Implementation and Application of a Well-Founded Configuration Management Ontology

Gleison Baiôco, Anilton Salles Garcia

Computer Science Department
Federal University of Espírito Santo, Vitória, Brazil
gleison.baioco@gmail.com and anilton@inf.ufes.br

***Abstract*—Information technology (IT) is becoming essential for organizations. In this context, IT management has evolved to include IT service management and governance, towards business driven IT management paradigms. Thus, the alignment between IT and business has been regarded as one of the leading factors to the effectiveness of such paradigms. Furthermore, contributions from management activities automation are mentioned. Following these evolution, configuration management plays a key role by providing accurate IT information to all related to management. However, due to this closed relationship among all management agents, interoperability between these agents has been characterized as one of the main research challenges in network and service management. This paper extends the contributions of [13], designing and implementing the conceptual models in implementation models. Thus, the main goal of this paper is to show how concepts modeled by ontologies can be implemented and applied in a computational system, allowing the conception of automated and semantically interoperable management solutions.**

***Index Terms* — Autonomic and Self-Management, IT Governance, IT Service Configuration Management, Ontology.**

I. Introduction

Information technology (IT) is becoming essential for organizations. In this context, IT management has evolved to include IT service management and governance, towards business driven IT management paradigms [1], as evidenced in [2].

According to [3], some factors are essential for the effectiveness of such paradigms. Among these factors, [3] highlight the alignment between IT and business. Furthermore, [3] [4] highlight the contributions that computer systems can provide towards an autonomic and self-management. However, as an emerging paradigm, these factors are still research challenges.

To support the development of the management discipline, frameworks such as ITIL [5] and standards such as ISO/IEC 20000 [6] have acquired notoriety and have been widely adopted in several organizations [1]. They provide a systematic approach for IT management, structuring the activities in an interrelated set of processes.

Among these processes, configuration management is the foundation for the entire management structure, promoting an effective and efficient IT control and providing accurate information to all involved with management. In IT service management and governance context, this closed relationship includes the interaction among main agents belonging to this domain such as business, people, processes, tools and technologies [5]. Therefore, the interoperability among these agents has been characterized as one of the major research challenges in network and services management [7].

According to [7], the use of semantic models eases interoperability between different management domains. To define semantic models in the management area, a formal definition of the knowledge used in this domain is required [7]. Ontologies, which are a formal and explicit specification of a shared conceptualization [8], can be used to establish a formal specification of the semantics of a certain domain [7]. Therefore, ontologies can be used to establish a common conceptualization about configuration management domain, in order to facilitate the exchange of information, including well-accepted semantics in management area [7]. Additionally, this conceptualization can be defined in a machine-readable format. In this way, the knowledge can be shared not only by human agents, but also by computer systems, allowing the processes automation [7]. However, despite the efforts of research initiatives, there is not an ontology that can be considered as a *de-facto* standard by the international community.

As discussed in [9], ontology development is a complex activity and thus to build high quality ontologies, it is necessary to adopt an engineering approach. Therefore, ontology development must use appropriate methods and tools. According to [10], ontology engineering, analogous to software and information systems engineering, should include phases of conceptual modeling, design and implementation. In a conceptual modeling phase, an ontology should strive for expressiveness, clarity and truthfulness in representing the domain conceptualization. These characteristics are fundamental quality attributes of a conceptual model responsible for the effectiveness of this model as reference framework for the tasks of reuse and semantic interoperability [11] [12]. The same conceptual model can give rise to different ontology implementations, in different languages such as OWL and RDF, in order to satisfy different non-

978-1-4244-6039-7/10/$26.00 ©2010 IEEE

functional requirements such as computational decidability, completeness etc [10]. Therefore, each phase shall produce different artifacts with different objectives and, as consequence, require different types of modeling languages and methods with specific characteristics. According to [12], semantic web languages, such as OWL and RDF, are languages focused on computation-oriented concerns and, therefore, improperly for ontology conceptual modeling phase. Philosophically well-founded languages, on the other hand, are committed with expressivity and conceptual clarity and thus suitable for this phase.

Following this approach, in [13] we propose a conceptual model of configuration management, in IT service management and governance context, based on foundational ontology. Besides addressing the need for semantic interoperability, this conceptual modeling is inserted in business driven IT management context, promoting integration between these two domains. Finally, this conceptual modeling subsidizes the management processes automation, since it derives implementation models, enabling not only the conventional use of tools to support the process, but also fomenting paradigms as autonomic and self-management. Thus, the main goal of the present paper is to extend the contributions of [13], showing how modeled concepts by ontologies can be implemented and applied in a computational system, allowing the conception of automated and semantically interoperable management solutions.

The remaining of this paper is organized as follows: section II presents a brief description of the conceptual model used in this paper to derive the implementation model; section III describes the implementation environment; section IV presents the configuration management implementation model; section V shows the application scenario; section VI presents the information inferences; section VII discusses related works; and, finally, section VIII presents the main conclusions of this paper and suggestions of future works.

