega} = N^{\hat{\omega}}$ and $o^{\omega} = o^{\hat{\omega}}$, 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. 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. Now, we inductively define a translation function τ over the concept expressions of the DL language $\mathcal{ALCQIO}$, from the alphabet $\langle \mathbf{A}, \mathbf{P}, \mathbf{O} \rangle$ to the alphabet $\langle \mathbf{A} \cup \{\mathsf{Top}\}, \mathbf{P}, \mathbf{O} \rangle$, as follows: $\tau(A) \triangleq A$, $\tau(C \sqcap C') \triangleq \tau(C) \sqcap \tau(C')$, $\tau((\bowtie \; n \; R \; C)) \triangleq (\bowtie \; n \; R \; \tau(C))$, $\tau(\{o\}) \triangleq \{o\}$, and $\tau(\neg C) \triangleq \mathsf{Top} \sqcap \neg \tau(C)$.

Proposition 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 $C^{\omega} = [\tau(C)]^{\hat{\omega}}$ for any concept expression C and $R^{\omega} = R^{\hat{\omega}}$ for any role expression R built using $\mathcal{ALCQIO}$ over the domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$.*

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 $\mathcal{T}$, $\tau(KB)$ includes a new axiom of the form $\tau(C) \sqsubseteq \tau(D)$; and for each ABox assertion $o : C$ in $\mathcal{A}$, $\tau(KB)$ includes a new axiom of the form $o : \tau(C)$.

Theorem 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 $\mathfrak{U}$.*

Since for the language $\mathcal{ALCQI}$ the *disjoint union model property* holds, we are able to establish the computational complexity of checking consistency of a world specification[3].

Corollary 1. *Given an $\mathcal{ALCQI}$ world specification $\mathcal{W}$, the problem of checking if it is consistent is* EXP*-complete.*

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 our 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: (1) create new objects, which means add (to the active domain of the resulting state) elements that are not included in the active domain of the

[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.

[3] Please refer to [16] for details about complexity classes.

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE

initial state (assuming that $\mathfrak{U} \setminus \Delta^{\omega} \neq \varnothing$), and that can be viewed as new; (2) add or remove elements to/from a concept extension; (3) add or remove links between elements. We point out that, while the extensional level can be altered by the actions performed, the intensional level, built essentially by the system specification and constraints, must be assumed as immutable. The devised framework allows for various complex primitives, generally available in such kind of settings, like: *variables* (encoded as singleton concepts or *nominals*) and *queries* (encoded as arbitrary $\mathcal{ALCQIO}$-concept expressions involving also variable names as parameters). Generally, an *assignment* binds each variable name to a domain element. We employ the variable construct also to model the object instantiation: *instantiation variables* are bound to newly instantiated objects or, in other words, 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.

4 Simple e-services

A simple e-service is an atomic update operator, described following the IOPE 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 non-deterministically chosen in order to mimic the *black-box* behavior of the provider. However, in this simple version we assume that an e-service has only one possible effect defined in its specification, which is realized once the service is invoked in a consistent way. For simple e-services, we exclude the possibility of any side-effect, that can potentially interfere with previous assumptions. However, we will remove this limitation in the following section. The key idea is that 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 set of formulas, stating a correspondence between the semantic properties of such a kind of logical formalization and the state transition system which is typically used to describe the semantics of an e-service.

