

**PROCEEDINGS OF THE 8<sup>th</sup>  
INTERNATIONAL WORKSHOP ON  
FORMAL ONTOLOGIES MEET  
INDUSTRY  
SEPTEMBER 21, BOLZANO-BOZEN, ITALY  
(PRE-PRINT VERSION)**

**Workshop PC chairs:**

- **Emilio M. Sanfilippo, ISTC-CNR Laboratory for Applied Ontologies, Italy**
- **Laura Daniele, Netherlands Organisation for Applied Scientific Research (TNO), The Netherlands**
- **Giorgio Colombo, Polytechnic University of Milan, Italy**

**Workshop PC members:**

- **Bob Young, Loughborough University**
- **Aleksandra Sojic, National Council of Research (CNR)**
- **Zahid Usman, Coventry University**
- **Walter Terkaj, Institute of Industrial Technologies and Automation (ITIA-CNR)**
- **Stefano Borgo, ISTC-CNT Laboratory for Applied Ontology**
- **Nicola Guarino, ISTC-CNR Laboratory for Applied Ontology**
- **Michael Grueninger, University of Toronto**
- **Riichiro Mizoguchi, Japan Advanced Institute of Science and Technology**
- **Lorenzo Solano, Polytechnic University of Valencia**
- **Pedro Rosado, University Jaume I**
- **Mathias Brochhausen, University of Arkansas for Medical Sciences**
- **Tiago Sales, University of Trento**
- **Joao Paulo Almeida, Federal University of Espirito Santo**
- **Marten van Sinderen, University of Twente**
- **Maria Poveda Villalon, Technical University of Madrid (UPM)**
- **Luiz Olavo Bonino, Dutch Techcentre for Life Sciences (DTL)**

# Semantic Ontologies and Financial Reporting: An Application of the FIBO

Oliver BROWNE <sup>a,1</sup>, Nenad KRZAVAC <sup>a</sup>, Philip O'REILLY <sup>a</sup>, Mark HUTCHINSON <sup>a</sup>

<sup>a</sup>*Accounting, Finance and Information Systems, University College Cork, Ireland*

**Abstract.** This paper illustrates the application of a developed global fund reporting ontology (GFRO) for efficient financial reporting. The GFRO extends Financial Industry Business Ontology (FIBO). Existing reporting financial information systems lack the ability to integrate data from heterogenic sources and provide unified and consistent financial reports that will comply with regulations. This study reveals that by integrating the power of XSLT and Semantic Web technologies, operationalised through the development of a scalable working prototype, allows financial services industry experts to build more flexible and consistent reports. Our research shows that the consistency of financial reports can be dramatically improved by using an appropriate inference engine.

**Keywords.** Ontologies, FIBO, financial reporting, information systems, reasoning

## Introduction

Data required for reporting to regulatory bodies is open to interpretation, from teams of legal experts to data analysts and senior management, interpretation of legal documents can leave experts arguing ad infinitum about minor nuances of language and terminology. These challenges have spurred the need for the development of a standardised language across financial instruments and institutions, a terms sets such that there is little room for interpretation and the regulator receives transparent and comparable data from all institutions for aggregation, and to be able to prove that the report is consistent with regulations which are constantly changing and developing [1].

Towards addressing this shortcoming, we have developed GFRO and implemented a framework that uses GFRO to provide consistent and unified financial reporting across heterogenic data from different sources. Our research aims to advance research on ontologies by illustrating their application for improved reporting capabilities over a broad subset of financial instruments; specifically, bonds and equities, and to perform reasoning over source data to infer new observations.

Bonds are a debt investment instrument in which an investor loans money to an entity, the entity borrows the money for a fixed period of time and repays moneys to the investor as incremental interest payments with a bullet payment of the principle at a set date. Many variations of bonds coupon and principle payments also exist. Equities are a piece of ownership and generally control of an entity, generally referred to as shares or stocks.

---

<sup>1</sup> Corresponding Author

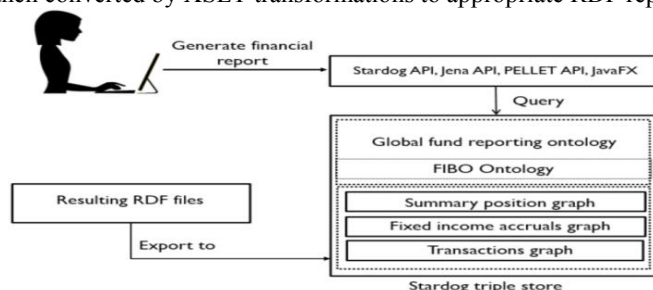
## 1. Main Ontology-based Features of the Framework for Financial Reporting

The most important task in adapting the FIBO [1] standard to serve as a core in financial reporting, is to bridge the gap between domain specific databases and FIBO ontologies. Table 1 contains a list of extensions that were made to FIBO in this research study to support GFRO.

**Table 1.** Sample list of extensions to FIBO

<b>FIBO: extended features</b>	<b>Description</b>
Full Service Fund	Service level provided by fund.
Mixed Fund Classification	The investment strategy for the asset allocation of the fund.
Real Estate Investment Trust	Type of equity instrument not represented in conceptual taxonomy.
Bond Lot Number	Lot number of the holding of the bond instrument, necessary in case purchases of same instrument made on different dates and aggregated in data.
Gain or Loss	Unrealized and realized gains and losses are paramount in the ongoing valuation of funds.
Accrued Interest Money Amount	Accrued interest represented as a monetary value as opposed to a percentage of par.
MMIF Yield	Indicator of implementation of specific regulator required formula for reporting of data.
European Market Infrastructure Reporting (EMIR) Indicator	Indicates that an asset must be included as part of an EMIR report.

Figure 1 presents an overview of the POC architecture. It reveals the process of generating the resulting RDF graph starting from raw data. The first step is to extract data necessary for reporting into csv files. Each csv file is converted into appropriate XML and then converted by XSLT transformations to appropriate RDF representation.



**Figure 1.** Architecture of consistent reporting using FIBO

We generate three types of rdf graphs. The first one covers point in time representations of the financial instruments i.e. month end data. The second transformation covers transactional data of financial instruments i.e. buying and selling of positions. The last one covers accrued interest to certain financial instruments such as bonds.

The second component of the implementation establishes communication between the end user and the triple store. It uses parametrized SPARQL queries to deliver data needed to generate financial reports, or conversely, return exceptions in case of inconsistencies. On top of this, is the GUI layer, which implements a graphical user interface that allows subject matter experts to create custom intelligent reports over financial data including regulatory reports, automatically populating regulatory reporting templates. The end user can produce reports across many funds or fund types over features of that fund. We use the *JavaFX* [2] library to generate different type of charts.

Figure 2 illustrates a UML class diagram of the implementation of all layers. The central class is *GenerateReport*. Method *getTransformation* runs the XSLT transformation of financial data stored in a folder and generates appropriate RDF file, and exports that file into the triple store, also loading GFRO into the triple store. The system runs SPARQL queries necessary for reporting of financial instruments. Some of these queries are parametrized, and some of them are not. For example, parametrized queries for bond reporting are reusable for any type of bond. All queries are stored in one folder.

When an end user submits a request for generating a report, then method *runQueries()* in *Query* class runs all queries over Stardog [3] triple store, and generates results that are stored in the output folder as a csv file as well as remembered results as properties in bean classes such as *FinancialInstrumentBean* class. The *Query* class allows a user to query more than one graph in the triple store. Methods in *GenerateReport* class such as *getPieChart()* uses csv files and methods in bean classes to generate and visualize report to end user.

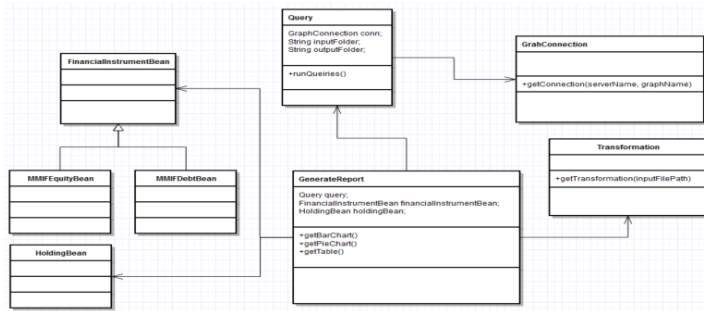


Figure 2. (B): UML class diagram of financial reporting service

## 2. POC Output: Illustrative Example

Figure 3 provides a view of a worked sample of a dynamic fund report. This report provides a summary view of the fund at a point in time represented in an easily digestible manner by finance subject matter experts.

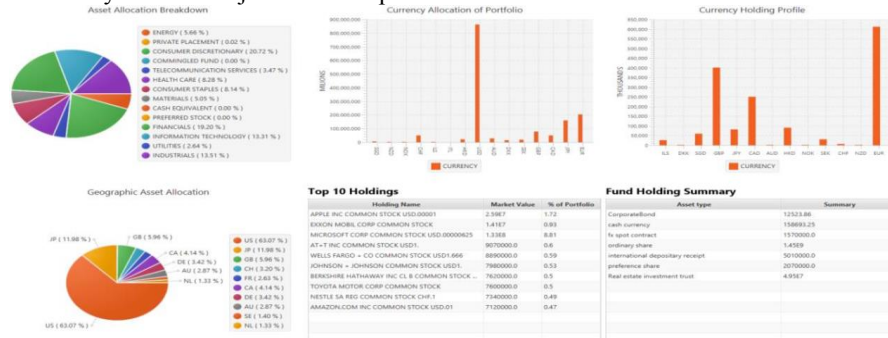


Figure 3. Sample representation of a fund at point in time.

Clockwise from bottom left; we represent the equity holdings of the fund, by country, and next by sectoral allocation of the equities as pie charts. The bar charts are used to represent currency allocation of the equities within the fund and the physical cash holding of the fund. Data in the tables' represent overall summary data. The representation of the data is standard in its presentation; however, the method of retrieval and querying is unique in its flexibility.

Databases find it difficult to query from a data end point and must be queried more generally and data collated and validated [3], this approach allows us to be flexible over querying by using shared characteristic over reasoning, removing the need for collation of spreadsheets and manual processes.

### 3. Conclusions and Future Work

The paper demonstrates a framework for consistent financial reporting that complies with regulations by adapting and extending FIBO to meet the requirements of the data. This a flexible approach to extending current reporting over legacy database systems rather than design new databases.

Although we tested the process described in the paper over a large amount of complex and varied data representative of fund level data, a limitation of our research is that we did not test beyond the scope of bonds and equities. The next stage of our research is to explore the utility of our framework in a bid data setting.

This application of regulatory and risk reporting over a complex set of data and the ability to automatically verify results over reasoning should allow further research into the benefits of ontologies. The adoption of a shared ontology could potentially dramatically reduce the timeframe for transaction processing. The financial ontology standards (FIBO) are at early stages of development, these standards are conceptually strong but lack the testing to allow for robust implementation. All findings were reported to the EDM Council and Object Management Group to help to improve the FIBO standard wherever possible [5].

### 4. Acknowledgements

The proof of concept outlined in this paper was funded by the State Street Corporation. In particular we would like to attribute the ongoing success and funding of this project to David Saul, Dáire Lawlor, Daragh McGetrick

### References

- [1] M. Bennett, The financial industry business ontology: Best practice for big data, *J. Bank. Regul.* **14** (2013) 255–268. doi:10.1057/jbr.2013.13.
- [2] Java FX, <http://docs.oracle.com/javase/8/javase-clienttechnologies.htm>, Accessed on September, 2017.
- [3] Stardog, 2017, <http://stardog.com/> Accessed on September, 2017.
- [4] G.W. Dickson, R.L. Leitheser, J.C. Wetherbe, and M. Nechis, Key Information Systems Issues for the 1980's, *MIS Q.* **8** (1984) 135–159. <https://doi.org/10.2307/248662>
- [5] FIBO, <https://www.edmcouncil.org/financialbusiness>

# BPMN 2.0 Choreography Language: interface or business contract?

Greta ADAMO<sup>a,c</sup> and Stefano BORGO<sup>b</sup> and Chiara DI FRANCESCO MARINO<sup>a</sup> and  
Chiara GHIDINI<sup>a</sup> and Marco ROSPOCHER<sup>a</sup>

<sup>a</sup>FBK-IRST, Via Sommarive 18, 38050 Trento, Italy

<sup>b</sup>ISTC-CNR Laboratory for Applied Ontology, Trento, Italy

<sup>c</sup>University of Genova, DIBRIS, via Dodecaneso 35, 16146 Italy

**Abstract.** Choreography diagrams have been introduced in the Business Process Model and Notation language 2.0 (BPMN 2.0), one among the most used languages for modelling and analyzing business processes in industry, in order to provide a view on the *interaction* between participants. Besides the intuitive definition of choreographies as *interfaces* among participants, the BPMN 2.0 specifications also define choreographies as *business contracts* among the parties. However, the adoption and the diffusion of the business contract nature of choreography diagrams seem to be hindered by the underspecification of the notation, which does not allow to model and formalize constraints and relationships among choreography entities, which would need to be specified in a business contract. In this paper we provide a preliminary investigation of some of the open issues characterizing BPMN 2.0 choreography diagrams when looking at the business contract nature of the notation, by focusing on those related to messages and participants.

## 1. Introduction

The Business Process Modelling Notation<sup>1</sup> (BPMN) is one of the most popular languages for business process modelling largely used in industry. Few works in the literature [9,6,10] focused on the ontological formalization of the language. For instance, [9] proposed the “BPMN Ontology” for formalizing *business process diagrams* of the BPMN 1.1 specifications. A similar approach is followed in [6] to formalize the structure of process diagrams expressed according to BPMN 2.0. A different approach is taken in [10], where the authors focus on the analysis of behavioral aspects of process models (activities and events) in order to investigate whether these constructs of BPMN commit to an ontological theory of the domain entities at hand. However, so far no initiatives have considered the ontological formalization of the additional typologies of diagrams introduced with the BPMN 2.0 specifications, namely *collaborations* and *choreographies*. Differently than process diagrams, collaborations focus on representing the interactions between two or more processes, while choreographies enable capturing the coordination between different business participants. In the quest for providing an ontological formalization of the whole BPMN specifications, we started tackling the analysis of choreography diagrams.

---

<sup>1</sup><http://www.omg.org/spec/BPMN>

The BPMN 2.0 specifications [7] provide several *definitions* of choreographies, each one capturing a different aspect. A first definition of choreography is “the way business participants *coordinate their interactions*” ([7, p. 345]). Starting from this assumption, it seems that the purpose of a choreography is to hook the *interaction* behavior between two or more participants, thus “making possible to derive the process interfaces of each partner”([7, p. 345]). In addition, the interaction is focused on the exchange of *information* conceived as *messages*. Another definition considers choreographies as “a type of *business contract* between two or more organizations” ([7, p. 345]).

Over the past years, different works have proposed languages and evaluation models for extending choreographies with business contract (deontic) constructs [1] and for capturing and assessing the value of choreographies [2,3]. Choreographies in BPMN 2.0 seem not to satisfy models as the Web Service Choreography Description Language proposal (WS-CDL [8]) and, at least not completely, the main evaluation model for the adequacy of choreographies based on a semiotic quality framework [11]. The conclusion of the evaluation is, indeed, that BPMN 2.0 choreographies present some issues [2].

In the attempt to better understand choreography diagrams for their formalization, we also clashed against a mismatch between the business contract nature of the BPMN 2.0 choreography language and the underspecification of the language when coming to define constraints and relationships at a business contract level.

In this preliminary work we focus on a few ontological elements (participants, roles, information) to highlight and discuss, by means of an example, some of the constraints and the relationships which a business contract could require to model, but that the language does not allow to explicitly represent. In detail, we focus on:

- message-related issues: how to constrain a message and its content?
- participant-related issues: how to specify relationships among participants?

By analyzing these issues we advocate the need of a finer-grained specification of the language able to capture these aspects or, alternatively, the adoption of semantic annotation mechanisms enabling the formalization of constraints and relationships among choreography entities.

After a brief introduction of the main BPMN 2.0 choreography constructs (Section 2), we illustrate through an example the issues of the language in formalizing contract level constraints and relationships (Section 3). Finally, conclusions and future work are reported in Section 4.

## 2. The BPMN 2.0 choreography modeling

The BPMN 2.0, which is the *de-facto* standard notation for business processes proposed by the Object Management Group (OMG), captures the coordination between different business participants through *choreography diagrams*.

Figure 1 shows an example of a choreography diagram in BPMN 2.0. The interaction behaviour among two or more participants is described by means of *choreography activities* i.e., *choreography tasks* (atomic activities) or *sub-choreographies* (compound activities), connected via *sequence flows*. Choreography activities are depicted as rectangles (decorated with a “+” symbol in case of sub-choreographies), while participants as bands on their top and bottom. For instance, the choreography diagram in Figure 1

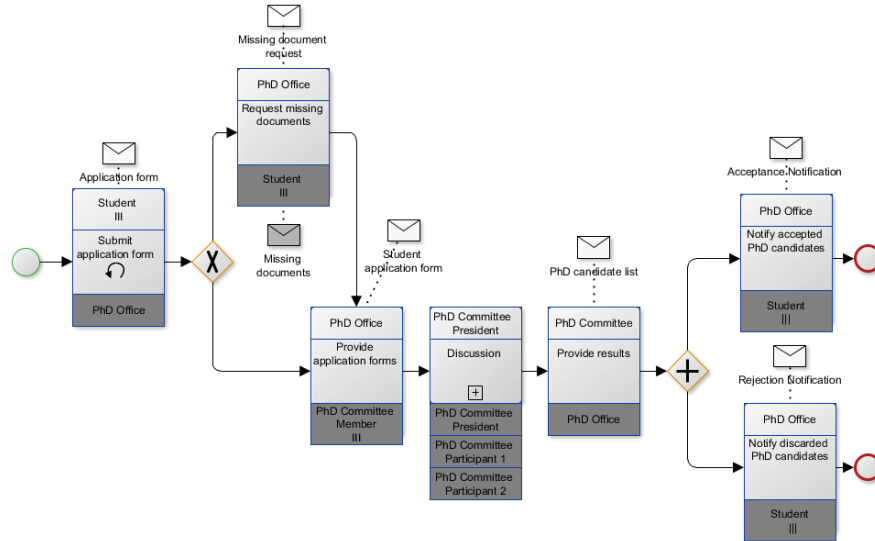


Figure 1. Choreography example: the PhD selection process

is modelled involving seven participants: the *student*, the *PhD Office*, the *PhD Committee*, the generic *PhD Committee Member* and the single members of the committee (*PhD Committee Participant 1*, *PhD Committee Participant 2* and the *PhD Committee President*).

Each choreography activity has only one initiator (depicted in white) sending the initial message and one or more receivers (darker bands associated to the activity). An envelope represents a message sent by the sender while return message envelopes of a two-way interaction are darkened. For instance, the atomic choreography task *Provide results* represents the interaction between the *PhD Committee* (sender) and the *PhD Office* (receiver): the *PhD candidate list* is sent from the *PhD Committee* to the *PhD Office*.

As for the control flow, the BPMN 2.0 choreography language inherits the gateways of the BPMN 2.0 language: the XOR, OR and AND gateways. For example, in Figure 1, the first XOR gateway models a decision point so that, if the application forms sent by the students are complete, they are directly passed to the PhD Committee; otherwise, the students are asked to send the office the missing documentation.

Moreover, BPMN 2.0 choreography diagrams allow for representing more than two participants (when dealing with sub-choreographies) and *multi-instance* participants. An example of choreography activity with several participants is the *Discussion* sub-choreography in Figure 1. This choreography activity involves three participants (one as initiator and three as receivers of the message). *Multi-instance* participation is denoted by three vertical lines and can refer both to the choreography activity and the participants. For the sake of this paper we provide here only an example of *multi-instance* participant: *Student* is used at the start of the diagram of Figure 1 for representing several instances of the role ‘Student’ (as several students can apply for PhD). Note that, in this case, the *PhD Office* will loop on the receipt of the application forms by each of the students



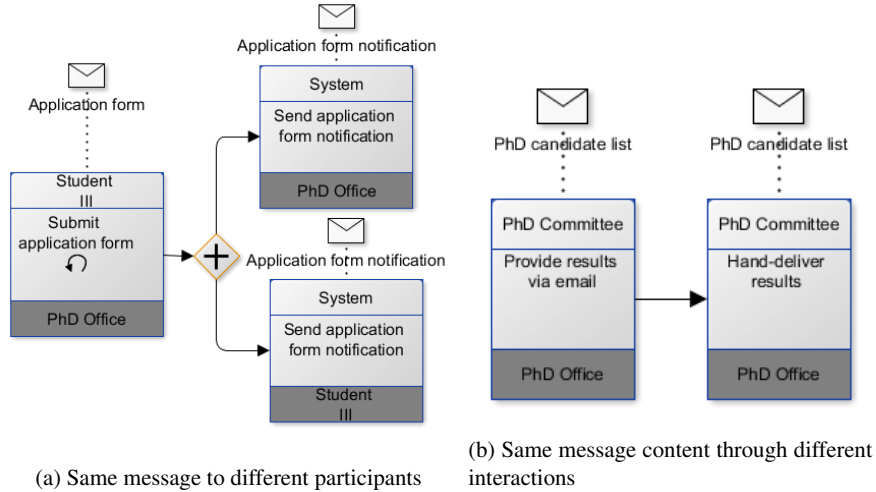


Figure 2. Message-related issues

until the PhD call deadline is over. This is denoted by the looping arrow symbol in the choreography activity *Submit application form*.

### 3. Open issues

In this section we describe a scenario we are interested to model as a choreography diagram and we use it for discussing the aspects of the BPMN 2.0 choreography language that could potentially hinder the interpretation of a choreography as a business contract.

*Ph.D Selection Scenario* The *Ph.D selection* scenario deals with the application, evaluation and applicants' notification in a PhD selection. In detail, students apply for a PhD through an online application form, composed of three parts: personal information details, CV, and motivation letter. The online application form is filled and submitted by the student to the PhD Office and to the student herself for notification. If documents are missing, the PhD Office requests them to the students. Once all the needed documents are available, the PhD Office sends the received application forms to the PhD Committee. The committee members discuss about each of the applicants and, at the end, the committee returns the final list of accepted candidates to the PhD Office both via e-mail and through hand-delivery. The PhD Office will finally notify both the accepted and the rejected PhD candidate applicants.

Figure 1 shows a choreography diagram modelling the PhD Selection Scenario at the level of abstraction that is generally used with choreography diagrams. However, by keeping in mind the definition of a choreography as a business contract and looking at the description of the scenario, we would like to be able to specify and formalize other constraints and relationships among choreography entities. We detail in the following the modelling issues we found in the scenario when detailing messages and participants, respectively.

*Message-related issues* For instance, we would like to specify that the application form is composed of three parts: the personal details, the CV and the motivation letter. As only a single message (per direction) can be exchanged in a choreography task, the only viable solution for expressing this kind of information in BPMN 2.0 choreographies seems to be the Message attribute - *item description* - which allows for specifying the structure of the exchanged message, i.e., the *form* of the message. However such attribute provides only an informal means to specify the structure of the document.

If the form of the message can be, at least informally, specified, this is not possible at all for the (expected) *content of a message*. For instance, if we wish to explicitly model the fact that the applicant student receives, as submission confirmation, the submitted form, we would probably need to explicitly model a new participant - the *system* - sending an *Application form notification* not only to the PhD Office, but also to the students themselves. In other terms we would need to model the *sending of the very same message to two different participants*. Unfortunately, however, BPMN 2.0 choreography specifications limit the number of participants of a choreography task to two. We hence need to resort to graphically modelling the notification messages from the system to the *PhD Office* and to the *Student* as different messages (and choreography tasks), only implicitly bound by the same name, as shown in Figure 2a. However, this does not enable us to formally specify that the two *Application form notification* messages are exactly the same message, as we wish.

A similar, although slightly different situation, is related to the inability of the language to provide a mechanism for *sending the same message content through different participant interactions*. For instance, in the selection scenario, we would like to model the fact that the committee returns to the PhD Office the list of PhD candidates both via email and through hand-delivery. To this aim we would like to specify that a message with the same content (the list of PhD candidates) is delivered twice to the PhD Office - first via email and then through hand-delivery (see Figure 2b). The BPMN 2.0 choreography language, however, does not allow us to specify that the content of a message is exactly the same of another.

In both the last two examples, we can observe that the same message or message content, respectively, can have different effects on the participant(s) receiving it. Instead of focusing on a general message content, we can capture this difference by separating the generic content in *information* and *knowledge*. Roughly speaking, here we use the term *information* to refer to the meaning of a message in isolation. That is, when the message is seen as a (possibly complex) statement taken out of any particular context. For example, consider a message composed of the statement “snow is white”. This message conveys the information that the object snow has color white. The fact that snow is an object and white is a value of the color-quality is ensured by the language itself. Thus, information is here seen as the ‘linguistic’ meaning of the statement. The same message has a different meaning, which here we call *knowledge*, if we consider the meaning of the statement in a specific context. Briefly, in this paper the knowledge of a message is the change in the epistemic state of the receiver as caused by the acquisition of the message information in the context provided by the choreography model. The “snow is white” message has a different impact on the receiver depending on whether this is represented within a choreography model relative to an agent that is moving from, say, Cuba to Norway, compared to the impact it has on an agent that lives in Norway and is learning English. In the first case this sentence conveys new knowledge about the object

snow; in the second case it conveys new knowledge about the English grammar or about the meaning of some English terms.

In the first example of the selection scenario - *sending the very same message to two different participants* - there is only one message (and thus one message information) and yet the knowledge it conveys to the PhD Office and to the student is different. Indeed the PhD Office acquires knowledge on the fact that an application form has been submitted and a new student has applied for PhD. The student, on the contrary, acquires knowledge on the fact that the application form has been successfully received by the specific University. In the second example - *sending the same message information through different participant interactions* - there are two message instances with the same message information sent to the same participant and still the knowledge they convey differs: while the first message conveys knowledge to the PhD Office about the outcome of the evaluation, the second message provides no knowledge as the PhD Office's context at this point has already been updated on this issue.<sup>2</sup>

This discussion suggests a further issue of the choreography language: lack of guidelines on the usage of message labels. The use of message labels seems to be a natural means for informally constraining the messages' information. When the language already provides a mechanism to specify that two messages have the same information, the label of the message could be used for providing information about knowledge. For instance, in the first case we could label *confirmation receipt* the application form notification received by the student and *submission request* the very same message received by the PhD Office. Similarly, in the second example, the PhD Office could have received first a *PhD candidate list* and then a *Copy of the PhD candidate list*, where here the term 'copy' specifies that no new knowledge is added to the PhD Office at this point.

*Participant-related issues* A second group of issues is instead related to the participants and their relationships. In the choreography diagram in Figure 1, for instance, the *PhD Committee* and its members have been modelled in different ways: as a single entity (for representing that a unique message is sent by the committee to the PhD Office), as a multi-instance participant (for denoting that each member of the committee received the student application forms by the PhD Office), or by means of different roles (for specifying the role of each of the members of the committee in the discussion). However, these participants are completely unrelated in the choreography diagram, as BPMN does not allow to specify any relationship between participants, while we would like to be able to state and specify such relationships.

On the other hand, we are aware of the potential complexity introduced by allowing to specify these relationships, not only in terms of choreography diagram but also in terms of process diagram. Indeed, since each participant could implement her process in her own pool, reconciling the processes of the *PhD Committee* and a generic *PhD Committee Member*, or even worse the process of a generic *PhD Committee Member* with the one of a specific *PhD Committee Participant 1*, could be far from trivial.

---

<sup>2</sup>Note that the knowledge acquired by the PhD Office on the fact that these activities have been executed due to the reception of the two messages is not knowledge obtained from the information in the messages themselves but from the fact that they (both) exist.

#### 4. Conclusions and Future Work

The preliminary analysis carried out on the BPMN 2.0 choreography language revealed that the language presents some issues with respect to its capability to describe constraints and relationships among its entities. These issues open the way to the possibility to enrich the language with the appropriate means for expressing properties on message form and content, as well as participant relationships. An alternative possibility could be enriching the diagram with semantic annotations, as already proposed for the BPMN process diagram [4], in order to be able to formalize and specify constraints on messages and relations among participants.

In the future, we plan, besides proceeding with the ontology-based choreography diagram formalization, to further investigate these open issues and to suggest possible solutions to support and guide users in the exploitation of choreographies as a means for specifying business contracts among parties.

