Using Ontologies for Enterprise Architecture Analysis

Gonçalo Antunes [†], Marzieh Bakhshandeh[†], Rudolf Mayer[‡], José Borbinha[*†], Artur Caetano[*†]
goncalo.antunes@ist.utl.pt, marzieh.bakhshandeh@ist.utl.pt, rmayer@sba-research.at,
jose.borbinha@ist.utl.pt, artur.caetano@ist.utl.pt
[*]Instituto Superior Técnico, Technical University of Lisbon, Portugal
[†]INESC-ID - Information Systems Group, Lisbon, Portugal
[‡]Secure Business Austria, Vienna, Austria

***Abstract*—Enterprise architecture aligns business and information technology through the management of different elements and domains. An architecture description encompasses a wide and heterogeneous spectrum of areas, such as business processes, metrics, application components, people and technological infrastructure. Views express the elements and relationships of one or more domains from the perspective of specific system concerns relevant to one or more of its stakeholders. As a result, each view needs to be expressed in the description language that best suits its concerns. However, enterprise architecture languages tend to advocate a rigid "one-model fits all" approach where an all-encompassing description language describes several architectural domains. This approach hinders extensibility and adds complexity to the overall description language. On the other hand, integrating multiple models raises several challenges at the level of model coherence, consistency and traceability. Moreover, EA models should be computable so that the effort involved in their analysis is manageable. This work advocates the employment of ontologies and associated techniques in EA for contributing to the solving of the aforementioned issues. Thus, a proposal is made comprising an extensible architecture that consists of a core domain-independent ontology that can be extended through the integration of domain-specific ontologies focusing on specific concerns. The proposal is demonstrated through a real-world evaluation scenario involving the analysis of the models according to the requirements of the scenario stakeholders.**

***Index Terms*—enterprise architecture, ontology, analysis, ArchiMate, OWL.**

I. Introduction

Enterprise architecture (EA) is defined by Lankhorst as "a coherent whole of principles, methods, and models that are used in the design and realization of an enterprise's organizational structure, business processes, information systems, and infrastructure", providing a basis for business-IT alignment [1]. Such alignment is typically achieved through the creation of models and other artifacts describing different aspects of the organisation, including business and IT elements. Managing the dependencies between and within the different models and other artifacts produced during the application of architecture development methods is crucial for supporting the communication between the different stakeholders, and the alignment and consistency between the development artifacts and other products [2], [3].

Despite the efforts for developing comprehensive approaches to architecture, it has been recognized that "one model fits all" approaches are not enough for addressing the specifics of each organization [4], [5]. Different organizations have different business needs and play in different markets, thus having different demands in terms of the information to be contained and to be extracted from EA descriptions. As such, it is still a challenge to create an architecture description that is representative of the specifics of each organization, a fact which is attested by situational approaches to enterprise architecture [6].

The integration and extension of different models and underlying meta-models is common in different practices, such as tool interoperability and integration [7], [8]. However, such approaches bring challenges at the level of coherence and model consistency [9], [10]. In terms of traceability, it becomes difficult to maintain links among elements from the different models, which is a problem that gets exacerbated as the models evolve. Consequently, model consistency is also hindered, namely at the level of the multiple viewpoints when supported by different tools. Moreover, the assessment and validation of the different artifacts becomes more challenging [11], as it usually implies using a set of specific tools and languages.

As a result, there is a need for an extensible approach to EA that ensures meta-model level coherence, which concerns encoding rules on the meta-model, and supports model and cross-model verification, which concerns the evaluation of the conformance of the models to the rules specified in the respective meta-models. Moreover, EA models should be computable so that the effort involved in their analysis is manageable.

This paper proposes an approach based on ontologies as a means to specify knowledge and reason about it. Ontologies and associated techniques are increasingly being recognized as valuable tools in the EA domain, as witnessed in [12], [13], [14], [15], [16]. In fact, there is a wide body of knowledge from which EA practice can benefit, including ontology matching [17], which can improve model creation, extension and validation, through an explicit semantic account of the concepts used in the different models when compared to approaches without an ontological foundation [18].

The main contributions brought by this proposal can be described as:

- improved extensibility and expressiveness of the enter-

prise architecture, through the management of the inclusion of new domain-specific meta-models in a standard way, with the aim capturing specific aspects of organisations;
- improved enforcement of meta-model coherence, through the addition of axioms to the ontologies enforcing semantic rules implicitly defined in the model specifications;
- improved meta-model conformance verification of models, since models specified according to a determined meta-model can be verified against the semantic rules specified in the ontology through the use of reasoners, allowing for the verification of logical inconsistencies present in models;
- and improved analysis for decision-making purposes, through the usage of computational inference and querying mechanisms, allowing for better information retrieval and processing.

This work is organized as follows. In section II we describe some of the current issues existing in enterprise architecture practice which this work intends to approach. Section III presents a proposal based on ontology techniques for improving the extension, analysis and conformance or architecture models. Section IV evaluates the proposal through its application to a concrete scenario. Finally, Section V concludes the paper and provides directions for future work.

II. Problem Statement

According to the ISO/IEC/IEEE 42010 [19], the usage of multiple views is fundamental in an architecture description. A system has multiple stakeholders, each with specific interests on the system and with different information needs. An architecture description should be an aggregation of multiple views, materialized in a set of models, formulated according to viewpoints that express the concerns of the stakeholders of the system-of-interest. In this way, architecture functions as a communication tool between different stakeholders, as each is presented with is own view on the system of interest.