Definition 4 (Simple e-service)**.** *Given a domain specification $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, and possibly a world specification $\mathcal{W}$ expressed using such alphabet, a simple e-service specification is a quadruple formed by: (1) a (possibly empty) finite set of input variable names $\mathbf{X}_S$; (2) a (possibly empty) finite set of output or instantiation variable names $\mathbf{Y}_S$; (3) a (possibly empty) finite set of invocation precondition constraint $\mathcal{P}_S$; (4) 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, the conditions under which it can be requested by a client, and the updates performed, if any. Generally speaking, a *precondition constraint* is a conjunction of positive (resp. negative) atomic conditions that are satisfied if the query result is not empty (resp. empty) given an input variable assignment and a world state. An atomic precondition is a pair $\langle s, Q(\mathbf{X}) \rangle$ where: (1) $s \in \{+, -\}$ is the sign of the precondition (positive or negative); (2) $Q(\mathbf{X})$ is a parameterized query over the domain specification in the variables $\mathbf{X} \subseteq \mathbf{X}_S$. A *simple effect* E is an arbitrary set of atomic concept and role effects built according to its variable names and domain specification. An atomic concept (resp. role) effect is a triple $\langle s, A, a \rangle$ (resp. a quadruple $\langle s, P, l, r \rangle$) s.t.: (1) $s \in \{+, -\}$ is the sign of the effect (insert or delete); (2) $A \in \mathbf{A}$ (resp. $P \in \mathbf{P}$) is the target concept (resp. role) name; (3) a (resp. l and r) is the argument of the update (positive or negative) according to the sign of the effect. A positive (resp. negative) effect argument is any element $Y \in \mathbf{Y}_S$ or any parameterized query $Q(\mathbf{X})$ over the domain specification in the variables $\mathbf{X} \subseteq \mathbf{X}_S$ (resp. any parameterized query $Q(\mathbf{X})$ over the domain specification in the variables $\mathbf{X} \subseteq \mathbf{X}_S$). Roughly speaking, a simple effect specifies which elements (or pair of elements) are inserted or removed from a set (or a binary relation). Generally, each atomic effect can affect more elements since its domain is denoted using queries[4].

Example 1. *Let $\mathcal{W}$ (Tab. 2) be an axiomatization of a simple domain, where people interact with e-services provided by public administrations. According to the given axiomatization, each goods 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. The preconditions can be expressed as $\{\{\langle +, x_1 \sqcap \mathsf{Citizen} \rangle, \langle +, x_2 \sqcap \mathsf{Town} \rangle\}\}$, while effects as $\{-\mathsf{resIn}(x_1, \exists \mathsf{resIn}^{-}.x_1), +\mathsf{resIn}(x_1, x_2)\}$. 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 any citizen, and allows to select any town as the new residence place. The town t_1 provides also the following version only to*

Table 2.

$\exists \mathsf{resIn}^{-}.\top$	$\sqsubseteq$	Town	$\mathsf{Citizen} \sqcap \mathsf{Town}$	$\sqsubseteq$	$\bot$
$\exists \mathsf{resIn}.\top$	$\sqsubseteq$	$\mathsf{Citizen}$	$\mathsf{Citizen} \sqcap \mathsf{Goods}$	$\sqsubseteq$	$\bot$
$\exists \mathsf{regIn}^{-}.\top$	$\sqsubseteq$	Town	$\mathsf{Goods} \sqcap \mathsf{Town}$	$\sqsubseteq$	$\bot$
$\exists \mathsf{own}^{-}.\top$	$\sqsubseteq$	$\mathsf{Citizen}$	$\mathsf{Vehicle}$	$\sqsubseteq$	Goods
$\mathsf{Citizen}$	$\sqsubseteq$	$(= 1\ \mathsf{resIn}\ \top)$	t_1	:	Town
$\mathsf{Vehicle}$	$\sqsubseteq$	$(= 1\ \mathsf{regIn}\ \top)$	t_2	:	Town
Goods	$\sqsubseteq$	$(\leq 1\ \mathsf{own}\ \top)$			

[4] Moreover, an atomic effect can be more concisely written also using the notation $+A(a), -A(a), +P(l, r), -P(l, r)$.

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE

its inhabitants that ask for a residence change that 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. The preconditions are $\{\{\langle +, x_1 \sqcap \exists \mathsf{resIn}.\{\mathsf{t}_1\}\rangle, \langle +, x_2 \sqcap \mathsf{Town}\rangle\}\}$*, while the effects are* $\{-\mathsf{resIn}(x_1, \exists \mathsf{resIn}^-.x_1), +\mathsf{resIn}(x_1, x_2)\} \cup \{-\mathsf{regIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{own}.x_1, \{\mathsf{t}_1\})\} \cup \{+\mathsf{regIn}(\mathsf{Vehicle} \sqcap \exists \mathsf{own}.x_1, x_2)\}$.