II. Configuration Management Conceptual Model

As aforementioned, to develop high quality ontologies, it is necessary to employ an engineering approach, which requires appropriate methods and tools in each one of its phases. Thus, the development of the conceptual model proposed in [13] is driven by the SABiO method (Systematic Approach for Building Ontologies) proposed in [14]. This method prescribes an iterative set of activities leading the ontology development process. Furthermore, this conceptual model is based on a philosophically well-founded ontology, proposed in [11], named UFO (Unified Foundational Ontology).

According to ITIL and ISO/IEC 20000, the configuration management process is responsible for providing the necessary information concerning IT infrastructure to all other management processes. In IT service management and governance context, the configuration management process must be able to answer in a clear, precise and unambiguous way, questions such as: *what are the business processes and how do they relate to IT services and components? [5]*.

Whereas the main goal of the ontology proposed in [13] is to describe an IT service configuration management theory, independent on specific applications, the defined competency questions reflect this intention. In this sense, these competency questions lead to a mapping between IT and business related concepts. Competency questions, according to SABiO method, concern the ontology competency, i.e., the questions on which the ontology is proposed to answer.

For addressing these questions, the following sub-ontologies were developed: (i) business process, (ii) IT service, (iii) IT component, and, lastly, (iv) configuration item. These sub-ontologies complement each other to constitute the IT service configuration management ontology. They are connected by relations between their concepts as well as by formal axioms. These axioms allow: (i) a rich expressiveness that can not be reached only by means of graphical models; (ii) inferences, carried out by means of implementation models; (iii) evaluation of the truthfulness with the ontology purpose and restrictions; and (iv) inconsistency identification. In short, we did not show here all the ontology diagrams nor all its axioms, since they are not part of the present paper scope. Thus, Figure 1 illustrates an indicative part of one of the IT Service Configuration Management sub-ontologies.

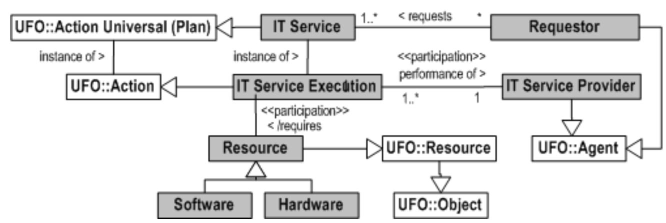

Fig. 1. IT service sub-ontology fragment [13].

The purpose of the conceptual model presented in [13] is to allow a common and shared understanding of the domain conceptualization between different involved agents such as business, people, processes, tools and technologies. A configuration management domain formal specification allows that modeled concepts are understood in a clear and explicit way, avoiding ambiguous and inconsistent domain interpretations by agents. It is important to highlight that UFO played a prominent role in the conceptual model development presented in [13]. It was helpful for building a conceptual model committed to maximize expressivity, clarity and truthfulness regarding configuration management domain. These characteristics are fundamental quality attributes of a conceptual model responsible for the effectiveness of this model as reference framework for the tasks of reuse and semantic interoperability [11] [12]. The conceptual model presented in [13], for example, makes a distinction: (i) between the participation of agents (e.g., IT Service Provider) and resources (e.g., hardware, software) in actions such as business process activity and IT service execution; (ii) between actions and their specific occurrences (e.g., business process activity and business process activity occurrence which denotes particular action that occur in specific time interval); (iii) between types of dependency relations (e.g., dependency relations between the business process activities occurrences as well as dependency relations between requestor and IT Service Provider that lead to a delegation relation) etc.

III. IMPLEMENTATION ENVIRONMENT

As discussed in [10], each ontology engineering phase requires the use of appropriate languages related to the context for which the model is designed. In the implementation phase, the choice of a language must be conducted by the end-application requirements. Therefore, due to the implementation models requirements, the used languages must possess intrinsic characteristics to computational issues. Thus, a language must allow, for example, that the implementation models are understandable when they are read by inference machines and computable in order to generate information in acceptable times. Inference accomplishment and information generation refer to formal languages that have both syntax and semantics unambiguous and exact formulations.

With Semantic Web advent it has grown the need to allow the semantic processing of the information in the Web and thus some ontology implementation languages have become standards adopted by W3C (World Wide Web Consortium). Among these languages, the RDF (Resource Description Framework) [15] and OWL (Web Ontology Language) [16] are highlighted. OWL stands out among these languages due to being more expressive in representation of ontology concepts and relationships. In order to enable the extension or reutilization of implementation models which were developed in this work, the OWL language was used.

To meet the various goals and requirements, OWL is divided into three sub-languages, namely OWL Lite, OWL DL and OWL Full. These sub-languages have incremental levels of expressivity. Considering factors such as decidability, completeness and expressiveness as fundamental requirements respecting configuration management process, due the role of this process to all others in service management, the OWL DL sub-language was adopted since it enables a greater degree of expressiveness, when compared to OWL Lite, while keeping computational guarantees such as completeness and decidability, features that are not guaranteed by the OWL Full.

In addition to OWL DL, the implementation of models also used the SWRL (Semantic Web Rule Language) [17]. SWRL language allows the representation of axioms which were defined in the conceptual models presented in [13] in an integrated way to the concepts and relations implemented by means of the OWL language.

Finally, for models implementation in OWL and axioms definition in SWRL, this work used the Protégé tool [18], an ontology editor that enables the integration of different languages such as OWL and SWRL in a same development environment.