Although the recommended practice recognizes the need for multiple views on the system, and even goes as far as defining correspondence rules (i.e., dependencies) between architecture description elements, the truth is that it is still a challenge to maintain these dependencies when multiple independent meta-models and models are involved. Languages such as ArchiMate [20] try to be as comprehensive as possible up to a certain level of abstraction, providing a meta-model that approaches the different layers of an organization, but the fact is that the specific aspects of each organization and its business still require extension and specialization of such models, as it is recognized in ArchiMate's specification. Thus, it is a challenge to be able create an architecture description that is truly representative of the specifics of each organization and, at the same time, being able to make the management of the dependencies between its different elements.

Multiple meta-models often means multiple tools and multiple validation mechanisms, making it especially hard to ensure the consistency in and across models and conformance with the respective meta-models, particularly when this verification has to be performed by manual means. The automatic or semi-automatic validation of the conformance of the models to the meta-models might be available or not, depending on whether the models are fully computable (abstract syntax and semantics) or not, or on the existing tool support.

EA should also support governance and decision making [1], [21]. As such, the ability to analyse the models for supporting decision making is also desirable [22]. Management should be able to obtain as much useful information as possible from the knowledge contained in the models, which might reach a great level of complexity when elaborated with detail [23]. Such analysis can of course be made without any automatism, however it can be difficult to perform in-depth analysis in such complex scenarios.

As such, creating computable representations for enterprise architecture models is a relevant problem. The combination of the computable models along with the enforcement of the dependencies brings benefits for EA, such as enhanced retrieval and management of information, and the use of information processing and analysis mechanisms. One example of such benefit is dependency analysis, which can be used for analysing the impact of changes on the business on the IT infrastructure and vice-versa.

Ontologies and associated techniques bring several benefits which potentially can be employed in solving the aforementioned issues[24]:

- Communication: ontologies can be used for ensuring interoperability between systems, humans, and systems and humans. Moreover, they can be use for uniquely identifying and disambiguating concepts through formal semantics, facilitating knowledge transfer.
- Computational inference: ontologies allow for deriving implicit facts and be used for analysis and detection of logical inconsistencies.
- Reuse and organization of knowledge: ontologies allow for systematic domain descriptions and for the reuse of knowledge models in new applications.

As such, ontologies have been adopted in different domains for solving different problems [24]. Integration, semantic search and retrieval, annotation, software engineering, and knowledge representation are some of the areas where the application of ontologies for solving problems has been cited. Moreover, the use of ontologies and associated techniques in EA is increasing, with proposals of EA based on ontologies for improving the models and their semantics [12], [13], [14], [15], [16].

The aim of this paper is thus advocating for the use of ontologies and associated techniques in EA for contributing in solving the aforementioned issues. Thus, an architecture based on the use of ontologies is proposed, so that advantage can be taken from the associated mapping and analysis techniques, with the purposes of improving the extensibility with domain-specific aspects, enforcing meta-model coherence through the addition of coherence rules, verifying the conformance of the models to the meta-models, and improving the analysis for reporting and decision-making.

III. Ontology-based Enterprise Architecture

In order to ground our proposal in good practice and allow for a structured and extensible approach, a set of architecture principles were defined. An architecture principle can be described as "a declarative statement that normatively prescribes a property of the design of an artifact, which is necessary to ensure that an artifact meets its essential requirements" [25]. As such, the following design principles were considered:

- Concern orientation: The architecture shall represent the concepts necessary and sufficient to address an explicit set of modeling concerns. This means that the model shall be derived from the questions that need to be addressed and to provide answers to those questions. This also means that the model shall not support any concepts that are not explicitly derived from stakeholder concerns.
- Expressiveness: The architecture shall be able to represent the domain concepts without ambiguity. This entails defining the minimum set of types and relationships to describe a domain
- Extensibility: The architecture must cope with extensions because context modeling entails using multiple concurrent perspectives on the same problem. This derives from being able to answer to multiple concerns. Therefore, domain-specific and domain-independent models must coexist and the overall architecture must cope with multiple model transformation and integration. A specific concern is that the architecture is extensible to new application domains.
- Viewpoint-orientation: The architecture must support defining views over subsets of its concepts. This serves to facilitate the communication and the management of the models as viewpoints act as a separation of concerns mechanism. Viewpoints will facilitate addressing multiple concerns and can improve decision-making by isolating certain aspects of the architecture according to the needs of decision makers.
- Modularity: The architecture must follow the principles of high-cohesion and low-coupling. Observing these principles contributes to expressiveness and extensibility of the architecture. It is especially important that adding new domain-specific aspects to the model does not interfere with the concepts already present in the model.

Based on these principles, our approach is based on the use of a core meta-model, also termed domain-independent ontology (DIO), which represents a domain-independent language (i.e., that does not address any specific domain-dependent concerns), containing the minimum set of concepts required for addressing the majority of scenarios. Then, that simple core meta-model can be extended in a plug-in fashion with other domain-specific meta-models, in a varying number depending on the situation at hand, which are termed domain-specific ontologies (DSOs). Each DSO represents a domain-specific language that addresses a particular set of concerns, and should also have the minimum set of concepts required for describing a determined domain. Low-coupling and high-cohesion are the principles guiding the addition of new DSOs: the number of dependencies between the DSOs and DIO should be minimal, and each DSO should deal only with a set of domain-specific concerns. In this way, the addition of DSOs should have a minimum impact on the DIO and existing DSOs.