A set of precondition constraints is interpreted as a disjunction of such constraints. Moreover, a service is *accessible* if there exists a legal world state and an assignment s.t. it can be activated consistently with its preconditions.

Theorem 2. *Given a world specification $\mathcal{W}$ and a simple e-service S, both defined over the same domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$, let ω be a world state and σ be an input variable assignment. Then, there exists a $\mathcal{ALCQIO}$ boolean knowledge base KB^P, having a length linearly bounded by the problem setting size, s.t. S is accessible from ω using σ iff there exists a model $\hat{\omega}$ of KB^P and the world state and the assigment are embedded into it.*

Informally, the result relies over an extension of the encoding previously devised, adding some axioms to enforce the singleton semantics of variable names (for each variable V, $V \sqsubseteq \mathsf{Top}$, $\sharp(V) = 1$) and to select only world states that allow for the instantiation of at least $n = \|\mathbf{Y}_S\|$ new objects in the active domain ($\top \sqsubseteq \forall \mathsf{aux}.\mathsf{New}$, $\top \sqsubseteq \forall \mathsf{aux}^-.\mathsf{New}$, and $\mathsf{spy} : \mathsf{New} \sqcap (\geq\ n\ \mathsf{aux})$), while the preconditions are encoded adding the following boolean axioms $\bigvee_{P \in \mathcal{P}_S} \bigwedge_{p \in P} \gamma(p)$, where γ is a function defined as $\gamma(\langle +, Q\rangle) \triangleq \alpha_p : \tau(Q)$ and $\gamma(\langle -, Q\rangle) \triangleq \tau(Q) \sqsubseteq \bot$, being α_p a new fresh constant name not appearing elsewhere. The following property relies on a result reported in [19].

Corollary 2. *Given a world specification $\mathcal{W}$ and a simple e-service S, the problem of checking if it is accessible is in* NEXP.

Now, we introduce the formal definitions related to the semantics of the state update adopted in our framework. Generally speaking, we denote sets of elements or element pairs affected by an update specification defined using the syntax previously introduced. The following is the definition for concept extension update: the corresponding definition for role extension update is analogous.

Definition 5 (Concept update sets)**.** *Let E be an 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. We define as $A^+(\omega, \sigma_{\boldsymbol{X}}) = \bigcup_{\langle +, A, Q\rangle \in E} Q^\omega(\sigma_{\boldsymbol{X}}) \setminus A^\omega$ the insert set, while $A^+(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}}) = \bigcup_{\langle +, A, Y\rangle \in E} \{\sigma'_{\boldsymbol{Y}}(Y)\} \cup A^+(\omega, \sigma_{\boldsymbol{X}})$ is the instantiation set and $A^-(\omega, \sigma_{\boldsymbol{X}}) = \left(\bigcup_{\langle -, A, Q\rangle \in E} Q^\omega(\sigma_{\boldsymbol{X}})\right) \cap A^\omega$ the delete set.*

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., [4]). An effect is *consistent* if 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 2. *Both services introduced in Ex. 1 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 $\langle -, x_2 \sqcap \exists \mathsf{resIn}^-.x_1\rangle$ and $\langle -, x_2 \sqcap \{\mathsf{t}_1\}\rangle$.*

Since we are interested only in services that have no contradicting effects, we need also to enforce such a kind of constraint introducing some specific axioms. In particular, we can state the following claim.

Theorem 3. *Given a world specification $\mathcal{W}$ and a simple e-service S, both defined over the same domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$, there exists a $\mathcal{C}^2$ sentence KB^E, having a length polynomially bounded by the problem setting size, s.t. the service effect E is consistently defined iff for each concept name $A \in \boldsymbol{A}$ we have that $KB^E \wedge \tau(\mathcal{W}) \models A^+ \sqcap A^- \sqsubseteq \bot$ and for each role name $P \in \boldsymbol{P}$ we have that $KB^E \wedge \tau(\mathcal{W}) \models \neg\exists x, y.P^+(x, y) \wedge P^-(x, y)$.*

In [8] we show an effective procedure that is able to build KB^E for an arbitrary problem instance. 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 no other interacting agent.

