An Ontology-based Diagnosis of Mainstream Service Modeling Languages

Julio Cesar Nardi[1], João Paulo A. Almeida[2], Paulo Henrique A. da Silva[1], Giancarlo Guizzardi[2,3]
[1]Informatics Department, Federal Institute of Espírito Santo, Colatina, Brazil
[2]Ontology & Conceptual Modeling Research Group (NEMO), Federal University of Espírito Santo, Vitória, Brazil
[3]Facoltà di Scienze e Tecnologie Informatiche, Free University of Bozen-Bolzano, Bozen-Bolzano, Italy
julionardi@ifes.edu.br, jpalmeida@ieee.org, pauloharaujos@gmail.com, giancarlo.guizzardi@unibz.it

***Abstract*— This paper presents a diagnosis of mainstream service modeling languages (SoaML, USDL, and ArchiMate) in light of UFO-S, a reference ontology for services. UFO-S is intended as a broad ontology for service phenomena, harmonizing different perspectives on services (e.g., "service as commitment", and "service as capability"), and addressing several phases of the service lifecycle (service offering, service agreement, and service delivery). As result, UFO-S is used as an "analysis theory" to identify choices in these languages concerning their focus and coverage of service phenomena. We identify a number of possible improvements concerning the representation of service participant (roles), the description of service offerings, service agreements and service delivery.**

Keywords: service modeling languages; service ontology; SoaML; USDL; ArchiMate.

I. INTRODUCTION

The notion of service has been approached from various perspectives, including: "service as commitment" [1], "service as interaction" [2], "service as value co-creation" [3], and "service as capability" [4]. These perspectives are influenced by many aspects, such as, the point of view, the level of maturity, and the practical problems faced by the respective application area. For example, in the Business and IT application areas, the perspective of "service as capability" has been widely adopted [5][4]. The perspective of "service as commitment", in turn, has been advocated as a useful way of dealing with service intangibility [4] and raising the low-level of abstraction of service-oriented architectures to reduce the gap between the Business and the IT [6]. We have observed that these perspectives end up being reflected in different ways in various service modeling languages, resulting in different language facilities for the representation of service relations. Despite the importance of the different representation facilities, we believe that service modeling languages need to be consistent to the essence of service relation dynamics as a way of avoiding semantic inconsistence and favor language interoperability.

This paper describes and analyses the service representation support of three service modeling languages that were subject to standardization. The analyzed languages are: the Service-oriented Architecture Modeling Language (SoaML) [7], the Unified Language for Service Description (USDL) [8], and ArchiMate [9]. The *SoaML* specification provides a metamodel and a UML profile for the representation of services within a Service-Oriented Architectures (SOA) [7]; *USDL* is a platform-neutral language for describing technical and business services by adding information useful to, e.g., providers, gateways, and consumers [8]; and, *ArchiMate,* as a graphical service-oriented modeling language, provides service representation throughout enterprise architecture layers [9]. Together, these languages represent a spectrum of concerns in service modeling, reflecting the broad scope of application of service notion: SoaML stems from a software-centric setting, focusing on the technical specification and design of services in UML; USDL is intended to complement web services technical languages addressing business concerns (e.g., pricing and policies in service provisioning); and ArchiMate was designed originally with the intent of bridging the gap between business and IT services, facilitating communication in various architectural domains.

We aim to answer the following main research question: "*What are the representational capacities of SoaML, USDL and ArchiMate with respect to the service phenomena?"*. For that, we have defined a research method focused on addressing a set of derived research questions based on the reference ontology UFO-S [10]. In this work, therefore, UFO-S is applied as a kind of "analysis theory" [11], grounding the research method definition and its execution phases. It is suitable for this task as it was developed with the purpose of harmonizing different service perspectives and addressing several phases of the service lifecycle (service offering, service agreement, and service delivery), revealing service relations and service participant roles.

The results of the diagnosis can support prospective users and language experts in future efforts of revision and redesign. We show there is a particular gap in representation of commitment-based aspects in the analyzed techniques.

This paper is further structured as follows: Section II presents the methodological aspects, clarifying the role of reference ontologies in the analysis of modeling languages; Section III presents the reference UFO-S ontology; Section IV refines the general research question into more six specific ones, in light of UFO-S; Sections V to VII briefly present and analyse the three service modeling languages addressing the research questions; Section VIII summarizes the analysis and position the modeling languages; and, finally, Section IX presents the final considerations.

II. METHODOLOGICAL CONSIDERATIONS

There is an established tradition of almost three decades of systematically analyzing and evaluating conceptual modeling languages by employing the results of formal ontological studies [12]. In ontological analysis (or "representational analysis" [13]), a rigorously defined *reference ontology* is used to assess a language (or its metamodel) to uncover representation gaps and ontological deficiencies in the language. The approach is based on the

observation that, since the main purpose of a language is to represent and communicate certain aspects of phenomena of interest, we must as best as possible understand and characterize the nature of those phenomena; a task which is undertaken systematically in reference ontology design. A reference ontology serves in this case an "analysis theory" [11] or "representation theory" [13].

Since the pioneering work of [12], a number of languages have been evaluated and (re)designed using this approach, whose effectiveness has been empirically demonstrated by a myriad of studies over the years [14][15].

As discussed in [13], [12], and in [16], ontological analysis is performed by "comparing the constructs of the chosen representation theory with the constructs of the modeling grammar and by identifying any representation equivalence between these". Two principal evaluation criteria are *ontological completeness*, i.e., the extent to which the modeling grammar has a deficit of constructs that map to the set of representation theory constructs, and *ontological clarity*, i.e. the extent to which the modeling grammar constructs are deemed overloaded, redundant, or excessive [12] (*apud* [13]). From the perspective of design science research, language grammars can be regarded as designed artifacts. In the rigor cycle, the design artifact is evaluated by employing a knowledge base (such as ontologies) [17].

By using as a reference ontology the Unified Foundational Ontology (UFO) [18], ontological analysis has been successfully employed over the years to analyze, (re)design and integrate conceptual modeling languages and standards in different domains (e.g., RM-ODP [19], TROPOS/i* [20], ARIS [21], BPMN [22], and ArchiMate [23]). Of particular interest of this paper, UFO has been used to develop core ontologies in the domain of Services (termed UFO-S). As discussed in the next section, in the case of service modeling, a suitable reference ontology must characterize the multifaceted notion of service, harmonizing its various perspectives. Given that services cross the Business-IT line, we argue that such a reference ontology should address the social nature of service relations, which are invariably linked to various aspects of the service life-cycle, covering from service offering to service delivery.