#### References

- [1] Andrew Berry and Zoran Milosevic. Extending choreography with business contract constraints. *International Journal of Cooperative Information Systems*, 14(02n03):131–179, 2005.
- [2] Mario Cortes-Cornax, Sophie Dupuy-Chessa, Dominique Rieu, and Marlon Dumas. *Evaluating Choreographies in BPMN 2.0 Using an Extended Quality Framework*, pages 103–117. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
- [3] Gero Decker, Oliver Kopp, Frank Leymann, and Mathias Weske. Bpel4chor: Extending BPEL for modeling choreographies. In *2007 IEEE International Conference on Web Services (ICWS 2007), July 9-13, 2007, Salt Lake City, Utah, USA*, pages 296–303, 2007.
- [4] Chiara Di Francescomarino, Chiara Ghidini, Marco Rospocher, Luciano Serafini, and Paolo Tonella. A framework for the collaborative specification of semantically annotated business processes. *Journal of Software Maintenance*, 23(4):261–295, 2011.
- [5] Pawel Garbacz and Oliver Kutz, editors. *Formal Ontology in Information Systems - Proceedings of the Eighth International Conference, FOIS 2014, September, 22-25, 2014, Rio de Janeiro, Brazil*, volume 267 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2014.
- [6] Christine Natschläger. *Towards a BPMN 2.0 Ontology*, pages 1–15. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.
- [7] OMG. Business Process Model and Notation (BPMN), Version 2.0, January 2011.
- [8] Zongyan Qiu, Xiangpeng Zhao, Chao Cai, and Hongli Yang. Towards the theoretical foundation of choreography. In *Proceedings of the 16th International Conference on World Wide Web, WWW '07*, pages 973–982, New York, NY, USA, 2007. ACM.
- [9] Marco Rospocher, Chiara Ghidini, and Luciano Serafini. An ontology for the business process modelling notation. In Garbacz and Kutz [5], pages 133–146.
- [10] Emilio M. Sanfilippo, Stefano Borgo, and Claudio Masolo. Events and activities: Is there an ontology behind bpmn? In Garbacz and Kutz [5], pages 147–156.
- [11] Terje Wahl and Guttorm Sindre. An analytical evaluation of bpmn using a semiotic quality framework. *Advanced topics in database research*, 5:94–105, 2006.

# A Method to generate a Modular ifcOWL Ontology

Walter TERKAJ<sup>a,1</sup>, and Pieter PAUWELS<sup>b</sup>

<sup>a</sup>*Institute of Industrial Technologies and Automation (ITIA-CNR), Milan, Italy*

<sup>b</sup>*University of Ghent, Ghent, Belgium*

**Abstract.** Building Information Modeling (BIM) and Semantic Web technologies are becoming more and more popular in the Architecture Engineering Construction (AEC) and Facilities Management (FM) industry to support information management, information exchange and data interoperability. One of the key integration gateways between BIM and Semantic Web is represented by the ifcOWL ontology, i.e. the Web Ontology Language (OWL) version of the IFC standard, being one of reference technical standard for AEC/FM. Previous studies have shown how a recommended ifcOWL ontology can be automatically generated by converting the IFC standard from the official EXPRESS schema. However, the resulting ifcOWL is a large monolithic ontology that presents serious limitations for real industrial applications in terms of usability and performance (i.e. querying and reasoning). Possible enhancements to reduce the complexity and the data size consist in (1) modularization of ifcOWL making it easier to use subsets of the entire ontology, and (2) rethinking the contents and structure of an ontology for AEC/FM to better fit in the semantic web scope and make its usage more efficient. The second approach can be enabled by the first one, since it would make it easier to replace some of the ifcOWL modules with new optimized ontologies for the AEC-FM industry. This paper focuses on the first approach presenting a method to automatically generate a modular ifcOWL ontology. The method aims at minimizing the dependencies between modules to better exploit the modularization. The results are compared with simpler and more straight-forward solutions.

**Keywords.** IFC, ifcOWL, Ontology, Modularization, EXPRESS

## 1. Introduction

BIM (Building Information Modeling) is gaining more and more relevance in the Architecture Engineering Construction (AEC) and Facilities Management (FM) industry to support the digitalization of the business process. Industry Foundation Classes (IFC) [16] is one of the standards in the BIM domain and it is widely used in industrial applications. However, there are barriers limiting its semantic interoperability and adoption on a larger scale [23]. Indeed, the IFC standard is provided as single schema written in EXPRESS language [14] that is extremely large and complex, being characterized by an almost monolithic structure. For instance, the IFC4\_ADD1 EXPRESS schema contains 768 En-

---

<sup>1</sup>Corresponding Author: Institute of Industrial Technologies and Automation (ITIA-CNR), Milan, Italy; E-mail: walter.terkaj@itia.cnr.it

tity data types, 206 Enumeration data types, 60 Select data types, 131 defined data types, 46 FUNCTION declarations, and 2 RULE declarations. The complex structure of IFC jeopardizes its exploitation by industrial domains outside the core AEC applications that may need a simple model of building, spaces, elements and their relations with geometry, topology, monitoring, automation and control, safety, etc.

Semantic Web offers opportunities to provide more effective solutions also for the BIM domain, by exploiting its typical enablers in terms of formal modeling language, data distribution, extensibility, and automatic reasoning. Possible BIM solutions based on Semantic Web technologies include:

1. an OWL version of IFC, named ifcOWL. Previous works [22,21] demonstrated how the ifcOWL can be automatically generated by converting the IFC EXPRESS schema to OWL. For example, this conversion leads to an ifcOWL ontology [21] for IFC4\_ADD1 with 1313 classes, 1580 object properties, 13867 logical axioms, and 1158 individuals.
2. the development of novel ontologies for BIM that are based on the semantic web principles and designed exploiting modularity and extendability since the beginning. Such approach is currently investigated by the World Wide Web Consortium (W3C) with the Linked Building Data (LBD) Community Group that is working on a set of loosely related ontologies for Building Topology (BOT) [24], Product, Geometry, Automation and Control [28], etc.

Given the original complexity of the IFC schema, also the resulting ifcOWL ontology is considerably large and complex to load and use. The ifcOWL ontology has many interdependencies that it becomes a huge challenge to exploit data distribution both at Tbox and Abox level. The ontology takes full advantage of OWL2 DL expressivity (*SHIQ(D)*), which can lead to a high number of assertions when handed to OWL reasoning engines because all axioms are loaded when the ontology is referenced by an RDF graph.

This paper will investigate how the ifcOWL ontology can be split into separate ontology modules, so that end users and applications only need to select the modules that are actually going to be used. The modularization is expected to reduce the complexity and provide enablers also for future extensions and integrations. Sect.2 briefly presents related works on ontology modularization, whereas Sect.3 addresses the specific problem of modularizing the ifcOWL ontology. Sect.4 presents the modularization algorithm and Sect.5 shows the results of the application of the algorithm to generate a modular ifcOWL ontology. Finally, conclusions are drawn in Sect.6.

## 2. Related Works

As defined by d'Aquin et al. [10], the task of partitioning an ontology is “the process of splitting up the set of axioms into a set of modules  $\{M_1, \dots, M_k\}$  such that each  $M_i$  is an ontology and the union of all modules is semantically equivalent to the original ontology  $O$ ”. The topic of ontology modularization has been largely addressed in the literature [27]. Indeed, modularity can be beneficial both during the design phase and during the deployment and usage. Some of the benefits of modularity can be mentioned as follows [19]:

September 2017

- scalability for querying data and reasoning on ontologies
- scalability for evolution and maintenance
- complexity management
- understandability
- context-awareness and personalization
- reuse

Modularity can be applied to pursue different goals [7] while using different strategies for modularity [19], some times also in a concurrent way:

- disjoint or overlapping modules
- semantics-driven strategies
- structure-driven strategies (e.g. using graph decomposition algorithms)
- machine learning strategies
- monitoring modularization and making it evolve

Various techniques have been proposed mainly to process large ontologies and extract modules from them, e.g. [5,9,11,13,15,18]. Modularization has been applied in various knowledge domains, for instance architectural design [6,3] and biomedical domain [20,26,29]. In some cases the ontologies were designed since the beginning in a modular way, for example the TOVE ontologies [12] supporting the enterprise integration. Furthermore, when addressing a modularization problem, the definition of the evaluation criteria [10] plays a key role and it must be consistent with the overall goals.

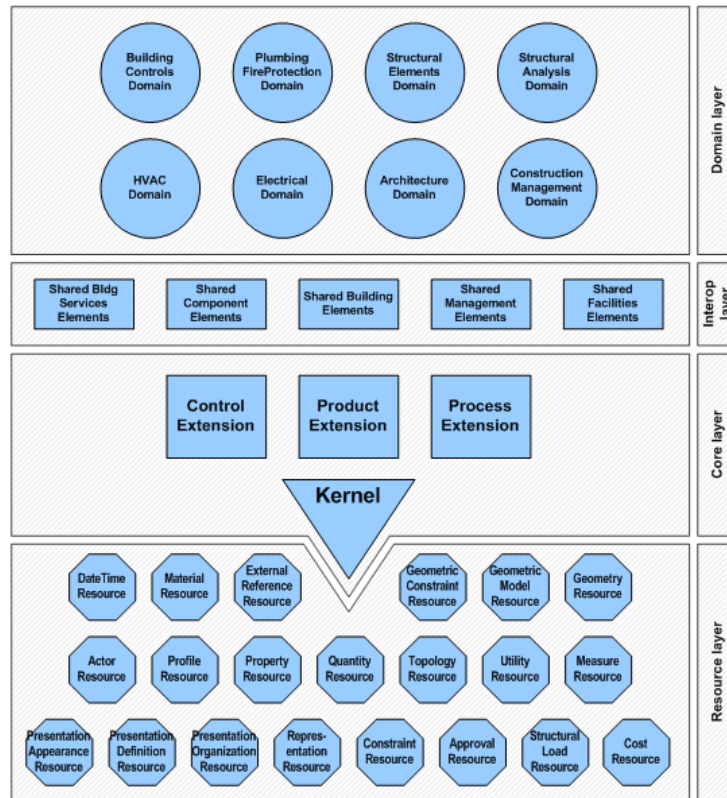
### 3. ifcOWL Ontology and Modularity

The modularization of ifcOWL is an important step in updating the ontology so that it can be more efficiently used in a web context. Indeed, a modular ifcOWL is expected to improve:

- usability
- performance (e.g. query and reasoning)
- ease of alignment with other ontologies, also reducing overlapping

At least two strategies can be envisioned to generate a modular ifcOWL:

1. modularization by content, i.e. the definition of classes and properties are separated based on the knowledge domain they are related to, e.g. geometry, units of measurement, building components, HVAC, etc.
2. modularization by axiom type, i.e. separating the different axioms that are included in ifcOWL, such as definition of classes, subsumption, data/object properties, domain/range of properties, equivalent classes, cardinality restrictions apart, etc. This option might even align with the idea of being able to load an ifcOWL in specific OWL profiles (OWL2 EL, OWL2 QL, OWL2 RL - see [17]). For example, an ifcOWL version not containing cardinality restrictions could be used to conform with the OWL2 EL profile. On the other hand, most OWL reasoners allow to specify to which level of expressiveness (RDFS, OWL2 EL, OWL2 QL, OWL2RL) an ontology should be loaded. Thus, when reasoning is concerned, this first option is already supported by using an OWL reasoner with appropriate settings.



**Figure 1.** The IFC data schema architecture with conceptual layers, as displayed in the introduction of the IFC specification (IFC4 ADD1) [16]

Herein the attention is focused on the first strategy that is supported by the fact that the IFC standard was developed in a modular way and each data type (i.e. entity, enumeration, select, defined) in the EXPRESS schema belongs to a specific sub-schema, as reported in its documentation [16]. Indeed, the IFC schema consists of four layers, each containing sub-schemas (see Figure 1) that define a part of all EXPRESS data types. For example, the `IfcActorResource` schema (bottom left in Figure 1) contains 3 enumerations and 8 entities. The corresponding OWL definitions could in theory be kept in a separate `IfcActorResource` ontology module, which would be significantly smaller than the complete `ifcOWL` ontology, thus resulting in better usability. However, many of the data types in the IFC schema are tightly interconnected with each other, not only within a sub-schema but also between different sub-schemas. In addition, in several cases there are reciprocal dependencies between sub-schemas, even belonging to separate layers. For example, `IfcApprovalResource` imports `IfcControlExtension` and vice versa. Hence, in order to make a useful modularization, a full investigation of the schema needs to be made, and the relation between the different sub-schemas (and therefore modules) would need to be reconsidered to a significant level and detail.



Beetz et al. 2009 [4] already proposed a modular ontology for an earlier version of ifcOWL. The authors addressed the problem of interwoven interdependencies by moving some axioms to additional modules, named *pivot ontologies*, that include a set of independent semantic clusters. However, the problem of cyclic references was not completely solved. Furthermore, the author addressed the problem of modularization of the Abox ontologies.

#### 4. Modularization Algorithm

The proposed modularization algorithm can be applied to convert any EXPRESS schema (e.g. IFC [16], but also ISO 15531, ISO 14649, etc.) to a modular OWL ontology. A novel algorithm was developed to exploit the peculiar problem settings, since the modularization takes place while converting an EXPRESS schema (e.g. IFC schema) to an OWL ontology (e.g. ifcOWL), instead of being executed on an already existing large monolithic ontology (e.g. the already generated full ifcOWL). Once the algorithm assigns an EXPRESS definition to a module, then the conversion to the corresponding OWL axiom is executed as stated in [21] and described with more details in [22]. It must be noted that an automatic conversion of a technical standard from an EXPRESS schema to an OWL ontology may lead to problems related to the lack of a precise definition and meaning of some concepts, thus hindering semantic interoperability. Therefore, a proper ontological analysis of the original standard should be carried out, as addressed in the works [2,25,8]. However, such analysis goes beyond the scope of this work.

The goal of the algorithm consists in finding the best way of implementing a given input modularization by minimizing the number of direct import relations between modules. Moreover, the algorithm must avoid to create reciprocal dependencies between modules because it would lead to circular import paths. Even though circular import is not forbidden according to OWL2, still it is not desirable because it would actually weaken the modularization. Indeed, a direct import of any node in a circular path will lead to indirectly importing all the nodes in the circle; thus the final effect is that all the modules in a circular path are merged.

In summary, the algorithm receives as input the following pieces of information:

- content of a parsed EXPRESS schema in terms of data types (i.e. defined data, entity, select, enumeration), subsumption relationships and attributes of each entity data type.
- input modularization in terms of mapping between EXPRESS data types and modules. This mapping can be the results of more or less sophisticated methodologies, or it can be provided in a technical documentation (as in the case of IFC [16]), or it can be simply set by the user based on his/her needs.
- priority level associated with each module. This priority is used to set import relations between modules. *Ceteris paribus*, the module with lower priority will import the module with higher priority. For instance, the priority may be associated with the layer in the whole IFC schema, giving highest priority to the modules in the Resource Layer and the lowest to the modules in Domain Layer.

The modularization algorithm is decomposed into two routines Algorithm 1 and Algorithm 2. Algorithm 1, via the function `GenModularETO`, elaborates the various EX-

PRESS definitions that must be converted to a corresponding OWL axiom. The OWL axiom is serialized as a set of triples that are added to a specific module based on the result of the function `SetModule` in Algorithm 2. Thus, the function `SetModule` incrementally adds import relationships between modules based on the actual needs derived from the inter-module dependencies between EXPRESS data types. After STEP 4 of Algorithm 1 all the OWL axioms required to convert the EXPRESS schema are assigned to a specific module. Moreover, the full set of dependencies (i.e. import relations involving the term `owl:import`) between modules is available and can be represented as a directed graph, where the modules are nodes and the import relations are arcs. With reference to the notation adopted in the algorithm, the graph can be defined as  $G = (M, I)$ , where  $M$  is the set of modules (i.e. nodes) and  $I$  is the set of direct import relations (i.e. arcs). If  $(w, z) \in I$ , then it means that module  $w \in M$  directly imports module  $z \in M$ .

---

**Algorithm 1** Modularization Algorithm
 

---

**Input:** set  $Ent$  of EXPRESS entities  
 set  $Enu$  of EXPRESS enumerations  
 set  $S$  of EXPRESS selects  
 set  $D$  of EXPRESS defined data types  
 set of supertypes  $sup(t)$  of EXPRESS data type  $t \in (Ent \cup Enu \cup S \cup D)$   
 set of items  $it(s)$  belonging to the EXPRESS select  $s \in S$   
 set  $attr(e)$  of attributes of entity  $e \in Ent$   
 data type  $ran(e, a) \in (Ent \cup Enu \cup S \cup D)$  being the range of attribute  $a \in attr(e)$   
 set  $M$  of modules  
 module  $mod(t) \in M$  to which the data type  $t \in (Ent \cup Enu \cup S \cup D)$  is assigned  
 set  $I$  of ordered pairs of modules defining direct import relations

**function** GENMODULEARETO( $Ent, Enu, S, D, sup, it, attr, ran, M, mod$ )

**for all**  $t \in (Ent \cup Enu \cup S \cup D)$  **do** ▷ STEP 1  
 add the OWL axiom defining  $c$  to module  $mod(t)$

**for all**  $t \in (Ent \cup Enu \cup S \cup D)$  **do** ▷ STEP 2  
**for all**  $a \in sup(t)$  **do**  
 add the OWL axiom defining the subsumption relationship to the module returned by `SETMODULE( $mod(t), mod(a), I$ )`

**for all**  $s \in S$  **do** ▷ STEP 3  
**for all**  $a \in it(s)$  **do**  
 add the OWL axiom defining the subsumption relation between  $s$  and  $a$  to the module returned by `SETMODULE( $mod(s), mod(a), I$ )`

**for all**  $e \in Ent$  **do** ▷ STEP 4  
**for all**  $a \in attr(e)$  **do**  
 add the OWL axiom defining the attribute relation (i.e. property definition and restrictions) between  $e$  and  $ran(e, a)$  to the module returned by `SETMODULE( $mod(e), mod(ran(e, a)), I$ )`

Apply the transitive reduction to the graph  $G = (M, I)$  ▷ STEP 5

---

The result of Algorithm 1) and 2) is a directed acyclic graph (DAG), i.e. cycles in the graph are avoided. It can be demonstrated that the resulting graph is a DAG by considering that a topological ordering is possible if and only if the graph has no directed cycles. A topological ordering can be generated from the resulting graph because each pair of nodes (i.e. modules) can be ordered, since Algorithm 2 guarantees that there is only one import direction (direct or indirect) between them. Axioms involving atoms belonging to two different modules are added always to the same module, thus solving the problem of circular imports without needing to merge modules.

The resulting graph can be further optimized by applying a transitive reduction [1] that allows to obtain a graph with fewer arcs but the same reachability (cf. STEP 5 of Al-

**Algorithm 2** Set Module Algorithm

---

**Input:** set  $M$  of modules  
priority  $p(m)$  of module  $m \in M$   
set  $I$  of ordered pairs of modules defining direct import relations

**Output:** selected module  
updated set  $I$

**function** SETMODULE( $x, y, I$ )  
**if**  $x = y$  **then**  
  **return**  $x$   
**else**  
  Calculate the transitive closure of graph  $G = (M, I)$  to obtain the set of reachability relations  $R$   
  **if**  $(x, y) \in R$  **then**  
    **return**  $x$   
  **else if**  $(y, x) \in R$  **then**  
    **return**  $y$   
  **else if**  $p(x) > p(y)$  **then**  
    add  $(y, x)$  to the set  $I$ , **return**  $y$   
  **else**  
    add  $(x, y)$  to the set  $I$ , **return**  $x$

---

gorithm 1). In case of a DAG the transitive reduction is unique and consists in a subgraph of the original graph that minimizes the number of arcs, i.e. the number of the imports.

## 5. Experiments

This section presents the experiments related to the generation of a modular ifcOWL ontology from the IFC4 EXPRESS schema<sup>2</sup>. As reported in Table 1, the input modularization is based on the 38 IFC sub-schemas (Figure 1), plus the ontology modules `express`<sup>3</sup> and `list`<sup>4</sup> that are automatically included during the EXPRESS to OWL conversion. Table 1 reports also the priority level associated with each module, as required to execute the algorithm. Three different versions of modularization algorithm have been tested to demonstrate the benefits of the full version presented in Sect.4:

1. *Simple* version, i.e. the modularization algorithm consisting of Algorithm 1 and Algorithm 3 that represents a simplification of Algorithm 2.
2. *Basic* version, the modularization algorithm consisting of Algorithm 1 and Algorithm 2, but without STEP 5 in Algorithm 1.
3. *Full* version, i.e. the modularization algorithm consisting of Algorithm 1 and Algorithm 2.

Algorithm 3, used in the *Simple* version, implements the selection of the module where the OWL axioms are added by looking at the incumbent need, without considering the already set module dependencies. This simplification leads to a higher number of direct import relations (189) compared to the *Basic* version (95). Moreover, the *Simple* version causes the realization of circular import patterns (e.g. modules 10 and 11 import each other), thus disabling the chance to execute a straightforward and deterministic transitive reduction.

<sup>2</sup><http://www.ontoeng.com/modularIfcOWL/>

<sup>3</sup><https://w3id.org/express>

<sup>4</sup><https://w3id.org/list>

**Table 1.** Modules of the ifcOWL ontology with definition of *id* and *priority* level.

Module	IFC Layer	Label	Priority
list	N/A	1	5
express	N/A	2	5
IFCACTORRESOURCE	Resource	3	4
IFCAPPROVALRESOURCE	Resource	4	4
IFCCONSTRAINTRESOURCE	Resource	5	4
IFCCOSTRESOURCE	Resource	6	4
IFCDATETIMERRESOURCE	Resource	7	4
IFCEXTERNALREFERENCERESOURCE	Resource	8	4
IFCGEOMETRICCONSTRAINTRESOURCE	Resource	9	4
IFCGEOMETRICMODELRESOURCE	Resource	10	4
IFCGEOMETRYRESOURCE	Resource	11	4
IFCMATERIALRESOURCE	Resource	12	4
IFCMEASURERESOURCE	Resource	13	4
IFCPRESENTATIONAPPEARANCERESOURCE	Resource	14	4
IFCPRESENTATIONDEFINITIONRESOURCE	Resource	15	4
IFCPRESENTATIONORGANIZATIONRESOURCE	Resource	16	4
IFCPROFILERESOURCE	Resource	17	4
IFCPROPERTYRESOURCE	Resource	18	4
IFCQUANTITYRESOURCE	Resource	19	4
IFCREPRESENTATIONRESOURCE	Resource	20	4
IFCSTRUCTURALLOADRESOURCE	Resource	21	4
IFCTOPOLOGYRESOURCE	Resource	22	4
IFCUTILITYRESOURCE	Resource	23	4
IFCKERNEL	Core	25	3
IFCCONTROLEXTENSION	Core	24	2
IFCPROCESSEXTENSION	Core	26	2
IFCPRODUCTEXTENSION	Core	27	2
IFCSHAREDBLDGELEMENTS	Interoperability	28	1
IFCSHAREDBLDGSERVICEELEMENTS	Interoperability	29	1
IFCSHAREDCOMPONENTELEMENTS	Interoperability	30	1
IFCSHAREDFACILITIESELEMENTS	Interoperability	31	1
IFCSHAREDMGMTLEMENTS	Interoperability	32	1
IFCARCHITECTUREDOMAIN	Domain	33	0
IFCBUILDINGCONTROLSDOMAIN	Domain	34	0
IFCCONSTRUCTIONMGMTDOMAIN	Domain	35	0
IFELECTRICALDOMAIN	Domain	36	0
IFCHVACDOMAIN	Domain	37	0
IFCPLUMBINGFIREPROTECTIONDOMAIN	Domain	38	0
IFCSTRUCTURALANALYSISDOMAIN	Domain	39	0
IFCSTRUCTURALELEMENTSDOMAIN	Domain	40	0

**Algorithm 3** Simple version of Set Module Algorithm

---

**Input:** set  $M$  of modules  
priority  $p(m)$  of module  $m \in M$   
set  $I$  of ordered pairs of modules defining direct import relations

**Output:** selected module  
updated set  $I$

- 1: **function** SETMODULE( $x, y, I$ )
- 2:   **if**  $(x, y) \notin I$  **then**
- 3:     add  $(y, x)$  to the set  $I$
- 4:   **return**  $x$

---

The comparison between the *Basic* and *Full* versions show the impact of the transitive reduction, since the total number of import relations becomes 46. A synthetic comparison of the three algorithm versions is reported in Table 2, showing that a great deal of unnecessary imports can be eliminated. The graph-based representation of the three modular ifcOWL solutions are shown in Figures 2 and 3 highlighting how strongly interconnected are the IFC sub-schemas. Analyzing Figure 3b, it can be noted that:

- there are modules in the Core layer (i.e. `IfcControlExtension` and `IfcProcessExtension`) that are not actually used in any of the modules in the upper levels;
- just two modules in the Resource layer are not imported (directly or indirectly) by `IfcKernel`, i.e. `IfcStructuralLoadResource` and `IfcMaterialResource`;
- just two modules in the Interoperability layer are imported by modules in the Domain layer, i.e. `IfcSharedBldgServiceElements` and `IfcSharedComponentElements`.

**Table 2.** Synthetic results of the three versions of modularization algorithm for the ifcOWL ontology, considering modules 3-40 defined in Table 1

	Simple	Basic	Full
n. modules	38	38	38
total n. imports	189	96	47
max n. imports per module	12	6	2
avg n. imports per module	4.97	2.5	1.21
circular imports	yes	no	no

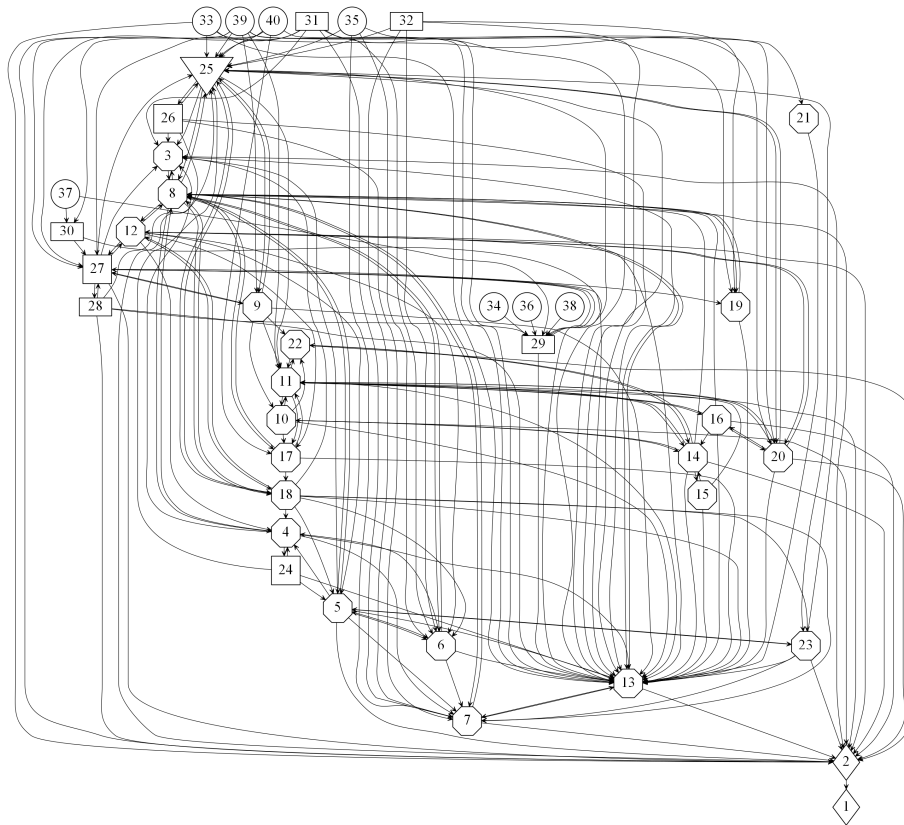
The experiments demonstrate how the proposed algorithm enables to optimize the number of import relations. This result is important because it leads to a decomposed ifcOWL ontology with a minimal number of inter-dependencies, thus easing the selection and extraction of a subset of modules that may better fit the requirements of a user.

Finally, even if the same *Full* version of the algorithm is adopted, different solutions can be obtained based on the priority assigned to the modules. Indeed, the final result of the algorithm is deterministic only if all modules have a different priority. On the other hand, if an import relation is required between two modules having the same priority, then the direction of the import depends on the order of the definitions that are elaborated by Algorithm 1. For example, since modules 14 and 15 need each other (see Figure 2) and have the same priority, the final solution could include that module 15 imports module 14, instead of vice versa as in the experiment (see Figures 3a and 3b).