Definition 6 (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 a simple e-service S according realizing the effect E and instantiating the set $\boldsymbol{Y}$, iff: (1) the interpretation domain $\Delta^{\omega'}$ of the successor state is the smallest subset of $\mathfrak{U}$ s.t. $\Delta^\omega \cup cod(\sigma'_{\boldsymbol{Y}}) \subseteq \Delta^{\omega'}$; (2) the interpretation of object names is preserved, i.e. $o^\omega = o^{\omega'}$; (3) for each concept or role name $N \in \boldsymbol{A} \cup \boldsymbol{P}$, the insert set is included in the successor state interpretation, i.e. $N^{\omega'} \supseteq N^+(\omega, \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}})$, and the delete set is excluded, i.e. $N^{\omega'} \cap N^-(\omega, \sigma_{\boldsymbol{X}}) \subseteq \emptyset$.*

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

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE

$\mathbf{Y}$ is the set of newly instantiated objects. Given a pair $\langle \mathbf{Y}, E \rangle$, the output variables occurring in the definition of the effect E must belong to the set $\mathbf{Y}$. 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, in a similar way to some data reconciliation approaches ([11]), based upon the *set symmetric difference*, applied to concept and role alphabets $\mathbf{A}$ and $\mathbf{P}$ and denoted as $d(\cdot, \cdot)$.

Definition 7 (Transition relation)**.** *Let ω and ω' be a pair of world states, s.t. the latter is resulting from the execution of a simple e-service S in the state defined by the former, realizing the effect E. Given an input and an output variable assignment $\sigma_{\boldsymbol{X}}$ and $\sigma'_{\boldsymbol{Y}}$ consistently defined, we say that there is a system state transition from ω to ω' using the specified effect iff: (1) ω' is a (potential) successor state of ω w.r.t. the given assignments; (2) there does not exist any other potential successor state ω'' of ω, w.r.t. the same assignments and service effect, s.t. $d(\omega, \omega'') < d(\omega, \omega')$.*

The *service enactment* set for a state ω and a consistent input variable assignment $\sigma_{\mathbf{X}}$, denoted as $S(\omega, \sigma_{\mathbf{X}})$, contains all the pairs $\langle \omega', \sigma'_{\mathbf{Y}} \rangle$ s.t.: (1) $\sigma'_{\mathbf{Y}}$ is a consistent instantiation assignment w.r.t. ω; (2) ω and ω' are in transition relation w.r.t. the assignment and the service effect. Also the embedding relation must be accommodated in order to deal with the transition definition that involves multiple world states (initial and final) and variable assignments. On such foundation, as in the previous cases, we obtain the following result:

Theorem 4. *Given a world specification $\mathcal{W}$ and an accessible and consistently defined simple e-service S, both defined over the same domain $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O} \rangle$, there exists a $\mathcal{C}^2$ sentence KB^U, having a length polynomially bounded by the problem setting size, s.t., every quadruple $\langle \omega, \omega', \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ embedded into a structure $\hat{\omega}$ is an enactment $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ of S, iff $\hat{\omega} \models KB^U$.*

As in the previous case, we can exhibit a procedure to actually build KB^U for a given service specification. The following claim enables us to extend the reasoning framework to any arbitrary enactment. Notice that the number of possible enactments is uncountably infinite, due to the nondeterministic choice on the instantiation assignment: however, it turns out that they are actually indistinguishable under the constraint language $\mathcal{ALCQI}$ ([10]).

Proposition 2. *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.*

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, and at the same time is able to fulfill its contract everytime it is activated consistently. In other terms, given a legal state where the service preconditions hold, the service invocation must result into a legal state where the declared effects are realized. The service contract is defined presuming that the invocation preconditions are sufficient in order to obtain one of the declared service effects by a service enactment. 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.