III. A Reference Ontology for Services

A number of works in Service Science [1][24] and Service Computing [25][26] explicitly mention commitments, promises and/or obligations for characterizing the service relations established between service participants. The benefits of a characterization based on commitments have been discussed from the perspective of business [24] as well as IT [27].

UFO-S is a core reference ontology for services based on the notion of social commitments [10]. As a *reference ontology* [28], UFO-S is intended to assist humans in meaning negotiation and shared understanding. It is grounded in a foundational ontology (the Unified Foundational Ontology – UFO [18]) from which it reuses foundational notions of objects, types, object properties, object relations, events/processes, and further social concepts that specialize the more general notions and account for social reality. The social layer of UFO includes important notions of social agents (e.g., enterprises), the objectives they pursue, the roles they play, the social relations they establish (commitments and corresponding claims), etc.

UFO-S focuses on the three basic phases of the service life-cycle: (i) service offer (when a service is presented and made available to a target customer community), (ii) service negotiation (when providers and customers negotiate for establishing an agreement), and (iii) service delivery (when actions are performed to fulfill a service agreement).

Figure 1 presents a UFO-S model fragment regarding service offer. A service offer is an event (e.g., the registration of a service provider organization in a chamber of commerce) that results in the establishment of a service offering, which mediates the social relations between the service provider and the target customer community. A service offering is composed of service offering commitments from the service provider towards the target customer community, and the corresponding service offering claims from the target community towards the service provider. Service offering commitments refer to commitments that can be established later in the negotiation phase. The content of the service offering commitments and claims may be described in service offering descriptions (e.g., folders, registration documents in a chamber of commerce, and artifacts in software service registries).

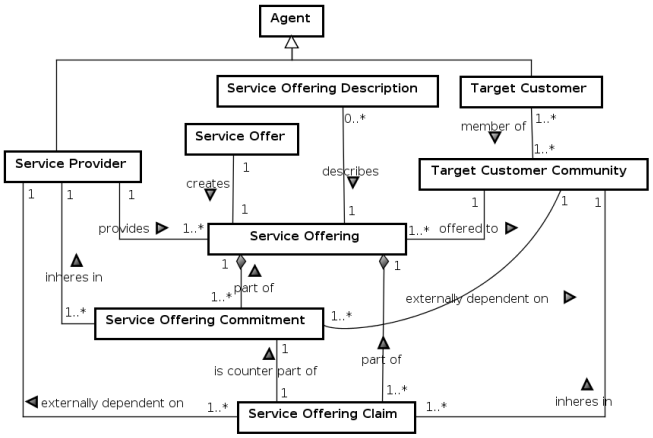

Figure 1: Service Offer.

Service provider is the role played by intentional agents (e.g., physical agents such as persons, and social agents such as organizations) when these agents commit themselves to a target customer community by a set of offering commitments. Target customer community is a collective that refers to the group of agents that constitute the community to which the service is being offered. Target customer is the role played by agents when become members of the target customer community, and, as a consequence, have claims for the fulfillment of the commitments established by the agent playing the role of service provider.

Once a service is offered, service negotiation may occur. Figure 2 presents UFO-S model fragment of this phase. Service negotiation is an event involving a target customer and a service provider. If service negotiation succeeds, a

service agreement is established, and the service provider starts to play the role of hired service provider, while the target customer starts to play the role of service customer.

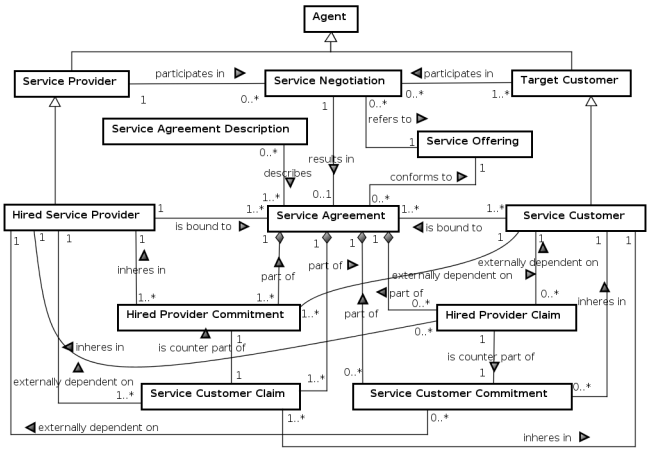

Figure 2: Service Negotiation.

A service agreement mediates the social relations between service customer and hired service provider, being composed of commitments and claims established among them. The content of commitments/claims of a service agreement may be described in a service agreement description (e.g., contract). The mutual service commitments/claims established in the service agreement will drive the service delivery.

Figure 3 presents UFO-S model fragment regarding service delivery. Service delivery is an event composed by actions performed by the hired service provider (hired provider actions), actions performed by the service customer (customer actions), and/or actions performed by both in an interaction (hired provider-customer interaction).

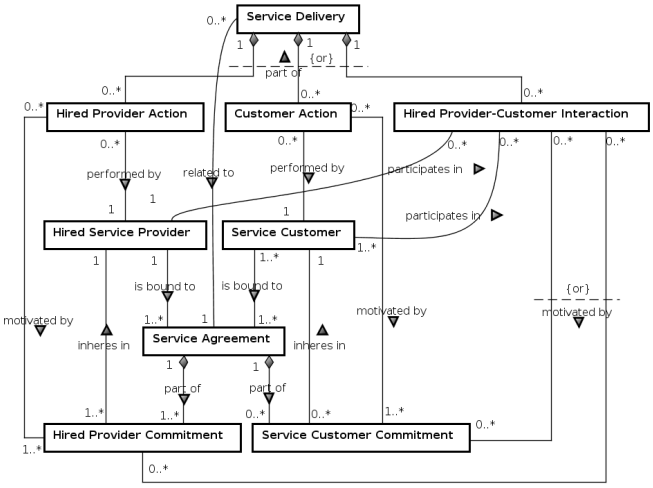

Figure 3: Service Delivery.

These actions are the manifestation of dispositions (and in particular capabilities) of service participants [29][23]. Service delivery concerns the execution of actions aiming at fulfilling commitments established in the service agreement. Depending on the business model, other agents can also perform actions. E.g., the service provider can delegate some actions to a third-party (e.g., actions performed by human resources, or actions performed by third-business partners). These actions are part of the service delivery process, but they are not explicitly represented in Figure 3.

Besides the aforementioned *commitment-based service view*, we can also find that one directly associated with the use or application of resources/capabilities [30]. This *capability-based service view* is discussed under different banners, e.g: "service as capability" (capability of a provider to produce benefits to customers) [23]; "service as application of competences" (manifestation of one party's capability in benefit of another party) [31], and "service as resource/capability applied in process" (integration between service as resource and service as process) [32].