IV. CONFIGURATION MANAGEMENT IMPLEMENTATION MODEL

Once defined the development environment, the implementation models were developed, as the structure presented in Table I.

TABLE I – IMPLEMENTATION MODELS IN OWL

Implementation Model	Description
UFO.owl	Implementation model derived from UFO conceptual model, which includes the UFO-A, UFO-B and UFO-C. This implementation model is used as the basis for IT Service Configuration Management implementation model development.
BusinessProcess.owl	Implementation model derived from Business Process conceptual model.
ITService.owl	Implementation model derived from IT Service conceptual model.
ITComponent.owl	Implementation model derived from IT Component conceptual model.
ConfigurationItem.owl	Implementation model derived from Configuration Item conceptual model.

Implementation models modularization, described in Table I, confers a more flexible structure with regard to reuse issues, since it does not require the reuse of the entire implementation model to other works that are only interested in a few concepts of the model. In additional, this modular structure provides better organization and, therefore, better understanding and visualization of implemented models, using the structure of the conceptual models defined in [13]. Figure 2 shows a graphical representation fragment of the IT Service Configuration Management implementation model.

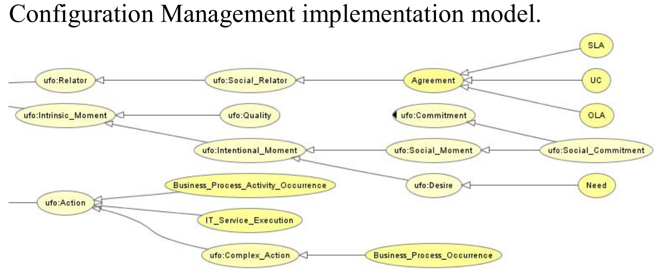

Fig. 2. Graphical representation of the IT Service Configuration Management implementation model.

As discussed in [13], the used languages in the conceptual models development are committed to conceptual clarity and expressiveness concerning to the modeled domain representation. These languages do not have computational commitments. On the other hand, the languages used in the implementation models development have computational commitments and, due to this, they do not have the same expressivity potential when compared to the languages employed in the conceptual modeling phase. Therefore, the development of the implementation models requires some considerations and definitions (i.e., a design) respecting expressivity restrictions of the adopted implementation languages. These restrictions refer to implementation models which do not represent faithfully the conceptual models from which they are derived. Regarding to the mapping issues, especially the aim for maintaining the expressivity reduction acceptable, the most relevant losses we have found are transforming all ontologically well-founded concepts and relations into OWL classes and properties, respectively. This mapping should consider the information contained in the notation used for the conceptual models development such as cardinality, transitivity, domain and range. Respecting cardinality and transitivity, in OWL it is not possible to represent both of them [16]. Thus, we have judged that representing cardinality restriction is more relevant to the implementation model presented in this work. In additional, to represent cardinality restriction in both directions we have used inverse relations. For example, the relation *Requests* is represented by the pair of relations that follows, namely *Requests* and *isRequestedBy*. However, according to [19], the use of inverse relations increases the reasoning complexity significantly. Therefore, they should be used only when necessary. Concerning domain and range, an issue that comes up regarding to how to organize and represent many generic relations. For instance, if a generic relation *isComposedOf* is created, it is not possible to restrict the domain and range. In this case, our choice was to use specific relations like *isComposedOf_BusinessProcessOccurrence_BusinessProcessActivityOccurrence* which is represented as a subrelation of a generic relation *isComposedOf*. Finally, with respect to SWRL restrictions, this language does not have either negation operators or existential quantifiers [17]. Besides that, SWRL might lead to undecidable implementation models. Nevertheless, this issue may be overcome by restricting the use of rules for handling only those which are DL-safe [20].

As an attempt to make the implementation tangible, consider the implementation of the competency question CQ4, originally defined in [13], that follows:

CQ4. What is the relationship between IT services and organization's business processes?

As defined in [13], a business process occurrence (BPO), instance of a business process (BP), is composed of business process activity occurrences (BPAO) which, in its turn, are instances of business process activity (BPA). Business processes and the activities that compose them are owned and performed by business (usually by business units - BU). This fact denotes a social commitment of the business unit to carry out its activities. This commitment, in its turn, has a propositional content, i.e., a goal. Thus, this agent has a commitment to perform a particular action that satisfies this goal. According to UFO ontology, an agent *a* depends on an agent *b* if, and only if, *a* has a goal *g* that it cannot achieve by itself (either by lack of capacity or by the fact that *g* contrasts with one of its other goals), and *b* can achieve *g*. Thus, in an automated environment, the business units' goals may represent needs that can be achieved by IT services [13]. This definition is formalized by the implemented axiom (CQ4.1) that follows.