Definition 8 (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: (1) the effect E is consistent; (2) for each legal world state ω, for each consistent input assignment $\sigma_{\boldsymbol{X}}$, s.t. the service is accessible in ω using it, there exists a legal state ω' in the enactment.*

Example 3. *Given the specification of Ex. 1, consider the following e-service that allows a customer to buy a new vehicle (i.e., to become the owner): $\langle \boldsymbol{X} = \{x\}, \boldsymbol{Y} = \{y\}, \mathcal{P} = \{\{\langle +, x_1 \sqcap \mathsf{Citizen} \rangle\}\}, E = \{+\mathsf{Vehicle}(y), +\mathsf{own}(y, x)\}\rangle$. Despite the service is accessible and its effects are consistently defined, it is not valid since its enactments violate the domain constraints: a vehicle must be recorded to the town authority. A valid version of the service effects is the following: $E = \{+\mathsf{Vehicle}(y)\} \cup \{+\mathsf{own}(y, x), +\mathsf{Goods}(y), +\mathsf{regIn}(y, \exists \mathsf{resIn}.x)\}$*

Theorem 5. *A consistent and accessible simple e-service S is valid w.r.t. a world specification $\mathcal{W}$ iff the following implication holds: $KB^U \wedge \tau(\mathcal{W}) \models \bar{\tau}(\mathcal{W})$, where $\bar{\tau}$ is a translation function defined w.r.t. the name mapping $m(x) \triangleq \bar{x}$.*

Based on the above property, since the complexity of reasoning in $\mathcal{C}^2$ ([17]), we are able to provide the following complexity upper bound:

Corollary 3. *Given a world specification $\mathcal{W}$ and an accessible and consistent simple e-service S, the problem of checking if S is also valid is in* coNEXP.

5 Incomplete Specifications and Repairs

Differently from other approaches, so far we have assumed that the effect specification of a service is completely

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE

defined: in other words, there is no left space for any kind of collateral effects. Now, we remove this limitation, allowing for a non-deterministic update repair capability in order to enforce a consistent behavior. In the field of update theory, the notion of repair is old at least as the notion of update itself ([21]). However, the problem of repairing even a simple update in the presence of a complex intensional knowledge base (or a complex constraint set) turns out to be very hard, since, given the complexity of the axiom language, non-local repair side-effects may arise. This 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 DL ([4]) and in relational database schemas ([1]). Generally speaking, since we can reduce, using some adjustments, our framework to the proposal of [4], the problem is also undecidable, but, if we renounce to the completeness of the repair search, limiting to a restricted, and finite, set of possible repairs, we can regain decidability. The devised approach relies on the syntactical generation of repairing additional effects starting from singleton values (like variables and constants) mentioned in the problem setting, performing a kind of local search in the space interpretation structure w.r.t. the set symmetric distance.

Example 4. *Given the following world specification of Ex. 1 and the following service specification:* $\langle \boldsymbol{X} = \emptyset, \boldsymbol{Y} = \{y\}, \mathcal{P} = \emptyset, E = \{+\mathsf{Vehicle}(y)\}\rangle$*, it is trivial to observe that the service is not valid, since the insertion of a new element in the extension of the concept* Vehicle *violates various axioms. In order to enforce this constraint, a repair like* $\{+\mathsf{Goods}(y), +\mathsf{regIn}(y, \{\mathsf{t}_1\})\}$ *is enough.*

A *simple repair* R for a simple 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 (i.e., atomic repair differing only by the sign). An *atomic repair* is a special kind of atomic effect, which arguments range over any nominal introduced in the specification. Restricting our attention to simple repairs, we can assume, as repair search space for a given e-service S, a set R_S s.t.: (1) it includes the null repair ($\emptyset$); (2) 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.

Proposition 3. *Given a domain specification* $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$ *and a simple e-service* $S = \langle \boldsymbol{X}_S, \boldsymbol{Y}_S, \mathcal{P}_S, E_S\rangle$*, there are at most* $O\left(2^{\|\boldsymbol{A}\|\cdot n + \|\boldsymbol{P}\|\cdot n^2}\right)$ *distinct simple repairs, where* $n = \|\boldsymbol{O}\| + \|\boldsymbol{X}_S\| + \|\boldsymbol{Y}_S\|$.