We consider the *commitment-based service view* and the *capability-based service view* as complementary, and both are required in a thorough account of services [33]. Concerning the application of resources (or the manifestation of their capabilities), we consider that hired service provider and service customer are mutually committed to apply their resources (and their capabilities) to fulfill the established service commitments. In the service delivery phase, those resources (and capabilities) are used (manifested) as agreed.

As an important consequence of this theoretical foundation is the fact that a (genuine) service relation is inevitably a social phenomenon between intentional agents. Therefore, only intentional agents can play the roles of service provider and service customer, since only this kind of agent can be committed to other agents. Enterprise resources (e.g., applications and devices) do not themselves play the role of service providers and customers. Instead, service provider and service customers (agents) employ resources as a means to fulfill their commitments.

Finally, when a service agreement is established, the service customer delegates a goal/plan to the hired service provider. A delegation in a service relation may be followed by further (service) delegations, too common in supply chains and economic networks. As such, a network of service commitments is established between service participants that acts as a "glue" and leads to the application of their resources/capabilities to fulfilling their commitments. Thus, this network of mutual service commitments drives a network of Business- and IT-capabilities of different enterprises for delivering the correspondent services [33].

IV. Research Questions

In line with our main research question ("*What are the representational capacities of SoaML, USDL and ArchiMate with respect to service phenomena?*") and in light of UFO-S, the following refined questions are defined for each of the three analyzed modeling languages:

RQ1 – To what service characterization(s) is the language committed?

RQ2 – What are the representational capacities for service participants (including their roles as target customer, provider, service customer, and hired service provider)?

RQ3 – What are the representational capacities for service offerings?

RQ4 – What are the representational capacities for service agreements?

RQ5 – What are the representational capacities for service delivery (including relationships between delivery actions and the correspondent motivational service commitments)?

RQ6 – What are the representational capacities for links among service relations (e.g., service networking, and service relations chaining)?

Below we present the analyzed language and report on these research questions. The summary of the analysis with recommendations are presented in Section VIII.

V. Service Modeling in SoaML

A. SoaML Overview

In the Service oriented architecture Modeling Language (SoaML), a *service* is "the delivery of value to another party, enabled by one or more capabilities" [7], and "[...] enable us to offer our capabilities to others in exchange for some value". "A service represents a feature of a Participant that is the offer of a service by one participant to others using well defined terms, conditions and interfaces" [7]. "Capabilities identify or specify a cohesive set of functions or resources that a service provided by [...] participants might offer" [7].

A *participant* is defined as "specific entities or kinds of entities that provide or use services. Participants can represent people, organizations, or information system components" [7]. Thus, in SoaML, service provider and consumer entities may be people, organizations, technology components or systems (being all these called participants).

A *service description* establishes how the participant interacts to provide or use a service. Thus, it "specifies how consumers and providers are expected to interact through their ports to enact a service, but not how they do it" [7]. It can be specified by means of a simple UML interface, a service interface, and a service contract. A *simple interface* focuses on a one-way interaction provided by a participant. It is used with "anonymous" callers and the participant makes no assumptions about the caller or choreography of the service [7]. A *service interface* "is defined in terms of the provider of the service and specifies the interface that the provider offers as well as the interface, if any, it expects from the consumer" [7]. Therefore, it defines the responsibilities of a participant to provide or consume a service. A consumer of a service, in turn, specifies the service interface she requires. Compatibility of service interfaces determines if the agreements are consistent and can be connected to accomplish the real world effect of the service [7].

A *service contract* is a "formalization of a binding exchange of information, goods, or obligations between parties defining a service", being a specification of an agreement between the parties for how the service is to be provided and consumed [7]. It includes the interfaces, choreography, and any other terms and conditions [7]. In SoaML, the "agreement" may be asserted in advance or arrived at dynamically, as long as an "agreement" exists by the time the service in enacted [7]. Each role, or party involved in a service contract is defined by an interface or service interface that is the type of the role.

SoaML supports, basically, two modeling approaches [7]: (i) a contract-based approach, and an interface-based approach. These approaches are, respectively, associated to the use of the "service contract" and the "service interface" elements. In the *interface-based approach*, we focus on specifying the interfaces offered and used by service consumer and service provider. In the *contract-based approach*, we work on offering a wide view being most applicable where an enterprise or a community SOA architecture is defined, or when there are more than two parties involved in the service. The fundamental differences between interface- and contract-based approaches is whether the interaction between participants are defined separately from the participants in a service contract, or individually on each participants' service and request [7]. Despite the stylistic difference, there is significant overlap in specification capacities and both may be used in some cases.

Figure 4 presents a service interface diagram that addresses the "Online Selling Books" service from the "Bookstore" provider towards the "Bookstore Consumer". The diagram focuses on the provider's view, thus the presented service interface realizes the service provider interface and uses the correspondent consumer interface. Figure 5, in turn, presents a service contract diagram, the "Online Selling Books (terms and conditions)", which focuses on representing the service relation established between the "Bookstore Consumer" and the "Bookstore".

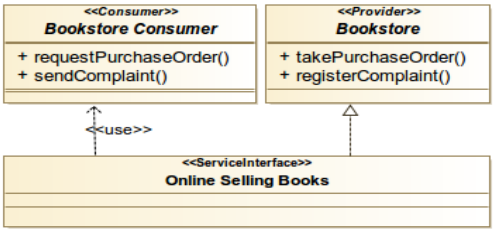

Figure 4: Bookstore scenario fragment: service interface diagram.

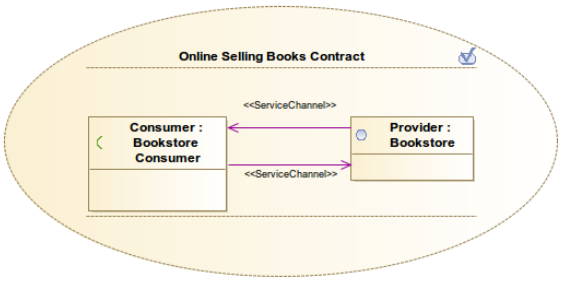

Figure 5: Bookstore scenario fragment: service contract diagram.

As we can notice, differently from the service interface, in the service contract diagram, the contractual aspects are represented separately from the participants, not individually. Finally, Figure 6 presents the "Online Selling Books Provisioning" service architecture diagram, which represents the *service contracts* established between consumers and providers involved in a services network, since the purchase order made by the "Bookstore Consumer" and addressed by the "Bookstore", until the goods shipping done by the "Shipping Company" (as provider).