For achieving traceability between the DIO and the DSO, it is necessary to integrate the ontologies. Thus, ontology integration deals with the combination of the different ontologies for ensuring consistency and maximum coverage of the domain being addressed. The simplest case is that of integrating the DSO with the core concepts represented in the DIO. Cross-DSO integration can also occur in cases where there is the need for adding more expressive power to specific domains. The ontology integration makes use of model transformation, which involves defining a mapping strategy from a source model to a target model [26], [27].

Considering the mapping between a DIO and a DSO, it might be as simple as a 1:1 correspondence between the concepts of the two ontologies. However, it is expected that mapping deficiencies might occur, as domain-specific languages might contain very specific concepts that are not mappable at all into the DIO. Using the Bunge-Wand-Weber representation model as an inspiration [28], one concept on the DSO might map to several concepts in the DIO (overload), a concept in the DSO might not map at all to the DIO (deficit), or, in the least common case, several concepts in the DSO might map to one concept in the DIO (redundancy). In this paper we are not dealing with this challenging aspect of ontology integration. Nonetheless, it is assumed that the mapping between some of the concepts of the DSO and some of the concepts of the DIO will always be possible as the kinds of problems we are dealing are situated in the information systems domain. Redundancy, overload and deficit cases are thus expected to occur frequently. Cases where doubts might occur concerning the mapping between two concepts are carefully judged, with the mapping not being considered in cases of incompatibility.

The architecture makes possible the usage of reasoning for performing analysis of the models: the core DIO and the extension DSOs. Four analysis configurations are possible:

- Intra-DIO reasoning, when inference is limited to the concepts of the DIO;
- Intra-DSO reasoning, when inference is limited to the concepts of the DSO;
- cross-DSO reasoning, whenever a mapping transformation between different DSOs is available, inference can use concepts from different DSOs;
- and cross DIO-DSO reasoning, when inference uses concepts from both the DIO and one or more DSOs, requiring a mapping transformation between each DIO-DSO pair.

One important aspect to refer is that this proposal would still acknowledge the use of existing tools and representation techniques for creating and managing the independent models. The models should then be converted to the DIO or DSOs (i.e., creation of individuals), depending on the case. As the models are being converted, they should be verified for any inconsistencies according to an already existing meta-model expressed in an ontological representation. After that, the models can be used for performing analysis and can even

be converted back into the original representation format if desired.

IV. Case Study Evaluation

This section describes the application of the proposed method to a concrete scenario, including the instantiation of the DIO, instantiation of a DSO, and some examples of the analysis that can be performed using the two.

The scenario used for the evaluation of this approach is concerns a civil engineering organization performing structural monitoring of large engineering structures for preventing accidents and ensuring their structural safety. This organization has a legal mandate for performing the monitoring of the structures. As such, each large structure has different types of sensors for measuring different kinds of physical phenomena that can be used for analysing and studying the behaviour of the structure along time.

Therefore, the organization is required to maintain a system for supporting the processes of acquiring and managing structure monitoring data generated by sensors installed in structures. The acquisition process involves automatically or manually retrieving raw data generated by the sensors and uploading it to the monitoring information management system. The system currently supports the following features:

- Instrumentation: supports the management of sensors, including meta-data, and the dynamic integration of new types of sensors.
- Transformation processes: manages the sensor specific algorithms that are responsible for transforming raw data into physical quantities, using instrument meta-data properties, such as calibration constants.
- Observation management: manages geodetic data, visual inspections data, and data acquired from monitoring systems.
- Data visualisation and exploitation: provides access to processes data through reports designed to support the required types of data analysis, depicting data using charts and diagrams.
- Synchronization: allows the deployment of the system in one or more locations, and the corresponding synchronisation of data.

Since the organization is obliged to acquire and maintain the monitoring data in an accurate shape, capturing information about the acquisition processes and supporting infrastructure becomes crucial, in the sense that this information can be used for confirming the provenance and authenticity of the monitoring data, and can also be used in the analysis of the behaviour of the dam along time. As such, EA becomes a valuable tool to be used in this scenario for capturing and managing this information. Moreover, the organization is interested in executing analysis on this data, for instance, for inferring the different dependencies between different components supporting the processes. Such information can become useful for change management processes.

This scenario was modelled using the standard ArchiMate 2.0 architecture description language. ArchiMate fits within the defined architecture principles as it provides a high-level of abstraction, is concern-oriented and viewpoint-oriented and was designed with extensibility in mind. But it does not address any domain-specific concerns so it can be used for demonstrating the extensibility of the architecture. Moreover, it includes a minimum set of concepts and relationships and the framework includes a minimum set of layers and aspects to enable modelling of the majority of cases. The extensions already described in ArchiMate's specification are also considered part of the domain-independent model as they address aspects that are traversal to all the organizations.An ArchiMate model was created using the Archi modelling tool[1]. The model was then converted into OWL accordingly to the meta-model, i.e., individuals where created for each element of the model.