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. Given a service with the update repair capability previously introduced, 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 ones.

Definition 9 (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 simple e-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.: (1) its interpretation domain is the same as the interpretation domain of the successor state* ($\Delta^{\omega'} = \Delta^{\omega'_R}$)*; (2) the interpretation of each concept or role name* $N \in \boldsymbol{A} \cup \boldsymbol{P}$ *is adjusted accordingly to the repair; (3) 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.: (1) it does not "undo" the effects of the service (i.e., deleting an element just inserted); (2) it actually updates the world state (i.e., it should not insert an element already present into a set or just inserted by the service enactment). Moreover, among multiple repaired successor states , the repairing strategy selects the one closest 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 10 (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 a simple 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, we say that there is a system state transition from* ω *to* ω'_R *using the specified effect iff: (1)* ω'_R *is a repaired successor state of the enactment* $\langle \omega', \sigma'_{\boldsymbol{Y}}\rangle \in S(\omega, \sigma_{\boldsymbol{X}})$*; (2)* $d(\omega'_R, \omega') \leq d(\omega', \omega'')$ *for any* $\omega'_{R'}$ *s.t.* $R' \in R_S$ *and* R' *is consistent.*

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. We can employ a name mapping function to cope with multiple repairs at the same time, using different names for each non-deterministic branch. So, let $R_i \in R_S$ be a repair, a suitable name mapping function in the case of simple e-service is $n_i(x) \triangleq x_i$ and $n_i(\mathsf{Top}) \triangleq \bar{\mathsf{Top}}$. We show that we can extend our reasoning framework also to keep into account the devised repair strategy.

Theorem 6. *Given a world specification* $\mathcal{W}$ *and an accessible and consistently defined simple e-service* S*, both defined over the same domain* $\langle \boldsymbol{A}, \boldsymbol{P}, \boldsymbol{O}\rangle$*, and a simple repair* R^* *for* S*, there exists a* $\mathcal{C}^2$ *sentence* $\Delta KB^R(R^*)$*, having a length polynomially bounded by the problem setting size, s.t. every*

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE

quintuple $\langle \omega, \omega', \omega'' \sigma_{\boldsymbol{X}}, \sigma'_{\boldsymbol{Y}} \rangle$ *embedded into a structure* $\hat{\omega}$ *is an enactment* $\langle \omega', \sigma'_{\boldsymbol{Y}} \rangle \in S(\omega, \sigma_{\boldsymbol{X}})$ *of* S *repaired applying* R^* *leading to state* ω''*, iff* $\hat{\omega} \models KB^U \wedge \Delta KB^R(R^*)$.

Definition 11 (Repairable simple e-service)**.** *Let* E *be the effect of a simple e-service* S*, and let* R_S *be the set of repairs,* S *is repairable w.r.t. a world specification* $\mathcal{W}$ *iff: (1) the effect* E *is consistent; (2) for each legal world state* ω*, for each consistent input assignment* $\sigma_{\boldsymbol{X}}$*, s.t. the service is accessible in* ω *using it, there exists a state* ω' *in the enactment and a repair* $R \in R_S$ *s.t. the repaired state* ω'_R *is legal.*

Theorem 7. *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:*

$$\tau(\mathcal{W}) \wedge KB^U \wedge \bigwedge_{i=1}^{r} \Delta KB^R(R_i) \models \bigvee_{i=1}^{r} \tau_i(\mathcal{W}) \wedge \Delta KB^C(R_i)$$

where τ_i *is the translation function defined w.r.t. the name mapping function* n_i *and* $\Delta KB^C(R_i)$ *is a suitable* $\mathcal{C}^2$ *sentence that encodes the repair consistency constraints for a given repair, having a length polynomially bounded by the problem setting.*

We have also devised an effective procedure to build both ΔKB^R and ΔKB^C for given service S and simple repair $R \in R_S$: please refer to [8] for details. Moreover, since an exponential number of possible repairs must be accordingly encoded, we obtain the following property:

Corollary 4. *Given a world specification* $\mathcal{W}$ *and an accessible and consistent simple e-service* S*, the problem of checking if it is also repairable is in* co2NEXP.