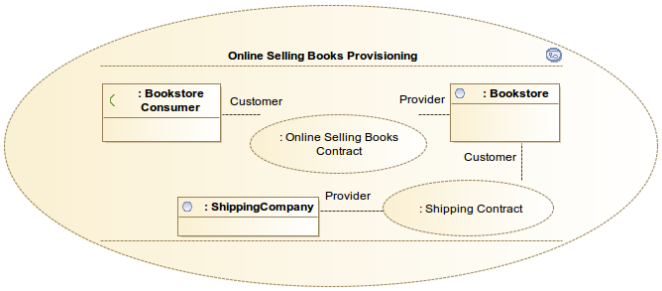

Figure 6: Bookstore scenario fragment: service architecture diagram.

B. Research Questions Analysis

RQ1 - Service characterization. In SoaML we can find three definitions for service, namely: (i) "delivery of value […] enable by […] capability", (i) "feature of a participant", and (i) "resource that enables access to […] capabilities". All of them are thus built on the notion of capability. However, considering the notion of *capability* in SoaML "as cohesive set of functions or resources", the emphasis is on capabilities related to behavioral aspects. In this context, SoaML offers the concept of *contract* that, as a UML collaboration, focuses on the behavioral aspects of service agreement type to be instantiated in a specific scenario. As a result, the notion of commitments and claims for characterizing (genuine) service relations, as advocated in UFO-S, is somewhat ancillary.

RQ2 – Service participant (roles). SoaML explicitly offers two basic service participant roles, namely: the *Service Consumer*, and the *Service Provider*. We could not find constructs for denoting the UFO-S notions of Target Customer, Hired Service Provider and Target Customer Community. Thus, SoaML's service participant roles, *Service Consumer* and *Service Provider*, seem to collapse, respectively: Target Customer and Service Customer, and, Service Provider and Hired Service Provider roles defined in UFO-S. We could find some textual references to "community" and "marketplace", which could be seen as possible Target Customer Community (in UFO-S). However, these terms appear only in the specification text informally, not corresponding to a language construct. Further, in SoaML there is no distinction between intentional and non-intentional participants, e.g.: "participants are either specific entities or kinds of entities that provide or use services […]", it "[...] can represent people, organizations, or information system components", and it "[...] may provide any number of services and may consume any number of services" [7]. As a result, even non-intentional participants (e.g., system software) with its computational capabilities put available could be considered a service provider. Neglecting social aspects of service relations does not contribute for a clear alignment between Business and IT, since typical organizational resources (as software systems, and processing nodes) may be grossly considered service providers [33]. These typical resources, differently from enterprises, departments, and persons cannot themselves commit, delegate and claim. In case the "Bookstore" hires a data storage service from the "Warehouse, Inc.", we cannot say that the data hosts/servers themselves provide the service, because the provisioning encompasses more than that. It is necessary, e.g., to guarantee the electric power supply, hardware and software upgrades. So, by means of the social aspects inherent to the Business level it is possible to address the application of capability/resources at IT level.

RQ3 and RQ4 – Service offerings and service agreement descriptions. In SoaML, service descriptions can be built by means of service interfaces, and service contracts. Thus, we understand that service offerings (in terms of UFO-S) could be represented in SoaML by means of service interfaces as well as service contracts. For that, however, there should be a way of representing the service provider and the correspondent target customer, for who the offering is established. In this case, service contracts and service interfaces would represent the service offering commitments from the service provider toward the target customers, and the correspondent expected conditions to be satisfied by possible actual service customers (in case of possible service agreement establishment). An agreement, in SoaML, between a consumer and provider may be also captured in a common service contract, which may constrain both the consumer's request service interface and the provider's service interface. Service contracts and service interfaces can also be used as a kind of patterns of services (service agreement type in UFO-S), which can be instantiated by specific individuals. However, considering the emphasis on behavioral aspects in SoaML, service interfaces and service contracts focus on interaction between service participants more than on quality of service parameters, for example. As such, service offering and agreement descriptions are based on the descriptions of operations or sets of operations that are typically characterized by a pair of interaction types and constraints on them. In this case, an operation invocation may count as (an implicit and trivial) service negotiation, with the establishment of an agreement whose type is pre-defined in the service offer. In any case, the notion of commitment is instrumental in explaining both the semantics of service description publication and the establishment of Service-Level Agreements (agreements in UFO-S) [34].

RQ5 – Service delivery representation. In SoaML, a service contract is represented as a UML Collaboration. The sequence of actions between provider and consumer are represented in service contract choreography, which is a UML behavior and, therefore, can be represented by UML behavioral diagrams (e.g., sequence diagram, and activity diagram). As aforementioned, the behavioral specification of

a service contract, as a UML collaboration, can be seen as a service agreement type in UFO-S encompassing the actions performed by service participants and resources. In SoaML, however, there is no explicit relationship between the actions/interactions performed by providers and customers and the usage of resources, and the correspondent service commitments established among these service participants, which motivate these actions/interactions and usages. We believe that establishing an explicit fine-grade relation between the delivery actions/interactions (type) to the correspondent motivational service commitments (not only on the level of contract, as in SoaML by means of the service contract choreography) would be useful. It would allow to identify, e.g., (i) why some actions must be performed in a such way, and (ii) which commitments may be broken if a specific action is not performed as agreed.

RQ6 – Service relation links. SoaML offers the service architecture diagram, by means of which it is possible to represent a specific scenario encompassing a network of service contracts linked to the correspondent consumers and providers. This diagram offers a view of what are the service agreements (types) that characterize the addressed service provisioning scenario. However, as Figure 1 illustrates, there is no specific relation among service contracts. Without these relations, we could not explore, e.g., dependency and conformance relationships among service contracts.

VI. Service Modeling in USDL

A. USDL Overview

The Unified Service Description Language (USDL) is a platform-neutral language for describing services [8]. It is designed for addressing technical services (e.g., WSDL and REST), and business services (business activity provided by a service provider to a service consumer) [8]. USDL aims at complementing the technical languages stack by adding business and operational information [8].

The USDL concepts are structured in 9 (nine) modules, which address functional and non-functional aspects of service provisioning [8]. They are described below and Figure 7 presents a bookstore scenario illustrated by representation of some USDL concepts.