## 6. Conclusions

This paper presented an approach to generate a modular version of an OWL ontology that is automatically converted from an EXPRESS schema. The attention was focused on the case of the IFC schema given the large size of the resulting ifcOWL ontology. A modular version of ifcOWL can help to solve practical problems related to its usability and the scalability of software applications based on it. Moreover, the modularization algorithm can be used also to extract fragments of the ifcOWL that are relevant for the specific applications. This can be achieved by missing to assign some EXPRESS data types to any module. Further developments will address:

- the generation of additional OWL modules to convert the IFC Property Sets that are currently not included in the IFC EXPRESS schema
- the investigation of other modularization strategies, e.g. the second one presented in Sect.3, and the introduction of criteria to at least partially control the definition of dependencies between modules, e.g. by optimizing their priorities
- testing the benefits of working with a subset of ifcOWL modules from a computational perspectives



**Figure 2.** Modular ifcOWL ontology resulting from the *Simple* version of the algorithm. The labels of the nodes are defined in Table 1

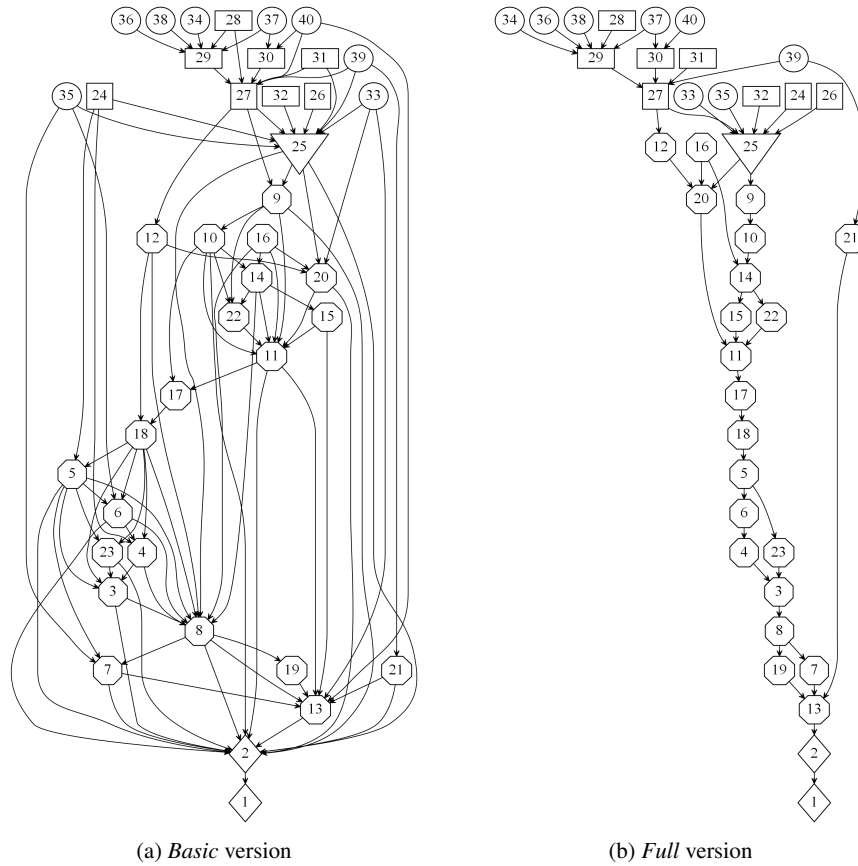
- the integration of a fragment of the ifcOWL ontology with other ontologies
- the comparison of the modular ifcOWL with other ontologies for BIM that are designed to be modular since the beginning [28]
- modularization strategies for Abox ontologies [4].

### Acknowledgments

This work has been partially funded by the Italian research project Smart Manufacturing 2020 within the Cluster Tecnologico Nazionale Fabbrica Intelligente.

### References

- [1] A. V. Aho, M. R. Garey, and J. D. Ullman. The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1(2):131–137, 1972.



**Figure 3.** Modular ifcOWL ontology resulting from the *Basic* (a) and *Full* (b) versions of the algorithm. The labels of the nodes are defined in Table 1

- [2] P. P. F. Barcelos, G. Guizzardi, A. S. Garcia, and M. E. Monteiro. Ontological evaluation of the itu-t recommendation g.805. In *2011 18th International Conference on Telecommunications*, pages 232–237, May 2011.
- [3] J. Bateman, S. Borgo, K. Lüttich, C. Masolo, and T. Mossakowski. Ontological modularity and spatial diversity. *Spatial Cognition and Computation*, 7(1):97–128, 2007.
- [4] J. Beetz, J. van Leeuwen, and B. de Vries. Ifcowl: A case of transforming express schemas into ontologies. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 23(1):89101, 2009.
- [5] C. Bezerra, F. Freitas, J. Euzenat, and A. Zimmermann. ModOnto: A tool for modularizing ontologies. In *Proc. 3rd workshop on ontologies and their applications (Wonto)*, page No pagination., Salvador de Bahia, Brazil, Oct. 2008. bezerra2008a.
- [6] M. Bhatt, J. Hois, and O. Kutz. Ontological modelling of form and function for architectural design. *Applied Ontology*, 7(3):233–267, 2012.
- [7] S. Borgo. Goals of modularity: A voice from the foundational viewpoint. In *WoMO*, pages 1–6, 2011.
- [8] S. Borgo, E. M. Sanfilippo, A. Šojić, and W. Terkaj. *Ontological Analysis and Engineering Standards: An Initial Study of IFC*, pages 17–43. Springer International Publishing, Cham, 2015.
- [9] B. Cuenca Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Extracting modules from ontologies: A logic-based approach. In H. Stuckenschmidt, C. Parent, and S. Spaccapietra, editors, *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, pages 159–186. Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

- [10] M. d'Aquin, A. Schlicht, H. Stuckenschmidt, and M. Sabou. Criteria and evaluation for ontology modularization techniques. In H. Stuckenschmidt, C. Parent, and S. Spaccapietra, editors, *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, pages 67–89, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
- [11] P. Doran, V. Tamma, and L. Iannone. Ontology module extraction for ontology reuse: An ontology engineering perspective. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management, CIKM '07*, pages 61–70, New York, NY, USA, 2007. ACM.
- [12] M. S. Fox and M. Gruninger. Enterprise modeling. *AI magazine*, 19(3):109, 1998.
- [13] B. C. Grau, I. Horrocks, Y. Kazakov, and U. Sattler. Just the right amount: Extracting modules from ontologies. In *Proceedings of the 16th International Conference on World Wide Web, WWW '07*, pages 717–726, New York, NY, USA, 2007. ACM.
- [14] International Organization for Standardization. ISO 10303-11: Industrial automation systems and integration - Product data representation and exchange - Part 11: Description methods: The EXPRESS language reference manual, 2004. Available online: [http://www.iso.org/iso/iso\\_catalogue/catalogue.tcl/catalogue\\_detail.htm?csnumber=38047](http://www.iso.org/iso/iso_catalogue/catalogue.tcl/catalogue_detail.htm?csnumber=38047) (Last accessed on 14 September 2017).
- [15] R. Kontchakov, L. Pulina, U. Sattler, T. Schneider, P. Selmer, F. Wolter, and M. Zakharyashev. Minimal module extraction from dl-lite ontologies using qbf solvers. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI'09*, pages 836–841, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.
- [16] T. Liebich, Y. Adachi, J. Forester, J. Hyvarinen, S. Richter, T. Chipman, M. Weise, and J. Wix. Industry Foundation Classes IFC4 official release, 2013. Available online: <http://www.buildingsmart-tech.org/ifc/IFC4/final/html/index.htm>. Last accessed on 14 January 2015.
- [17] B. Motik, B. C. Grau, I. Horrocks, Z. Wu, A. Fokoue, and C. Lutz. Owl 2 web ontology language profiles (second edition), W3C Recommendation 11 December 2012. Available online: <https://www.w3.org/TR/owl2-profiles/> (Last accessed on 12 July 2017).
- [18] S. Oh and H. Y. Yeom. Evaluation criteria ontology modularization tools. In *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 365–368, Aug 2011.
- [19] C. Parent and S. Spaccapietra. An overview of modularity. In H. Stuckenschmidt, C. Parent, and S. Spaccapietra, editors, *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, pages 5–23, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.
- [20] J. Pathak, T. M. Johnson, and C. G. Chute. Survey of modular ontology techniques and their applications in the biomedical domain. *Integr. Comput.-Aided Eng.*, 16(3):225–242, Aug. 2009.
- [21] P. Pauwels, T. Krijnen, W. Terkaj, and J. Beetz. Enhancing the ifcowl ontology with an alternative representation for geometric data. *Automation in Construction*, 80:77 – 94, 2017.
- [22] P. Pauwels and W. Terkaj. EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. *Automation in Construction*, 63:100–133, 2016.
- [23] P. Pauwels, S. Zhang, and Y.-C. Lee. Semantic web technologies in aec industry: A literature overview. *Automation in Construction*, 73(Supplement C):145 – 165, 2017.
- [24] M. H. Rasmussen, P. Pauwels, C. A. Hvid, and J. Karlshøj. Proposing a Central AEC Ontology that allows for Domain Specific Extensions. In *LC3 2017: Volume I Proceedings of the Joint Conference on Computing in Construction (JC3)*, pages 237–244, Heraklion, Greece, July 2017.
- [25] F. Ruy, R. Falbo, M. Barcellos, and G. Guizzardi. An ontological analysis of the iso/iec 24744 meta-model. *Frontiers in Artificial Intelligence and Applications*, 267:330–343, 2014. cited By 4.
- [26] A. Sojic, W. Terkaj, G. Contini, and M. Sacco. Modularising ontology and designing inference patterns to personalise health condition assessment: the case of obesity. *Journal of Biomedical Semantics*, 7(1):12, May 2016.
- [27] H. Stuckenschmidt, C. Parent, and S. Spaccapietra. *Modular ontologies: concepts, theories and techniques for knowledge modularization*, volume 5445. Springer, 2009.
- [28] W. Terkaj, G. F. Schneider, and P. Pauwels. Reusing domain ontologies in linked building data: the case of building automation and control. In *Proceedings of the 8th International Workshop on Formal Ontologies Meet Industry*, 2017.
- [29] P. Wennerberg, K. Schulz, and P. Buitelaar. Ontology modularization to improve semantic medical image annotation. *Journal of Biomedical Informatics*, 44(1):155 – 162, 2011. Ontologies for Clinical and Translational Research.



# Reusing Domain Ontologies in Linked Building Data: the Case of Building Automation and Control

Walter TERKAJ<sup>a,1</sup>, Georg Ferdinand SCHNEIDER<sup>b</sup> and Pieter PAUWELS<sup>c</sup>

<sup>a</sup>*Institute of Industrial Technologies and Automation (ITIA-CNR), Milan, Italy*

<sup>b</sup>*Fraunhofer Institute for Building Physics IBP, Nuremberg, Germany*

<sup>c</sup>*Ghent University, Ghent, Belgium*

**Abstract.** Linked data and semantic web technologies are gaining impact and importance in the Architecture, Engineering, Construction and Facility Management (AEC/FM) industry. Whereas we have seen a strong technological shift with the emergence of Building Information Modeling (BIM) tools, this second technological shift to the exchange and management of building data over the web might be even stronger than the first one. In order to make this a success, the AEC/FM industry will need strong and appropriate ontologies, as they will allow industry practitioners to structure their data in a commonly agreed format and exchange the data. Herein, we look at the ontologies that are emerging in the area of Building Automation and Control Systems (BACS). We propose a BACS ontology in strong alignment with existing ontologies and evaluate how it can be used for capturing automation and control systems of a building by modeling a use case.

**Keywords.** Linked Data, Semantic Web, Building Data, Building Automation Systems, Control Logic

## 1. Introduction

One of the major challenges in the engineering of complex cyber-physical systems is the interoperability between software applications that are used to engineer and manage these systems. Buildings constitute one of these systems and their engineering requires the interaction of a multitude of stakeholders over several stages of the life cycle [8]. Vast amounts of data are generated during this process. These data are typically stored across various formats and information silos being both in machine-readable formats (e.g. XML, STEP), or analog formats (e.g. drawings printed on paper). These data may include the design specification of the actual layout of a building, product data on the commissioned technical equipment, weather data [26] and sensor data obtained through the installation and operation of a Building Automation System (BAS) [23].

The ontology-based modeling approach and its related use of semantic web technologies seem to be a promising path towards addressing the prevalent heterogeneity of

---

<sup>1</sup>Corresponding Author: Institute of Industrial Technologies and Automation (ITIA-CNR), Milan, Italy; E-mail: walter.terkaj@itia.cnr.it

data in the Architecture, Engineering, Construction and Facility Management (AEC/FM) industry [5,29,15]. This is mostly undertaken by providing a common semantically well defined layer enabling seamless information exchange and linkage across domains. However, a plethora of ontologies is defined to provide conceptualizations of this domain (see Section 2). These ontologies can be used to represent data about buildings in a structured manner and help facilitating the information exchange between different stakeholders.

However, the existence of a multitude of ontologies that partly overlap in the scope of AEC/FM inhibits the wide-spread adoption of the technology throughout the industry. Several ontologies without standardization or consensus can only partially support the AEC/FM industry. Under the umbrella of the World Wide Web Consortium (W3C), the Linked Building Data Community Group (LBD) was initiated in 2016 to address this problem. This group aims at becoming a Working Group and thus effectively develop standard ontologies for the AEC/FM sector. When becoming a Working Group, the following mission will prevail, as listed in the editor draft of the charter [10]:

- to determine how building information can best be integrated with other data on the Web;
- to determine how machines and people can discover that different facts in different datasets relate to the same building, especially when *building* is expressed in different ways and levels of granularity;
- to identify and assess existing methods and tools and then create a set of best practices for their use;
- where desirable, to complete the standardization of informal technologies already in widespread use.

The community group aims at achieving this mission through a number of deliverables. This includes a central ontology (not an upper ontology) that allows to express the building topology of any building (Site - Building - Storey - Space - Element): the Building Topology Ontology (BOT) [20]. In addition, a PRODUCT ontology, a GEOM ontology, and a PSET ontology are being defined, allowing to represent product data, geometric representations, and properties, respectively. The set of four ontologies form the core of the work that is aligned with W3C recommendations for adjacent domains, e.g. the geospatial ontologies [24], SAREF [6], DogOnt [1], and the Semantic Sensor Network (SSN [4]). The group also aligns with the *W3C Data on the Web Best Practices* [9] for the adoption of Semantic Web Technologies (SWT) in the domain of building data.

One subgroup of the LBD Community Group focuses on the formal modeling of Building Automation and Control Systems (BACS). This partially reflects the need to integrate tools supporting the monitoring and automation of buildings being in the need of smart systems to automatically control technical equipment and improve building operation in terms of energy efficiency and indoor comfort.

As a first result of this work we present in this paper a modular ontology where we integrate information from Building Information Modelling (BIM) and BAS: notably building elements, sensors and actuators, devices of BAS and control logic. The work mainly reuses and aligns existing domain ontologies to comprise domain information in one common knowledge base for the control and automation of buildings.

In Sect.2, we provide an overview on existing domain ontologies regarding smart appliances and BAS. Then in Sect.3, we present the BACS Ontology. In Sect.4, we apply the BACS ontology to model an application scenario of state-based room control in a fictional BAS. Finally, we draw the conclusions in Sect.5.

## 2. Related Works

In this section we review existing ontology modeling approaches with a special focus on smart appliances and building automation systems. A comprehensive review on the usage of Semantic Web Technologies in the building domain is presented in [15], also including a discussion on ontologies in the BAS domain.

### 2.1. General Systems and Building Information Modelling (BIM)

The Industry Foundation Classes (IFC) standard<sup>2</sup> comprises a well accepted, open model for the information exchange when applying BIM in the AEC/FM-industry. The data model has already been converted to the Web Ontology Language (OWL) as the ifcOWL ontology [14]. Another set of ontologies including building information is available in the knowledge model for Smart Energy Aware Systems (SEAS) [13]. This includes ontology modules related to building automation and control, such as modules *DeviceOntology*, *OptimizationOntology*, and *FailableSystemOntology*.

An approach to align existing ontologies in the AEC domain is represented by the Building Topology Ontology (BOT) [20] that is part of the work conducted by the LBD Community Group. This ontology can be aligned with several of the mentioned ontologies, using some of the ontology alignment approaches proposed in [20]. In this scope, a BACS ontology is needed as well to model how appliances and devices can be used to automate and control the building and its components.

### 2.2. BACS and Energy-related

Several works focused on the energy management of facilities [12], such as airports [29]. An approach for domotics intelligent devices (DogOnt) was proposed by [1]. The integration of buildings with grid and energy market information is tackled in the ThinkHome ontology [21]. An approach to integrate device descriptions on BAS devices, functional specification with adjacent domains of BIM was developed in the BASont ontology [16]. An approach to formalize semantic tags by means of ontology is described in the Haystack Tagging Ontology (HTO) [3].

Among the various ontologies related to BACS and smart appliances, the Smart Appliances REference (SAREF) ontology [6] unifies common accepted conceptualisations into one reference ontology. SAREF4BLDG [17] is an extension for the building domain.

### 2.3. Sensor data and Control Logic

A well established ontology for the formal specification of sensor data is the W3C Semantic Sensor Network (SSN) ontology [4]. The SSN ontology has been recently updated [11] by including also the more general module SOSA (Sensor, Observation, Sample, and Actuator) making it possible to model key concepts also for the BACS domain, including sensor, actuator, observations, observable properties and results. The SSN ontology is included in the proposed BACS ontology (see Section 3).

Within the reported approaches it may be observed that typically taxonomies are used to describe the actual control behaviour of a certain control logic in a BACS. A

---

<sup>2</sup><http://www.buildingsmart-tech.org/ifc/IFC4/Add1/>

September 2017

modelling effort to specify UML state machines, a well known modeling approach for state-based control logic, as an OWL ontology was presented by Dolog [7].

#### 2.4. Summary

The reported ontologies form a comprehensive board of ontologies that can be reused to cover the BACS domain by spanning the description of building elements and equipment, sensors and actuators, BAS, and control logic. The ontology reuse, even if often applied in a limited fashion, represents a key best practice that was followed in this work.

### 3. The Building Automation and Control System Ontology

The proposed BACS ontology aims at supporting the modeling of the following information requirements:

- control behavior in a BAS (both discrete and continuous), including the sense-comprehend-actuate pattern in closed-loop control logic and the comprehend-actuate pattern in open-loop control logic;
- physical devices of Building Automation Systems (BAS) and their location in the building as well as affiliation to technical equipment;
- smart appliances;
- logical topology in a BAS.

The architecture of the BACS ontology exploits the ontology *reuse* and *modularity* principles [25]. The modular architecture consists of the following Terminological (Tbox) modules, as shown in Figure 1:

- *statistics*, defining basic concepts about probability distributions and descriptive statistics
- *fsm*, the ontology for Finite State Machine
- *sosa*, the Sensor, Observation, Sample, and Actuator (SOSA) Ontology
- *ssn*, the Semantic Sensor Network Ontology
- *expression*, a novel ontology formalizing algebraic and logical expressions
- *osph*, a novel ontology modeling object states and performance history
- *list*, ontology defining the set of entities used to describe the OWL list pattern
- *express*, ontology that maps the key concepts of EXPRESS language to OWL
- *ifcmr*, fragment of the ifcOWL ontology (version IFC4) generated from the EXPRESS sub-schema *IfcMeasureResource*
- *bot*, the Building Topology Ontology
- *bacs*, a novel domain ontology for building automation and control

All these modules are available on the web (see Table1). Such composition of modules was selected by carefully reviewing candidate ontologies (see Sect. 2) against the following criteria: (1) minimum overlap with each other, (2) possibly W3C standards (e.g. *ssn*) and (3) meeting the defined information requirements.

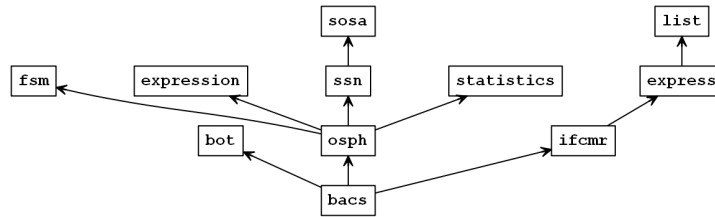
The ontology modules *fsm*, *sosa*, *ssn*, and *bot* were already mentioned in Sect.2.

The group of modules *list*, *express*, and *ifcmr* supports the definition of quantity values. This ontology is a fragment of the ifcOWL ontology [27] generated from the IFC

sub-schema `IfcMeasureResource` according to the method presented in [27]. These modules could be replaced by other similar ontologies, such as the Ontology of Units of Measure (OM) [22] or the QUDT ontology [19].

The `statistics` module is a minimal ontology defining the basic classes and properties needed to model data related to descriptive statistics and probability distributions. As an alternative, the larger and more complex STATO ontology [2] could be adopted.

The three novel modules proposed in this article (i.e. `expression`, `osph`, `bacs`) are described in Sect.3.1, whereas the overall integration and alignment is presented in Sect.3.2. The prefixes of namespaces employed in the next sections are defined in Table 1.



**Figure 1.** Modular architecture of the BACS Ontology. Arrows represent import relations.

**Table 1.** Namespaces and prefixes

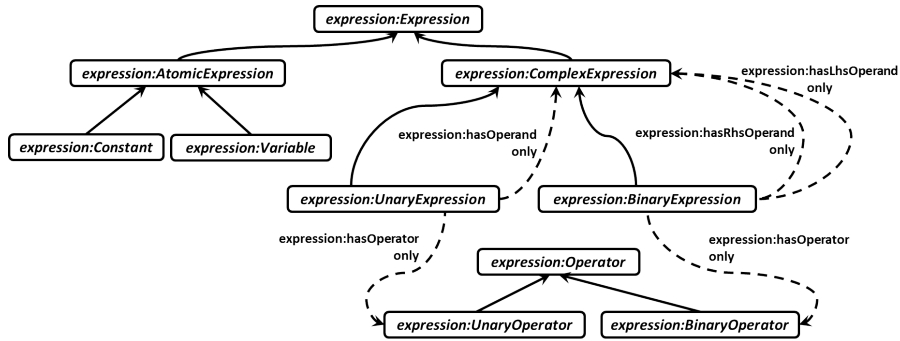
Prefix	Value	Prefix	Value
bacs	<a href="http://www.ontoeng.com/bacs#">http://www.ontoeng.com/bacs#</a>	owl	<a href="http://www.w3.org/2002/07/owl#">http://www.w3.org/2002/07/owl#</a>
bot	<a href="https://w3id.org/bot#">https://w3id.org/bot#</a>	rdf	<a href="http://www.w3.org/1999/02/22-rdf-syntax-ns#">http://www.w3.org/1999/02/22-rdf-syntax-ns#</a>
expression	<a href="http://www.ontoeng.com/expression#">http://www.ontoeng.com/expression#</a>	rdfs	<a href="http://www.w3.org/2000/01/rdf-schema#">http://www.w3.org/2000/01/rdf-schema#</a>
expr	<a href="https://w3id.org/express#">https://w3id.org/express#</a>	sosa	<a href="http://www.w3.org/ns/sosa/">http://www.w3.org/ns/sosa/</a>
fsm	<a href="http://www.learninglab.de/~dolog/fsm/fsm.owl#">http://www.learninglab.de/~dolog/fsm/fsm.owl#</a>	ssn	<a href="http://www.w3.org/ns/ssn/">http://www.w3.org/ns/ssn/</a>
ifcmr	<a href="http://www.ontoeng.com/IFC4-IfcMeasureResource#">http://www.ontoeng.com/IFC4-IfcMeasureResource#</a>	statistics	<a href="http://www.ontoeng.com/statistics#">http://www.ontoeng.com/statistics#</a>
list	<a href="https://w3id.org/list#">https://w3id.org/list#</a>	xsd	<a href="http://www.w3.org/2001/XMLSchema#">http://www.w3.org/2001/XMLSchema#</a>
osph	<a href="http://www.ontoeng.com/osph#">http://www.ontoeng.com/osph#</a>		

### 3.1. Novel Ontology Modules

The `expression` ontology aims at formalizing algebraic and logical expressions. A generic expression can be decomposed into atomic, unary, and binary expressions. An atomic expression can be a constant or a variable. This ontology was developed to provide a simple formal definition of expressions that can be used also as conditions to be met before a transition is triggered (see Sect.3.2). The classes and properties of the `expression` ontology are sketched in Figure 2.

The `osph` ontology is the evolution and generalization of an early proposal that was based on the `ifcOWL` ontology [28]. This ontology plays a key role because it models Object States and Performance History (OSPH), while integrating the `fsm`, `statistics`, `ssn`, and `expression` modules. The following classes are defined in the `osph` ontology:

- `osph:ObjectDefinition` is an abstract class whose definition resembles that of `IfcObjectDefinition` in the IFC standard.
- `osph:ObjectHistory` defines a history interval in the lifecycle of a generic object that is assigned via the object property `osph:isHistoryOf`. An interval can be de-

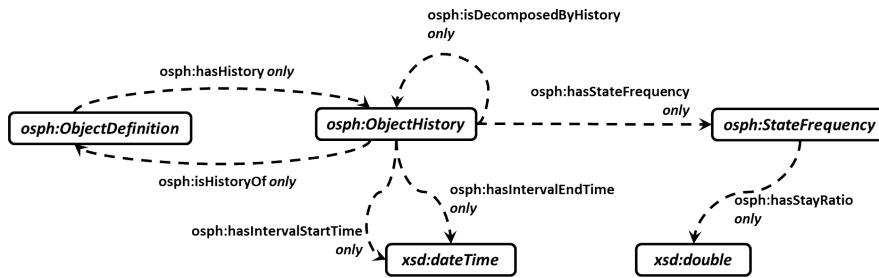


**Figure 2.** Classes and relations in the expression ontology. Dashed lines represent OWL restrictions, whereas solid lines represent subsumption relations.

composed into other intervals via the property `osph:isDecomposedByHistory`. A history interval can be characterized by a start and end time as `xsd:dateTime` via the properties `osph:hasIntervalStartTime` and `osph:hasIntervalEndTime`, respectively. A history interval can be related to individuals of `osph:StateFrequency` via the property `osph:hasStateFrequencies`.

- `osph:StateFrequency` describes the stay of an object in a specific state during a history interval.
- `osph:UnitOfMeasurement` is a class defining a generic unit of measurement.

The relations between these classes are shown in Figure 3, whereas the relations with classes defined in other modules are described in Sect.3.2.

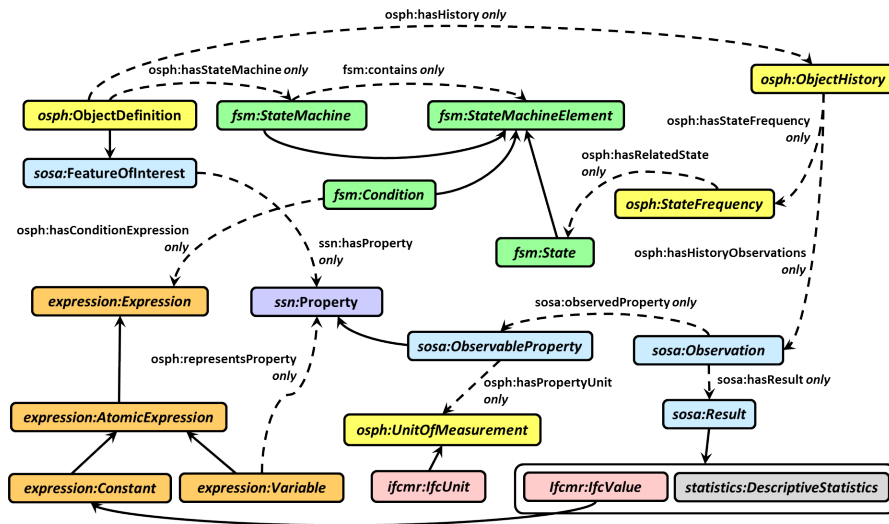


**Figure 3.** Classes and relations in the osph ontology, where dashed lines represent OWL restrictions.

The `bacs` module defines specializations of classes to directly support the instantiation of the use case (Sect.4). Classes `bacs:LightSensor` and `bacs:TemperatureSensor` represent light and temperature sensors, respectively. The class `bacs:SpaceProperty` and its subclasses `bacs:SpaceIlluminanceProperty` and `bacs:SpaceTemperatureProperty` model specific properties of a space (e.g. a room). The class `bacs:SpaceObs` and its subclasses `bacs:SpaceIlluminanceObs` and `bacs:SpaceTemperatureObs` are designed to represent observations made in a space. In the future, all these classes will be re-assessed to check if they can be replaced by definitions imported from other dedicated domain ontologies.

### 3.2. Integration

The integration of the various ontology modules represents one of the main contributions in the proposed ontology architecture. Indeed, the novel modules *osph* and *bacs* mainly play the role of ontology mediator creating links between different domains, as it can be noticed in the diagram of Figure 1. The alignments between the various ontology modules are reported in Table 2 by specifying which module is the mediator (i.e. where the alignment is defined), which are the aligned modules, which are the involved classes and properties, and finally providing a description. A generic object (*osph:ObjectDefinition*) can be characterized by a state machine (*fsm:StateMachine*) and also by one or more histories (*osph:ObjectHistory*) that are able to capture the evolution of the object in terms of observations and state. An expression (*expression:Expression*) can be composed by constant values (*ifcmr:IfcValue*) and variables (*expression:Variable*) that are related to measurable properties (*ssn:Property*). The result of an observation (*sosa:Observation*) can be a descriptive statistics (*statistics:DescriptiveStatistics*) or a quantity value (*ifcmr:IfcValue*). The relevant alignments are also graphically represented in Figure 4.



**Figure 4.** Excerpt of classes and relations in the BACS Ontology showing the key alignments. Dashed lines represent OWL restrictions, whereas solid lines represent subsumption relations.