(CQ4.1) Business_Process(?BP) ∧ Business_Process_Occurrence(?BPO) ∧ ufo:isInstanceOf_ComplexAction_Action(?BPO, ?BP) ∧ Business_Process_Activity(?BPA) ∧ Business_Process_Activity_Occurrence(?BPAO) ∧ ufo:isInstanceOf_Action_ActionUniversal(?BPAO, ?BPA) ∧ isComposedOf_BusinessProcessOccurrence_BusinessProcessActivityOccurrence(?BPO, ?BPAO) ∧ Business_Unit(?BU) ∧ performanceOf(?BPAO, ?BU) ∧ Need(?n) ∧ characterizes_Need_Requestor(?n, ?BU) ∧ ufo:Proposition(?p) ∧ ufo:isPropositionalContent(?p, ?n) ∧ ufo:Situation(?s) ∧ ufo:isPostState(?s, ?BPAO) ∧ ufo:satisfies(?s, ?p) ∧ IT_Service(?IT-Service) ∧ canAchieve(?ITService, ?n) → query:select(?BP, ?BPA, ?BU, ?IT-Service) ∧ query:orderBy(?BP, ?BPA)

Following the UFO ontology definition, these factors (lack of capacity or contrast with other goals) may be a reason why agent *a* decides to delegate such goal accomplishment to agent *b*. Thus, the UFO ontology defines that a delegation is associated with a dependency but as a material relation, a delegation is founded on something more than its connected elements. In this case, the connected elements are two agents (delegator and delegatee) and a goal (delegatum). The foundation of this material relation is the social relator (a commitment/claim pair) established between the two agents involved in this delegation. In other words, the UFO ontology states that when agent *a* delegates a goal *g* to agent *b*, besides the fact that *a* depends on *b* regarding *g*, *b* commits itself to accomplish *g* on behalf of *a*. Therefore, as defined in [13], the business' need may lead to a delegation. In this case, an IT service request, followed by its hiring, denotes an IT service delegation (ITSD). This delegation is associated with an agreement (social relator), which mediates the delegation process. As a type of social commitment to be fulfilled by the delegatee (in this case, an IT service provider – ITSP) this agreement causes the IT service execution (ITSE). This definition is formalized by the implemented axiom (CQ4.2). The conditional part of the axiom (CQ4.1) is also considered in the axiom that follows. It is not repeated in (CQ4.2) due to space limitations.

(CQ4.2) requests(?BU, ?IT-Service) ∧ Hired_ITSP(?ITSP) ∧ hires(?BU, ?ITSP) ∧ Agreement(?Agreements) ∧ mediates_Agreement_Requestor(?Agreements, ?BU) ∧ mediates_Agreement_ITSP(?Agreements, ?ITSP) ∧ IT_Service_Execution(?ITSE) ∧ isInstanceOf_ITSE_ITService(?ITSE, ?IT-Service) ∧ isCausedBy_ITSE_Agreement(?ITSE, ?Agreements) ∧ IT_Service_Delegation(?ITSD) ∧ ufo:isAssociatedTo_Delegation_SocialRelator(?ITSD, ?Agreements) ∧ ufo:isDelegatorOf(?BU, ?ITSD) ∧ ufo:isDelegateeOf(?ITSP, ?ITSD) → query:select(?BP, ?BPA, ?BU, ?IT-Service, ?ITSE, ?ITSP, ?Agreements) ∧ query:orderBy(?BP, ?BPA)

As described in [13], the IT department receives the delegations from the business and provisions the necessary resources to achieve the IT services hired by the business. According to UFO ontology, as we say that a certain agent can achieve a goal, this means that such agent is able to do it itself or can delegate to another agent that can accomplish it on its behalf. Therefore, when a provider receives a delegation through a service level agreement this provider analyzes the service delegated and, if necessary, delegates such service to other entities such as internal service providers (e.g., IT infrastructure department) or external service providers (e.g., suppliers). This definition is formalized by the implemented axiom (CQ4.3) that follows.

(CQ4.3) IT_Service_Execution(?Complex-ITSE) ∧ ufo:Complex_Action(?Complex-ITSE) ∧ IT_Service_Execution(?ITSE) ∧ ufo:isComposedOf_CAction_Action(?Complex-ITSE, ?ITSE) ∧ Agreement(?Agreements) ∧ isCausedBy_ITSE_Agreement(?ITSE, ?Agreements) ∧ IT_Service_Delegation(?ITSD) ∧ ufo:isAssociatedTo_Delegation_SocialRelator (?ITSD, ?Agreements) ∧ IT_Service_Provider(?Delegator) ∧ ufo:isDelegatorOf(?Delegator, ?ITSD) ∧ IT_Service_Provider(?Delegatee) ∧ ufo:isDelegateeOf(?Delegatee, ?ITSD) → query:select(?Complex-ITSE, ?ITSE, ?Delegator, ?Delegatee,

?Agreements) $\wedge$ query:orderBy(?Complex-ITSE, ?ITSE)

Also, consider the implementation of the competency question CQ3, originally defined in [13], that follows:

CQ3. What is the relationship between services and IT components, such as hardware and software?

In [13] is defined that an IT service may require resources, such as hardware and software, to be executed. However, it is important to highlight that if an IT service execution requires a software and this software is processed by a hardware then this IT service execution also requires this hardware, as formalized by implement axiom CQ3 that follows:

(CQ3) IT_Service_Execution(?ITSE) $\wedge$ Software(?Sw) $\wedge$ requires(?ITSE, ?Sw) $\wedge$ Hardware(?Hw) $\wedge$ process(?Hw, ?Sw) $\rightarrow$ query:select(?ITSE, ?Sw, ?Hw) $\wedge$ query:orderBy(?ITSE)

The rule CQ3 denotes, at least, two observations with regard to ontological models capabilities. On the one hand the ability to represent the relations, restrictions etc of the universe of discourse. On the other hand the capability of knowledge acquisition. As mentioned earlier, the rule CQ3 infers new relations (in this case the hardware required by IT service execution) based on pre-existing information (in this case the software required by IT service execution and processed by hardware).