Figure 1 is one of the high-level views produced displaying the process and the services supporting it. The process starts with the capturing and uploading of the sensor readings which might be done automatically, via the uploader component and upload service, or manually, via a Portable Data Terminal (PDT), which is a hand-held device for registering readings. After that, the readings are processed and validated via an application component termed *Observations*, which provides a series of services for manipulating readings. The raw readings are then transformed into engineering quantities through a service which applies a transformation algorithm according to the reading format. The transformation is then validated, and if deemed successful, it is archived. Two additional services and respective components implementing them are used in the process: a messaging service, which messages the system administrators in case of errors happening along the process; and a document management service, which documents the errors taking place during the validations performed along the process.

Figure 2 depicts a high-level view of the application components and infrastructure supporting the acquisition process. The monitoring information management system runs on an application server and uses a DBMS for storing, managing and providing access to the readings. Information gathered from automatic sensors is uploaded to the system through an external application installed at the production site. Information gathered by manual means uploaded through the PDT application, using a gateway application for interfacing with the system. Other more detailed views were produced but are not going to be shown here for reasons of brevity.

Figure 3 depicts the resulting OWL representation displayed in Protegé[2], with the description of the class Business process on the right side and respective individuals in the lower left corner.

A. The ArchiMate Domain-Independent Ontology

ArchiMate is used to describe the domain-independent aspects of the architecture. To create the DIO, the ArchiMate meta-model, along with the Motivation and Implementation and Migration extensions were specified in OWL-DL. OWL-DL is designed to represent rich and complex knowledge