## 4. Application Scenario

### 4.1. Description

The applicability of the proposed BACS ontology was tested against an application scenario that includes the instantiation of building elements, sensors and actuators, automation systems, and control logics. The scenario is motivated by the deployment of auto-

**Table 2.** Alignments between the modules of the BACS ontology.

Mediator	Modules	Classes and Properties	Description
osph	sosa,osph	osph:ObjectDefinition, sosa:FeatureOfInterest	osph:ObjectDefinition is defined as a subclassOf sosa:FeatureOfInterest.
osph	fsm,osph	osph:ObjectDefinition, fsm:StateMachine, osph:ObjectHistory; osph:hasStateMachine, osph:hasHistory	An individual of osph:ObjectDefinition can be linked with an individual of fsm:StateMachine via the property osph:hasStateMachine, and with individuals of osph:ObjectHistory via property osph:hasHistory.
osph	fsm,osph	osph:StateFrequency, fsm:State; osph:hasRelatedState	An individual of osph:StateFrequency is related with an individual of fsm:State via the property osph:hasRelatedState.
osph	sosa,osph	osph:ObjectHistory, sosa:Observation; osph:hasHistoryObservations	A history interval osph:ObjectHistory can be related to individuals of sosa:Observation via the property osph:hasHistoryObservations.
osph	expression, fsm,osph	fsm:Condition, expression:Expression; osph:hasConditionExpression	An individual of class fsm:Condition can be related to an individual of expression:Expression via the object property osph:hasConditionExpression.
osph	expression, ssn,osph	expression:Variable, ssn:Property; osph:representsProperty	An individual of class expression:Variable can be related to an individual of ssn:Property via the object property osph:representsProperty, i.e. a variable can be used to represent the property of a feature.
bacs	bot,osph	osph:ObjectDefinition, bot:Building, bot:Space, bot:Element, bot:Storey	bot:Building, bot:Space, bot:Element, bot:Storey are defined as subclassOf osph:ObjectDefinition.
bacs	bot,sosa	bot:Element, sosa:Sensor	sosa:Sensor is subclassOf bot:Element.
bacs	sosa,bacs	sosa:Sensor,bacs:LightSensor, bacs:TemperatureSensor	bacs:LightSensor and bacs:TemperatureSensor are subclassOf sosa:Sensor.
bacs	bacs,bot, osph	osph:ObjectHistory, bot:Space, bacs:SpaceObs, bacs:SpaceHistory; osph:hasHistoryObservations, osph:isDecomposedByHistory, osph:isHistoryOf	bacs:SpaceHistory is subclassOf osph:ObjectHistory and further specializes the restrictions characterizing osph:ObjectHistory by means of properties osph:hasHistoryObservations, osph:isDecomposedByHistory, osph:isHistoryOf and classes bacs:SpaceObs, bot:Space, bacs:SpaceHistory.
bacs	osph,ifcmr	osph:UnitOfMeasurement, ifcmr:IfcUnit	ifcmr:IfcUnit is subclassOf osph:UnitOfMeasurement.
bacs	sosa,bacs	sosa:ObservableProperty, bacs:SpaceProperty	bacs:SpaceProperty is subclassOf sosa:ObservableProperty.
bacs	sosa,bacs	sosa:Observation,bacs:SpaceObs	bacs:SpaceObs is subclassOf sosa:Observation.
bacs	statistics, ifcmr, sosa	statistics:DescriptiveStatistics, ifcmr:IfcValue, sosa:Result	sosa:Result is a subclassOf the union of ifcmr:IfcValue and statistics:DescriptiveStatistics, i.e. the result of an observation must be either a value or a statistics.
bacs	expression, ifcmr	expression:Constant, ifcmr:IfcValue	ifcmr:IfcValue is a subclassOf expression:Constant.

matic control in the room of a building. The focus is on control logic because it can significantly impact on the energy consumption and comfort conditions.

In the scenario, a room is equipped with a window, a controllable sunblind, a room air temperature sensor and an outdoor illuminance sensor. A finite state machine is designed to control the sunblind depending on the room air temperature and outdoor illuminance, as presented in [18]. The sunblind can be in one of the following states: noShade, nightShadeDeployed, dayShadeDeployed (Figure 5). The following observations have been made in the system:

- room temperature: 20°C at 2017-03-09T08:00:00; 24°C at 2017-03-09T10:00:00
- room illuminance: 90 lx at 2017-03-09T08:00:00; 200 lx at 2017-03-09T10:00:00
- sunblind state: noShade at 2017-03-09T09:30:00



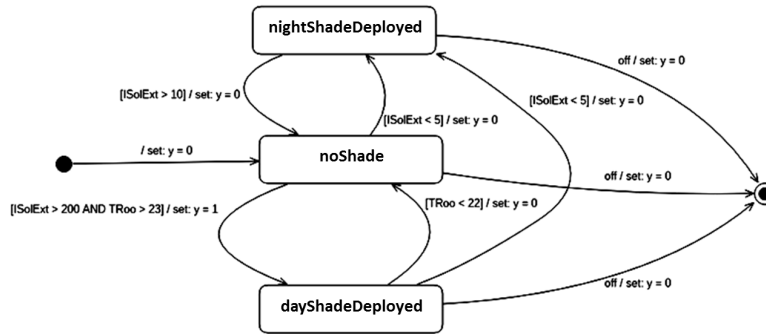


Figure 5. State Machine to control a sunblind in a room [18].

#### 4.2. Instantiation

The Abox ontology module named `bacs_test`<sup>3</sup> imports the `bacs` module and instantiates the application scenario in terms of OWL individuals defined in its namespace. For sake of simplicity, the following fragments in turtle format present only a subset of definitions included in `bacs_test`.

Fragment 1 presents the definition of room, temperature sensor and sunblind. The room is associated with two properties: room temperature and room illuminance. Other definitions include building, building storey, window, and illuminance sensor.

Fragment 2 shows a part of the definition of the finite state machine modeling the control logic of the sunblind. As an example, the details of the transition from state `dayShadeDeployed` to `nightShadeDeployed` are reported. This transition can be triggered only if the guard condition is met, i.e. if the room illuminance is less than 5.0 lx.

Fragment 3 provides an example of how the history of the room can be characterized by observations of its properties thanks to the sensors. In addition, the history of the sunblind considers the evolution of the sunblind state.

Finally, a couple of SPARQL queries are presented to show how the contents of the ontology can be extracted to support business processes, while referring to the prefixes defined in Table 1.

---

```

1 :room a owl:NamedIndividual , bot:Space ;
2   bot:containsElement :lightsensor , :sunblind , :tempsensor ;
3   ssn:hasProperty :room_illuminance_prop , :room_temperature_prop ;
4   bot:adjacentElement :window ; osp:hasHistory :room_history .
5 :tempsensor a owl:NamedIndividual , bacs:TemperatureSensor ;
6   sosa:observes :room_temperature_prop .
7 :sunblind a owl:NamedIndividual , bot:Element ;
8   osp:hasHistory :sb_history ; osp:hasStateMachine :sb_StateMachine .
9 :room_temperature_prop a owl:NamedIndividual , bacs:SpaceTemperatureProperty ;
10  osp:hasPropertyUnit :temperature_unit .
11 :room_illuminance_prop a owl:NamedIndividual , bacs:SpaceIlluminanceProperty ;
12  osp:hasPropertyUnit :illuminance_unit .

```

---

Fragment 1: Excerpt of space and elements instantiation

<sup>3</sup>[http://www.ontoeng.com/bacs\\_test](http://www.ontoeng.com/bacs_test)

September 2017

---

```
1 :sb_StateMachine a owl:NamedIndividual , fsm:StateMachine ;
2   fsm:contains :sb_InitialState, :sb_dayShadeDeployed, :sb_nightShadeDeployed,
3     :sb_noShade, :sb_trDayNight, :sb_trDayNo, :sb_trInitNo, :sb_trNightNo,
4     :sb_trNoDay, :sb_trNoNight, :sb_trDayNight_guard, :sb_trDayNo_guard,
5     :sb_trNightNo_guard, :sb_trNoDay_guard, :sb_trNoNight_guard.
6 :sb_dayShadeDeployed a owl:NamedIndividual , fsm:Simple .
7 :sb_nightShadeDeployed a owl:NamedIndividual , fsm:Simple .
8 :sb_trDayNight a owl:NamedIndividual , fsm:Transition ;
9   fsm:Source :sb_dayShadeDeployed ; fsm:Target :sb_nightShadeDeployed ;
10  fsm:TransitionGuard :sb_trDayNight_guard .
11 :sb_trDayNight_guard a owl:NamedIndividual , fsm:Guard ;
12   fsm:GuardCondition :sb_trDayNight_condition .
13 :sb_trDayNight_condition a owl:NamedIndividual , fsm:Condition ;
14   osph:hasConditionExpression :sb_trDayNight_expr .
15 :sb_trDayNight_expr a owl:NamedIndividual , ex:BinaryExpression ;
16   ex:hasLhsOperand :room_illumiance_var ;
17   ex:hasOperator ex:LESSTHAN ; ex:hasRhsOperand :illumA .
18 :illumA a owl:NamedIndividual , ifcmr:IfcIlluminanceMeasure ;
19   express:hasDouble "5.0"^^xsd:double .
```

---

**Fragment 2:** Excerpt of sunblind state machine instantiation

---

```
1 :room_history a owl:NamedIndividual , bacs:SpaceHistory ;
2   osph:isDecomposedByHistory :room_history_int1 , :room_history_int2 ;
3   osph:hasIntervalStartTime "2017-03-09T08:00:00"^^xsd:dateTime .
4 :room_history_int1 a owl:NamedIndividual , bacs:SpaceHistory ;
5   osph:hasHistoryObservations :room_illum1 , :room_temp1 ;
6   osph:hasIntervalStartTime "2017-03-09T08:00:00"^^xsd:dateTime .
7 :room_illum1 a owl:NamedIndividual , bacs:SpaceIlluminanceObs ;
8   sosa:hasResult :illum1 ; sosa:madeBySensor :lightsensor ;
9   sosa:observedProperty :room_illumiance_prop .
10 :illum1 a owl:NamedIndividual , ifcmr:IfcIlluminanceMeasure ;
11   express:hasDouble "90.0"^^xsd:double .
12 :sb_history a owl:NamedIndividual , osph:ObjectHistory ;
13   osph:hasStateFrequencies :statefreq1 ;
14   osph:hasIntervalStartTime "2017-03-09T09:30:00"^^xsd:dateTime .
15 :statefreq1 a owl:NamedIndividual , osph:StateFrequency ;
16   osph:hasRelatedState :noShade ; osph:hasStayRatio "1.0"^^xsd:double .
```

---

**Fragment 3:** Excerpt of room and sunblind history instantiation

The query in Fragment 4 gets the elements in the building and the associated state machine (if existing). The query in Fragment 5 explores the finite state machine of any element in a room and returns the state machine components (e.g. states, transitions, guards) and further details about transitions.

---

```
1 SELECT distinct ?building ?storey ?room ?elem ?statemach
2 WHERE {
3   ?building rdf:type/rdfs:subClassOf* bot:Building.
4   ?building bot:hasStorey ?storey. ?storey bot:hasSpace ?room .
5   ?room bot:adjacentElement|bot:containsElement ?elem .
6   OPTIONAL{ ?elem osph:hasStateMachine ?statemach .}
7 }
```

---

**Fragment 4:** SPARQL query to get the elements in the building

---

---

```
1 SELECT distinct ?statemach ?fsmelem ?class ?source ?target ?guard ?condition
2 WHERE {
3   ?statemach rdf:type/rdfs:subClassOf* fsm:StateMachine .
4   ?statemach fsm:contains/fsm:hasStateMachineElement* ?fsmelem .
5   ?fsmelem rdf:type ?class . FILTER ( ?class != owl:NamedIndividual ) .
6   OPTIONAL{
7     ?class rdfs:subClassOf* fsm:Transition .
8     ?fsmelem fsm:Source ?source . ?fsmelem fsm:Target ?target .
9     OPTIONAL{
10      ?fsmelem fsm:TransitionGuard ?guard . ?guard fsm:GuardCondition ?condition .}
11  }}
```

---

---

**Fragment 5:** SPARQL query to get the components of a state machine

## 5. Conclusions

This paper presented a modular ontology to model the domain of building control and automation, demonstrating its applicability in a test case. The architecture of the ontology will be further tested, aiming at becoming a W3C recommendation. Reusing ontologies through integration and alignment is promising to tackle the building control and automation domain, but best practices and guidelines from ontology engineering will be further investigated. The proposed test case focuses on the building automation domain, but, as automation is an essential component of many industries (process industry, manufacturing industry, etc.), the reuse of the ontology in other domains will be studied. In addition, future works will address:

- testing more complex cases requiring the interaction between smart objects;
- preparation of a library of general purpose SPARQL queries and update to support the use of the BACS ontology (e.g. extraction of object history, extraction of expressions, triggering and execution of a control action);
- integration with other ontologies specializing elements, sensors and actuators, and defining concepts related to geometry (e.g. placement, representation of objects).

## Acknowledgments

This work has been partially funded by the Italian research project “Efficientamento dei processi di produzione e gestione integrata di utenze energivore con fonti rinnovabili e sistemi di accumulo mediante periferiche ICT in un contesto Smart District” within the “Piano Triennale 2015-17 della Ricerca di Sistema Elettrico Nazionale”.

## References

- [1] D. Bonino and F. Corno. DogOnt - Ontology Modeling for Intelligent Domestic Environments. *Lecture Notes in Computational Science*, 5318:790–803, 2008.
- [2] O. Burke, A. Gonzalez-Beltran, and P. Rocca-Serra. STATistics Ontology, 2016. Available online: <http://bioportal.bioontology.org/ontologies/STATO> (Last accessed on 21 July 2017).
- [3] V. Charpenay, S. Kabisch, D. Anicic, and H. Kosch. An ontology design pattern for iot device tagging systems. In *2015 5th International Conference on the Internet of Things (IOT)*, pages 138–145, Korea.

- [4] M. Compton, P. Barnaghi, L. Bermudez, and et al. The SSN ontology of the W3C semantic sensor network incubator group. *Journal on Web Semantics*, 17:25 – 32, 2012.
- [5] E. Curry, J. O'Donnell, E. Corry, S. Hasan, M. Keane, and S. O'Riain. Linking building data in the cloud: Integrating cross-domain building data using linked data. *Advanced Engineering Informatics*, 27(2):206–219, 2013.
- [6] L. Daniele, F. den Hartog, and J. Roes. Created in Close Interaction with the Industry: The Smart Appliances REference (SAREF) Ontology. In R. Cuel and R. Young, editors, *Formal Ontologies Meet Industry*, volume 225, pages 100–112. Springer International Publishing, Cham, Switzerland, 2015.
- [7] P. Dolog. Model-Driven Navigation Design for Semantic Web Applications with the UML-Guide. In *Proc. ICWE*, pages 75–86, 2004.
- [8] C. Eastman, P. Teicholz, R. Sacks, and K. Liston. *BIM Handbook: A guide to building information modeling for owners, managers, designers, engineers, and contractors*. Wiley, Hoboken NJ, USA, 2008.
- [9] B. Farias Lóscio, C. Burle, and N. Calegari. Data on the Web Best Practices. <https://www.w3.org/TR/dwbp/>, 2017. Last accessed: 2017-07-14.
- [10] L. C. Group. Building Data on the Web Working Group Charter. <https://w3c-lbd-cg.github.io/lbd/charter/>, 2017. Last accessed: 11 July 2017.
- [11] A. Haller, K. Janowicz, S. Cox, D. L. Phuoc, K. Taylor, and M. Lefrançois. Semantic Sensor Network Ontology. <https://www.w3.org/TR/2017/CR-vocab-ssn-20170711/>, 2017. Last accessed: 2017-07-22.
- [12] J. Kaiser and P. Stenzel. eeEmbedded D4.2: Energy System Information Model - ESIM. Technical report, eeEmbedded Consortium, Brussels, Belgium, 2015.
- [13] M. Lefrançois, J. Kalaoja, T. Ghariani, and A. Zimmermann. D2.2: The SEAS Knowledge Model. Technical report, ITEA2 I2004 Smart Energy Aware Systems, Brussels, Belgium, 2017.
- [14] P. Pauwels and W. Terkaj. EXPRESS to OWL for construction industry: Towards a recommendable and usable ifcOWL ontology. *Automation in Construction*, 63:100–133, 2016.
- [15] P. Pauwels, S. Zhang, and Y.-C. Lee. Semantic web technologies in AEC industry: A literature overview. *Automation in Construction*, 73:145–165, 2017.
- [16] J. Ploennigs, B. Hensel, H. Dibowski, and K. Kabitzsch. BASont - A modular, adaptive building automation system ontology. In *IECON 2012 - 38th Annual Conference of IEEE Industrial Electronics*, pages 4827–4833, Montreal, Canada.
- [17] M. Poveda-Villalón and R. Garcia Castro. SAREF extension for building devices. <https://w3id.org/def/saref4bldg#>, 2017. Last accessed: 2017-07-11.
- [18] C. Ptolemaeus, editor. *System Design, Modeling, and Simulation using Ptolemy II*. Ptolemy.org, 2014.
- [19] qudt.org. QUDT. <http://www.qudt.org/>, 2017. Last accessed: 2017-07-22.
- [20] M. H. Rasmussen, P. Pauwels, C. A. Hvid, and J. Karlshøj. Proposing a Central AEC Ontology that allows for Domain Specific Extensions. In *LC3 2017: Volume I Proceedings of the Joint Conference on Computing in Construction (JC3)*, pages 237–244, Heraklion, Greece, July 2017.
- [21] C. Reinisch, M. J. Kofler, F. Iglesias, and W. Kastner. ThinkHome Energy Efficiency in Future Smart Homes. *EURASIP Journal on Embedded Systems*, (104617):1–18, 2011.
- [22] H. Rijgersberg, M. van Assem, and J. Top. Ontology of units of measure and related concepts. *Semant. web*, 4(1):3–13, Jan. 2013.
- [23] T. Sauter, S. Soucek, W. Kastner, and D. Dietrich. The Evolution of Factory and Building Automation. *IEEE Industrial Electronics Magazine*, 5(3):35–48, Sept. 2011.
- [24] SDWWG. Spatial Data on the Web Working Group Charter. <https://www.w3.org/2015/spatial/charter>, 2017. Last accessed: 11 July 2017.
- [25] E. Simperl. Reusing ontologies on the semantic web: A feasibility study. *Data & Knowledge Engineering*, 68(10):905–925, 2009.
- [26] P. Staroch. A weather ontology for predictive control in smart homes. Master's thesis, TU Vienna, Vienna, Austria, <https://paul.staroch.name/thesis/thesis.pdf>, 8 2013. Faculty for Computer Science.
- [27] W. Terkaj and P. Pauwels. A method to generate a modular ifcOWL ontology. In *Proceedings of the 8th International Workshop on Formal Ontologies meet Industry*, 2017.
- [28] W. Terkaj and M. Urgo. Ontology-based modeling of production systems for design and performance evaluation. In *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, pages 748–753, July 2014.
- [29] N. M. Tomašević, M. Č. Batić, L. M. Blanes, M. M. Keane, and S. Vraneš. Ontology-based facility data model for energy management. *Advanced Engineering Informatics*, 29(4):971–984, 2015.

# Business process languages: an ontology-based perspective

Greta ADAMO<sup>a,c</sup> and Stefano BORGO<sup>b</sup> and Chiara DI FRANCESCO MARINO<sup>a</sup> and Chiara GHIDINI<sup>a</sup> and Nicola GUARINO<sup>b</sup> and Emilio M. SANFILIPPO<sup>b</sup>

<sup>a</sup>*FBK-IRST, Via Sommarive 18, 38050 Trento, Italy*

<sup>b</sup>*ISTC-CNR Laboratory for Applied Ontology, Trento, Italy*

<sup>c</sup>*University of Genova, DIBRIS, via Dodecaneso 35, 16146 Italy*

**Abstract.** Business process modelling (BPM) notations describe processes using a graphical representation of process-relevant entities and their interplay. Despite the wide literature on the comparison between different modelling languages, the BPM community still lacks an ontological characterisation of process constructs. Purpose of this paper is to start filling this gap by providing a first ontological analysis of the main business process entities. The analysis and the resulting characterisation aim at illustrating the different perspectives that BPM languages implicitly take on business processes, as well as guiding the modellers in making an appropriate choice when selecting among different notations.

## 1. Introduction

Business process modelling (BPM) notations describe processes using a graphical representation of process-relevant entities and their interplay. If we focus on typical business-to-consumer (B2C) scenarios, examples of languages include well-known imperative languages such as the Business Process Model and Notation (BPMN), the Unified Modeling Language Activity Diagram (UML-AD) and the Event-driven Process Chain (EPC) as well as declarative notations such as the Case Management Model and Notation (CMMN) and DECLARE.<sup>1</sup> Despite the wide literature on process execution semantics and on the comparison between the graphical constructs of different languages [25, 14, 12, 16], the BPM community still lacks a robust ontological characterisation of the entities involved in process models.<sup>2</sup> While some initial efforts have been done towards this direction (see, e.g., [20]), they focus on the analysis of behavioral aspects of process models, thus neglecting other central modelling constructs such as those denoting process participants (data objects, actors and so on). As a result, process participants are exposed to a paradox: on the one hand, they are explicitly referred to within process diagrams; on the

---

<sup>1</sup>Imperative paradigms aim at producing models that describe all allowed flows: every flow that is not specified in the model is implicitly disallowed. Declarative process modelling notations instead allow the production of flexible models obtained by describing constraints on the allowed flows: all flows are allowed provided that they do not violate the specified constraints.

<sup>2</sup>We will interchangeably use the notions of ‘process model’ and ‘process diagram’.

other hand, they are emblematically neglected when explaining or illustrating the very notion of process in the BPM community.

The purpose of the paper is to remove the above paradox, offering an ontological analysis of the various kinds of process participants, with a specific focus on B2C scenarios that will hopefully contribute to the ontological foundations of BPM. The paper is structured as follows. In Section 2, we provide an illustration of different constructs used by imperative and declarative modelling notations; then we identify in Section 3 the constructs referring to process participants, and, after analysing the definition of process in Section 4, we discuss the ontological properties of process participants in Section 5, providing in this way an initial comparison of (some of) the modelling constructs used in different modelling notations (Section 6). This represents a first step toward the illustration of how an ontological analysis enlarged to process participants can support the interpretation of business process diagrams, the comparison between modelling notations, the illustration of the different perspectives that BPM languages implicitly take on business processes, as well as guiding the modellers in making an appropriate choice when selecting among different notations.

## 2. Background

In this section we illustrate the graphical elements of the BPM languages taken into account throughout the paper.<sup>3</sup> As a starting point, we select five amongst the most popular languages that follow the imperative (BPMN, UML-AD, EPC) and declarative (CMMN and DECLARE) paradigms. To support the presentation, we make use of process diagrams illustrating the simple scenario of a customer buying a flight ticket from a travel agency. For the sake of clarity, we “annotate” the diagrams with speech balloons to explicitly indicate the graphical constructs.

*BPMN.* It is a standard language, proposed by the Object Management Group (OMG), to design business processes.<sup>4</sup> BPMN defines a Business Process Diagram (BPD) which includes a set of graphical constructs divided in: (i) flow objects, (ii) data, (iii) connecting objects, and (iv) swimlanes. Flow objects define the behavior of a business process, as the one reported in Figure 1. They divide into *events*, *activities* and *gateways*. Events represent things that happen during a process; they divide into *start*, *intermediate* and *end* events. An activity is a generic term that is used for work to be performed. It can be either atomic (*task*) or compound (*sub-process*). A gateway determines the forking, merging or joining of paths. BPMN 2.0 allows for the explicit modelling of data by means of constructs denoting *data objects*, *data inputs*, *data output* and *data stores*. Flow objects are inter-linked through *connecting objects* which are not further discussed here. *Swimlanes* are used to specify who is responsible for the execution of a certain process.<sup>5</sup> Looking at Figure 1, for example, two swimlanes specify that ‘Customer’ and ‘Travel agency’ will carry out the depicted processes.

---

<sup>3</sup>Note that our analysis focuses on the graphical elements used to draw models and leave out further notions that may be included in the language specification.

<sup>4</sup><http://www.omg.org/spec/BPMN/2.0/>

<sup>5</sup>Because of lack of space we do not introduce the distinction between *pools* and *lanes* here.

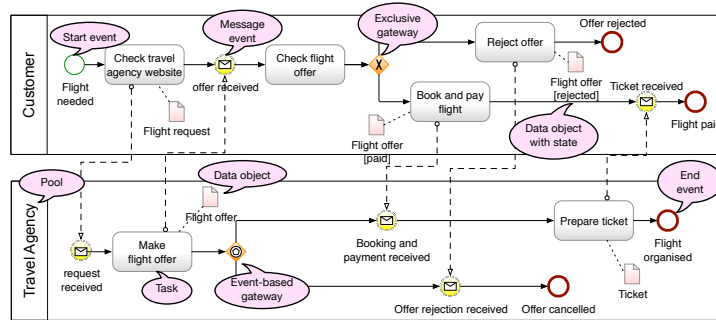


Figure 1. A Business Process Diagram in the BPMN language.

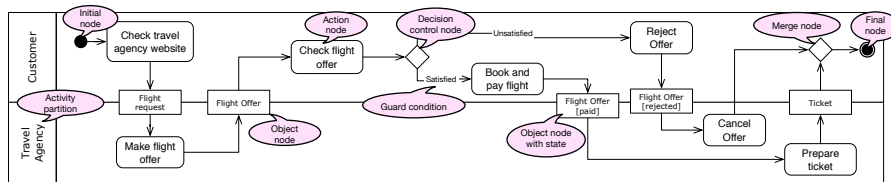


Figure 2. A Business Process Diagram in the UML AD language.

*UML-AD.* It is one of the diagram families of the OMG standardized UML language<sup>6</sup>, whose purpose is to describe the control and data flow as a sequence of activity nodes connected by activity edges (see example in Figure 2).

The nodes responsible of describing the control flow are the *action nodes* and the *control nodes*. While the former represent atomic steps within an activity, the latter allow for controlling the execution flow by means of the AND, OR or XOR logical operations. Additional control flow nodes are used to depict the initial and final nodes of process models. *Object nodes* and *object flows* are the main UML-ADs constructs describing the data flow. The former represent objects at a given point of the flow and, as such, they can also have an associated *state*. The latter are instead used for connecting object nodes to actions. *Activity partitions* are a mechanism for grouping activity nodes that have common characteristics. They are mainly used to define organizational units. Finally, the notation allows for specifying activity pre- and post-conditions, for instance, by annotating activity edges with guards.

*EPC.* It is a modelling language developed in the early 1990s as part of the Architecture of Integrated Information Systems (ARIS) framework [22].

Three types of nodes are responsible for describing the control flow: *function*, *event* and *logical operators* (see Figure 3). Function nodes represent atomic activities and can be considered as the “active” part of a control flow; event nodes stand for the states in which a process happens to be and can be therefore considered as the “passive” part of the control flow. Functions and events alternate, capturing the intuition that states lead to activities, while activities generate states. Finally, the XOR, AND and OR logical operators allow for controlling the execution flow.

<sup>6</sup><http://www.omg.org/spec/UML/>

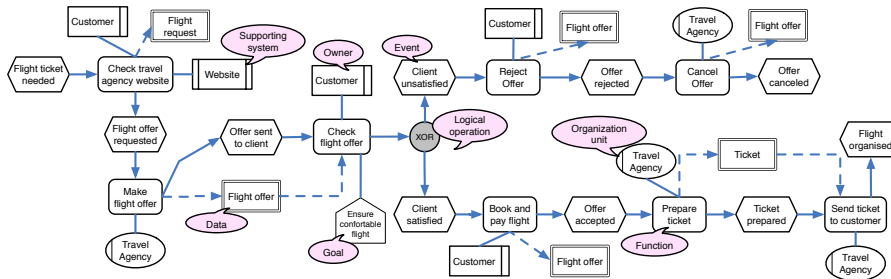


Figure 3. A Business Process Diagram in the EPC language.

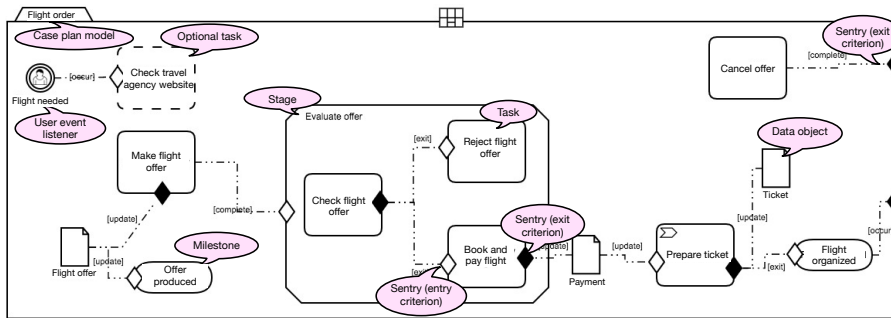


Figure 4. A Business Process Diagram in the CMMN language.

Functions within the control flow can be connected to objects belonging to the other views of an ARIS model, namely the organizational, data, function and product service views. While the exact number of objects differs across different versions of the language,<sup>7</sup> the core modeling constructs usually denote: (i) input and output *data, material, services or resource objects* required or produced by a function; (ii) *owners* who are responsible for a specific function; (iii) *organization units* responsible for a specific function (e.g., a department); and (iv) *supporting systems* upon which a function acts (e.g., a database). Some versions include *goals* that can be connected to specific functions.

**CMMN.** It is a OMG standard for the declarative representation of process models.<sup>8</sup> Its main modelling construct is the *case*, which is described by a case diagram (see Figure 4). Differently from the previous languages, CMMN follows a declarative approach. Thus, rather than describing all the allowed flows of a process from the start to the end, it models cases as composed of process segments (called *stages*) and *tasks*.