V. Application Scenario

This section describes an application scenario as concept proof of the developed ontology. Furthermore, through this application scenario, it is possible to understand how the implementation models can be applied in a computational environment. Due to the context of this work, namely business driven IT management, the adopted application scenario is based on a model proposed by an IT service management best practices library, namely ITIL. Figure 3 shows this scenario.

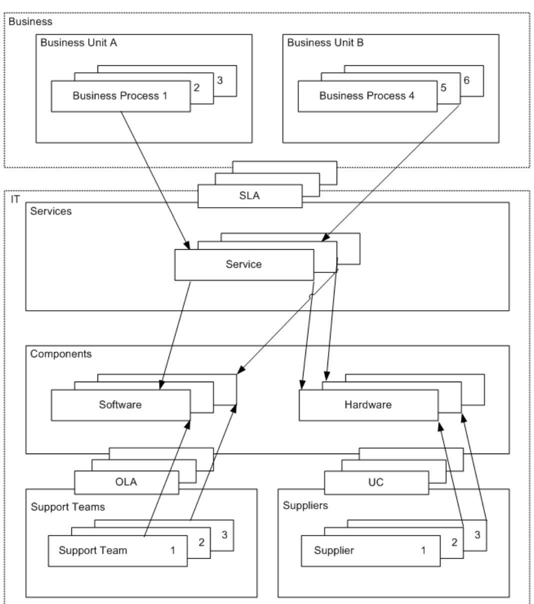

Fig. 3. Application scenario based on [5].

As described by ITIL, IT infrastructure purpose is to allow the business reach its goals through the efficient and effective use of computer systems. Therefore, the business' needs should be translated into the technologies that support them. In this context, the main objective of the IT service management is to ensure that IT services are aligned with business' needs, actively supporting these needs. As illustrated in Figure 3, IT service management should consider the following entities: (i) Business Process; (ii) IT Service; (iii) SLA (Service Level Agreement); (iv) IT Component such as hardware and software; (v) Support Team; (vi) OLA (Operational Level Agreement); (vii) Supplier; (viii) and, lastly, UC (Underpinning Contract).

In additional, an effective service management must consider not only each aforementioned entity in isolation, but must also consider the interrelationships between each entity as well as their interactions and interdependencies in order to provide an effective and comprehensive solution that satisfies business' needs [5]. Figure 3 illustrates the situation where the business is divided into business units. Business units have business processes and are responsible for these processes. When a business unit needs IT services to carry out its activities, this business unit delegates its necessity to IT department. This delegation is mediated through an agreement, namely SLA. In this context, the IT department subsumes the IT service provider role, more specifically, internal IT service provider, while business unit subsumes the client role of this provider. IT services are based on the use of information technology, composed of components such as hardware and software that are provided by internal support teams or external suppliers. Thus, IT department, responsible for the IT service provision to business, delegates the necessary resources provision to internal support teams or external suppliers. In this context, IT department delegates IT sub-service to support teams, mediated through agreements such as OLA, or to suppliers, mediated through agreements such as UC.

VI. Information Inference

There are numerous contributions of ontology engineering to build applications more flexible, autonomous, intelligent and, above all, interoperable. Although in a different area, the work of [21] presents an application in order to interpret the results of an electrocardiogram which the use of an ontology provides a graphic simulation of the behavior of an individual's heart and the correlation of behavior of the heart with known diseases. As aforementioned, configuration management provides the necessary information about the IT infrastructure to all other management processes. Thus, this section aims to demonstrate how the models implemented in the present paper may be applied in a computing environment. Furthermore, the results provide a concept proof of the developed ontology. For testing purposes, we described individuals consistent with the case study presented earlier. The information inferences were made in Protégé with the aid of the rules inference system named Jess [22].

First part includes the mapping between the business process, the activities that compose it, as well as the business unit responsible by the process. In additional, this inference relates the IT services that can achieve the needs of the business unit. Figure 4 shows the results of this inference.

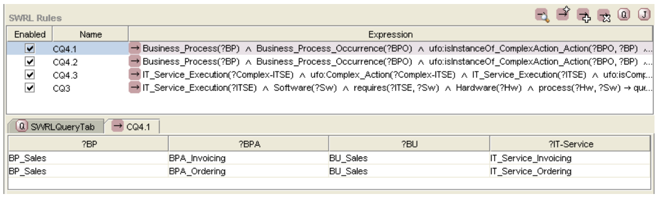

Fig. 4. Management information inference inherent to Business.

To produce the result shown in Figure 4 the system inferred the business processes of the organization, the activities that make up these processes as well as the business units responsible for each process. In additional, the system inferred the IT services that can support every business process activity, based on the needs of each unit. This inference was performed by implemented axiom CQ4.1. As shown in Figure 4, the *BP_Sales* business process is composed by *BPA_Ordering* and *BPA_Invoicing* activities. The responsible for these activities is the *BU_Sales* business unit. Based on the needs of that unit, the *IT_Service_Ordering* IT service can support the *BPA_Ordering* activity as well as the *IT_Service_Invoicing* IT service can support the *BPA_Invoicing* activity. Information of this nature are fundamental to processes such as Service Level Management that refer to the Configuration Management requiring information concerning IT services that can achieve customers' needs.