[1] http://archi.cetis.ac.uk/

[2] http://protege.stanford.edu/

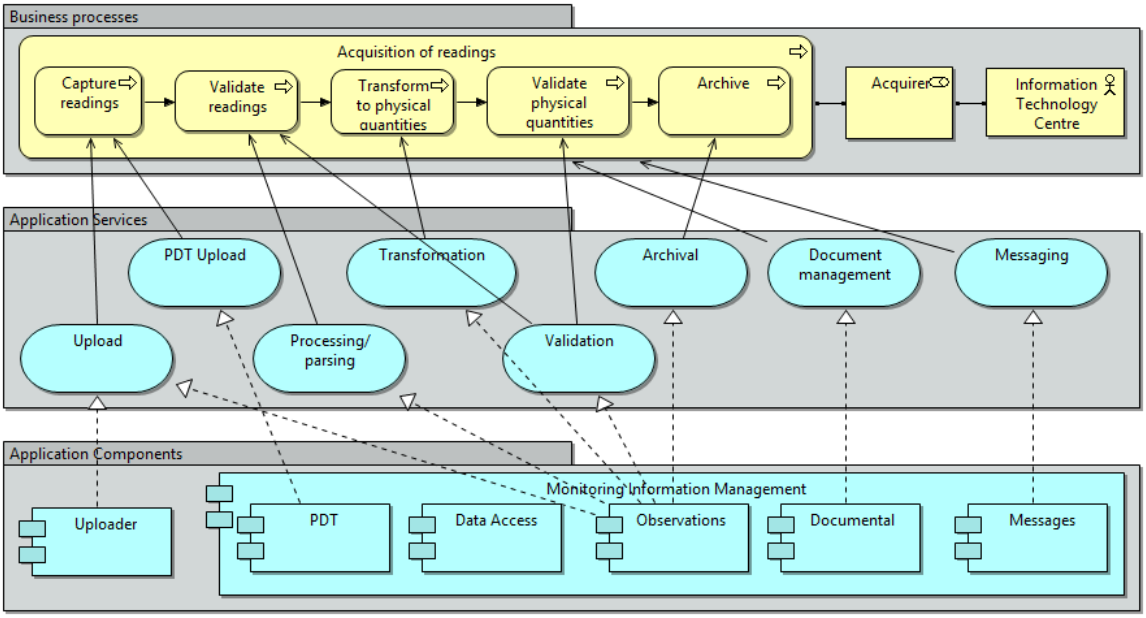

Fig. 1. The scenario process and service support

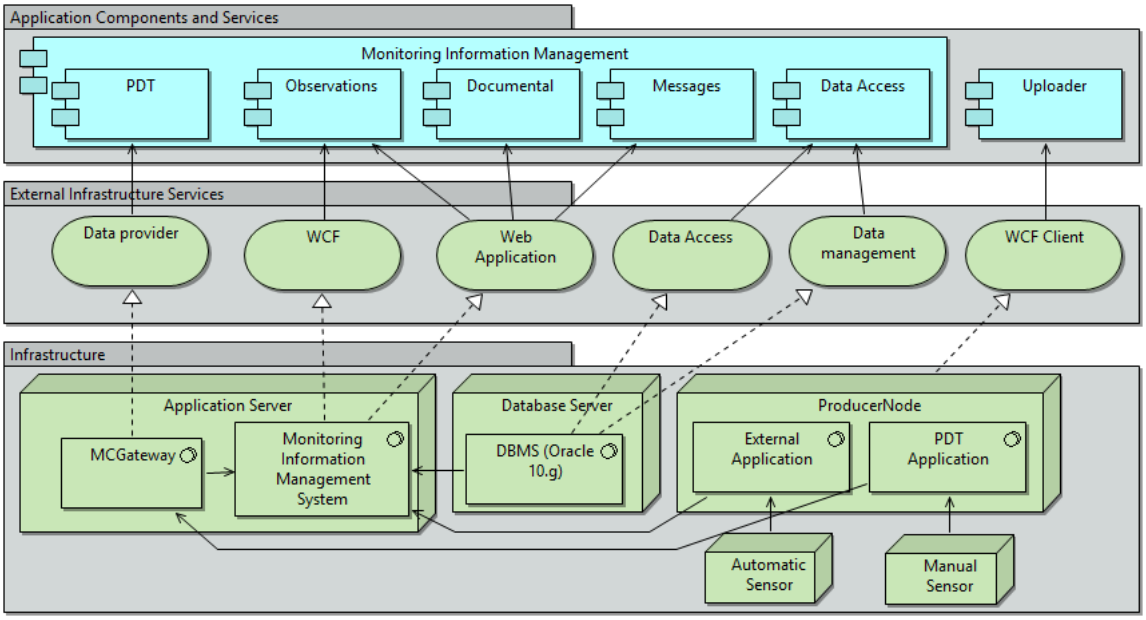

Fig. 2. The scenario application and infrastructure support

about things, groups of things, and relations between things. This choice enables taking advantage of the different computational inference and querying mechanisms already existing, and as a result, being able to perform analysis of the model for assessing the consistency of models against rules, verify the completeness of models, and for decision making through the production of reports based on contents of the model. Hence, an analysis of ArchiMate's meta-model was performed concept-by-concept, including the relations with other concepts and the constraints existing in those relations. The result was the mapping of concepts into OWL classes and the mapping of relations into OWL *ObjectProperties*. Restrictions were added to the properties, such as *InverseObjectProperties* and *SuperObjectProperties* axioms were added to the OWL ontology, so that derived relationships can be inferred. Cardinality was also added to ensure the compliance against the ArchiMate specification shows some of the aspects of ArchiMate. Figure 4 depicts a subset of the ontology restrictions.

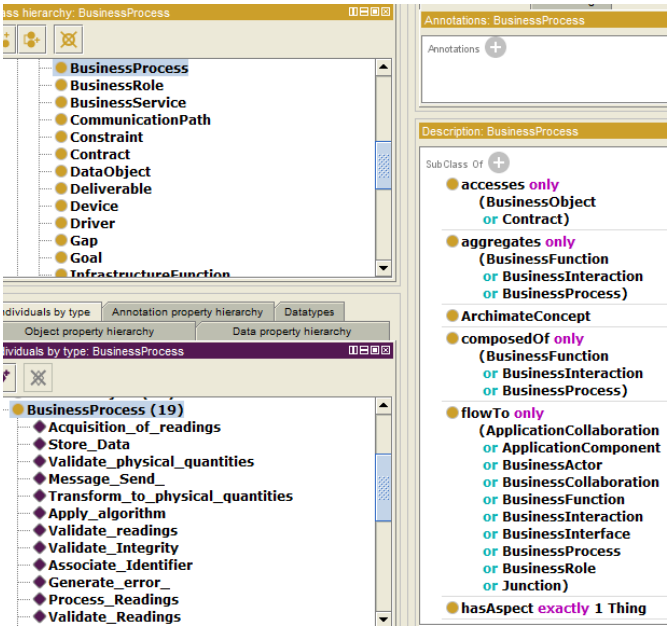

Fig. 3. OWL representation of the scenario

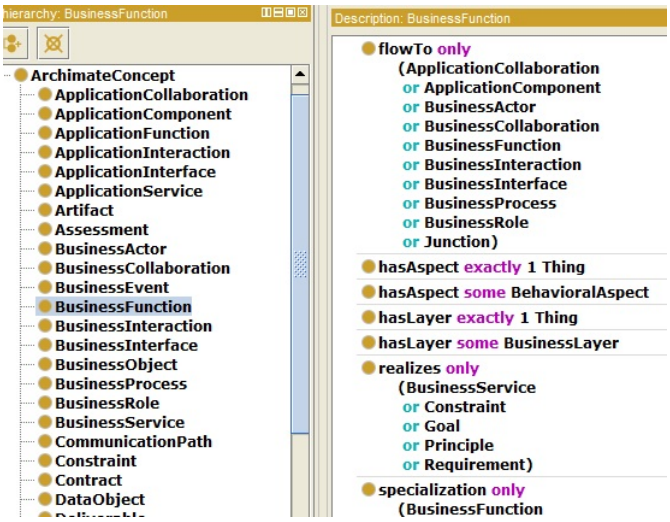

Fig. 4. Meta-model coherence enforcement

B. Extension through Domain-Specific Ontologies

The organizational stakeholders required modelling and analysing specific information about sensors since they play an important role in the structural monitoring and safety of civil engineering structures. However, the ArchiMate language lacks the expressiveness to capture the specifics of this domain. As a result, a specific description language needs to be defined. Sensors measure values that can be processed and used in structural behaviour analysis, where structural behaviour can be predicted and safety measures can be taken, if needed. Different types of sensors exist for measuring different types of engineering quantities, which are derived according to determined transformation algorithms and calibration constants. Sensors are installed in determined locations on the structure and in absolute terms, and have a determined acquisition rate. Different domain-specific languages were analysed such as SensorML[3] and TransducerML[4]. However, the stakeholders noted the complexity demonstrated of these languages and proposes using a sensor language that catered for the specific requirements of the scenario. Figure 5 depicts a class diagram depicting the concepts of the DSO.