A case plan model contains: (possibly discretionary) *tasks, stages, milestones, event listeners, connectors, and sentries*. A task is a unit of work. Stages are plan fragments which can be composite or atomic. A milestone represents an accomplishment which occurs during the process of a case. Events represent something that can happen to a plan construct (e.g., a task cancelled) or in general (timer and user event listener). Connectors

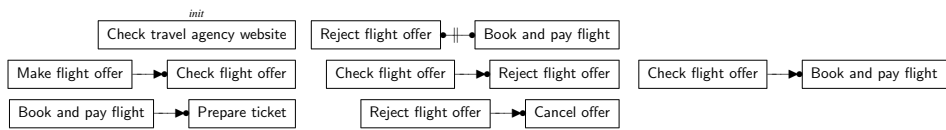
<sup>7</sup>The analysis and diagrams contained in this paper refer to the description provided in [23].

<sup>8</sup><http://www.omg.org/spec/CMMN/1.1/>



TEMPLATE	NOTATION	DESCRIPTION
init(A)	$\overset{init}{\boxed{A}}$	A must occur as a first activity
not coexistence(A,B)	$\boxed{A} \parallel \boxed{B}$	If A occurs, B must not occur and viceversa
precedence(A,B)	$\boxed{A} \rightarrow \boxed{B}$	B can occur only if A has occurred before
response(A,B)	$\boxed{A} \Rightarrow \boxed{B}$	if A occurs, then B occurs after A

**Table 1.** Graphical notation of some DECLARE templates.



**Figure 5.** A DECLARE process diagram.

are used to link different plan items. Finally, sentries represent the entry / exit criteria for path items and can direct the control flow mimicking the AND and OR logical operators.

*Declare.* It is one of the most popular declarative languages for modelling business processes [18]. It grounds on the finite-trace semantics of LTL and aims at capturing variable cases by means of the so-called *patterns*. These are particular LTL formulae that have been singled out for process modelling taking inspiration from [8]. Table 1 reports the graphical notation and a brief description of the patterns used across the paper; Figure 5 provides a DECLARE version of the flight ticket example using the patterns in the table. As we can see from the figure, DECLARE focuses only on the temporal relations between (atomic) activities and does not provide any support for data or actors.<sup>9</sup> Combinations of patterns, such as *precedence* and *non-coexistence*, can be used to direct the control flow using the AND, OR, and XOR logical operators.

### 3. A brief comparison of business process language constructs

We present now a short categorization and comparative summary of the main modelling constructs of the five languages. Modelling constructs are grouped into the three basic categories of process modelling languages, namely the *behavioural* (BEV) category related to the control flow, the *data* (DT) category related to the data flow, and the *organizational* (ORG) category related to process actors. Since the behavioural category is the most articulated, we further describe it in terms of *Functional* (executable pro-active actions), *Event* (what happens), *Flow* (how actions are connected and routed), and *State* (of the world) categories. The result of this grouping is summarized in Table 2.

First, we can observe that all the imperative languages, namely BPMN, EPC, and UML-AD, provide distinctive constructs to indicate the start and the end of a process.

<sup>9</sup>Some proposal to extend DECLARE in order to incorporate non-atomic activities and data centric patterns are presented in [6] and [2], respectively. They are nonetheless more focused on the logical properties of the specification rather than on the definition of modelling constructs suitable for business analysts, and are therefore not considered in this paper.

	BPMN	UML-AD	EPC	CMMN	DECLARE	
BEV	Func	Task Subprocess	Action node Activity	Function Process path	Task Stage	
	Event	Start/End Intermediate Send/receive	Start/End node Accept event action Send signal action	–	Timer User Event Listener	
	Flow	Gateway Sequence Flow Message Flow	Control node Control Flow Object Flow	Logical operators Control Flow Info Flow	Connector Sentry	Connector Pattern
	State	Guard on gateway	Guard on control node Pre- Post-condition on activity	Event Start/End event	Sentry Milestone	–
DT	Data input data output data store	Object node	(I/O) data object	Case file item	–	
ORG	Pool, Lane	Activity Partition	Organization Activity Owner	–	–	

**Table 2.** A comparison among modelling languages

CMMN specifies only exiting conditions while DECLARE allows (but does not force) to specify only initial activities. Not surprisingly, all five languages have graphical symbols for atomic activities. Instead, subprocesses and generic groups of activities are foreseen in all languages but EPCs and DECLARE. Other common constructs are routing nodes, connectors and data objects. CMMN (DECLARE) does not have explicit construct for routing nodes; nonetheless a combination of sentries and connectors, (DECLARE patterns) can be used to route the control flow mimicking the logical operators. The level of details of connectors can vary. Besides the one used to denote connections of the control flow, common to all languages, BPMN, EPC, and UML-AD provide symbols to denote the connections between actors (data) and activities, or the messages exchanged between different activities. Also, the level of detail of data objects can vary; e.g., EPC is particularly rich in defining a taxonomy of data objects. Alternative (OR, XOR) routing nodes can incorporate guards, i.e., conditions that specify which branch to follow, in all languages but EPC, where this role can be taken by states. Actors and organizational constructs are present in imperative languages, although exploiting different notations.<sup>10</sup>

A distinction that is present in BPMN (to some extent also in CMMN) is between active tasks, explicitly performed by the actor specified in a corresponding swimlane, and passive events that occur independently from the actor itself. Other distinctive aspects are (i) the explicit presence of pre- (activation) and post-conditions on activities, which is one of the characteristic features of CMMN and is also foreseen in UML-AD; (ii) the explicit presence of a state, which is a characteristic feature of EPCs where states and functions (tasks) have to interleave, and is also present in CMMN in the form of milestones.

#### 4. On the definition of business process

Davenport [5] defines a business process as “a structured, measured set of activities designed to produce a specific output for a particular customer or market. [...] A process is thus a specific ordering of work activities across time and space, with a beginning and an end, and clearly defined inputs and outputs”. Similar definitions are provided by

<sup>10</sup>Note that CMMN allows to associate organizational entities to cases during the run-time phase.

Hamer and Champy [11], and Johansson et al. [13]. The first states that a business process is “a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer”; the latter says that it is “a set of linked activities that take an input and transform it to create an output. Ideally, the transformation that occurs in the process should add value to the input”. The most modern and popular definition in the BPM literature is likely the one provided by Weske [26], who defines a business process as “a set of activities that are performed in coordination in an organizational and technical environment. These activities jointly realize a business goal. Each business process is enacted by a single organization, but it may interact with business processes performed by other organizations.”

From these definitions, it is rather spread the idea of a business process as a set of **activities** that, together, contribute to achieve a certain **goal** or, more in general, contribute to transform an **input** into a desired **output**. Besides the emphasis on these core aspects, the literature underlines the importance of additional characteristics such as the **value** brought about from the realisation of a certain goal, as well as the **organisational boundaries** in which a process is embedded.

Despite this general agreement, the BPM literature does not provide in depth explanations for what a process is or what its components (e.g., activities) are. Let us consider two illustrative examples. First, it remains unclear whether processes are defined at the type or at the instance (execution) level. A process execution happens in time and has a specific duration. Differently, process types are descriptions and do not unfold in time.<sup>11</sup> Examples of process types are depicted in process diagrams like the ones in the previous sections that describe how the various activities are connected to produce a goal, which is in turn a description of a desired state (e.g., that a certain ticket is purchased). The carrying out of these activities in real-time (possibly monitored by so-called event logs) provides the execution of the process type. As an example, process models may be only descriptive (and not prescriptive) and process executions non compliant with the process model are often considered executions of that process by people from the BPM community. Indeed all the process mining activities tend to define processes at the execution level more than at the process diagram level.

Second, the BPM literature does not provide in depth explanations of the intended semantics of the various modelling constructs, e.g., what an activity is, what its participants are, and how the latter relate to the former. An emblematic example is the lack of characterisation of the relations that occur between the activities in a process. While a business process is invariably understood as an ordered collection of activities, it remains unclear, e.g., whether such activities are only temporally or also causally related, or whether the effects of activities need to be explicitly represented.

## 5. An ontological analysis of some business process constructs

In this section, we provide an analysis of some of the modelling constructs we encountered in the previous sections, with particular emphasis on (the lack of) an ontological characterisation of their properties. We rely on previous works (e.g., [21,20]) for the characterisation of process-like entities (activities, tasks, and events), whereas we shall

---

<sup>11</sup>Concerning the type-execution dichotomy, see the distinction between `Activity` and `ActivityOccurrence` in the *Process Specification Language* (PSL) [9], respectively.

dig into the analysis of process participants. The results of the analysis are used in Section 6 to provide an initial ontologically grounded characterisation of the business process modelling languages summarised in Table 2.

*Activities* In the BPM literature, activities are understood as (atomic or compound) *actions*, consisting of intentional transformations from some initial state (the input) to some other state (the output). The participants to such actions are the entities that take part in these transformations. From the ontological point of view, actions are (specific kinds of) *events*, while their participants are *objects* [4].

A first aspect that needs to be considered concerns the relation between activities, and more in general the way the activities contribute to the achievement of the goal. In general terms, the precedence relation between activities can be a temporal, causal or dependence relation, perhaps depending on the context or scenario to be modelled. For instance, assume that in a slight variation of the example of Section 2, the travel agency splits the activity ‘Make flight offer’ in two subsequent steps ‘Send flight offer to customer’ and ‘Archive offer’ which, for purely organisational reasons, must be executed in this order. This would be a pure temporal relation between the activities in this specific setting and could easily be modified in a revision of the process model. Instead the activity of ‘Paying for a flight’ may be a strong precondition for the ‘Preparation of the ticket’, and so it should necessarily occur before the latter. Nonetheless these relations would be denoted by means of the same connector symbol.

Another aspect that needs to be considered is the (explicit or implicit) representation of the world’s states affected by the designed process. While the definitions of process in Section 4 describe business processes as a way to move from an input (state) to an output (state), they do not specify whether it is desirable to explicitly represent the world’s states affected by the designed process. Nonetheless, the description of intermediate states of affairs can be often found in business process models. Think for instance to functions in EPC diagrams, the status of data objects in BPM and UML-AD, and sentries and milestones in CMMN. Thus, a question that one may ask is whether the (explicit or implicit) representation of the world’s states is necessary to fully characterise a process (model) and what are the characteristics of this representation.

*Participants* From a general perspective, participants can be *physical* or *non-physical* depending on their properties. In fact, the very same activity may involve several types of objects as participants: physical objects (e.g., the knife used to cut a piece of bread); information objects (e.g., personal data involved in submitting a request); agents and/or organizations playing certain roles (e.g., an administrative employee receiving a form).

A physical participant, whenever it exists<sup>12</sup>, is located in the physical space. Differently, non-physical participants lack physical locations, although they are present in time. A person is an example of physical participant, whereas the content of a person’s ID, which is an information object (see below), is a non-physical participant. Information objects are rather common in business processes and are represented by means of data objects modelling constructs. In applied ontology, only a few systems [24,15,3,17] have attempted a formal treatment of information. These ontologies agree in distinguishing between information objects and their physical carriers like paper sheets or pdf files; also, the same information object may be encoded in multiple carriers while retaining its

---

<sup>12</sup>We use the expressions ‘to exist’ and ‘to be present in time’ as synonyms.

identity. For example, John's and Mary's copies of the *Divine Comedy* are two different carriers of the same information object. Information objects and their physical carriers have an important role in business processes. For example, in the scenario of buying a flight ticket, we may consider the flight offer an information object whose physical carrier is not that relevant. Instead, the physical carrier (e.g., a pdf file) is of fundamental importance to exchange the ticket. Again, a question that one may ask is whether this distinction needs to be represented in a process (model) and what are the characteristics of this representation.

Another crucial distinction in BPM is the one between *agentive* and *non-agentive* participants. For the purposes of our work, we consider a process participant as agentive depending on whether it has sensors, actuators and the capability to act on itself or on the environment. For instance, a lathe machine is an agent when, e.g., it has sensors by which it acquires data from the objects to be manufactured and acts upon them by elaborating the data through some software. Additionally, we consider a second notion of agentive participant characterising an agent that can be ascribed with intentions, including beliefs or desires. Non-agentive participants that undergo a change during an action are usually called *patients* of the action. It is easy to see that a process such as the one in Section 2 contains both agentive (e.g., the customer paying for the flight) and non-agentive participants (e.g., the offer whose status changed from created to rejected).

Apart from the classification of participants, we need to spend some words on their roles. From an ontological perspective, the latter are properties that objects only contingently satisfy within certain contexts, e.g., processes (e.g., *to be a resource* during a drilling process) or organisations (*to be professor* at MIT). In this sense, an object can loose or acquire a role while remaining the same entity. We assume that roles can be ascribed to any type of participant, including information objects. The ability to constrain the way an object playing a role participates in a process may be also of interest to the BPM field, not only at the execution but also at the type level. Let us focus, for example, on organisational/business roles, which in BPMN are usually described by means of pools, such as the pools 'customer' and 'travel agent' in Figure 1. While it seems plausible to assume that the customer does not change during the entire duration of the process (otherwise this would be another process instance), the same constraint would not apply to the 'travel agent'. In fact, any employee of the travel agency playing the 'travel agent' role would perfectly fit the specification of this process. Enriching BPM languages with the ability to distinguish between these cases may be useful to constrain and reason on the identity of process instances.

The second column of Table 3 summarises the main characteristics of business processes and of business process constructs described here and in Section 4. In the next section we provide some insights on whether the modelling notations described in Section 2 enable to represent these characteristics, and how. The results are summarised in the right hand side of Table 3.

## 6. Discussion

In this section, we discuss the ontological aspects of business processes presented in Section 5 in the light of the modelling constructs presented in Section 2, with the help of the flight purchase example. The results are summarised in Table 3

	CHARACTERISTIC	BPMN	UML-AD	EPC	CMMN	DECLARE
PROCESS	Set of activities	Yes	Yes	Yes	Yes	Yes
	Clear Input/Output	Yes	Yes	Yes	Somehow	Somehow
	Goal/Value	No	No	Somehow	Somehow	No
	Organizational boundaries	Yes	Yes	Yes	No	No
ACTIV.	Different types of relations between activities	No	No	No	No	Somehow
	State of the word	Somehow	Somehow	Yes	Somehow	No
PART.	Agentive vs non agentive	Somehow	No	Somehow	No	No
	Information vs carrier	No	No	No	No	No
	Object vs role	Somehow	Somehow	Somehow	No	No

**Table 3.** A comparison among modelling languages

By looking at the diagrams, we observe that all the notations enable to represent structured / coordinated sets of activities. This is not surprising: the specification of the control flow is indeed the top priority of a process model. The situation changes as soon as we move to the clear specification of input and output states. Here we observe that all the notations but DECLARE enable / require an explicit initial and final state. Not surprisingly, the imperative modelling languages (BPMN, UML-AD and EPC) strictly require explicit start and end symbols / states. These languages, in fact, model business processes in a prescriptive manner specifying all the allowed flows from a start (the input) to an end (the output) state. Despite its declarative nature, CMMN also facilitates the modellers to specify input and output states by means of input and output sentries and by explicitly asking for exit criteria. Instead, DECLARE is, in our opinion, the weakest language in terms of driving the modeller to explicitly represent input and output states. In fact, it provides a (optional) pattern for the initial activity, and it does not foresee any pattern for the exit / last activity, thus lacking also a way to express - even if implicitly - the goal, or desired state, of a business process. Moving to the specification of the business goal or added value that the business process realises, none of the modelling languages force the modeller to make them explicit. The only language that explicitly contains a ‘goal’ construct is (one of the variants of) EPC. Thus, we can say that most BPM languages leave implicit in the modeller’s (and the reader’s) mind the goal the activities contribute to realise. Finally, organization boundaries can be easily described in notations such as BPMN, UML-AD and EPCs, using notions such as pools/lanes, activity partitions, and organization/activity owner, respectively, while they are absent in CMMN and DECLARE.

Moving to the relation between activities, we can easily notice that almost all the languages only enable a connection between activities without specifying the nature of such connection. A notable exception is DECLARE, whose main focus is indeed the representation of (temporal) relations between activities. While it would be incorrect to say that DECLARE patterns have the aim of specifying the kind of relation existing between different activities, it is also true that some (temporal) patterns may be better suited to model causal vs. temporal vs. dependent relations. As an example, let us consider the ‘response’ and ‘precedence’ patterns in Table 1. While the precedence pattern may be suited to express that an activity happens before another, and therefore can be considered as a pure temporal constraint, the ‘response’ pattern conveys the meaning that *B* is a consequence of *A* being true (happening). Thus we may say that, from an ontological perspective, DECLARE somehow guides the modellers to think about the type of relation existing between activities, besides the simple sequencing.

A key difference among the modelling notations we took into account concerns the representation of the (state of the) world in response to a process execution. Figure 3 emphasises this as one of the focuses of EPCs. UML-AD and CMMN lie in the middle by exploiting data objects and sentries for describing how the world is changed because of the process execution. BPMN, instead, only provides (optional) constructs for representing the state of data objects. Finally, DECLARE does not offer any construct at all for representing the status of the world. From an ontological perspective, we would say that, differently from DECLARE and BPMN, EPC drives the modeller to explicitly represent the world's states affected by the designed process, while UML-AD and CMMN guide the modeller to implicitly represent the world's states through data objects and sentries.

The participants relevant in the flight purchase example are the customer, the travel agency and various information objects. The process thus includes different types of participants, material and immaterial ones. A first observation we have to make is that DECLARE does not offer any support to the modelling of entities that participate to the activities. It is therefore ignored in the remaining of the discussion. By looking at the diagrams, we observe that no explicit distinction is made between agentive and non-agentive participants. Nonetheless, the specification of activity owners in EPCs seems to suggest that they have the ability to act. Also, pools in BPMN are understood as participants in collaboration, and therefore exhibiting the capability to collaborate. In case of UML-AD, the activity partitions may be used for grouping activities with different purposes, i.e. not only according to the activity performer; however, when used for this purpose, also the UML-AD notation tends to suggest the ability of the performer to act. CMMN, instead, does not seem to distinguish between these types of participants.

In Figures 1–3, *Check travel agency website* results in the *Flight request* which is sent to the agency. On the basis of our analysis, one has to distinguish between the request-information-object and the request-support(s); to some extent, the former is more relevant than the latter, since it represents the customer's information to book the flight. However, one cannot avoid referencing the support. Hence, what the customer sends to the agency is a (copy of a) physical object displaying an information object. The distinction between information object and support is not addressed by the languages we considered; rather, it is blurred in the notion of data object.

No actors appear in CMMN and neither BPMN nor UML-AD specify whether 'customer' explicitly refers to a single individual (e.g., John) or to an organisation. In both cases, it reasonably stands for the *role* of a participant, who desires to book a flight ticket. This consideration reveals, besides the lack of declarative language graphical constructs for specifying actors, the underspecification of both BPMN and UML-AD with respect to our analysis, since pools and activity partitions can be used to refer to different types of participants, but also to their roles.<sup>13</sup> Differently, the distinction between single actors and organisations can be explicitly conveyed in EPC, although the difference between participants and their roles is blurred.

To conclude, the analysis of process entities needs to be extended to identify the different modelling approaches in the languages at hand. Once we recognise the ontological assumptions underlying the different languages, including the lack of characterisation of several properties, we can better understand how to correctly use the language to convey

---

<sup>13</sup>Although some support is given in the meta-models of the different languages, what we aim at emphasising here is the lack of support provided by the graphical notations of the different languages.

a well characterised meaning. This latter topic however deserves more attention and is left for future work.

## 7. Related and Future Work

Focusing on *ontology-based* BPM, which is the context of our paper, disparate ontologies have been proposed to semantically enrich process models. Among these, some ontologies axiomatise the properties that graphical constructs satisfy according to modelling notations (see, e.g., [19]). In a more general setting, an upper-level ontology for business processes is proposed in [7]. In these works, however, the authors do not attempt an ontological clarification of the modelling notations at stake. Some initial works towards the analysis of BPMN based on foundational ontologies are presented in [10,20]. These however focus only on constructs like activities and events, while leaving aside the analysis of participants, which is the focus of the presented work. Related work focused on the provision of semantic foundations for role-related concepts in the context of enterprise modelling is presented in [1].

In the future we plan to extend our preliminary analysis in order to deepen the investigation of the ontological commitments of modelling notations by further inspecting the different perspectives that they implicitly take on business processes and their participants, as well as by providing modellers with guidelines to make an appropriate choice when selecting among different notations. We also aim at extending our analysis including further languages that encompass the B2C view, such as ArchiMate, Petri Nets or the Integrated DEFINITION Methods (IDEF) language.

## References

- [1] J. P. A. Almeida, G. Guizzardi, and P. S. Santos, Jr. Applying and extending a semantic foundation for role-related concepts in enterprise modelling. *Enterp. Inf. Syst.*, 3(3):253–277, Aug. 2009.
- [2] A. Artale, M. Montali, S. Tritini, and W. van der Aalst. Object-centric behavioral constraints: Integrating data and declarative process modelling. In *Proceedings of the 30th International Workshop on Description Logics, Montpellier, France, July 18-21, 2017*, CEUR Workshop Proceedings, 2017.
- [3] C. Bekiari, M. Doerr, P. Le Bœuf, and P. Riva. FRBR object-oriented definition and mapping from FRBRER, FRAD and FR SAD (version 2.4). *International Working Group on FRBR and CIDOC CRM Harmonisation*, 2015.
- [4] S. Borgo and C. Masolo. Foundational choices in DOLCE. In S. Staab and R. Studer, editors, *Handbook on Ontologies*. Springer Science & Business Media, 2013.
- [5] T. Davenport. *Process Innovation: Reengineering work through information technology*. Harvard Business School Press, Boston, 1993.
- [6] R. De Masellis, C. D. Francescomarino, C. Ghidini, and F. M. Maggi. Declarative process models: Different ways to be hierarchical. In *Proc. of the 14th Int. Conf. on Service-Oriented Computing (IC-SOC2016)*, volume 9936 of LNCS, pages 104–119. Springer, 2016.
- [7] A. De Nicola, M. Lezoche, and M. Missikoff. An ontological approach to business process modeling. In *3th Indian International Conference on Artificial Intelligence 2007*, pages ISBN–978, 2007.
- [8] M. B. Dwyer, G. S. Avrunin, and J. C. Corbett. Patterns in property specifications for finite-state verification. In *Proc. of the 1999 International Conf. on Software Engineering (ICSE)*. ACM Press, 1999.
- [9] M. Grüninger. Using the PSL ontology. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, pages 423–443. Springer-Verlag Berlin Heidelberg, 2009.
- [10] G. Guizzardi and G. Wagner. Can BPMN be used for making simulation models? In *Workshop on Enterprise and Organizational Modeling and Simulation*, pages 100–115. Springer, 2011.



- [11] M. Hammer and J. Champy. *Reengineering the Corporation: A Manifesto for Business Revolution*. Harper Business, 1993.
- [12] F. Heidari, P. Loucopoulos, F. M. T. Brazier, and J. Barjis. A meta-meta-model for seven business process modeling languages. In *IEEE 15th Conference on Business Informatics, CBI 2013*, pages 216–221. IEEE Computer Society, 2013.
- [13] H. J. Johansson, P. McHugh, A. J. Pendlebury, and W. A. Wheeler. *Business Process Reengineering: Breakpoint Strategies for Market Dominance*. John Wiley & Sons, 1993.
- [14] B. List and B. Korherr. An evaluation of conceptual business process modelling languages. In *Proc. of the 2006 ACM Symposium on Applied Computing, SAC '06*, pages 1532–1539. ACM, 2006.
- [15] C. Masolo, L. Vieu, E. Bottazzi, C. Catenacci, R. Ferrario, A. Gangemi, and N. Guarino. Social roles and their descriptions. In *Principles of Knowledge Representation and Reasoning*, pages 267–277. AAAI Press, 2004.
- [16] H. Mili, G. Tremblay, G. B. Jaoude, E. Lefebvre, L. Elabed, and G. E. Boussaidi. Business process modeling languages: Sorting through the alphabet soup. *ACM Comput. Surv.*, 43(1):4:1–4:56, 2010.
- [17] R. Mizoguchi. Yamato: yet another more advanced top-level ontology. In *Proceedings of the Sixth Australasian Ontology Workshop*, pages 1–16, 2010.
- [18] M. Pestic, H. Schonenberg, and W. van der Aalst. DECLARE: Full Support for Loosely-Structured Processes. In *EDOC*, pages 287–300, 2007.
- [19] M. Rospoche, C. Ghidini, and L. Serafini. An ontology for the business process modelling notation. In *FOIS*, pages 133–146, 2014.
- [20] E. M. Sanfilippo, S. Borgo, and C. Masolo. Events and activities: Is there an ontology behind bpmn? In *FOIS*, pages 147–156, 2014.
- [21] P. S. Santos Jr., J. P. Almeida, and G. Guizzardi. An ontology-based semantic foundation for aris eps. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 124–130. ACM, 2010.
- [22] A. Scheer. *ARIS - vom Geschäftsprozess zum Anwendungssystem*. Springer, Berlin [u.a.], 4., durchges. Aufl. edition, 2002.
- [23] A.-W. Scheer, O. Thomas, and O. Adam. *Process-Aware Information Systems: Bridging People and Software Through Process Technology*, chapter Process Modeling Using Event-Driven Process Chains, pages 119–146. Wiley, October 2005.
- [24] B. Smith and W. Ceusters. Aboutness: Towards foundations for the information artifact ontology. In *Proceedings of the International Conference on Biomedical Ontology (ICBO) 2015*, 2015.
- [25] E. Söderström, B. Andersson, P. Johannesson, E. Perjons, and B. Wangler. *Towards a Framework for Comparing Process Modelling Languages*, pages 600–611. Springer Berlin, 2002.
- [26] M. Weske. *Business Process Management. Concepts, Languages, Architectures*. Springer, 2012.

# Encountering the Physical World

Bahar AAMERI<sup>a</sup>, Michael GRÜNINGER<sup>a</sup>

<sup>a</sup>*Department of Mechanical and Industrial Engineering, University of Toronto, Ontario,  
Canada M5S 3G8*

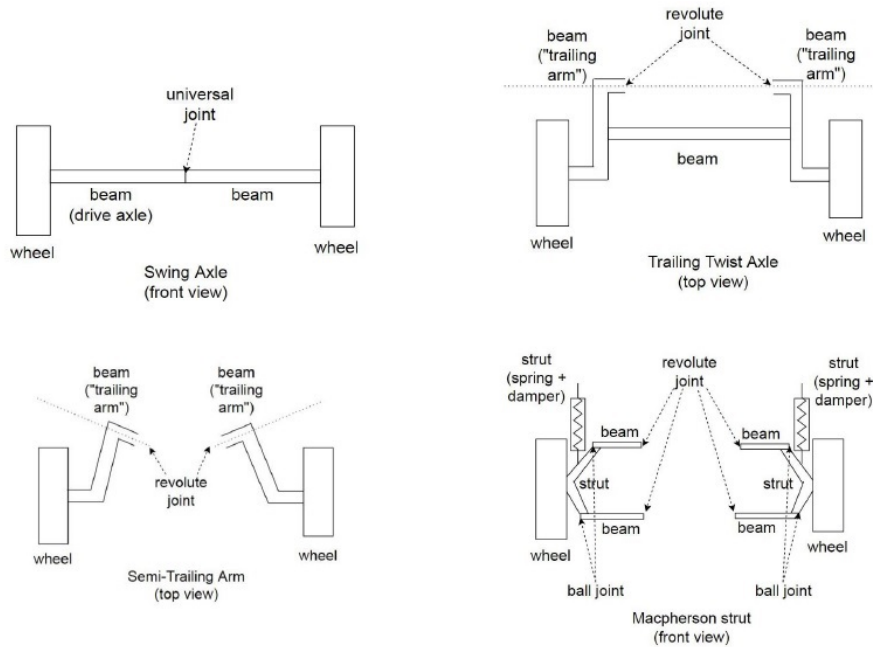
**Abstract.** Manufacturing and product design are grounded in the physical world. The entire product lifecycle involves a wide range of integrated tasks that focus on the properties of physical objects, beginning with begins with the design of physical objects and the specification of the materials from which the physical objects are made. Thus, reasoning tasks within manufacturing and product design requires an ontology of physical world. In this paper, we present the current status of a project that is developing a suite of ontologies, which are modules of an overarching ontology called the PhysicalWorld Ontology. Each module of the PhysicalWorld Ontology captures a particular class of physical phenomena or property of physical objects, such as shape, location, connectedness, parthood, and kinetic and kinematic behaviour.

**Keywords.** manufacturing ontologies, physical objects, mereotopology, shape

## 1. Introduction

Although it may be obvious, manufacturing and product design are essentially grounded in the physical world. Manufacturing is concerned with processes that involve the creation of physical objects, such as assembly, joining, fastening, fabrication, machining, coating, and more recently in additive processes such as 3D printing. From a wider perspective, manufacturing begins with the design of these physical objects (i.e. products) and the specification of the materials from which the physical objects are made. The entire product lifecycle (conceptual design, detailed design, manufacture, maintenance, disposal) involves a wide range of integrated tasks that focus on the properties of physical objects. The supply chain of the manufacturing enterprises spans the sourcing of raw materials and the delivery of products in logistics. All of this is supported by a vast ecosystem of product data management software as well as international standards. It is instructive to consider the scope of ISO 10303, also known as STEP (Standard for the Exchange of Product Data): standard data definitions for geometry (wire frame, surfaces and solid models), product identification, product structure, configuration and change management, materials, finite element analysis data, drafting, visual presentation, tolerances, kinematics, electrical properties, and process plans [2].