Service module [8]. This module can be considered the core of USDL. It addresses service description aspects that crosscut the other modules. This module contains concepts such as Service, ServiceBundle, CompositeService, and AbstractService. *Service* concept encapsulates a functionality offered by a provider. *ServiceBundle* allows to specify bundles of services offered/hired in tandem. *CompositeService* is a kind of service that establishes a relation of execution. It can be ordered (e.g., a business process phases) or non-ordered. *AbstractService* is used to describe class of service (e.g., "Car Wash"), offering as a model that can be instantiated in a specific situation.

Interaction module [8]. This module allows to describe the sequence of individual functions or interactions with other services and agents, including, therefore, the sequence of interactions between consumer and provider and actors involved in the service delivery. This module contains concepts as *Interaction Protocol* (as mandatory or optional interactions between service participants), *Function*, *Parameters*, *Conditions*, etc.

Legal and Pricing modules [8]. The Legal module describes legal aspects of service provisioning, e.g., terms and conditions, licenses, rules of using, etc., while the pricing module addresses segmentation rules within the price structure, i.e., rules determining when and how different consumers are charged different prices. It allows to model scenarios with alternative price plans that may be assigned to an offered service or bundle (e.g., by *PricePlan* concept).

Functional and Technical modules [8]. In order to enable the description of human and automated services, these modules capture service functionality in a conceptual way. Conceptual, in this context, means independent of the ways to technically access functionality. Thus, the concept of service itself and service's interface are addressed separately, since a service can be accessed by means of various interfaces. Interface means a set of concrete technologies through which the service can be accessed. E.g., an automated service can be available by means of a WSDL interface, whereas a manual service can be available *in-loco*.

Participants module [8]. The Participants module deals with information about service agents, such as: *Business Owner*, *Provider*, *Intermediary*, *Stakeholder* and *Target Consumer*. It also describes dependencies between these agents and their relations with the Service module.

Service Level module [8]. This module deals with service specifications addressing guarantees of quality of service operation, which are claimed/requested by different actors involved in the provisioning. A set of service level specifications can be combined into one profile and are offered, negotiated, or agreed upon as a whole. Different profiles are used to specify different options of how service levels may be grouped (e.g., as gold, silver, bronze profiles).

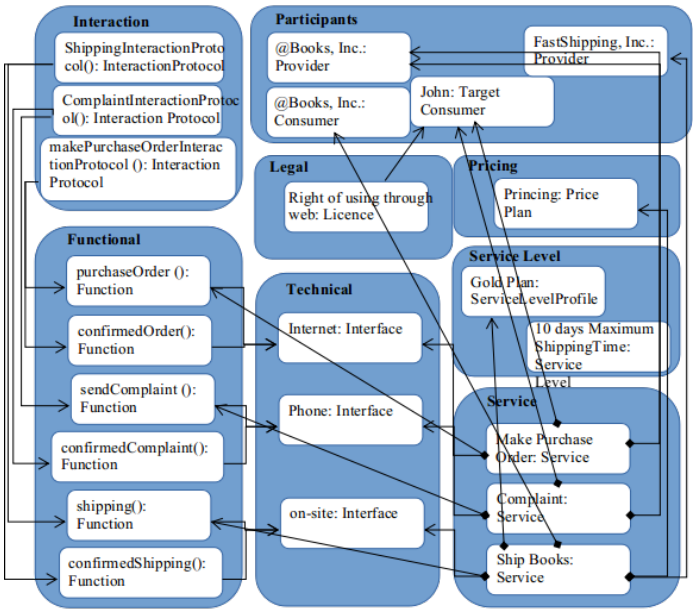

Figure 7: Bookstore scenario fragment represented by means of USDL.

Foundation module [8]. This module captures concepts common among several aspects or that cannot be organized in other module, e.g., concepts of naming and identification, or concepts that are completely independent of "service", e.g., *Organization*, *Person*, *Resource,* and *Artifact*.

By analyzing the Figure 7, we can notice the service participant roles ("Provider", "Target Customer", and "Consumer") and the correspondent individuals that instantiate them; the services provided/consumed, the interfaces by means of what services are provided/consumed; and the functional/interactive aspects (which represent the dynamics of service delivery). Finally, we can see the legal, pricing and service level elements that addresses non-functional properties of the service provisioning.

B. Research Questions Analysis

RQ1 - Service characterization. In USDL, a service is told to "encapsulate a functionality from prior instrumental artifacts". This functionality, therefore, can be understood as a capability inherent to an artifact. USDL grounds the notion of service to the capability, similar to SoaML (despite the latter focuses on behavioral aspects). A particularity of USDL is to offer a set of specific concepts to represents aspects related to terms of provisioning and usage of services (e.g., *Licence*, and *PricePlan*). In terms of UFO-S, we could say that these specific concepts would be useful to represent the content of some service commitments, especially those one related to non-functional provisioning aspects.

RQ2 – Service participant (roles). In USDL, we can find the "Target Consumer" and the "Consumer" concepts, which seem to have correspondence to the "Target Customer" and "Service Customer" concepts in UFO-S, respectively. As a result, it is possible to represent the notion of a service offered to target customers, as well as, a service already hired by an specific service consumer. In the language, however, we could not identify a specific concept for representing what in UFO-S is referred as "Hired Service Provider". As a consequence, the concept of "Provider" in USDL seems to collapse both: the "Service Provider" (who offers a service towards target customers) and the "Hired Service Provider" (who is already committed to a specific service provisioning agreed with a service consumer). USDL offers other participant roles as a way of addressing a wider setting of a service ecosystem (e.g., business networks), such as [8]: *service owners* (cost center owners typically having governance responsibility of services); *stakeholders* (having regulatory, commercial or other designated interests in the service); and *intermediaries* (having specialist provisioning, such as a broker or cloud provider, beyond the original provisioning). With these other participant roles, we understand that new kinds of (social) relations arise among them. This requires, therefore, a specification of these relations and the dynamics among them, including, e.g., types of delegations between owners and intermediaries, and between intermediaries and providers). These aspects could be grounded on the notions of "open and close delegations" or even "legal relations" in UFO-S [10][33][35]. Finally, USDL presents a clearer distinction between intentional and non-intentional aspects, if compared to SoaML. It establishes a structural organization between participants (as organizational actors) concerned with service provisioning, delivery and consumption, and artifacts/resources used to in the services.

RQ3 and RQ4 – Service offering and service agreement descriptions. USDL offers a number of concepts that can be applied to build service specifications, which can be used along the service life-cycle, i.e., they are offered, negotiated and agreed. For example, the *ServiceLevel* concept captures a unique service level specification, and a set of service levels can be combined into service level profiles (e.g., as gold, silver, bronze profiles). Also, a particular service offer's price plan can be specified together with functional, interface and interaction details. All of these elements may be used in service descriptions, which are applied in service offer events, when, in terms of UFO-S, service commitments are published from the service provider towards target customers. Regarding the service agreement description, a service level profile also resembles the notion of a service level agreement "template" (e.g., WS-Agreement) [8]. This agreement "template", when instantiated, could be seen as a service agreement (in terms of UFO-S) between provider and consumer individuals.