A mapping between this DSO and the DIO was created using an *equivalentClass* declaration. Sensors can be considered computational nodes, as they mix hardware and software for performing transformations. As such, the *Sensor* class in the Sensor DSO is considered equivalent to the *Node* class in the DIO. The *GeoLocation* class in the sensor DSO and the *StructuralLocation* class in the sensor DSO were considered equivalent to the *Location* class in the DIO, as *Location* is defined in ArchiMate as a point or extent in space, thus being more generic than the two concepts in the Sensor DSO. The *Algorithm* class in the Sensor DSO is considered equivalent to the *ApplicationComponent* class of the DIO. The *Value* class of the Sensor DSO is considered equivalent to the *DataObject* class of the DIO, the same happening with the *AcquisitionRatePerYear* class.

However, in order for this mapping to be correct, data properties needed to be added to DIO classes that are part of a mapping. For instance, the *Node* class of the DIO, along with a data property restriction on the equivalence declaration, which declares that the equivalence exists when the *hasType* data property is present with the value *"sensor"* assigned to it. In this way, sensor nodes are disambiguated from other types of nodes. The mappings are the only exception where any changes are made to the DIO, in this case the addition of a data property, otherwise the principles behind this architecture would be defeated.

C. Model Analysis

The organization in focus aimed the execution of analysis on the models for the purposes of determining the impact of changes in the different elements on the remainder of the architecture. As such, and taking advantage of the possibility of introducing semantics on the models, two *SuperObjectProperty* chains were created for modelling dependencies between different elements. The *dependsDown ObjectProperty* is thus a *SuperObjectProperty* of the aggregation, composition, assignment, usage, and realization ObjectProperties that resulted from the conversion of the ArchiMate relations, while the *dependsUp ObjectProperty* fills the same purpose for the counterpart InverseObjectProperties. Moreover, these properties are transitive, which means that a graph of dependencies can be created. The following examples can be used for demonstrating how the analysis can be performed.

The following example can be used for exemplifying intra-DIO reasoning: what are the technological entities supporting the process *acquisition of readings*?. As can be seen in Figure 6, such question can be formalized into a description logic query, which due to the creation of the *ObjectProperty* chains can provide the depicted results. For exemplifying cross DIO-DSO reasoning, an instance of the DSO was created containing

[3]http://www.ogcnetwork.net/SensorML
[4]http://www.ogcnetwork.net/infomodels/tml