Even with this cursory inspection, we can see that a rich set of ontologies about the physical world is needed for manufacturing. Within this paper, we present the current status of a project that is developing such a suite of ontologies, which are modules of an overarching ontology that we refer to as the PhysicalWorld Ontology. Based on the idea that solid physical objects are self-connected objects that are made of matter, have



**Figure 1.** Schematics of swing axle, trailing swing axle, semi-trailing arm, and Macpherson strut suspension systems [1]

a shape with boundaries, and are located in space, the ontologies will support reasoning about physical objects, their behaviors and interaction. Each module of the PhysicalWorld Ontology captures a particular class of physical phenomena or property of physical objects.

We begin in Section 2 by identifying semantic requirements for the representation of physical objects and their behaviour. We then explore the primary modules of the PhysicalWorld Ontology in Section 3 – Shape, Multidimensional Mereotopology, and Location. We finish with a look forward to the remaining ontology modules that will focus on physics and axiomatize the fundamental concepts required for representing kinetic and kinematic behaviour of physical systems.

## 2. Extracting Requirements for PhysicalWorld Ontology

We will begin by delineating the requirements for representing properties and behavior of physical objects. Throughout this section, we consider suspension design systems (see Figure 1) as our main use cases for extracting requirements. A suspension system consists of wheels, beams, struts, springs and dampers, related to each other by different types of joints.

## 2.1. Shape

Perhaps shape is the first feature that comes to mind when thinking about a physical objects; it is also a key concept in representing physical domains. As an example, consider the different suspension systems shown in Figure 1. One of the features that distinguishes these systems from each other is the shape of the beams between the two wheels. Later in Section 3.4.1, we will raise the problem of distinguishing between different rivet fastening methods, and we will show that a description of the shape of rivets is required in order to make this distinction.

The focus of the majority of the existing shape formalisms is on representing convexity and curvature (see [7,13]). In cases where information about convexity and curvature are not required, a shape can be described based on the adjacency and order of its points, edges, and surfaces. This approach has been taken in [10] for developing first-order ontologies for two-dimensional and three-dimensional shapes. We will discuss this approach further in Section 3.1.

## 2.2. Connection and Parthood between Physical Objects

Formalizing the part-whole relationship between a physical system (e.g., a suspension system) and its parts, as well as the physical relationships among the different parts of the system, requires a mereotopology for physical objects. A mereotopology is a formal theory which combines topology with mereology. The topological subtheory expresses connection relations between a set of individuals, while the mereological subtheory expresses parthood relations.

Mereotopological systems differ in their basic assumptions about supplementation, atomicity, extensibility, and closure under sum and product of spatial entities. Ground Mereotopology (MT) [6] is the weakest theory among the existing mereotopological theories, and does not take any of these assumptions. The signature of the MT theory consists of two primitive binary relations, parthood and connection. The axioms of the theory state that connection is a reflexive and symmetric relation, while parthood is a reflexive, transitive, and anti-symmetric relation. In addition, if one individual is connected to another, then the first one is also connected to any individual which the second is part of.

[9] shows that the MT theory is logically synonymous with a non-conservative extension of the RCC8 theory, called RCC8\*, meaning MT and RCC8\* are semantically equivalent, and only differ in signature (i.e., the non-logical symbols). In other words, MT is the mereotopology that underlies RCC8.<sup>1</sup>

The RCC8 relations have widely been used for describing spatial relationships within physical settings. This means that the underlying mereotopology (i.e., MT) used in such settings does not include any of the basic mereotopological principles (i.e., supplementation, atomicity, extensibility, and closure under sum and product). In the following, we provide examples of physical objects that do not satisfy atomicity, extensibility, and closure under sum. It remains, however, an open question whether physical mereotopologies should be supplemental and/or closed under product.

Many mereotopological theories, such as the Region Connection Calculus (RCC) [14], entail that domain entities are atomless. Within a domain, individuals are atomless

---

<sup>1</sup>RCC8 is a set of eight jointly exhaustive and pairwise disjoint binary relations representing mereotopological relationships between ordered pairs of individuals.

if every element has a proper part. However, physical objects are not necessarily atomless. For assembling a bookshelf, we do not care about proper parts of shelves and divider. In many applications we want to have a finite domain, meaning that the elements of the domain are not atomless. Thus, using atomless mereotopologies for representing physical objects is inappropriate as the additional unnecessary constraints result in the elimination of valid models. On the other hand, there might be domains which require an atomless representation of some classes of physical objects. Thus, a general physical mereotopology should not make any commitment about the atomicity of objects, and the atomicity assumption should be taken with respect to domain-specific requirements.

A relation  $R$  is said to be extensional if it satisfies the following sentence

$$(\forall z) (R(z, x) \equiv R(z, y)) \supset x = y.$$

Mereotopologies like RCC assume that the connection relation is extensional. However, extensionality does not always apply to physical objects. Consider a model with two elements. The two elements are obviously connected to the same set of elements, but they are not identical. In fact, any model with finite number of elements (which is the case in many physical domains) may not be extensional.

The following axiom entails that if two (self-connected) entities are connected, they add up to a self-connected whole ( $C(x, y)$  denotes ‘ $x$  is connected to  $y$ ’ and  $P(x, y)$  denotes ‘ $x$  is part of  $y$ ’):

$$(\forall x, y) C(x, y) \supset (\exists z) P(x, z) \wedge P(y, z).$$

However, there are physical domains that do not satisfy this axiom. For example, within the geospatial applications, two neighboring countries are connected, but their summation is not an entity in the domain. If a glass is placed on a desk, their sum does not make a new entity. With a similar example, we can also argue that physical domains are not necessarily closed under the summation of two underlapping objects.

### 2.3. Location

Mereotopologies alone are not sufficient for describing different configurations of physical objects. In standard mereotopologies, overlap relation between two individuals is defined with respect to their common parts; that is, two individuals  $a, b$  overlap if and only if there exists an individual which is part of both  $a$  and  $b$ . When two physical objects overlap, they do not necessarily have a common part. A book on a shelf, for example, overlaps with the shelf, however they do not have a part in common. In this case, the overlap relationship between physical objects should not be defined based on common parts; it should be defined based on a common abstract region that two physical objects occupy.

There are also mechanical objects with components which coincide but do not share parts. For example, while cartridges inside the cylinder of a revolver coincide with the cylinder, they are not part of the cylinder. Similarly, the ball of a ball joint coincides with the space that the hole of the joint surrounds. Definition of relations such as coincide requires a logical theory that axiomatizes relationships between physical objects and the spatial region they occupy. This is what is called a location ontology.

A location ontology is also required to distinguish between different types of spatial change. A moving object, for example, occupies a region that overlaps with the region the object originally occupies, while a shrinking object occupies a region that is a proper part of its original region. So, for distinguishing between movement and shrinkage of an object we need to know the relationships between the spatial region they occupy at each state.

#### 2.4. Joints and Attachment

With topological connection it is only possible to express contact between objects. However, in many application domains we require to distinguish between being “in contact” and being “attached”. A book on a shelf of a bookshelf, for example, is only in *contact* with the shelf, while the shelf is *attached* to the side panels of the bookshelf. In addition, in some domains, such as manufacturing and assembly, representations for different ways that physical objects can be join together are required.

There has been previous work on the ontologies for attachment. [12] suggests a set of first-order definitions describing mechanical joints. They use Smith’s mereotopology [15] for describing the relationships between mechanical parts. The descriptions they have provided are, however, incomplete in the sense that it does not capture the intended specifications of different types of joints. Consider the joining methods depicted in Figure 2. [12] proposes the following definition for threaded fasteners:

$$J_{jf}(x, y) \equiv (\exists u, v) (X(u, x) \wedge X(u, y)) \wedge (T(v, x) \vee T(v, y)) \wedge (P(u, f_s) \wedge P(v, f_s)) \wedge X(f_s, j) \quad (1)$$

and the following definition for fastening by rivets:

$$J_{rf}(x, y) \equiv (\exists u, v) (X(u, x) \wedge X(u, y)) \wedge T(v, x) \wedge T(v, y) \wedge P(u, f_s) \wedge P(v, f_s) \wedge X(f_s, j) \quad (2)$$

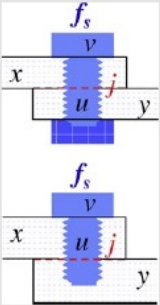
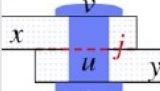
Here,  $P(x, y)$  denotes ‘ $x$  is part of  $y$ ’,  $X(x, y)$  denotes ‘ $x$  crosses  $y$ ’, and ‘ $T(x, y)$  denotes  $x$  tangents  $y$ ’.

The description for rivet fastening relies on the shape of the rivet (denoted by  $v$ ): a part like the one shown in Figure 2 would satisfy both Definition 1 and Definition 2. In fact, to have a sound and complete description for rivets fastening, and so be able to distinguish it from threaded fasteners, we need to be able to describe the shape of the rivet. That is, in addition to mereotopological relations, a complete ontology of mechanical joints requires an ontology for specifying shapes of parts involved in joining methods.

#### 2.5. Boundary

Specifying properties of mechanical joints and physical attachments requires a formal representation of the notion of boundary.

Consider, for example, a ball joint. Each of the ball and the hole of a ball joint have their own boundary surfaces, and one of the boundary surfaces of the ball is connected (in the topological sense) to one of the boundary surfaces of hole. However, if we weld

Joining method	Illustrative example	Description
Mechanical fastening by threaded fasteners		Mechanically fastened joints. The most common method of mechanical fastening is by using bolts, nuts, screws, pins, and a variety of other fasteners $u$ = threaded body $v$ = head of fastener and/or nut
Riveting or mechanical fastening by rivets		Installing a rivet consists of placing the rivet in the hole and deforming the end of its shank by upsetting or heading $u$ = body of rivet $v$ = head and upset tail of rivet

**Figure 2.** Examples of mechanical fastening methods [12].

two three-dimensional objects, the welded surfaces of the objects will be transformed into a single surface, and more importantly, the surface will not be a boundary surface anymore.

[15] defines boundary based on the interior of entities, using the closure operators. In Smith's theory a boundary is a region which has empty interior. That is, unlike other existing approaches, boundaries are not considered as lower-dimensional entities. Moreover, each boundary is a part of the region it bounds, and is a boundary of itself.

The alternative approach, adopted in GFO-Space theory [3] and CODIB [11], is to consider boundary as a lower-dimensional entity which is part of the bounded entity. A model of the GFO-Space theory is partitioned into four categories: *space regions*, *surface regions*, *line regions* and *point regions*, corresponding to three-, two-, one-, and zero-dimensional space entities, respectively. A boundary is a lower-dimensional entity which does not exist independently of the entity it bounds. Moreover, a boundary always bounds an entity with a higher dimension. A boundary does not necessarily fully cover the entity it bounds, and in that case, is part of another boundary which covers more of the entity. Within the GFO-Space theory it is assumed that boundaries are not connected (in topological sense) to other entities (including other boundaries). Rather, two boundaries may be coincide, meaning that they are congruent and there is no distance between them. [11] takes a similar approach, but provides a stronger specification of properties of boundaries, and their relationships to the corresponding bounding entity

Note that the multi-dimensional approach for axiomatizing boundary requires a multidimensional mereotopology. Since physical objects are multidimensional themselves, it seems (even without considering which approach for representing boundary is taken) that using multidimensional mereotopologies is more adequate than equidimensional mereotopologies.

## 2.6. Kinematic and Kinetic Behaviour

In addition to the static properties of a physical system, one might be interested in the kinematic and kinetic behaviour of a system. Consider again a suspension system. Axiomatizing the behaviour of springs and dampers requires an axiomatic representation of force, which in turn requires axiomatic theories of mass, acceleration, velocity, time, and displacement.

## 3. Design of the PhysicalWorld Ontology

The PhysicalWorld Ontology is being developed in an ongoing project that aims to axiomatize concepts and properties required for representation and reasoning about physical domains. The PhysicalWorld Ontology consists of five main modules, namely the Multi-Dimensional Mereotopology, the Occupation Ontology, the Shape Ontology, the Attachment Ontology, and the Physics Ontology. Each of these ontologies have their own modules, and captures one or more of the required concepts described in Section 2. Figure 3 shows the relationship between modules of the PhysicalWorld Ontology.

### 3.1. The Shape Ontology

The Shape Ontology is a qualitative representation of shape of physical objects. The Shape Ontology is an extension of the BoxWorld Ontology presented in [10], which is based on Hilbert's axiomatic theory of geometry. Hilbert's theory consists of three sub-theories: the first subtheory axiomatizes properties of the incidence relation; the second one is a theory of betweenness; and the third one describes congruence relationships. The focus of the Shape Ontology is on the incidence and betweenness relations, and ignores geometrical notions such as length and relative alignment of lines, or curvature and areas of surfaces.

The Shape Ontology consists of three main modules: CardWorld, BoxWorld, PolyWorld. Within the domain of a model of the Shape Ontology there are four disjoint categories of entities – *points*, *edges*, *surfaces*, and *boxes*, where they respectively correspond to zero-, one-, two-, and three-dimensional objects. CardWorld captures the relationship between points, edges, and surfaces. Describing properties of a single box and its parts (i.e., its edges and surfaces) are the focus of the BoxWorld Ontology, whereas the PolyWorld Ontology axiomatizes the relationships between multiple boxes.

The signature of the Shape Ontology includes a binary relation, *part*, that captures the incidence relations between different categories of objects. A lower-dimensional entity cannot exist independently and is always part of a higher dimensional object. For each box, there exists at least one surface which is part of the box. Similarly, for each surface exists at least an edge, and for each edge exists at least one point.

The set of edges in a surface, and the set of surfaces in a box, form cyclic orderings. The edges in a surface are partitioned into disjoint cyclic orderings so that one of these orderings is formed by the outer edges of the surface, and the remaining cycles represent holes within the surface. For each surface, there exists a unique set of outer edges that are all elements of the same cycle.



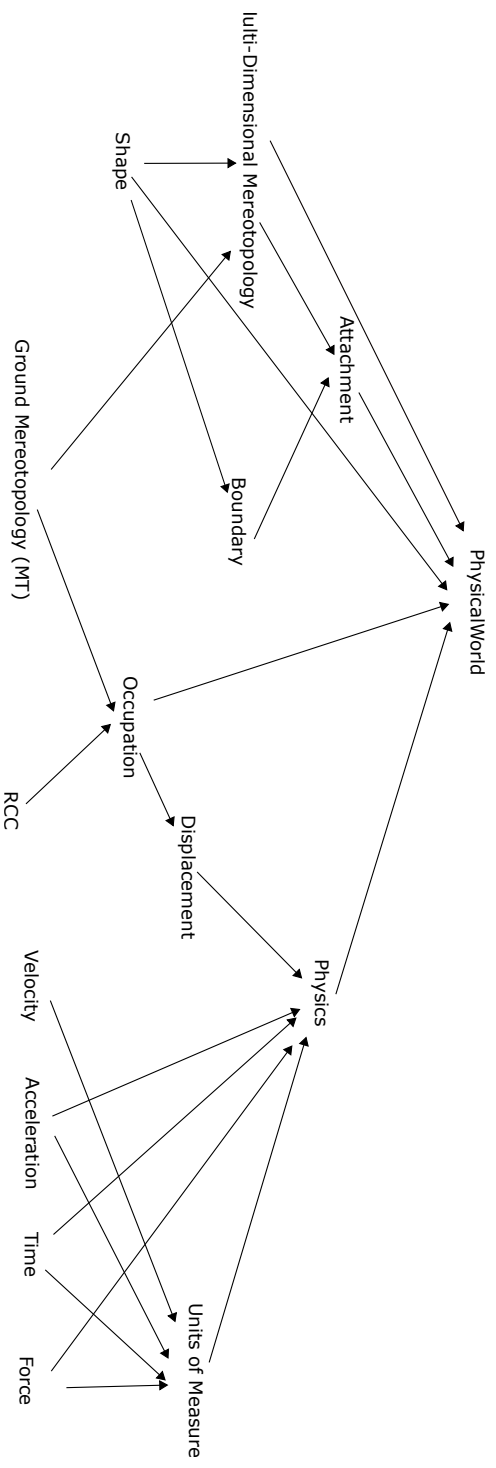


Figure 3.: Modules of the PhysicalWorld Ontology

Round objects (like circles) are the simplest two-dimensional object that can be described by the Shape Ontology. A Round object is a surface which has exactly one edge.

$$(\forall s) \text{round}(s) \equiv \text{surface}(s) \wedge (\exists e) \text{edge}(e) \wedge \text{part}(e, s) \wedge \\ (\forall e_1) \text{edge}(e_1) \wedge \text{part}(e_1, s) \supset (e_1 = e).$$

The Shape ontology, however, cannot represent the difference between a circle and an oval (i.e. curvature is not definable).

A box is a three-dimensional entity that contains at least one surface. For a box with multiple distinct surfaces, each surface will contain edges, called *ridges*, that are part of exactly two surfaces. In a polyhedron, every edge is a ridge. There are also models that are not polyhedra; in such models, there exist edges that are parts of unique surfaces. An edge that is part of a unique surface is a *border*.

Using the Shape Ontology, a sphere can be described as a box which has exactly one surface:

$$(\forall x) \text{sphere}(x) \equiv \text{box}(x) \wedge (\exists s) \text{surface}(s) \wedge \text{part}(s, x) \wedge \\ (\forall s_1) \text{surface}(s_1) \wedge \text{part}(s_1, x) \supset (s_1 = s).$$

And a cylinder can be described as a box that has three surfaces such that two of these surfaces are round objects, and the round objects do not have a common edge:

$$(\forall x) \text{cylinder}(x) \equiv \text{box}(x) \wedge (\exists s_1, s_2, s_3, l_1, l_2) \text{surface}(s_1) \wedge \text{round}(s_2) \wedge \\ \text{round}(s_3) \wedge \text{edge}(l_1) \wedge \text{edge}(l_2) \wedge \text{part}(l_1, s_1) \wedge \text{part}(l_1, s_2) \\ \wedge \text{part}(l_2, s_1) \wedge \text{part}(l_2, s_3) \wedge \text{part}(s_1, x) \wedge \text{part}(s_2, x) \wedge \text{part}(s_3, x).$$

### 3.2. Multidimensional Mereotopology

We use MT as the mereotopological theory for expressing connection and parthood between physical objects since MT is the weakest theory among the existing mereotopologies (recall from Section 2.2 that stronger mereotopologies impose constraints that may not be applicable to all classes of physical objects). However, as we explained in Section 3.1, there are four classes of physical entities in the PhysicalWorld Ontology, namely points, lines, surfaces, and boxes. Therefore, a multidimensional mereotopology is required.

Relationships between equidimensional individuals are captured by MT, while each class of object is mereotopologically independent of other classes. Individuals with different dimensions are only related by *part*, which is an incidence relation (see Section 3.1).

### 3.3. The Occupation Ontology

All of the existing axiomatic theories of location (including [6,8,4,5]) use a mereotopology stronger than MT over non-region entities. Thus, as we discussed in Section 2.2, they are not desirable for representing locative properties of some classes of physical objects. Moreover, some of these theories ([6]) allow mereotopological relationship between abstract regions and non-abstract objects, which leads to the existence of models that physically do not make sense. To overcome these shortcomings, we developed a new location theory called the Occupation Ontology.

The Occupation Ontology specifies physical location. Within the Occupation Ontology space is considered as an abstract entity in which other elements are located. For example, Canada is an object which is located on the abstract region between Atlantic Ocean and Pacific Ocean. The following is the list of ontological commitments the Occupation Ontology satisfies:

- Spatial regions and physical objects are distinct entities.
- There is no mereotopological relationship between spatial regions and physical objects. That is, a physical object is not part of (or connected to) an spatial region, or vice versa. Instead, physical objects occupy spatial regions.
- Occupation is a relation between a physical object and an spatial region. In other words, we assume that a spatial region does not occupy itself, or other spatial regions.
- There is no mereotopological relationships between spatial regions and physical objects; that is, a physical object is neither part of nor connected to a spatial region.
- The mereotopological relations between physical objects must be mirrored in the mereotopological relations between the corresponding spatial regions. That is, if a physical object  $a$  is part of (connected to) another physical object  $b$ , then the region occupied by  $a$  is part of (connected to) the region occupied by  $b$ .

Considering these commitments, the Occupation Ontology consists of three modules: a mereotopology over abstract regions, namely the Region Connection Calculus (RCC) [14], the MT theory relativised to physical objects, and the following axioms that specify the occupation relationship between abstract regions and physical objects:

$$obj(x) \supset \neg region(x).$$

$$occupy(x,y) \supset obj(x) \wedge region(y).$$

$$occupy(x,y) \wedge occupy(x,z) \supset (y = z).$$

$$obj(x) \supset (\exists y) occupy(x,y).$$

### 3.4. Remaining work

In this section we discuss the design of modules of the PhysicalWorld Ontology that have not been axiomatized yet.

### 3.4.1. The Attachment Ontology

The Attachment Ontology consists of definitions, based on relations specified by the Shape Ontology and Multidimensional Mereotopology, for describing different types of physical attachments and joints.

Currently, two types of attachment are included in the ontology, namely strong attachment and weak attachment. We define two boxes to be strongly attached if they are connected and have a common surface (i.e., there exists an surface which is incident with the two boxes). Two boxes are weakly attached if they are connected but do not have a common surface.

$$(\forall x,y)strong\_attach(x,y) \equiv C(x,y) \wedge (\exists z) surface(z) \wedge part(z,x) \wedge part(z,y).$$

$$(\forall x,y)weak\_attach(x,y) \equiv C(x,y) \wedge \neg strong\_attach(x,y).$$

In order to specify properties of other types of joints, we require to incorporate the notion of boundary into the PhysicalWorld Ontology. The existing theories of boundary, discussed in Section 2.5, only consider boundaries in abstract regions, and cannot be applied for representing boundaries of physical objects. It is part of the remaining work to apply ideas from these theories, and develop an ontology of physical boundaries. In particular, we need to identify an axiomatic specification for boundaries of three-dimensional objects (i.e., boxes).

### 3.4.2. Physics Ontology

The Physics Ontology axiomatizes fundamental concepts required for representing kinetic and kinematic behaviour of physical systems. These concepts include time, displacement, velocity, acceleration, mass, and force. The Physics Ontology includes a module for each of these fundamental concepts. Considering the quantitative formulation of these concepts, the Force ontology depends on the Mass and Acceleration Ontologies, and the Acceleration Ontology is axiomatized using the Time and Velocity Ontologies. The Velocity Ontology itself is specified with respect to the Time and the Displacement Ontologies. Note also that for representing displacement we require a representation for physical location, that is, the Displacement Ontology depends on the occupation relation specified by the Occupation Ontology.

In addition to axiomatizing fundamental concepts, the Physics Ontology includes a module, called Units of Measure, that specifies how units of measure corresponding to each concept is manipulated. More specifically, the ontologies explicitly axiomatize how units can be added, subtracted, and multiplied. Moreover, the Units of Measure Ontology utilizes existing ontologies for time, mereotopology, location, and constitution to axiomatize the relationship between units of measure and the concept being measured.

## 4. Summary

Any ontology that supports reasoning about the design and manufacturing of products must be rooted in a set of more foundational ontologies that represent the commonsense intuitions about the physical world. Starting with the idea that solid physical objects

are self-connected objects that are made of matter, have a shape with boundaries, and are located in space, we have designed a suite of ontologies which are modules of an overarching ontology that we refer to as the PhysicalWorld Ontology. The current status of the development of the PhysicalWorld Ontology and its modules is summarized in Table 1.

Concept	Ontology	Development Phase
Connection and Parthood	Multidimensional Mereotopology	Axiomatized
Location	Occupation Ontology	Verified
Qualitative Shape	Shape Ontology	Axiomatized
Joints and Attachment	Attachment Ontology	Under development
Kinematic and Kinetic Behaviour	Physics Ontology	Under development

**Table 1.** Current status of the development of modules of the PhysicalWorld Ontology.

In addition to supporting automated reasoning about manufacturing and product design, the PhysicalWorld Ontology also provides a possible foundation for the ontological analysis of relevant existing standards and to integrate the ontologies within those standards, in particular ISO 18629 (PSL), OWL-Time, ISO 10303 (STEP), and ISO 15531 (MANDATE).

## References

- [1] Autodesk. Use cases and model examples from autodesk. 2016.
- [2] Raphael Barbau, Sylvere Krifa, Sudarsan Rachuri, Anantha Narayanan, Xenia Fiorentini, Sebti Fofou, and Ram D. Sriram. Ontostep: Enriching product model data using ontologies. *Computer-Aided Design*, 44(6):575 – 590, 2012.
- [3] R. Baumann, F. Loebe, and H. Herre. Towards an ontology of space for gfo. In *FOIS*, pages 53–66, 2016.
- [4] T. Bittner. Logical properties of foundational mereogeometrical relations in bio-ontologies. *Applied Ontology*, 4(2):109–138, 2009.
- [5] S. Borgo, N. Guarino, and C. Masolo. An ontological theory of physical objects. In *Proceedings of Qualitative Reasoning 11th International Workshop*, pages 223–231, 1997.
- [6] R. Casati and A.C. Varzi. *Parts and places: The structures of spatial representation*. MIT Press, 1999.
- [7] A. Cohn. A hierarchical representation of qualitative shape based on connection and convexity. In *Proceeding of COSIT95, LNCS 988*, pages 311–326. Springer Verlag, 1995.
- [8] M. Donnelly, T. Bittner, and C. Rosse. A formal theory for spatial representation and reasoning in biomedical ontologies. *Artificial Intelligence in Medicine*, 36(1):1–27, 2006.
- [9] M. Gruninger and B. Aameri. A new perspective on the mereotopology of rcc8. In *COSIT*, 2017.
- [10] M. Gruninger and S. Bouafoud. Thinking outside (and inside) the box. In *Proceedings of SHAPES 1.0: The Shape of Things. Workshop at CONTEXT-11*, volume 812. CEUR-WS, 2011.
- [11] T. Hahmann. *A reconciliation of logical representations of space: from multidimensional mereotopology to geometry*. PhD thesis, PhD thesis, Univ. of Toronto, Dept. of Comp. Science, 2013.
- [12] K. Kim, H. Yang, and D. Kim. Mereotopological assembly joint information representation for collaborative product design. *Robotics and Computer-Integrated Manufacturing*, 24(6):744–754, 2008.
- [13] R. Meathrel and A. Galton. A hierarchy of boundary-based shape descriptors. In *Proceedings of IJ-CAI’01 - Volume 2*, pages 1359–1364. Morgan Kaufmann Publishers Inc., 2001.
- [14] D. A. Randell, Z. Cui, and A. Cohn. A spatial logic based on regions and connection. In *Proceedings of the KR92*, pages 165–176. Morgan Kaufmann, 1992.
- [15] B. Smith. Mereotopology: a theory of parts and boundaries. *Data & Knowledge Engineering*, 20(3):287–303, 1996.

# Where Enterprise Architecture and Early Ontology Engineering Meet: A Case Study in the Public Security Domain

Archimedes A. DETONI<sup>a,b1</sup>, Gabriel M. MIRANDA<sup>a</sup>, Laylla D. C. RENAULT<sup>a</sup>,  
João Paulo A. ALMEIDA<sup>a</sup>, Ricardo A. FALBO<sup>a</sup>,  
Giancarlo GUIZZARDI<sup>a,c</sup>, Fernanda A. BAIÃO<sup>d</sup> and Renata GUIZZARDI<sup>a</sup>

<sup>a</sup>*Ontology and Conceptual Modeling Research Group (NEMO)  
Federal University of Espírito Santo (UFES), Vitória, Brazil*

<sup>b</sup>*Federal Institute of Espírito Santo (IFES), Santa Teresa, Brazil*

<sup>c</sup>*Free University of Bozen-Bolzano, Italy*

<sup>d</sup>*Federal University of the State of Rio de Janeiro (UNIRIO), Rio de Janeiro, Brazil*

**Abstract.** This paper explores the use of ‘process-related models’ – such as Enterprise Architecture (EA) models – as non-ontological resources (NORs) in the Ontology Engineering (OE) trajectory. These models are commonly available in enterprise repositories in process-rich social domains (e.g., e-Government, finance, software engineering, manufacturing), and serve as valuable sources of consolidated knowledge. We focus on the role of EA models in supporting what we are naming here *Early Ontology Engineering*, comprising the phases of purpose and scope identification as well as the identification of functional requirements for creating domain ontologies. This is because these models characterize, among other aspects, the organizational context and the business motivations/goals. Therefore, they may facilitate the identification of intended uses/purpose of an ontology to be integrated to the EA, as a means to address goals of the organization stakeholders. We show how this approach is being applied in a real-world e-Government project in the Public Security Domain.

**Keywords.** Ontology engineering, requirements elicitation, purpose identification, competency questions, enterprise architecture, knowledge acquisition, non-ontological resource, e-government, public security.