RQ5 – Service delivery representation. USDL provides elements of the Interaction and Functional modules (interaction protocol, phase, function, etc.) that allow to represent the actions that service participants and resources perform along service delivery. Also, USDL offers a number of concepts, as aforementioned, that allow to specify non-functional requirements relative to service delivery actions (e.g., *PricePlan*, *UsageRight*, *TimeRestriction*), which represent, in some sense, the content of service commitments between service participants.

RQ6 – Service relation links. To support enterprises be aware of dependencies across their layers and beyond its boundaries (e.g, in service marketplaces), USDL offers a set of dependency relationships that goes beyond the traditional functional, compositional and bundling dependencies, namely: *includes*, *requires*, *enhances*, *mirrors*, *cansubstitute*, and *canconflict.* By means of these dependency relationships, it is possible to link service provisionings. From that, USDL widens the support to address third-party service provisioning and intermediation traditionally found in Web Services Architecture and SOA platforms [8]. Besides the service network established by means of these relationships, USDL intends to address an orthogonal and adjacent issue of service dependency, the set of resources applied to supporting a service's delivery. As discussed in Section III, the dependency relationships arise in service relations insofar provider and customer establish mutual commitments and put their resources available to each other. Differently from SoaML, which does not offer a well-defined approach for representing dependencies between service contracts, the aforementioned dependence relationships in USDL seems to constitute a strategy of explicitly representing the different dependencies among the service provisioning relations (e.g., agreements).

VII. SERVICE MODELING IN ARCHIMATE

A. ArchiMate Overview

ArchiMate is a visual service-oriented modeling language that provides a uniform representation through enterprise architecture layers [9]. The Business Layer depicts business services offered to customers, which are realized in the organization by business processes, having an actor responsible for it. The Application Layer depicts application services that support the business layer, and the application components that realize them. The Technology Layer depicts technology services (e.g., processing, and storage services) needed to run the software applications.

In ArchiMate, a *service* is "an explicitly defined exposed behavior" [9]. This concept is specialized in ArchiMate into business service, application service and technology service. The realization of services in all layers is described by means of behavior elements (business processes; business, application, and technology functions; business, application, and technology interactions, etc.). Services are provided and consumed through interfaces.

The service provisioning aspects are regulated by the *contract* element, which "represents a formal or informal specification (rights and obligations) of an agreement between a service provider and a service consumer" [9]. Services can be combined in *product*, which "represents a coherent collection of services" that is offered as a whole to (internal and external) customers [9].

A *business actor* is a business active element "capable of performing behavior". It can play a business role, by means of which it gains the "responsibility for performing specific behavior" [9]. An *application component* "represents an encapsulation of application functionality" and "exposes services, and makes them available through interfaces". A *node* "represents a computational or physical resource that hosts, manipulates, or interacts with other computational or physical resources" [9]. Application components and nodes are IT active element used as resources to support service provision. A *capability* is an ability that an active structure element (e.g., an organization, person, or system) possesses.

Figure 8 illustrates the usage of the aforementioned elements in a bookstore scenario.

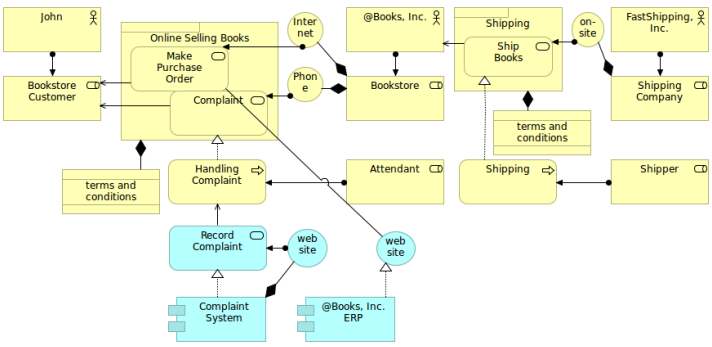

Figure 8: Bookstore scenario: ArchiMate example.

The "@Books, Inc." actor, as a bookstore, provides the "Make Purchase Order" and "Complaint" business services which are used by a "Bookstore Customer". The bookstore uses "Ship Books" service provided by the "FastShipping, Inc." company. All of these services are realized/provided by means of resources. E.g., the "Complaint" business service is realized by means of the "Handling Complaint" process that uses an application service realized by the "Complaint System" application component. (For the sake of simplicity, we do not represent the whole scenario).

B. Research Questions Analysis

RQ1 - Service characterization. In ArchiMate, a service "represents an explicitly defined exposed behavior" [9], which ultimately represents the capability of an individual (e.g., organization, person, application component, and processing node) to realize that behavior. This capability-based perspective, therefore, is reflected in the structure of the modeling language, as discussed below.

RQ2 – Service participant (roles). In ArchiMate, there is no clear distinction among the service participant roles in service relations (as discussed in UFO-S). The modeler can only represent (i) who *uses* the service, and (ii) who is *assigned to* it (not necessarily who is responsible for the service provisioning). As a consequence, a number of semantic misunderstandings may arise, e.g. [36][37]: (i) who are the target customers?, (ii) who are the actual customers hiring services?, and (iii) who are the providers hired to provider an specific service?. For addressing these "limitations", the modeler needs to create representation mechanisms. Some of these mechanisms can be found in [36] and [37], being structured in modeling patterns. In ArchiMate, there seems to be a clearer distinction between intentional and non-intentional service participants, if compared to SoaML. We understand that this is a result of the structure of the language that offers different constructs for modeling, e.g., business actors and business roles, as well as, application components, and devices. However, there are some aspects in the language that need attention. As discussed in [37], specially at Application and Technology layers, where the focus in on Application and Technology services provisioned and used by means of IT resources (e.g., a "File Hosting" application service realized by an application component and used by another software application), ArchiMate does not offer a clear way of representing the service provider and the service customer involved in the service relations, focusing solely on the non-intentional resources that contribute to the service delivery.

RQ3 and RQ4 – Service offerings and service agreement descriptions. In ArchiMate, there is no specific construct for representing service offerings (in terms of UFO-S). In case of service agreements/contracts, the language presents the "contract" construct, however, as described in detail in [36][37] we can find a number of service modeling limitations when trying to represent service agreements in ArchiMate (the "contract" construct seems to overload service offering types, service offerings, service

agreements). (A problem that was approached with specialized modeling patterns in [36][37]).