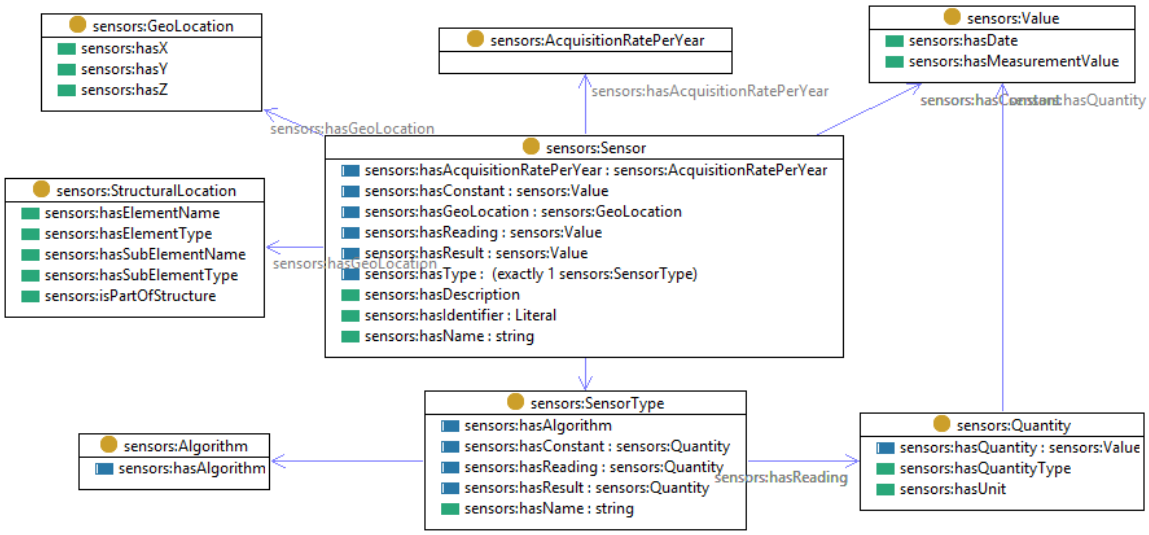

Fig. 5. Overview of the structure of the sensors ontology

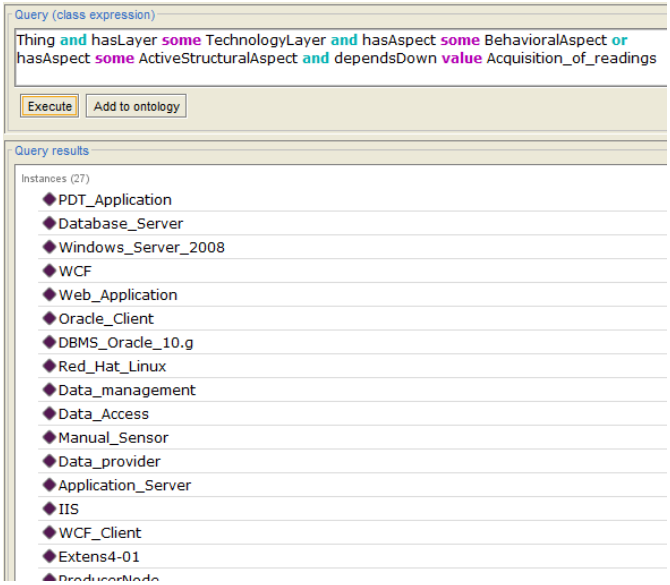

Fig. 6. Intra-DIO query results

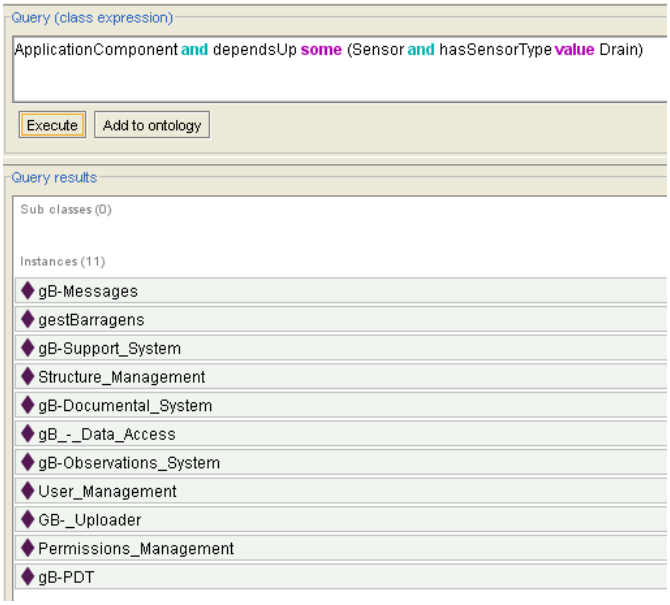

Fig. 7. Cross DIO-DSO query results

information about sensors of the type *Drain*, with individuals for the different classes. As the mapping between the DIO and DSO exists, it is possible to do cross DIO-DSO reasoning, as described in Section III. As such, an example could be that of the need to know which *ApplicationComponents* were responsible for performing the acquisition and transformation of the readings for *SensorType Drain*. This need was formalized in a description logic query. The result of the query lists all *ApplicationComponent* individuals, as Figure 7 shows.

These examples show the usefulness of having computable EA descriptions that can be processed for decision making. In this way, the organization might assess the impact that changes on the application or technology layer can have on the business, and vice-versa. Other types of semantics might be included on the models so that other kind of decision-making analysis can be performed. Additionally, if the models can be enriched with automatically captured data from the environment, the impact might even be quantified.

V. Conclusions

This paper proposes using ontologies and associated techniques to improve EA practice, contributing for improved extensibility and expressiveness of the enterprise architecture, improved enforcement of meta-model coherence, improved meta-model conformance verification of models, and improved analysis for decision-making purposes. Such techniques can be used in decision making through model analysis that takes advantage of the embedded semantics. This hypothesis was tested through an architecture that consists of a core ontology (DIO) that can be extended using domain-specific languages (DSO) to capture the specifics of each addressed scenario.

This architecture supports model reasoning that can be used to analyse the domain-independent and the the domain-specific languages.

The proposal was evaluated using ArchiMate as the DIO. To do that, we converted the ArchiMate meta-model to OWL-DL along with the required axioms that ensure the model coherence. A scenario was used to capture knowledge about specific aspects of the business that could not be addressed adequately by the DIO. As a result, we developed a DSO that captures those specific aspects and mapped into the DIO thus extending the architecture description. Example of model analysis for assessing the dependencies between different elements of the architecture was demonstrated at the level of the DIO, DSO and cross DIO-DSO.

Future work will focus on the application of this approach to new scenarios in order to explore the analysis possibilities, considering the usage of different reasoning and querying techniques. Since the individuals for the analysed case where created manually, another line of work will be the creation of extractors for obtaining real data which can then be used for validating the conformance of the models to reality and even for the semi- or even automatic creation of individuals in the different ontologies, similarly to what is already done in the process mining area [29].

ACKNOWLEDGEMENTS

This work was supported by national funds through FCT - Fundação para a Ciência e a Tecnologia, under project PEst-OE/EEI/LA0021/2013 and the grant (SFRH/BD/69121/2010) to Gonçalo Antunes, by COMET K1, FFG - Austrian Research Promotion Agency, by the Vienna Science and Technology Fund (WWTF) through project ICT12-046 (BenchmarkDP), and by the European Commission under the 7th Framework Programme for research and technological development and demonstration activities (FP7/2007-2013) under grant agreement no. 269940 (TIMBUS project).

REFERENCES

[1] M. Lankhorst, *Enterprise Architecture at Work: Modeling, Communication, and Analysis*. Springer, 2005.

[2] M. Galster, "Dependencies, traceability and consistency in software architecture: towards a view-based perspective," in *ECSA '11 Proceedings of the 5th European Conference on Software Architecture: Companion Volume*, 2011.

[3] J. Romero, J. I. Jaen, and A. Vallencillo, "Realizing correspondences in multi-viewpoint specifications," in *Proceedings of the 2009 IEEE International Enterprise Distributed Object Computing Conference*, 2009.

[4] J. Saat, U. Franke, R. Lagerstrom, and M. Ekstedt, "Enterprise architecture meta models for it/business alignment situations," in *2010 14th IEEE International Enterprise Distributed Object Computing Conference*, 2010.

[5] S. Buckl, F. Matthes, and C. M. Schweda, "Conceptual models for cross-cutting aspects in enterprise architecture modeling," in *2010 14th IEEE International Enterprise Distributed Object Computing Conference Workshops*, 2010.