By means of the information inferred in the first part, the second part includes the services that the business has effectively hired from the IT department, establishing a mapping between the business processes of the organization, their activities, the business units responsible for these activities, the hired IT services, the hired IT department, the IT service execution and the established agreement. In this context, the business unit subsumes the client role and the IT department subsumes the IT service provider role. The agreement established between the business unit and IT department is represented by the SLA. Figure 5 shows the result of this inference. This inference was performed by implemented axiom CQ4.2. Figure 5 shows, for example, that the *IT_Service_Ordering* IT service was hired to support the *BPA_Ordering* activity, which belongs to the *BP_Sales* process. This service delegation is governed by an agreement between the *BU_Sales* business unit and the *IT-Department* service provider, namely, the *SLA_Ordering* agreement.

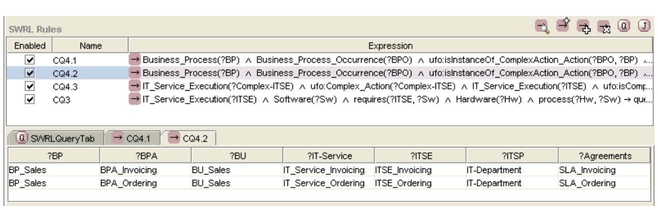

Fig. 5. Management information inference inherent to IT Service.

As aforementioned, the IT department receives the delegations from the business and provisions the necessary resources to achieve the IT services hired by the business. Thus, to produce the result shown in Figure 6, the system inferred the support groups hired by the IT department, the IT service executions delegated to each support group, the agreements established between the IT department and other support groups. This inference was performed by implemented axiom CQ4.3. Figure 6 shows, for instance, that the *ITSE_Ordering* IT service execution, which represents an occurrence of the *IT_Service_Ordering* IT service, consists of *Execution_Ordering* and *Execution_Ordering_Printing* executions. These executions were delegated by *IT-Department* to *IT-System-Department* and *IT-Support-Department* support groups, respectively. The *Execution_Ordering* execution was delegated by means of the *OLA_Ordering* operational agreement. The *Execution_Ordering_Printing* execution, in its turn, was delegated by means of the *OLA_Ordering_Printing* operational agreement.

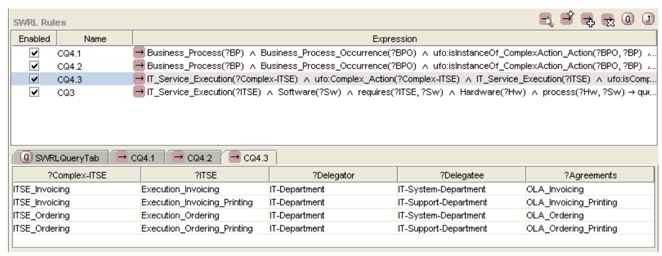

Fig. 6. Management information inference inherent to IT service provision.

As mentioned earlier, the IT services are based on IT components, such as hardware and software. Thus, the implemented axiom CQ3 maps the IT components required by each IT service execution. Figure 7 shows, for example, that the *Execution_Ordering* IT service execution uses the *System_Ordering* system, processed by the *Server_System_Sales* server. The *Execution_Ordering_Printing* IT Service execution, in its turn, uses the *System_Ordering_Printing* system, processed by the *Server_Printing* server.

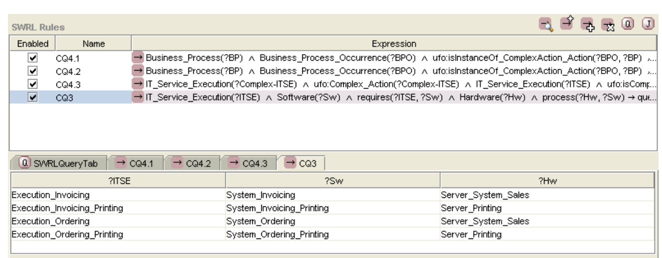

Fig. 7. Management information inference inherent to IT service execution and IT component utilization.

Thus, the quarter concludes the mapping between business and IT. This mapping is fundamental to other management processes. For example, configuration management process of this case study is able to correlate and determine that a downtime particular event on the *Server_System_Sales* server affects *System_Ordering* and *System_Invoicing* systems, which are used, respectively, by *Execution_Ordering* and *Execution_Invoicing* service executions. These executions belong, respectively, to *IT_Service_Ordering* and *IT_Service_Invoicing* IT services, that support activities related to *BP_Sales* business process. This correlation, provided by the implementation model, is the basis for processes such as Event Management. Identified the failure point, in this case, the *Server_System_Sales* server, Configuration Management process can provide workaround informations to the Incident Management process in order to restore the IT service as quickly as possible, until the Problem Management process identifies the root cause of the problem. This activity of identifying the root cause can also be supported by the Configuration Management, for example, by providing information about the reason for the server unavailability. With respect to other processes such as Change Management, the implementation model presented in this case study, for example, provides vital information for decision making, being able to answer, for instance, what impact a particular change in a computational system can offer to business. As illustration, a change in the *System_Invoicing* system causes direct consequences on the *BPA_Invoicing* activity. Similarly, processes such as Security Management may be supported by the model presented in this case study, for example, relating to what impact a particular vulnerability in a computational system can offer to business. This support is also provided to other processes such as Availability Management, Capacity Management etc. Finally, the support given to the Financial Management is illustrated, in which the organization can relate to the costs each business unit, business process activity, or even the business itself, represent with respect to the use of the IT resources.