RQ5 – Service delivery representation. Since, in ArchiMate, the concept of service is based on the notion of behavioral capability, the behavioral modeling elements of this language (e.g., business process, and application function) are useful for representing the realization of this behavior. Thus, by means of business processes, e.g., one can represent the sequence of actions to be performed by human beings, social agents, and other organizational resources. However, despite ArchiMate offers ways of achieving a contract from the behavioral elements that realizes the correspondent service, similar to SoaML and USDL, ArchiMate does not provide a refined traceability between delivery actions and the service commitments that have motivated their performance (as discussed in [35]).

RQ6 – Service relation links. By means of some modeling elements in ArchiMate (e.g., serving, realization, and assignment relationships), the modeler can define ways of linking some elements (e.g., services, business roles, business process, and application components) in a service provisioning scenario. The serving relationship, e.g., represents a control dependency relationship. However, since ArchiMate does not offer direct relationships between *contracts*, it lacks a clear way of representing, in a service network, which contracts influence the other contracts.

VIII. Discussion

Service characterizations. A common aspect regarding service characterization in the three analyzed languages is the notion of "capability". SoaML, in turn, has a strong concern on behavioral specification of service contracts. Therefore, the notion of capabilities is central in these three languages, and all of them focus on the *capability-based service view*. As discussed in [10], UFO-S reveals an important distinction between (i) possessing a capability to perform certain actions or to produce certain outcome, and (ii) employing capabilities in order to fulfill service commitments. The former is not sufficient for characterizing service provisioning, since the capability of an organization to wash cars does not automatically make it a car wash service provider. For characterizing a (genuine) service relation, thus, there should be a set of service commitments from the provider towards costumer for "guaranteeing" the systematicity of applying such capabilities.

Service participants (roles). If we compare the service participants roles offered by the analyzed languages to those one defined by UFO-S, we can note some differences specially related to degree of expressiveness: whereas USDL presents at least five different participant roles, SoaML presents two (*Service Provider* and *Service Consumer*) and ArchiMate offers means of representing only who *uses* the service and who is *assigned to* the service. We could also identify some construct overload in those languages: in SoaML, *Service Provider* and *Service Consumer* roles overlap, respectively, Service Provider and Hired Service Provider, and Target Customer and Service Customer (in UFO-S); whereas in ArchiMate who *uses* the service could be interpreted as Target Customers as well as Service Customers with no distinction. One of the possible mechanisms to overcome these limitations is the definition of modeling patterns, as suggested in [36][37]. By being a lightweight approach, it does not require any language redesign. Finally, regarding the intentional characteristic of service participants in (genuine) service relations (in terms of UFO-S), we could identify that SoaML makes no distinction between intentional and non-intentional participants, differently from USDL and ArchiMate, which address some distinctions, specially at Business level. By making a clear distinction between theses aspects, it is possible to establish a clear representation between capabilities/resources used in service provisioning and agents committed to apply them.

Service Offering and Service Agreement Descriptions. We consider that the analyzed languages present a partial coverage concerning the representation of service relations established along service life-cycle (especially service offerings and agreements). This reflects the purpose of the languages: more focused on the capabilities used, and less focused on the social relations that guarantee the application of such capabilities. Regarding representation of offering and agreements elements, the analyzed languages offer concepts that could be applied for representing both: SoaML offers *service interface* and *service contract* constructs (focused on specifying behavioral concerns); USDL offers concepts (service level, profile, etc.) used to specify offering and agreements; and ArchiMate provides the *contract* construct, which may cover offerings and agreements indistinctively.

Service Delivery Representation. It is useful for describing service choreography and the interactions between provider and customers. The analyzed modeling languages provide useful, and, in some degree, equivalent elements/diagrams for representing service delivery actions types. However, none of these languages explicitly offers means to relate the (required) actions/interactions performed by providers and customers to the correspondent service commitments established among them and that ultimately motivate these actions/interactions. This would allow us to identify, e.g., (i) why some actions must be performed in a specific way, and (ii) which commitments may be broken if a specific action is not performed as agreed.

Service relation links. The analyzed languages present different support for representing service networks. SoaML offers the service architecture diagram, ArchiMate offers, specially, the *serving* relationship as a *control dependency relationship*, and USDL offers the richer set of dependency relationships (*includes*, *requires*, *enhances*, *mirrors*, *cansubstitute*, and *canconflict*) that allows to represent relationships between services and applied resources.

IX. Final Considerations

The diagnosis presented in this paper reinforces that the service phenomena is complex and encompasses a number of service perspectives. We have shown that emphasis on a particular service perspective may coincide with neglect of other perspectives. Even with the focus on a specific service perspective, it is important to be consist/aware to other related perspectives as a way of avoiding language inconsistence and favoring languages interoperability.

We have noticed little attention to important social aspects in service relations. Service offerings and service agreements, are prior to and regulate the existence of service as behavior and/or application of capabilities and resources, which occurs in service delivery phase. Offerings and contracts are key elements to address the social nature of commitments and claims established among intentional agents in highly developed (and formal) social contexts.

Specifically, we reveal some aspects concerning (i) the adopted service characterizations and its consequences; (ii) representation of service participant (roles) for better dealing with the different intentional agents involved in service relations; (iii) description of service offering and agreements; and (iv) representation of service delivery actions and their relationships to the correspondent motivational aspects.

As future work, we intend to propose improvements in the analyzed languages considering the diagnosis presented here. Some of our observations concerning ArchiMate already led to improvement proposals (see [37][33][36]). We intend to extend these improvements to the other two service modeling languages analyzed here, and design a conceptual framework for supporting design and analysis of service modeling languages in light of the aforementioned aspects.

ACKNOWLEDGMENT

This work has been supported by CNPq (407235/2017-5, 312123/2017-5), CAPES F. Code 001 (23038.028816/2016-41), FAPES (69382549) and FUB (OCEAN Project).

REFERENCES

[1] R. Ferrario and N. Guarino, "Commitment-based Modeling of Service Systems," in *Third Intern. Conference, IESS*, 2012, pp. 170–185.

[2] D. A. C. Quartel, M. W. A. Steen, S. Pokraev, and M. J. van Sinderen, "COSMO: A conceptual framework for service modelling and refinement," *Inf. Syst. Front.*, vol. 9, no. 2–3, pp. 225–244, 2007.

[3] P. P. Maglio, S. L. Vargo, N. Caswell, and J. Spohrer, "The service system is the basic abstraction of service science," *Inf. Syst. E-bus. Manag.*, vol. 7, no. 4, pp. 395–406, 2009.