## 1. Introduction

When an ontology is being developed, it is essential to firstly identify the domain categories and properties that it should represent [1]. In most OE methods, this is achieved by means of a set of initial tasks usually concerning: (i) purpose and scope identification, which settles why the ontology will be developed and determines its domain of enquiry; and (ii) requirements elicitation, which identifies (functional and non-functional) requirements that the ontology and its implementations should satisfy. Frequently, these tasks determine, among other things, what is relevant to the ontology

---

<sup>1</sup> Corresponding Author, Archimedes A. Detoni, NEMO – Informatics Department, Federal University of Espírito Santo, Av. Fernando Ferrari, 514, Vitória - ES, Brazil; E-mail: archimedes@ifes.edu.br.

in the form of the so-called Competency Questions (CQs), as occurs in methods such as NeOn [1], Methontology [2], UPON [3], and SABiO [4].

Given their level of generality, many OE methods often focus on defining what-to-do, but lack more prescriptive guidelines for specific tasks (how-to-do). This is particularly true in the case of the aforementioned tasks, which are challenging and complex in at least two aspects [1][4]: (i) they depend on intense collaboration between ontology engineers, domain experts and potential ontology users, wherein the specialized terminology makes communication difficult; (ii) they encompass selection and reuse of the most suitable non-ontological resources (NORs), from an amount of available knowledge resources used by a particular community, and that must have already achieved some consensus level between experts. The reuse of such NORs may lead to a complex re-engineering process, which is called *ontologization* in [1].

As observed in [1][2][4], NORs used for Knowledge Acquisition (KA) during the OE initial phase include classical books, standards, glossaries, lexicons and thesauri, as far as they are available in the context. In addition, many OE methods deal with information or data models/schemas as NORs to be reused.

We have noticed, however, that none of the investigated approaches have explicitly explored the use of ‘process-related models’ such as Business Process Management (BPM) and EA models as NORs. These models are commonly available in process-rich social domains (e.g., e-Government, finance, software engineering, manufacturing), and may be employed as valuable sources of consolidated knowledge about the domain(s) in which an enterprise operates. They are often kept in enterprise repositories and reflect the result of significant modeling and validation efforts.

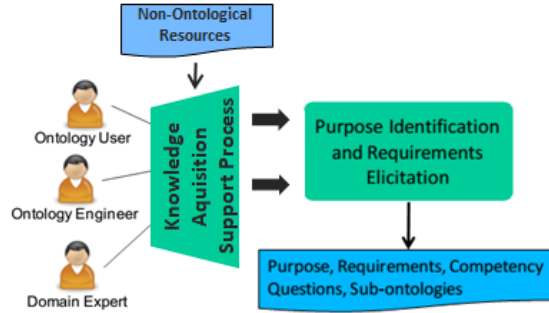
In this paper, we present an approach that employs such type of models as NORs in the OE trajectory. While these models may play a role in KA in general, we focus on their role in supporting the definition of the purpose, scope and functional requirements for an ontology. This is because these models characterize, among other aspects, the organizational context and the business motivations/goals [5]. Therefore, they facilitate the identification of intended uses/purpose of an ontology to be integrated to the EA, as a means to address goals of the organization stakeholders. Furthermore, such models are typically presented in a graphical notation (e.g., BPMN for business process models, and ArchiMate for EA models), favoring communication between domain experts and ontology engineers about the subject domain to be modeled. We show how this approach is being applied in a real-world e-Government project, where we need to develop a network of ontologies to deal with interoperability problems regarding data from public security Information Systems (ISs).

The remainder of this paper is structured as follows: Section 2 introduces the concepts to understand our approach, namely OE initial tasks (purpose and scope identification and requirements elicitation) and EA models and their main elements. Section 3 presents the approach, which is illustrated in Section 4. Section 5 discusses related work, and Section 6 presents some final considerations.

## **2. EA Models as Non-Ontological Resources**

In several OE methods (e.g., SABiO, NeOn and Methontology), most of the KA occurs during the OE initial phase, i.e., in the phases comprising what we term here *Early Ontology Engineering*. As shown in Figure 1 (adapted from SABiO method), in order to perform this activity, ontology engineers need the collaboration of ontology users

and domain experts, and they should also select and reuse suitable knowledge resources, including the so-called non-ontological resources (NORs) [1].







**Figure 1.** Knowledge acquisition support process using NORs in OE initial phase (adapted from [4])

In this paper, we consider EA models developed in the diagrammatic language ArchiMate [6] as NORs. ArchiMate comprises an EA modeling framework and an homonymous EA modeling language [5], whose main objective is to promote the integration of the various viewpoints of the organization, promoting communication between stakeholders and analysis of various aspects of the organization.

The main graphical elements provided by ArchiMate are disposed in three architectural layers: (i) the business layer – which concerns the products and services produced by business processes executed by actors or roles; (ii) the application layer – which concerns the application software that supports the business layer; and (iii) the technology layer – which concerns infrastructural elements. In this paper, we focus on the elements of the business layer.






We employ two viewpoints in this layer: a *motivation viewpoint* and a *business process viewpoint*. The motivation viewpoint explores elements concerning motivational aspects that capture and justify why the business wants to do something, what it aims to achieve, how it plans to get there. The business process viewpoint captures actors and roles and their assignments to tasks/activities within one or across several business processes. The description of the EA modeling elements used in this paper and their notation in ArchiMate are described in Table 1 (the elements of EA motivation models) and Table 2 (the elements of EA business layer). (The descriptions listed in these tables are originated from the ArchiMate specification [6]).

**Table 1.** EA motivation elements description and ArchiMate notation

Description	Notation
<b>Stakeholder</b> - is the role of an individual, team, or organization (or classes thereof) that represents their interests in the outcome of the architecture.	
<b>Driver</b> - represents an external or internal condition that motivates an organization to define its goals and implement the necessary changes to achieve them.	
<b>Assessment</b> - represents the result of an analysis of the state of affairs of the enterprise with respect to some driver.	
<b>Goal</b> - represents a high-level statement of intent, direction, or desired end state for an organization and its stakeholders.	



**Table 2.** EA business elements description and ArchiMate notation

Description	Notation
<b>Business actor</b> - is a business entity that is capable of performing behavior. A business actor may be assigned to one or more business roles. It can then perform the behavior to which these business roles are assigned.	
<b>Business role</b> - is the responsibility for performing specific behavior, to which an actor can be assigned, or the part an actor plays in a particular action or event.	
<b>Business process</b> - represents a sequence of business behaviors that achieves a specific outcome such as a defined set of products or business services.	
<b>Business object</b> - represents a concept used within a particular business domain. Business objects may be accessed (e.g., in the case of information objects, they may be created, read, written) by a business process.	
<b>Representation</b> - represents a perceptible form of the information carried by a business object. A single business object can have a number of different representations. A single representation can realize one or more business objects.	

### 3. Approach using EA models in Early Ontology Engineering

In this section, we discuss how EA models can be used for guiding the ontology engineers in the identification of: (i) domain experts and potential users of the ontology, (ii) knowledge resources, (iii) ontology purpose aligned with organizational goals, (iv) ontology scope and requirements. Therefore, the approach consists of the activities that should be performed by ontology engineers helping the specification of the expected ontology from the analysis of EA models.

Concerning (i), to guide the identification of the domain experts and potential users of the ontology, the ontology engineer should analyze the motivation element *stakeholder*, and the business elements *actor* and *role* in the EA models. This is because the *actors* and *roles* represent the entities responsible for performing organizational processes, and *stakeholders* represent entities interested in the organizational context, both of which can be considered potential domain experts and/or ontology users.

Concerning (ii), the ontology engineer may identify additional knowledge resources from the analysis of the business elements *representation* and *business object*. This is because *representations* (e.g., documents, messages, models, database or spreadsheet tables) capture the perceptible carriers of information related to *business objects*, and may correspond to types of NORs proposed for reuse in the literature.

Once the knowledge resources have been identified, the ontology engineer initiates the identification of the ontology purpose (iii), which consists of exploring the motivation elements *driver*, *assessment* and *goal*. In general, the *stakeholders* are motivated by external or internal conditions, called *drivers*. It is common for organizations to undertake an *assessment* of these *drivers*, which may reveal weaknesses and threats that affect such *drivers*. Thus, the *stakeholders* are often motivated to define organizational *goals* and implement the necessary changes to achieve them. In this step, the ontology to be developed is considered a prospective artifact to take part in the EA, and as such, should have an intended purpose fit to contribute to the achievement of organizational *goals*.

Furthermore, concerning (iv), and following the recommendations of the cited OE methods to elicit the ontology requirements, the ontology engineer should write them in form of CQs. Considering some business elements in the EA models, the ontology engineer can directly identify ‘process-related’ CQs, as explained below:

- Given that *business roles* and *business actors* may represent active entities in charge of executing *business processes*, a competency question should be elaborated for each specific actor executing a task. These CQs follow the structure: “Which individual playing the <<*business role / actor*>> is responsible for the occurrences of a particular <<*business process*>>?”
- Given that *business roles* and *business actors* also represent active entities impacted by *business processes*, the following CQs can be structured: “Which individual playing the <<*business role / actor*>> is impacted by the occurrences of a particular <<*business process*>>?”
- Given that *business objects* represent information assets accessed by *business processes*, the CQs that deal with such relation may follow the structure: “Which <<*business object*>> is accessed by a particular <<*business process*>>?”
- Unlike previous questions, which can be directly inferred from relations among business elements, there may be relevant questions concerning sequences of interrelated *business processes*. These may require a more complex analysis about: (i) the relations between *business roles* and/or *business actors* and different *business processes*; or (ii) the *business objects* accessed by a sequence of interlinked *business processes*. An example of the structure of such kind of CQ is: “Which <<*business role*>> is affected by a particular <<*business process*>> as consequence of a previous <<*business process*>> that triggers the first one?”

Finally, it is important to notice that the ‘process-related’ CQs produced by this approach should be taken as complementary to those suggested by existing OE approaches, which consider other kinds of NORs.

The next section explores each of the steps discussed here in the context of a concrete setting of an e-Government interoperability project.

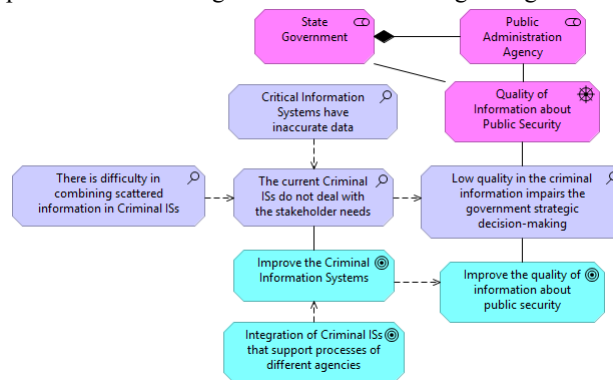
#### 4. A Case Study in the Public Security Domain

According to the Brazilian Health Ministry, between 1996 and 2010 there were almost 1.9 million violent deaths in Brazil, including 710 thousand homicides; and 174 thousand deaths whose basic cause could not be determined by the State. That is, violent death incidents of undetermined cause represent 9.6% of all violent events. In developed countries, this proportion is a residue with less than 1% of all violent cases [8]. As Cerqueira noted [8], the lack of consistent and qualified information on crimes and violent deaths in Brazil is caused in part by deficiencies concerning the sharing and dissemination of information among public administration (PA) agencies. Although these agencies have a large amount of information in their ISs, the various ISs function in isolated silos, failing to support overall decision making.

These are characteristic problems of e-Government interoperability [7], going beyond the security sector to many other areas of the PA. Integration efforts in this setting are challenging mainly because the involved IS are often: (i) commissioned and maintained by different agencies; (ii) designed to address different tasks; and (iii) positioned to support different business processes in isolation.

#### 4.1. Available EA models about the Public Security Domain

Figure 2 presents an EA motivation model that depicts the current scenario of the public security domain addressed in this work. By analyzing this model, one can observe that interoperability is a key element to improve an existing Criminal IS and to allow the cooperation and exchange of information among PA agencies.



**Figure 2.** EA Motivation model about the public security domain.

In order to understand the current process followed by PA agencies to deal with violent crimes, we used the model presented in Figure 3. It represents the current aspects (a so called *as-is* model) of the PA agencies involved in the public security sector, by means of: their roles in the violent crime processes, the subprocesses performed by each PA agency, the IS infrastructure that supports these processes and the information flow, and the information artifacts.

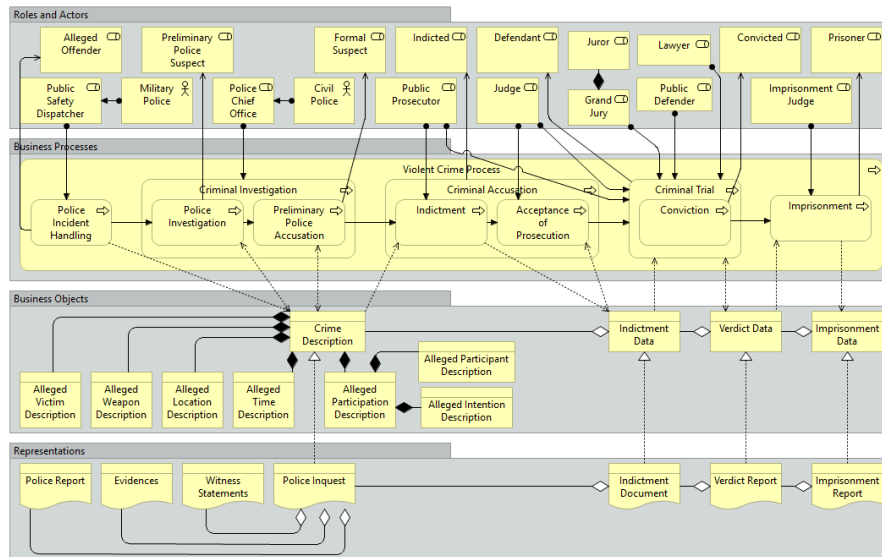
The core of the EA model is the “Violent Crime Process” (VCP). The VCP is a complex business process composed by other business processes. These VCP subprocesses are performed by different public agents of the PA agencies as explained in the sequel. Initially, in the “Police Incident Handling” sub-process, the “Military Police”<sup>2</sup> receives a request and performs the necessary procedures, while the “Public Safety Dispatcher” creates a “Crime Description”, recording information about the police incident (e.g., possible location, time, victim) in the “Police Report”, which is used as part of the “Police Inquest”. Next, in order to determine authorship of the alleged crime, the “Civil Police”<sup>3</sup> establishes a “Criminal Investigation” process, which is composed by: (i) a “Police Investigation” to gather “Evidences” (e.g. crime scene, autopsy and death reports) and “Witness Statements” that are attached to the “Police Inquest”; and (ii) a “Preliminary Police Accusation” based on the “Police Inquest”, if the details in the “Crime Description” suffice to name “Formal Suspects” for the crime.

Then, the “Public Prosecutor” analyzes the “Police Inquest” and defines whether to initiate the “Criminal Accusation” process, which starts with the “Indictment”, a formal complaint that causes the “Formal Suspect” to become “Indicted”. After that, in the “Acceptance of Prosecution” sub-process, the “Judge” analyzes the “Indictment Document” and may accept it, thus starting a “Criminal Trial” in the judicial sphere.

<sup>2</sup> The Military Police is the state police charged with maintaining order. It patrols the streets and imprisons suspects of criminal activity. It is a “militarized” institution (gendarmerie).

<sup>3</sup> The Civil Police is the state police with criminal law enforcement duties. It investigates crimes committed in violation of Brazilian criminal law. It does not patrol the streets.

During the “Criminal Trial”, the “Grand Jury” hears the prosecution (as conducted by the “Public Prosecutor”) and the defense (“Defendant” lawyer), in addition to witness(es) and victim(s), and then announces the verdict.



**Figure 3.** EA Processes Model about violent crime process (VCP)

In case of a guilty verdict, the “Judge” pronounces the “Conviction” imposing a penalty to the “Convicted”. Thereafter, the “Imprisonment” process initiates, the guilty party (i.e. the “Prisoner”), has to comply with the sentence.

The business objects “Crime Description”, “Indictment Data”, “Verdict Data” and “Imprisonment Data” represent the information accessed (and changed) by business processes and stored in the related physical objects (*representations*).

#### 4.2. Applying our approach in a concrete e-Government interoperability project

In this section, we apply the approach proposed in section 3 in the context of a real-world e-Government interoperability project. As this section is based in the models presented in section 4.1, the terms “motivation model” and “business model”, when used, refer to Figure 2 and Figure 3 respectively.

##### 4.2.1. Identifying information artifacts

The first step consists in the identification of the domain experts and the potential users of the ontology. In the public security domain, these entities are the “State Government” and the “Public Administration Agency”, depicted in the motivation model. In the business model, these are the “Public Safety Dispatcher”, “Police Chief Office”, “Public Prosecutor”, “Judge” and “Imprisonment Judge”.

Furthermore, other potential sources of domain knowledge can be inferred from the analysis of the *business objects* and *representations* in the business model. Thus, by analyzing these models, the following such potential resources were identified: “Police

Report”, “Evidences”, “Witness Statements”, “Police Inquest”, “Indictment Document”, “Verdict Report” and “Imprisonment Report”.

The second step consists in the identification of the ontology purpose. As seen in motivation model, the *driver* “Quality of Information about Public Security” motivates the *stakeholders* “State Government” and “Public Administration Agency” to define organizational *goals* and implement the necessary changes to achieve such *goals*. However, there are problems (depicted by *assessments*) that prevent the *stakeholders* from having quality information about public security. For example, the problem “Low quality in the criminal information impairs the government strategic decision-making”, associated with this *driver*, reveals itself as a threat to the public security domain. Here, the ontology, to be introduced as a new artifact in this architecture, has the purpose of providing a semantic basis to allow the integration of different criminal ISs. This clarifies the intended role of the ontology in the architecture, and can also serve to communicate this to the *stakeholders* considering their *goals* and *drivers*.

Further, the approach guides the elicitation of ontology requirements in the form of competency question (CQs). From the analysis of the business model, the following information may be extracted:

- By analyzing the *business roles* and *business actors*, we obtain active entities (e.g., “Public Safety Dispatcher”, “Police Chief Officer”, “Public Prosecutor”) in charge of realizing some *business process* (e.g., “Police Incident Handling”, “Police Investigation”, “Indictment”). These elements allow formulating CQs, such as “Which individual playing the role of ‘Police Chief Officer’ is responsible for conducting a particular ‘Police Investigation’?” or, also, “Which individual playing the role of ‘Public Prosecutor’ is responsible for conducting a particular ‘Indictment’?”
- By analyzing the *business roles* and *business actors*, we also obtain active entities (e.g., “Formal Suspect”, “Convicted”, “Prisoner”) affected by some *business process* (e.g., “Preliminary Police Accusation”, “Conviction”, “Imprisonment”). These elements allow formulating CQs, such as “Which individual playing the ‘Formal Suspect’ role is impacted by the occurrences of a particular ‘Preliminary Police Accusation’?” or, also, “Which individual playing the role of ‘Convicted’ is impacted by the occurrences of a particular ‘Conviction’?”
- By analyzing the *business objects* (e.g., “Crime Description”, “Alleged Victim Description”, “Imprisonment Data”) that are accessed by *business process* (e.g., “Police Investigation”, “Indictment”, “Imprisonment”) we can formulate CQs, such as “Which ‘Alleged Victim Description’ is accessed by a particular ‘Police Investigation’?” or, also, “Which ‘Imprisonment Data’ are accessed by a particular ‘Imprisonment’?”

In addition, we identify CQs that transcend *business processes* performed by different public agencies. For instance: “Which police investigations conducted by a police chief officer led to the effective conviction of the formal suspects?” In this case, we need to understand which are the *business roles* involved in a police investigation and in a conviction, and how they relate to one another. Beyond extracting the *business roles* involved in *business processes*, we analyze the relation between *business roles* and *business processes*. For example, the information regarding a possible formal suspect is carried by the *business object* “Alleged Participant Description” (part of the “Crime Description” and created in “Police Investigation”), and the “Conviction” process accesses this information through the “Verdict Data” (which aggregates the

“Indictment Data” and consequently the “Crime Description”). Therefore, if the “Conviction” and the “Police Investigation” share the same information about a suspect, this brings about additional evidences that the “Convicted” and the “Formal Suspect” in a specific crime are the same individual, thus providing a more effective semantic support for deciding that this “Police Investigation” led to an effective “Conviction”.

#### 4.2.2. Criminal Investigation Domain Ontology

In order to exemplify how these identified CQs support the development of domain ontologies, we present a fragment of the VCP domain ontology, represented in the OntoUML language [9], concerning “Criminal Investigation” (which encompasses the subprocesses of “Police Investigation” and “Preliminary Police Accusation”). This fragment was built using the following CQs:

**Table 3.** Identified CQs of the Criminal Investigation Domain Ontology

ID	Description
CQ01	Which individual playing the role of Police Chief Officer is responsible for conducting a particular Police Investigation (or Preliminary Police Accusation)?
CQ02	Which individual playing the role of Preliminary Police Suspect is impacted by the occurrences of a particular Police Investigation?
CQ03	Which individual playing the role of Formal Suspect is impacted by the occurrences of a particular Preliminary Police Accusation?
CQ04	Which Criminal Description is accessed by a particular Police Investigation (or Preliminary Police Accusation)?
CQ05	Which Alleged Victim Description is accessed by a particular Police Investigation (or Preliminary Police Accusation)?
CQ06	Which Alleged Weapon Description is accessed by a particular Police Investigation (or Preliminary Police Accusation)?
CQ07	Which Alleged Location Description is accessed by a particular Police Investigation (or Preliminary Police Accusation)?
CQ08	Which Alleged Time Description is accessed by a particular Police Investigation (or Preliminary Police Accusation)?
CQ09	Which Alleged Participant Description is accessed by a particular Police Investigation (or Preliminary Police Accusation)?
CQ10	Which Alleged Intention Description is accessed by a particular Police Investigation (or Preliminary Police Accusation)?

In Figure 4, the “Police Investigation” is a *relator* (roughly, an objectified relational context) connecting the “Investigator” (the *role* played by a “Police Chief Officer” when performing an investigation) to the “Preliminary Police Suspects” (the *role* played by a “Person” being investigated). The “Police Investigation” is characterized by an investigation content, which refers to a “Crime Description”. This description grounds the indication of some participant as “Preliminary Police Suspect”, justifying the relation “refers to” holding between an “Alleged Participant Description” and a “Preliminary Police Suspect”.

A “Preliminary Police Accusation” is based on a “Police Investigation” (hence the *historical dependence* relation [9]). The “Preliminary Police Accusation” mediates the “Accuser” and the “Formal Suspect” (specialization of the “Preliminary Police Suspect”). A “Preliminary Police Accusation”, similarly to a “Police Investigation”, is characterized by some content, which is captured by the notion of “Crime Description”.

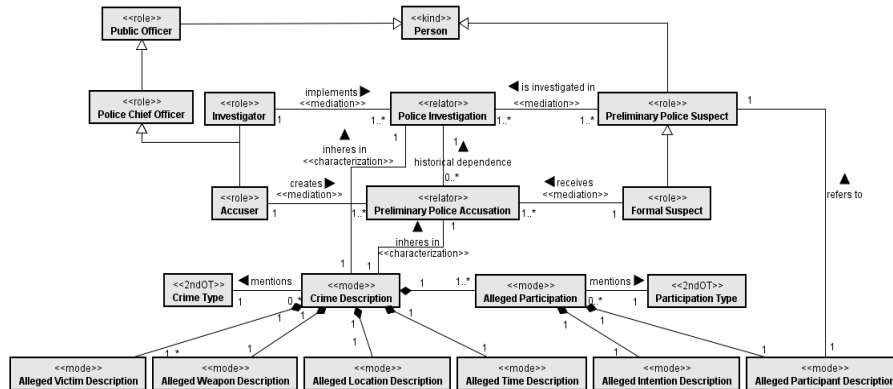


Figure 4. Fragment of the Criminal Investigation Domain Ontology

The two relations, namely the *historical dependence* relation between “Preliminary Police Accusation” and “Police Investigation” and the generalization relation between “Formal Suspect” and “Preliminary Police Suspect”, capture two important notions: the former captures the idea that a “Police Investigation” is required for a “Preliminary Police Accusation” to exist; the latter that the “Formal Suspect” must be a “Preliminary Police Suspect” in the scope of a “Police Investigation”. We have observed that similar patterns are manifested throughout the whole VCP process.

## 5. Related Work

As previously mentioned, many of the OE methods include the use of CQs to identify requirements. A brief description of some methodologies and their support to CQs definitions is presented in [10], pointing out that most of those methods presuppose the existence of this set of questions (not focusing on specifying how to formulate them) and concentrate their efforts in defining the next stages of ontology development. Despite the importance of eliciting ontology requirements within the field of OE, the existing methodologies make little use of elicitation and modeling techniques [10]. In order to fill this gap between the definition of CQs and the ontology modeling per se, we rely here on the use of EA models.

The approach presented in [10] is the closest to the one presented here that we found. The authors in [10] proposed an OE approach with three main objectives: (i) defining the ontology scope; (ii) deciding the ontology’s applicability; and (iii) specifying what questions the ontology must answer. To achieve these objectives they applied Tropos [11] as the goal modeling methodology. The first activity of that approach is to develop early requirements, which permits to understand the organizational setting as it is. The output of this activity is an organizational model, which includes relevant actors, their goals and interdependencies. This model provides a context for the definition of the ontology scope, helping to identify ontology’s applicability. In late requirements activity of Tropos, the system is modeled as an actor that is dependent of other actors in the organization. These dependencies define the functional and nonfunctional requirements of the system. Such requirements detail what kind of support a system (in our case, an ontology-based system) should provide. Similar to our proposal, they concentrated their efforts in defining the first stage of

ontology development. Differently from them, our work is not only based on analysis of the goals depicted in motivation models, but also on the process models. We argue that, using EA process models, it is possible to analyze other organizational elements that impact the ontology specification. We did not find studies that explicitly propose the use of EA models as knowledge resource to support ontology development.

Concerning requirements elicitation, a systematic review reported in [12] has shown that there are many proposals using EA models, but all of these concern requirements for software development. The review focused in the requirements extraction from business process models represented in the BPMN notation. In this context, one study that we consider relevant is proposed in [13]. They propose acquiring requirements for software intensive systems according to the business objectives and base lining business processes. The approach consists of the following activities: (i) concepts exploration and orientation; (ii) analysis and modeling of current business processes; (iii) modeling of target business processes; and (iv) requirements generation for the target system. Although they do not focus on ontology development, there are similarities with our work. First, they use the models to understand the current business processes with their business flows, inputs, outputs, and responsible bodies. They also proposed guidelines to get knowledge about the domain using the BPM model, closer to our proposal. Considering the process of requirements elicitation, we focused on the semantics of the elements of EA models and their relations. In contrast, they explain that functional requirements of the target system were elicited by analyzing the business processes, but they do not show how these requirements can be explicitly identified. An advantage of our proposal is focusing on how to do the requirements identification by detailing guidelines of using EA model elements in this activity.

## 6. Conclusion

This paper presents an approach for Early Ontology Engineering, in particular for KA support process. We do that by (re)using EA models as a suitable process-based NOR in process-rich social domains. These EA models provide not only a mechanism to systematically structure knowledge about the subject domain, but also provide resources to analyze and understand the organizational elements from different viewpoints.

The proposed approach uses the knowledge assets contained in the (pre-existing) EA models (diagrammatic models about institutional processes, structure, information technology architecture and business motivational aspects) as inputs to Early Ontology Engineering activities in way that: (i) helps ontology engineers to comprehend the overall subject domain; (ii) makes their communication with the domain experts easier and clearer; and (iii) provides guidelines to identify the purpose and scope of domain ontologies as well as to elicit their requirements.

This approach proposes to reuse EA models in enterprise contexts where they are readily available. In cases that these models do not yet exist, the development of new EA models for organizations has become a common practice. In such cases, the EA modeling stage can be a beneficial activity for developing structured knowledge assets in (and about) the organization, enhancing its own EA.

In addition to these aspects, we have also perceived other benefits exploring the use of EA models as NORs in other stages of the ontology development process, which are not explained in this paper for the sake of space. For example: (i) by analyzing



application layer elements (e.g., application services, application components and data objects) and technology layer elements (e.g., technology nodes, artifacts, services and softwares), it is possible to point to other useful NORs that can be used to support the identification of concepts and relations for the ontology, and to provide data for the instantiation and evaluation of the ontologies produced; and (ii) by analyzing how the underlying application layer elements support the various *business processes*, ontology engineers can identify IS interoperability gaps, e.g., observing that the relations between business processes do not have correspondent relations between the application layer elements that support such processes. From this finding, the ontology engineers can design an ontology that addresses this issue.

In future work, we intend to investigate: (i) how EA models may support identifying non-functional requirements, e.g. desirable computational properties in order to implement an operational ontology; (ii) the role of reference ontology models to improve EA, i.e., how a developed ontology introduced as an new artifact in the organizational architecture can improve such architecture, providing a precise semantics to domain concepts, thus contributing to achieve the organizational *goals*.

## Acknowledgements

This research is partly funded by FAPES (69382549), CNPq (grants number 311313/2014-0, 461777/2014-2 and 407235/2017-5), CAPES (23038.028816/2016-41), as well as the Free University of Bozen-Bolzano (OCEAN Project).