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.

6 Conclusions

The main contribution of this work is to show how an expressive and suitable operational semantic model for e-service, based upon the enforcing of the service contract, can be traced to a logic characterization as a foundation for the analysis of a number of relevant functional properties in the design of a service-oriented integration solution. Moreover, the model accounts for both complete and incomplete specifications of e-services. To the best of our knowledge, this is the only framework which is able to address the following aspects: (1) allowing for effective reasoning about action consequences w.r.t. a complex domain specification, without any significant limitation regarding the constraint language; (2) adopting a semantics of minimal-change; (3) taking into account more interesting, and semantically well-founded, repairing options.

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] G. Alonso, F. Casati, H. Kuno, and V. Machiraju. *Web Services: Concepts, Architectures and Applications*. Springer-Verlag, Berlin, Germany, 2004.

[3] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2003.

[4] F. Baader, C. Lutz, M. Milicic, U. Sattler, and F. Wolter. Integrating description logics and action formalisms for reasoning about web services. Technical report, Desden University of Technology, 2005.

[5] D. Berardi, D. Calvanese, G. De Giacomo, R. Hull, and M. Mecella. Automatic composition of transition-based semantic web services with messaging. In *Proc. of the 31st VLDB Conference*, Trondheim, Norway, 2005.

[6] M. Carman, L. Serafini, and P. Traverso. Web Service Composition as Planning. In *Proceedings of ICAPS'03 Workshop on Planning for Web Services*, Trento, Italy, June 2003.

[7] A. Deutsch, L. Sui, and V. Vianu. Specification and verification of data-driven web services. In *Proc. of the PODS 2004 Conference*, Paris, France, 2004. ACM.

[8] L. Dragone and R. Rosati. Modeling and reasoning about e-services. Technical report, DIS, Università di Roma "la Sapienza", 2006.

[9] ebXML Technical Commitee. ebXML – electronic business using XML. Specification, OASIS, Oct 2003.

[10] N. Immerman and E. Lander. Describing graphs: A first-order approach to graph canonization. In *Alan L. Selman, Editor, Complexity Theory Retrospective*, volume 1. 1990.

[11] J. Lin and A. Mendelzon. Merging databases under constraints. *International Journal of Cooperative Information Systems*, 7(1):55–76, 1996.

[12] A. Martens. Usability of web services. In *Proc. of the 1st Web Service Quality Workshop*. IEEE Press, 2003.

[13] S. A. McIlraith and T. C. Son. Adapting Golog for composition of semantic web services. In *Proc. of the KR 2002 Conference*, Toulouse, France, 2002.

[14] B. Meyer. *Object-Oriented Software Construction*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[15] T. D. Noia, E. D. Sciascio, F. M. Donini, and M. Mongiello. A system for principled matchmaking in an electronic marketplace. In *Proc. of the WWW 2003 Conference*, Budapest, Hungary, 2003.

[16] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[17] I. Pratt-Hartmann. Complexity of the two-variable fragment with counting quantifiers. *J. of Logic, Lang. and Inf.*, 14(3):369–395, 2005.

[18] M. K. Smith, C. Welty, and D. L. McGuinness. OWL web ontology language guide. Recommendation, World Wide Web Consortium (W3C), Feb 2004.

[19] S. Tobies. The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. *J. Artif. Intell. Res. (JAIR)*, 12:199–217, 2000.

[20] W. M. P. van der Aalst. The application of Petri Nets to workflow management. *J. of Circuits, Systems and Computer*, 8(1), 1998.

[21] M. Winslett. *Updating logical databases*. Cambridge University Press, New York, NY, USA, 1990.

2007 IEEE International Conference on Services Computing (SCC 2007)
0-7695-2925-9/07 $25.00 © 2007 IEEE