[6] S. Buckl, C. M. Schweda, and F. Matthes, "A design theory nexus for situational enterprise architecture management," in *2010 14th IEEE International Enterprise Distributed Object Computing Conference Workshops*, 2010.

[7] S. Burmester, H. Giese, J. Niere, M. Tichy, J. P. Wadsack, R. Wagner, L. Wendehals, and A. Zundorf, "Tool integration at the meta-model level: the fujaba approach," *Int J Softw Tools Technol Transfer*, vol. 6, pp. 203–218, 2004.

[8] E. Kapsammer, T. Reiter, and W. Schwinger, "Model-based tool integration - state of the art and future perspectives." in *Proceedings of the 3rd International Conference on Cybernetics and Information Technologies, Systems and Applications (CITSA 2006)*, 2006.

[9] R. France and B. Rumpe, "Model-driven development of complex software: A research roadmap," in *Future of Software Engineering, 2007. FOSE '07*, 2007.

[10] M. Bjekovic, E. Proper, and J.-S. Sottel, "Towards a coherent enterprise modelling landscape," in *Sandkuhl, K. (ed.), Emerging Topics in the Practice of Enterprise Modeling : short paper proceedings of 5th IFIP WG8.1 Working Conference on the Practice of Enterprise Modeling, Rostock, Germany, November 7-8, 2012*, 2012.

[11] J. P. A. Almeida, M. E. Iacob, and P. van Eck, "Requirements traceability in model-driven development: Applying model and transformation conformance," *Inf Syst Front*, vol. 9, pp. 327–342, 2007.

[12] J. Kang, D. ande Lee, S. Choi, and K. Kwangsoo, "An ontology-based enterprise architecture," *Expert Systems with Applications*, vol. 37, pp. 1456–1464, 2010.

[13] G. Wagner, "Ontologies and rules for enterprise modeling and simulation," in *2011 15th IEEE International Enterprise Distributed Object Computing Conference Workshops*, 2011.

[14] C. L. B. Azevedo, J. P. A. Almeida, M. van Sinderen, D. Quartel, and G. Guizzardi, "An ontology-based semantics for the motivation extension to archimate," in *2011 15th IEEE International Enterprise Distributed Object Computing Conference*, 2011.

[15] J. P. A. Almeida and G. Guizzardi, "An ontological analysis of the notion of community in the rm-odp enterprise language," *Computer Standards & Interfaces*, vol. 35, no. 3, pp. 257 – 268, 2013, ¡ce:title¿RM-ODP: Foundations, Experience and Applications¡/ce:title¿. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0920548912000402

[16] H. H. Hoang, J. J. Jung, and C. P. Tran, "Ontology-based approaches for cross-enterprise collaboration: a literature review on semantic business process management," *Enterprise Information Systems*, pp. 1–17, 2013. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/17517575.2013.767382

[17] P. Schvaiko and J. Euzenat, "A survey of schema-based matching approaches," *Journal on Data Semantics IV, Lecture Notes in Computer Science*, vol. 3730, pp. 146–171, 2005.

[18] P. S. Santos Jr., J. P. A. Almeida, and G. Guizzardi, "An ontology-based semantic foundation for organizational structure modeling in the aris method," in *2010 14th IEEE International Enterprise Distributed Object Computing Conference Workshops*, 2010.

[19] *ISO/IEC/IEEE 42010:2011 - Systems and Software Engineering - Architecture Description*, International Organization for Standardization, International Electrotechnical Commission and Institute of Electrical and Electronic Engineers Std.

[20] The Open Group, *ArchiMate 2.0 Specification*. Van Haren Publishing, 2012.

[21] P. Johnson and M. Ekstedt, *Enterprise Architecture: Models and Analyses for Information Systems Decision Making*. Lightning Source Incorporated, 2007. [Online]. Available: http://books.google.pt/books?id=2LdxPQAACAAJ

[22] M. Buschle, J. Ullberg, U. Franke, R. Lagerstrom, and T. Sommestad, "A tool for enterprise architecture analysis using the prm formalism," in *CAiSE Forum 2010, Hammamet, Tunisia, June 7-9, 2010, Selected Extended Papers*, 2010.

[23] T. Binz, F. Leymann, A. Nowak, and D. Schumm, "Improving the manageability of enterprise topologies through segmentation, graph transformation, and analysis strategies," in *2012 16th IEEE International Enterprise Distributed Object Computing Conference (EDOC)*, 2012.

[24] T. Burger and E. Simperl, "Measuring the benefits of ontologies," in *On the Move to Meaningful Internet Systems: OTM 2008 Workshops*, 2008.

[25] D. Greefhorst and E. Proper, *Architecture Principles: The Cornerstones of Enterprise Architecture*. Springer, 2011.

[26] G. Guizzardi, "Ontological foundations for structural conceptual models," Ph.D. dissertation, University of Twente, Enschede, The Netherlands, 2005.

[27] M. Rosemann, P. Green, and M. Indulska, "A reference methodology for conducting ontological analyses," in *Proceedings of the 23rd International Conference on Conceptual Modelling (ER 2004)*, 2004.

[28] M. A. Bunge, *Treatise on Basic Philosophy Volume 3: Ontology I - The Furniture of the World*. Kluwer Academic Publishers, 1977.

[29] W. M. P. van der Aalst, "Business alignment: using process mining as a tool for delta analysis and conformance testing," *Requirements Eng*, vol. 10, pp. 198–211, 2005.