VII. Related Work

Several works claims that ontologies are a promising way to achieve interoperability between different management domains. However, ontology-based model and formalization of the IT Service Configuration Management still is a research challenge. Regarding limitations, it should be mentioned that ontologies are under development in the management area. In fact, the technology is not yet mature, and there is not an ontology that can be considered as a *de-facto* standard by the international community [7].

In general, research initiatives do not employ an systematic approach in the ontologies development. According to [9], the absence of a systematic approach, ignoring appropriate methods, techniques and tools, makes the ontology development process an art rather than an engineering activity. According to [10], the ontology engineering, as well as software and information systems engineering, should include phases of conceptual modeling, design and implementation. Each phase has its specific objectives and, therefore, requires appropriate languages to achieve these goals. However, most times, the research initiatives are engaged to the use of technologies and tools such as OWL and Protégé. On the one hand, these technologies and tools are used in conceptual modeling phase, which might result in various problems of semantic interoperability, as demonstrated in [12]. On the other hand, these technologies and tools are used in the implementation phase, however ignoring antecedent phases such as conceptual modeling and design. Consequently, they are obliged to rely on low expressivity models. Furthermore, in their majority, such initiatives propose the use of these technologies and tools to formalize the network management data models (e.g., MIB, PIB, CIM schema etc). Data models are closer to the underlying protocols used to transport the management information and the particular implementation in use. Information models, in contrast, work at a conceptual level. They are intended to be independent of any particular implementation or management protocol. Working at a higher level, information models usually provide more expressiveness [7]. More information about the difference between data models and information models can be found in [23]. Following this approach, in [24], the authors propose the integration of all concepts that actually belongs to different network management data models (e.g., MIB, PIB, CIM schema etc) in a unique model, formalized by ontology languages such as OWL. In a more specific context, in [25], the author presents an ontology based network configuration management system. In his work the ontology was developed according to the MIB data model concepts. As MIB is limited to describing a single system, a view on the entire infrastructure and the relationships between its components is not supported by the model. In practice, this gap is often filled by functionality provided by SNMP-based network management tools (management platforms) that, for example, support viewing network topologies. Also, an MIB is concerned only with the operational state of a resource. An MIB is not designed to support tracking the lifecycle status of a resource [26].

Moreover, the research initiatives are characterized with specific purposes regarding specific applications in information systems that restrict their conceptualizations. In [27], a common ontology-based intelligent configuration management model for IP network devices is presented, aiming the use of ontology in automation of this process.

Finally, there are approaches that seek to establish semantic interoperability among existing ontologies by means of ontological mapping techniques, as evidenced in [28]. However, it is not scope of such approaches, to develop an ontology, but rather, to integrate existing ontologies.

Therefore, considering the main challenges discussed in this paper and analyzing the surveyed works, it is observed that there are gaps to be filled, as noted by [7]. In other words, was not identified, among the investigated works, an ontology that is consistent with the IT service management and governance context, i.e., that maps the business and IT concepts and relationships. Furthermore, among the examined works was not identified an ontology that is committed to the expressivity and conceptual clarity and free of a specific application scenario, essential factors for semantic interoperability as well as the widespread adoption and reuse of model. In this scenario, in [13] we propose a conceptual model of the IT service configuration management domain, based on foundational ontologies, committed to maximize the expressiveness, clarity and truthfulness of the concepts belong to this domain and independent of a specific context of application. The purpose is to promote interoperability between the various involved agents. Continuing this work, the goal of the present paper is to design and implement the models developed in [13], demonstrating how these conceptual models can be derived and implemented on computational systems with a view to process automation.

VIII. CONCLUSION

In this paper an implementation model of the IT Service Configuration Management domain was presented. This implementation model is derived from a conceptual model based on a philosophically well-founded ontology, named UFO (Unified Foundational Ontology). Despite the inherent limitations of the languages used, with regard to the expressivity of concepts and relations representation, which limits the faithful implementation of the conceptual models, implementation models allowed to verify the adherence of conceptual models concerning the universe of discourse and, thus, to perform a concept proof of the conceptual model developed in [13]. Furthermore, considering the automated support as an important factor for the management processes success, this work presented how implementation models, derived from conceptual models, can be used in computational systems, supporting IT management activities in an automated manner.