[4] OASIS, "Reference Model for Service Oriented Architecture 1.0: OASIS Standard." OASIS, pp. 1–31, 2006.

[5] OMG, "Service oriented architecture Modeling Language (SoaML) Specification," 2012.

[6] M. P. Singh, A. K. Chopra, and N. Desai, "Commitment-Based Service-Oriented Architecture," *Computer (Long. Beach. Calif).*, vol. 42, no. 11, pp. 72–79, 2009.

[7] OMG, "Service oriented architecture Modeling Language (SoaML)," 2012. [Online]. Available: http://www.omg.org/spec/SoaML/..

[8] D. Oberle, A. Barros, U. Kylau, and S. Heinzl, "A unified description language for human to automated services," *Inf. Syst.*, vol. 38, no. 1, pp. 155–181, 2013.

[9] The Open Group, "ArchiMate 3.0 Specification," 2016.

[10] J. C. Nardi *et al.*, "A Commitment-based Reference Ontology for Services," *Inf. Syst.*, vol. 51, 2015.

[11] S. Gregor, "The Nature of Theory in Information Systems," *MIS Q.*, vol. 30, no. 3, pp. 611–642, 2006.

[12] R. Weber, *Ontological Foundations of Information Systems*. Melbourne: Coopers & Lybrand and the Accounting Association of Australia and New Zealand, 1997.

[13] M. zur Muehlen and M. Indulska, "Modeling languages for business processes and business rules: A representational analysis," *Inf. Syst.*, vol. 35, no. 4, pp. 379–390, 2010.

[14] M. Verdonck, et al. "Comparing Traditional Conceptual Modeling with Ontology-driven Conceptual Modeling: An Empirical Study," *Inf. Syst.*, vol. 81, pp. 92–103, 2019.

[15] J. Recker, M. Rosemann, P. Green, and M. Indulska, "Do Ontological Deficiencies in Modeling Grammars Matter?," *MIS* 35, 1, 2011.

[16] G. Guizzardi, "Ontology-based evaluation and design of visual conceptual modeling languages," in *Domain Engineering*, Springer Berlin Heidelberg, 2013, pp. 317–347.

[17] A. R. Hevner and S. Chatterjee, "Design Research in Information Systems," in *Design Research in Information Systems: Theory and Practice*, Springer Science - Business Media, 2010, p. 320.

[18] G. Guizzardi, "Ontological Foundations for Structural Conceptual Models," University of Twente, 2005.

[19] J. P. A. Almeida and G. Guizzardi, "An Ontological Analysis of the Notion of Community in the RM-ODP Enterprise Language," *Comput. Stand. Interfaces*, vol. 34, pp. 257–268, 2012.

[20] R. Guizzardi, X. Franch, and G. Guizzardi, "Applying a Foundational Ontology to Analyze Means-end Links in the i* Framework," in *IEEE Intern. Conf. on Research Challenges in Inf. Sciences*, 2012, pp. 1–11.

[21] P. S. Santos, J. P. A. Almeida, and G. Guizzardi, "An Ontology-based Analysis and Semantics for Organizational Structure Modeling in the ARIS Method," *Inf. Syst.*, vol. 38, no. 5, pp. 690–708, 2013.

[22] G. Guizzardi and G. Wagner, "Can BPMN be used for making simulation models?," in *Workshop on Enterprise and Organizational Modeling and Simulation*, 2011, pp. 100–115.

[23] C. L. B. Azevedo, et al. "An Ontology-Based Well-Founded Proposal for Modeling Resources and Capabilities in ArchiMate," in *17th IEEE International EDOC Conference*, 2013.

[24] S. Alter, "Service System Fundamentals: Work System, Value Chain, and Life Cycle," *IBM Syst. J.*, vol. 47.1, pp. 71–85, 2008.

[25] L. da S. Santos, "A Goal-based Framework for Semantic Service Provisioning," University of Twente, 2011.

[26] M. Dumas, J. O'Sullivan, M. Hervizadeh, D. Edmond, and A. H. M. Ter Hofstede, "Towards a Semantic Framework for Service Description," in *IFIP TC2WG26 Ninth Working Conf. on Database Semantics Semantic in ECommerce Systems*, 2001, pp. 277–291.

[27] P. R. Telang and M. P. Singh, "Business Modeling via Commitments," in *Service-Oriented Comp.: Agents, Semantics, and Eng.* 2009.

[28] G. Guizzardi, "On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models," in *Frontiers in Art. Intellig. and Appl., Databases and IS IV*, vol. 15, O. Vasilecas, J. Edler, and A. Caplinskas, Eds. Amsterdã: IOS Press, 2007, pp. 18–39.

[29] G. Guizzardi, G. Wagner, R. de A. Falbo, R. S. S. Guizzardi, and J. P. A. Almeida, "Towards Ontological Foundations for the Conceptual Modeling of Events," in *32th Intern. Conf. ER 2013*, pp. 327–341.

[30] T. Ruokolainen, "A Model-Driven Approach to Service Ecosystem Engineering," University of Helsinki, 2013.

[31] S. L. Vargo and R. L. Lusch, "Evolving to a new dominant logic for marketing," *J. Mark.*, vol. 68, pp. 1–17, 2004.

[32] B. Anderson, M. Bergholtz, and P. Johannesson, "Resource, Process, and Use – Views on Service Modeling," 2012, pp. 23–33.

[33] J. C. Nardi, R. de A. Falbo, and J. P. A. Almeida, "Revealing Service Commitments in Service-Oriented Enterprise Architecture," in *The Sixth Workshop on Service oriented Enterprise Architecture for Enterprise Engineering*, 2014, pp. 286–295.

[34] J. C. Nardi *et al.*, "Towards a Commitment-based Reference Ontology for Services," in *17th IEEE International Enterprise Distributed Object Computing Conference (EDOC)*, 2013, pp. 175–184.

[35] C. Griffo, J. P. A. Almeida, G. Guizzardi, and J. C. Nardi, "From an Ontology of Service Contracts to Contract Modeling in Enterprise Architecture," in *The 21st IEEE EDOC* 2017.

[36] J. C. Nardi, R. de A. Falbo, and J. P. A. Almeida, "An Ontological Analysis of Service Modeling at ArchiMate's Business Layer," in *18th IEEE International EDOC*, 2014, pp. 92–100.

[37] J. C. Nardi *et al.*, "Service Commitments and Capabilities Across the ArchiMate Architectural Layers," in *20th IEEE EDOC*, 2016.