Regarding adherence aspect, it is worth noting the ability of the model to map the customer aspects, inherent to the business, and the provider aspects, intrinsic to IT. This capability validates the model for its ability to respond to major issues for business driven IT management: *what are the business processes and how do they relate to IT services and components?*

The ontology engineering approach allowed developing conceptual models, that are application-independent artifacts, which can be used as a reference ontology in further phases of a development process for deriving a number of implementation models addressing different end-application purposes. In addition, it was shown how the use of ontologies, foundational ontologies in particular, can contribute to building models consistent and unambiguous, addressing semantic interoperability problems, preponderant factor with regard to configuration management process. Moreover, the concepts inherited from foundational ontologies are important to promote the development of autonomic and self-management paradigm, by making explicit, for computational agents, concepts expressing everyday reality issues, such as intentions, goals, actions etc.

Finally, in addition to future work discussed in [13], is important to consider the contributions that artificial intelligence techniques can provide when applied with the implementation models developed in this work, in order to fomenting paradigms as autonomic and self-management.

REFERENCES

[1] Moura, A., Sauve, J., Bartolini, C., "Business-Driven IT Management - Upping the Ante of IT: Exploring the Linkage between IT and Business to Improve Both IT and Business Results", IEEE Communications Magazine, vol. 46, issue 10, October 2008.

[2] Pavlou, G., Pras, A., "Topics in Network and Service Management", IEEE Communications Magazine, 2008.

[3] Moura, A., Sauve, J., Bartolini, C., "Research Challenges of Business-Driven IT Management". In: BDIM, 2007.

[4] Tosic, V., "The 5 C Challenges of Business-Driven IT Management and the 5 A Approaches to Addressing Them". In: BDIM, 2006.

[5] Office of Government Commerce (OGC), "ITIL Core Books", 2007.

[6] International Organization for Standardization, "ISO/IEC 20000", 2005.

[7] Pras, A. et al., "Key research challenges in network management", IEEE Communications Magazine, vol. 45, issue 10, October 2007.

[8] Gruber, T. R., "A Translation Approach to Portable Ontology Specifications", Knowledge Acquisition, vol. 5, issue 2, June 1993.

[9] Falbo, R. A., "Knowledge Integration in an Software Development Environment" (in portuguese), Doctoral Thesis, COPPE/UFRJ, 1998.

[10] Guizzardi, G., "On Ontology, ontologies, Conceptualizations, Modeling Languages, and (Meta)Models", In Frontiers in Artifical Intelligence and Applications, Databases and Information Systems IV, 2007.

[11] Guizzardi, G., "Ontological Foundations for Structural Conceptual Models", Ph.D. Thesis, University of Twente, The Netherlands, 2005.

[12] Guizzardi, G., "The Role of Foundational Ontology for Conceptual Modeling and Domain Ontology Representation", In 7th International Baltic Conference on Databases and Information Systems, 2006.

[13] Baiôco, G., Costa, A. C. M., Calvi, C. Z., Garcia, A. S., "IT Service Management and Governance – Modeling an ITSM Configuration Process: a Foundational Ontology Approach". In: BDIM, 2009.

[14] Falbo, R. A., "Experiences in Using a Method for Building Domain Ontologies". In: SEKE, 2004.

[15] RDF, "RDF primer", W3C Recommendation. In: http://www.w3.org/TR/rdf-primer/.

[16] OWL, "OWL Web Ontology Language Reference", W3C Recommendation. In: http://www.w3.org/TR/owl-ref/.

[17] Horrocks, I. et al., "SWRL: A Semantic Web Rule Language Combining OWL and RuleML", DAML. In: http://www.daml.org/2003/11/swrl/.

[18] Protégé, "The Protégé Ontology Editor and Knowledge Aquisition System". In: http://protege.stanford.edu/.

[19] Rector, A., Welty C., "Simple part-whole relations in OWL Ontologies", W3C Recommendation. In: http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhole.

[20] Motik, B., Sattler, U., Studer, R., "Query Answering for OWL-DL with Rules", Journal of Web Semantics, Vol. 3, No. 1, pp. 41-60, 2005.

[21] Gonçalves, B. et al., "An Ontology-based Application in Heart Electrophysiology: Representation, Reasoning and Visualization on the Web". In: ACM Symposium on Applied Computing, 2009.

[22] Jess, "Jess: the Rule Engine for the JavaTM Platform". In: http://www.jessrules.com/.

[23] Pras, A., Schönwälder, J., "On the Difference between Information Models and Data Models", RFC 3444, January 2003.

[24] Lopez de Vergara, J.E., Villagra, V.A., Asensio, J.I., Berrocal, J., "Ontologies: Giving Semantics to Network Management Models", IEEE Network, special issue on Network Management, Vol. 17, No. 3, 2003.

[25] Santos, B.C.L., "O-bCNMS: An Ontology-based Network Configuration Management System" (in portuguese), Master Dissertation, Computer Science Department, Federal University of Espírito Santo, Brazil, 2007.

[26] Brenner, M., Sailer, M., Schaaf, T., Garschhammer, M., "CMDB - Yet Another MIB? On Reusing Management Model Concepts in ITIL Configuration Management". In: DSOM, 2006.

[27] Xu, H., Xiao, D., "A Common Ontology-Based Intelligent Configuration Management Model for IP Network Devices". In: ICICIC, 2006.

[28] Ralha, C. G., Gostinski, R., "A methodological framework for business-IT alignment", In: BDIM, 2008.

ULRs were last checked in January, 2010.