## References

- [1] M.C. Suárez-Figueroa, A. Gómez-Pérez, E. Motta, A. Gangemi, (Eds.), *Ontology engineering in a networked world*, Springer Science & Business Media, 2012.
- [2] M. Fernández-López, A. Gómez-Pérez, & N. Juristo, Methontology: from ontological art towards ontological engineering, *AAAI Technical Report SS-97-06* (1997), 33-40.
- [3] A. De Nicola, M. Missikoff, & R. Navigli, A proposal for a unified process for ontology building: UPON, *Int. Conf. on Database and Expert Systems Applications*, Springer Berlin Heidelberg (2005), 655-664.
- [4] R.A. Falbo, SABiO: Systematic Approach for Building Ontologies, *ONTO. COM/ODISE@ FOIS*, 2014.
- [5] M. Lankhorst, Enterprise architecture at work: Modelling, communication and analysis, *The Enterprise Engineering Series* (2009), 2nd ed. Springer, Germany, Berlin, 2009.
- [6] The Open Group, ArchiMate 3.0 specification, United Kingdom, 2016.
- [7] H.J. Scholl, & R. Klischewski, E-government integration and interoperability: framing the research agenda, *International Journal of Public Administration*, **30.8-9** (2007), 889-920.
- [8] D. Cerqueira, Map of hidden homicides in Brazil (text translated), *Text for Discussion*, **1848**, IPEA, Rio de Janeiro, Brazil, 2013.
- [9] G. Guizzardi, *Ontological foundations for structural conceptual models*, Fundamental Research Series, Centre for Telematics and Information Technology, Netherlands, 2005.
- [10] P.C.B. Fernandes, R.S.S. Guizzardi, & G. Guizzardi, Using goal modeling to capture competency questions in ontology-based systems, *Journal of Inf. and Data Management* **2.3** (2011), 527-540.
- [11] P. Bresciani, et al. Tropos: An agent-oriented software development methodology. *Autonomous Agents and Multi-Agent Systems* **8.3** (2004), 203-236.
- [12] A.S. Bitencourt, D. Paiva, & M.I. Cagnin, Requirements Elicitation from Business Process Model in BPMN: A Systematic Review. *Proceedings of the XII Brazilian Symposium on Information Systems on Brazilian Symposium on Information Systems: Information Systems in the Cloud Computing Era-Volume I*. Brazilian Computer Society, 2016.
- [13] O. Demirörs, C. Gencel, & A. Tarhan, Utilizing Business Process Models for Requirements Elicitation, *EUROMICRO*, 2003.

# Financial Reporting by a Shared Ledger

Ivars Blums <sup>a,1</sup> and Hans Weigand <sup>b</sup>

<sup>a</sup>*SIA ODO, Riga, Latvia.* <sup>b</sup>*University of Tilburg, The Netherlands.*

**Abstract.** Among models and information about economic phenomena which help to understand how enterprises produce value, Accounting and Financial Reporting still play a leading and regulative role. The regulative role is established by enforceable International Financial Reporting Standards (IFRS). Ontology engineering methods, which have proven to cope with difficult standardization issues, are seldom used in developing the abovementioned standards. Furthermore, the standard setting should more increasingly account for the influence of advanced information technologies for capturing and reporting financial information, such as [blockchain-based] shared ledgers and data analytics. This paper proposes an initial version of the Core Ontology of Financial Reporting Information Systems for a Shared Ledger Environment (COFRIS) grounded on the Unified Foundational Ontology (UFO) network, and a preliminary analysis of the IFRS.

**Keywords.** UFO, REA, IASB, IFRS, COFRIS, Shared Ledger

## 1. Introduction

Ontology engineering methods, which have proven to cope with difficult standardization issues [14], are seldom used in developing standards of international financial reporting (IFRS). Consistency, completeness and clarity of recent editions of Conceptual Framework (CF) for Financial Reporting (FR) [1] and reworked standards [2] by the International Accounting Standards Board (IASB) still need to be improved [6]. We see the following areas of improvement of this framework and standards:

- usage of the ontology engineering tools for standard setting, grounding on upper level ontologies;
- elimination of the repetitions and inconsistencies among IFRS standards;
- elimination of the inconsistencies with other enterprise standards and enterprise ontologies;
- generalization of the conceptualization of economic contracts and their progression events [12];
- accounting of the impact of modern information technologies, such as data analytics and shared ledger [11].

Our research attempts to contribute to these improvements by using and extending the general patterns of upper ontologies and by extracting patterns common to all standards. These patterns facilitate the reuse, understanding and precision of IFRS standards and their accompanying documents, which now comprise thousands of pages.

The main contribution of this paper is an initial version of the Core Ontology of Financial Reporting Information Systems for a Shared Ledger Environment (COFRIS)

---

<sup>1</sup> Corresponding Author, Ivars Blums; E-mail: Ivars.Blums@odo.lv.

grounded on the Unified Foundational Ontology (UFO) [3] network. The proposed ontology is partially extracted and validated analyzing the Financial Reporting Conceptual Framework, Standards and Illustrative examples. Such an analysis should lay the groundwork for a modern Ontology driven IS environment including data analytics, a shared ledger and smart contracts. The results are OntoUML [15] supported financial reporting information systems conceptual model development principles and Conceptual Framework for Financial Reporting improvement suggestions.

This paper is structured as follows: Section 2 reviews UFO foundational and domain related patterns and other previous work. Section 3 introduces the main economic phenomena used in financial reporting, and their presentation in OntoUML [15]. Section 4 outlines a possible information processing in a shared ledger environment for financial reporting. As a partial validation, Section 5 makes a preliminary ontological analysis of Conceptual Framework for Financial Reporting. Conclusions finalize the paper.

## 2. UFO Grounded Ontology Engineering and Other Previous work.

Our approach [12] is grounded on the Unified Foundational Ontology (UFO) network because of its comprehensive coverage of enterprise and FR relevant social concepts and cases that exceeds other foundation ontologies, and availability of the OntoUML ontology engineering tools and methodology.

UFO Foundational Ontology Patterns (FOP) [19] distinguishes types and individuals. *Individuals* are specialized in endurant FOP and event FOP. *Endurants* are entities that, whenever they exist, they are wholly present, i.e., whenever they are present, they are present with all their parts, i.e., they persist in time during their *life*. Endurants are further specialized in objects and modes (moments). *Objects* are *non-agentive objects*, agents and situations. *Modes* are intrinsic and relational (called relators). *Relator* FOP is existentially dependent on two or more endurants. *Agent* FOP represents an object that can bear intentional modes, such as beliefs, desires, and intentions.

In UFO-B [16], *Events* are individuals composed of temporal parts, they happen in time, in the sense that they extend in time and accumulate temporal parts. *Actions* are intentional events with the specific purpose of satisfying a goal. An action achieves a goal if the action brings about a situation in the world which satisfies that goal. *Disposition* FOP represents modes that are only manifested in particular situations, but that can also fail to be manifested. When manifested, they are manifested through an occurrence of events. Dispositions are described by reference to the types of processes which would realize them under certain conditions, e.g., fragility, and *liability* [16].

*Types* in UFO are [sub]kinds, roles[mixin] and phases[mixin], categories, and collectives. *Phase* FOP is the extension of the endurant type due to a change in intrinsic properties. When mediated by a relator, an endurant plays a *Role* FOP in a certain context. *Rolemixin* FOP, is used for roles of different kinds.

In UFO-C [4], *Social relator* FOP is a relator composed of one or more pairs of social commitments and social claims among social agents. UFO *Pattern for Representing Events in Structural Business Models* [16] is used in COFRIS in combination with disposition and social relator, employed for modeling economic phenomena. Social relators are founded by events. As other events, creation events begin and end at certain time points. The creation moment of a relator, and objects in general, is derived from the termination time point of its creation event (e.g. in FR domain: initial recognition of an asset, contract inception, incorporation).

Objects have a causally active phase (e.g., a contract, an acquired property, an accrued liability, an going-concern enterprise) during which the qualities and dispositions of the, e.g., relator are manifested through several *life events* (e.g., a fulfillment of an obligation, a depreciation of a property, a revaluation of a foreign currency liability) that accumulate to constitute, at each point, a different process that represents the *current life of the relator* (e.g., the performance of an enterprise during the reporting period). Relators have a causally inactive phase (e.g., a sold property, a settled liability). In this latter phase, the properties of that relator can no longer be manifested, its qualities are immutable regarding their values and we can refer to the *final life of the relator* as the total accumulation of all *events in the life of the relator*. OntoUML gives us a support for deciding for which types in a model of endurants we should specify a behavioral model of changes [16], depicting its phases (e.g., an *offering* phase, a *fulfilment* phase of a contract) and including temporal OCL constraints prescribing the phase transitions, at the same time having possibility to specify all the structural details.

Within the UFO ontology network, UFO-S is a *core reference ontology on services* [13, 5], which characterizes the service phenomena by considering service commitments and claims established between service provider and customer along the service life-cycle phases: offering, negotiation/agreement and delivery. UFO-S presents general concepts spanning across several applications domains so that its conceptualization can be broadly reused. The service concept in UFO-S is rather broad, allowing the use of concepts and relations of this ontology as a base for a generic economic agent interaction life-cycle [12]. Resource exchanges can be added. However, the concepts defined in UFO-S sometimes have a different meaning as those from the financial reporting domain. For instance, car rental is a *service* as illustrated by UFO-S [13], but it is a *right to use* per [2], a Netflix subscription is a right to use per [5], but is a service per [2].

Legal aspects of service contracts are further elaborated in [18] within the UFO-L Legal ontology, that is based on Hohfeld's theory of fundamental legal concepts. The *legal positions* of UFO-L include not only those corresponding to commitments and claims from UFO-S (i.e., right and duty), but also other elements: no-right, permission, power, subjection, disability and immunity). All these legal relators are from two classes of *entitlement* and *burden/lack*, that we refer further as rights and obligations respectively.

Concerning FR related, but not UFO grounded ontologies, the Resource-Event-Agent ontology (REA) was originally formulated in [8] and announced as an Accounting ontology in [9]. REA is based on the core concepts of [physical] resources and *duality* of their transfer actions, performed by the economical parties, fulfilling commitment *reciprocity* within contracts. REA approach is different from FR primarily because it tries to conceptualize events, while FR accounts for the effects of the events. We support this aspect of REA, but otherwise we find its entities insufficient for conceptualizing FR. Primarily, the resource concept lacks consistent approach of presenting rights and obligations, leading to conceptualizing liabilities and assets as formal relations. Essential concepts of roles, such as of owners, and phases of objects, resource/obligation disposition, timing, uncertainty, impairment and value, as well as contract manipulation and revaluation/reclassification economic events are not elaborated in REA. UFO-S and REA include only "Happy Path"; contract and delivery phases in their models, while "Expired/Violated" phases are relevant for the FR.

Two others, not UFO grounded ontologies are OWL based. The Financial Industry Business Ontology (FIBO) [17] covers concepts and relations in a particular industry, relevant for important part of FR, regulated by IFRS 7 and 9 *Financial instruments* [2]. FIBO is grounded on semantic web principles and, in a sense, its own foundational

ontology. While many FIBO concepts overlap with COFRIS, their alignment is planned at the stage of developing a reference ontology for IFRS standard. The domain ontology of Conceptual Framework for Financial Reporting [6] is a fundamental transliteration of the part of the Conceptual Framework for FR concepts in OWL, using Protégé tools. It differs from our approach that ultimately aims at the whole set of the IFRS standards.

### 3. Economic Phenomena Depicted in COFRIS.

**Financial reporting** (FR) provides information relevant to investors and lenders about the *reporting enterprise's economic situation - economic relationships* with *economic agents* and the *changes* of those relationships. *Reporting period* is used to decompose the changes of the whole as separate one-period flows. FR may be divided into *operating segments* of an enterprise, and comprises of several *financial statements*, such as balance sheet and income statement, in *functional currency* (see Fig. 1).

**Economic agent** is a category of *persons* and *enterprises, contractual groups* of people and enterprises, or the *society at large* (see Fig. 1 in yellow). Economic agents are capable of [self] committing and fulfilling economic actions. According to UFO-C [4] communicative acts, of *incorporation* in our case, are used to distinguish between physical agents (e.g., a person) and social agents (e.g., an organization, a society). At this stage we model society and groups as collectives, but not functional complexes. Society at large forms *economic environment* that is the context for economic agents and is represented in a *shared ledger*. Economic agents in a shared ledger are represented by accounts.

**Reporting enterprise** (subject of FR and an institutional agent per UFO [4]), is an *incorporated* contractual group with some inherent goals. An *enterprise* has *control* to *allowed* and *intended actions* upon *economic resources* to attain its goals and fulfill its obligations for the benefit of its owners and other related agents. The *Enterprise owner* *controls* or *has a non-controlling interest* in an enterprise as per the *articles* contract. FR normally assumes that the reporting enterprise is a going concern. An enterprise itself is a resource/obligation and a prototype of any resource/obligation, in a sense that it comprises mediating agents, an internal or external input, a process and an output, as described in IFRS 3 [2].

**Economic relationship** is a UFO social relator [4] existentially dependent on involved economic agents playing the roles of the *party* (e.g., by the reporting enterprise) and the *counterparty* and having two or more pairs of mutually dependent commitments/claims quantified in monetary terms, regarding some *underlying object*.

**Economic contract** represents intended, suspended or recognized economic relationships and establishes a right and an obligation to exchange economic resources/obligations. In other words, a party has a commitment to transfer some resource/obligation to the counterparty in exchange for a claim to receive another resource/obligation. In the *smart contracts* commitments may be fulfilled automatically [11]. We distinguish *contract phases* progressed by *economic events*. These events and their effects become parts of the contract. Normally a contract in FR has no exchange value at inception. For economic contract conceptualization, we ground on UFO-S and UFO-L. We refine exchange pattern of [12], extracting common concepts of IFRS 9, 15-17 [2].

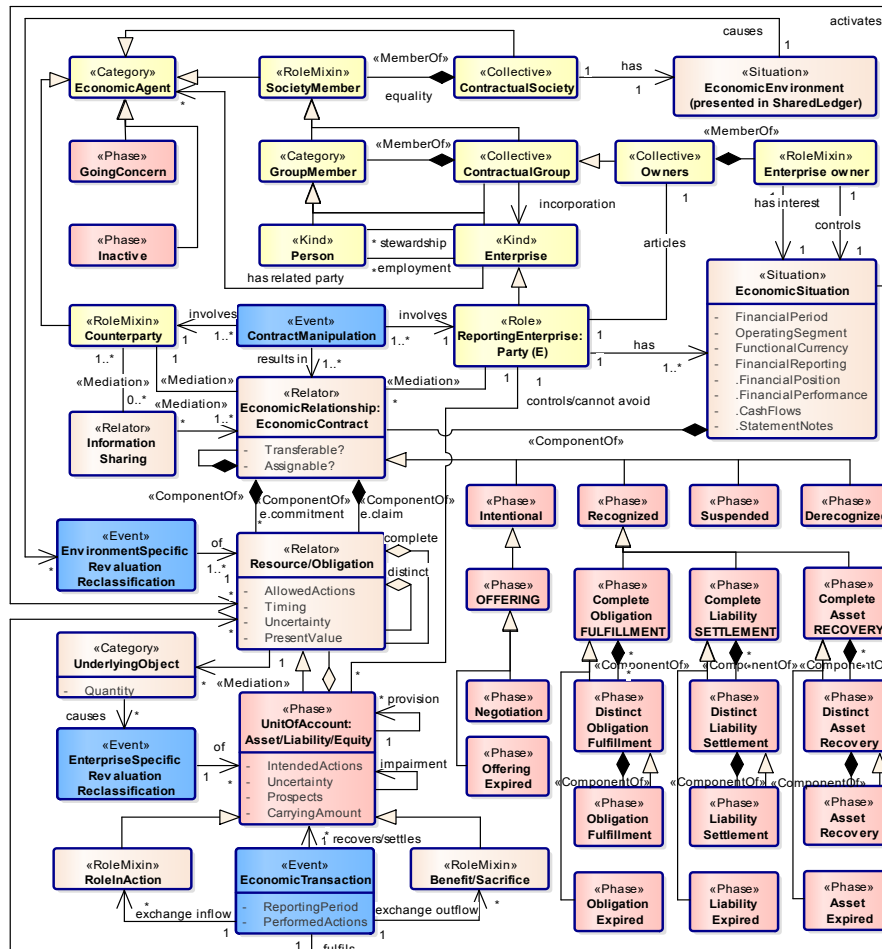


Figure 1. A fragment of the OntoUML diagram of COFRIS.

**Resource** is a right that has the disposition to produce [possibly in conjunction with other resources] *economic benefits*. It is an economic relationship of a party including, at a minimum, enforceable permissions against all economic agents for some underlying object. In [14] UFO regards the rights aspect of resources that is understood “as being available for the organization, e.g., by an employment contract between employers and employees, or by having the right/ownership over a certain object”, i.e., the rights are contractual or are permissions [18]. Our comment is that contractual rights are also permissions, because they may be transferred. The *allowed actions*, by law, contract or nature rights, prescribe permissions of economic agents to use economic resources. The prescription includes: *exchange action type*; *role of the object* in this action; resulting *benefits/sacrifice* objects; *prospects* - potential or contracted counterparties and their roles; *timing*; and *uncertainty* of the actions and *present value* of the resources. Exchanges, which types are not within the allowed actions, should be prohibited in a smart contract system. An example of such violation would be a sale of a leased object by a lessee. *Timing* is a condition indicating when the resources are to be used/obligations

fulfilled, such as: on condition or triggering event; on fixed or determinable dates or on demand; at the end of the process or useful life of a resource; on default (expiration, violation); at will or a changed economic situation (dividends, put, call); at liquidation. Timing also broadly classifies relationships into current and noncurrent; together with economic events, it determines the transition of the phases of the economic relationships; characterizes the priority of the exchange relative to other exchanges. Timing is essential for all IFRS standards. *Present (exchange) value* is a price that exists independent from the enterprise, and is used as a measurement unit for FR.

**Underlying object** is a physical or intellectual object; or amount of matter, including human and natural environment energy; or an obligation/right for another underlying object. The *quantity* of an underlying object or its characteristic is used to depict resources required/available for exchange (e.g. currency units, commodity units, or a share of the net assets of the enterprise). Quantity is absent from the concepts of Conceptual Framework for FR, but is present in most IFRS standards.

**Obligation** is a resource transfer action to which an economic agent is legally or constructively bound. Obligations and resources may be complex objects – bundles combined from several separate parts, for transfer/use we distinguish *complete* and *distinct* obligations and resources. *Distinct obligation* fulfillment creates a usable resource for a counterparty and its distinct liability and allows for the enterprise to recognize revenue. *Complete obligation* fulfillment creates an unconditional right for the enterprise to receive and a complete liability for the counterparty.

**Unit of account:** is a group<sub>s</sub> and a phase - a result of past events, of enforceable/constructive rights/obligations, presently recognized by an enterprise, classified by their intended use and valuation, with assessed uncertainty and impairment. A unit of account represents a stock of resource/obligation objects.

**Asset** is a resource controlled by the enterprise. As defined above an asset is a subtype of a resource. This is in contrast with other sources, e.g., mentioned in [11,14] where the resources could be understood as subtypes of “assets that can be drawn on by” the enterprise. In COFRIS, a resource has the general potential to produce economic benefits, while an asset has the potential to be realized by the enterprise who controls the resource. In other words, a resource is transferable, while an asset is a controlled resource. *Intended actions* refer to the primary actions, and assets/liabilities used in those actions in which an enterprise is engaged and capable, e.g., selling goods/services (via a specified distribution *channel*; for a specified *region*), manufacturing, or administration.

*Carrying amount (use value)* depicts the account value after deducting any accumulated depreciation and impairment losses. *Uncertainty* of receiving economic benefits. Assessed through *provisions* and mitigated by hedging. Valuation and uncertainty also include methods, assumptions and judgements used for their ascription.

*Impairment* is a condition that exists when the carrying amount exceeds the present value. IAS 36 [2].

*Role in an action* refers to the economic characteristics that distinguish assets and liabilities used in actions that do not respond similarly to alike economic events, such as materials, labor. For example, flour, owned by E, is intended as a raw material [role] for baking [action]. In the case of a sale of that flour, the exchange could be classified as *unordinary*. It is compliant with a general concept of resource modeling in UFO: “resource models both the role [mixin] an object plays in a particular context of usage as well as its allowed type” [14]. The context of usage, in UFO is defined “in the scope of a material relation (or in the scope of an event)”, i.e., in economic relationship or exchange respectively. *Benefit/Sacrifice* refers to the outcome of the intended or

performed action, which increases/decreases equity. Additional identification of a portion of a resource may be required by the economic event affecting the resource/obligation; or location of the resource.

**Liability/Equity** is an obligation of the enterprise that cannot be avoided, to transfer a resource. Per UFO-L [18], legal modes are related to each other by a *correlation* association and are essential and inseparable parts of a legal relator. We assume the same association for the economic relator.

**Economic event:** is an UFO-B: Event that affects economic relationship characteristics relevant for FR (see Fig. 1 in blue).

**Contract manipulation** is a communicative act [4] (special kind of action) that can be used to create social modes. In COFRIS it includes contract *offer*, *inception*, *modification*, [un]*suspension* and *cancellation* events [10], performed by consensus or power.

**Revaluation** of economic relationships due to the changes in the environment or underlying object impairment.

**Reclassification** of economic relationships due to changes in the environment or in the allowed or intended actions of the enterprise.

**Economic transaction** is an exchange whereby an economic agent *transfers* one resource/obligation to obtain another for a gain in use value. Contains two opposite processes of partial, distinct and complete transfer, usually between the provider, that fulfills its contract obligations and the customer who settles its liabilities – customer’s obligations enforced by provider’s transfer. Generally, it is impossible to precisely allocate all the input and output resources/obligations for the exchange, therefore, only direct flows are accounted for within the exchange hierarchy, the upper level of which is the exchange activities of an enterprise with society in large for a reporting period. Particular exchange either *fulfils* an obligation, *settles* a liability or *recovers* an asset in either the planned or an alternative way. The agents of the parties of exchange are those of the contract or their assigned agents who act in planned or alternative roles, such as debtor and creditor.

*Example 1.* Flour owned by an economic agent C1, has an exchange value, and is transferable. That constitutes a disposition for the exchange manifested by a purchase by enterprise E. The exchange fulfills C1’s contractual obligation to deliver and E’s obligation to pay. The purchase creates a resource for E, with a new disposition for usage in the baking activity, and this resource, in the *role* of a raw material, is consumed in exchange for producing a pizza, that is then in turn sold to the economic agent C2, manifesting the sale disposition. Here the purchase, production and sale are exchange events, while flour and pizza are endurants – underlying objects of other endurants – economic relators representing ownership and contractual rights and potential economic benefits.

**Relator Phases** are listed in Fig. 1 in pink. In accordance with [7] status classes modeling constrains the evolution of an instance’s membership in a type along its lifecycle and generally includes four phases: *intended* (scheduled), *recognized* (active), *suspended*, *derecognized* (disabled/inactive).

**Contract phases:** The reciprocity economic relationship in its lifecycle progresses through phases by exchange events as depicted in Fig. 1. Neither UFO-S, nor REA regard liabilities as separate objects as required by FR. Our model of commitments/claims initially [12] was grounded on the service lifecycle model in UFO-S [13], adapting UFO-S patterns and including the partial and complete (realization) transfer phases. However, our interpretation of [economic] commitment is different from that in UFO-S and REA.



We regard commitment as a reciprocity and a disposition of exchange, but not as a separate promise of a standalone transfer. While the later are possible (as e.g., altruism), in the economical realm they are regarded as exceptional. The economic actions are motivated not by standalone commitments as stated in UFO-S and REA, but by the expected result of future exchanges. A further distinction from REA is that the liabilities or REA: Claims are not “imbalances of a duality” [9], but separate objects that may be revaluated or reclassified for FR. Phase dispositions describe all further phases, e.g., the offerings describe not only delivery, but also negotiation.

**Offering phase** is formed by a contract offer event as a meta-commitment by a provider to a customer, to exchange. The offering may further enter into the *negotiation* phase or become *expired*. Offerings are also a source of market valuation (prices) and should be part of [often public] smart contract offerings in the shared ledger.

**Fulfillment phase** begins with the contract inception event and persists while the partial provider and customer obligation fulfillment events reach *distinct obligation fulfillment* and ends when the provider reaches *complete obligation fulfillment* specified by the contract. Per IFRS, a provider may recognize *revenue* for transferring distinct resources. To illustrate, we will use the example from IFRS 15 [2] with some modification:

*Example 2.* E, a software developer, contracts with a C to transfer a *complete* bundle that comprises: a software licence; an installation service; updates and technical support for a one-year period. E sells the licence, installation service and technical support separately. The software remains functional without the updates and the technical support. Per [2], E determines that the obligation to transfer the licence is not separately identifiable from the customized installation service. Thus, the software licence transfer only partially fulfils the obligation and forms a contract asset, but not an unconditional liability of the customer (receivable); and the payment for the licence only partially fulfils the customer’s obligation and forms a contract liability, but not a payable. The license and the customized installation service are not *distinct*. At the same time, E concludes that the software updates and technical support are *distinct* from the other obligations in the contract, because they may be performed by other enterprises.

**Settlement phase** begins when the provider has accomplished complete fulfillment of its contract obligations and an unconditional *complete customer liability* is accrued. This phase is exemplified by accounts “Payables for purchase of energy”, “Receivables from the rental of properties”. Liabilities may also arise without the preceding phases (e.g. by statutory or court decisions). Liability accrued is settled in the *Liability settlement* phase; and occasionally may have a *Distinct liability settlement* phase. Customer liability usually refers to the most homogeneous resource - cash. The determination of liabilities can be subtle, as laid out in [2]. The *Obligation or Liability expired phase* occurs when those are not fulfilled/settled at a specified timing.

**Recovery phase** begins with the complete settlement of the liability and means that the asset recovery (use) may be started. Within a complete asset lifetime, the distinct parts that have their own lifetime may be recognized with particular recovery plan as e.g. property, plant and equipment that needs to be reported per IAS 16 [2].

#### 4. Shared Ledger and the Financial Reporting Process

In a **Shared ledger**, [multiple copies of] the economic contract and transaction ledgers are held by different parties, with data being added by consensus. As a result, a shared

ledger can provide gains in efficiency, trust and data reconciliation across all ledger participants. The shared ledger is a part of a common ledger, where data may be added by power or for information. In [11] the Essential business ontology for Blockchain (shared ledger) was developed, that is further extended for financial reporting purposes and grounded on COFRIS in this article. *Disintermediation*: Economic agents, can interact directly, without the need for an intermediary, including the ability to initiate direct transactions on digitized resources, which may be a cryptocurrency or a digital representation of real-world resources, such as land titles and collaterals.

**Information sharing** in the ledger is selective, ranging from global, i.e., among all members of society in large, to particular – among contractual group members, or a party and a counterparty, or participants within an enterprise. Sharing may be informative, i.e., unilaterally disclosed by an economic agent, or obtained in consensus. Sharing may be enabled on the type or instance level. Parties may have unshared sensitive or subjective information. *A [Digitized] resource or token*: represent the valued rights of an agent (for an underlying object) which can be transferred to a counterparty by simply transferring the token. For referenced [not Digitized] resource the transfer can be a representation of another action of rights transfer or it can effectuate the rights transfer itself (depending on legal context). Digitized resources and consensus are eliminating the need for reconciliation. Economic relationships are represented by referenced or digitized resources, and reciprocities by smart contracts or their offerings in a shared ledger.

The accounting interpretation of the contracts may be different for each party. The goal should be to obtain more consensus for assets/liabilities and resource/obligation interpretation in the contracts. Accounting in many cases does not directly capture the counterparty as the transferee/transferor of resources, but only as the debtor/creditor of the reciprocal liabilities. In a shared ledger, the requirements are stricter – the captured exchanges and reciprocities should be in mutual and possibly - public consensus.

We outline the process of FR in a shared ledger environment for a participating enterprise as consisting of the following activities:

1. Smart contracts (and contract offerings) of economic agents, containing mutual (unilateral) commitments, including information sharing specification, and IFRS relevant characteristics are added to a shared ledger by consensus of the parties.
2. Transactions of digitized resources/obligations are unchangeably recorded by consensus in a shared ledger, completely or partially fulfilling the smart contracts. Transfers together with accrual of liabilities caused by transfers, are accounted within smart contracts, including information sharing and IFRS relevant characteristics.
3. The effects of transactions involving resources/obligations are [de]recognized as assets/liabilities per IFRS requirements and enterprise policies in the shared or in the individual ledger part, according to information sharing specification.
4. FR relevant information gathered in activities 1 through 3 is abstracted to the type level, hiding sensitive instance details and forming an enterprise's multi-dimensional cube within global FR system.
5. The multi-dimensional cube is then aggregated, calculated, viewed, and mined per the IFRS Taxonomy requirements and financial reports are issued.

*Example 3.* E, a construction company, enters into a cost-plus smart contract with a customer D to build an object. D reimburses E for all its *allowed* expenses plus an additional variable payment that allows E to make a profit. E contracts with the subcontractors and vendors Ss and allows these contracts and contract transactions [complying to IFRS requirements] to serve as inputs to the contract with D, sharing with

D [and the global FR system] all the required details in consensus with Ss, possibly omitting the names of Ss. Furthermore, in consensus with D, shares all the required and non-sensitive details of the contract with D with the FR system. During the warranty period, D shares all relevant transactions involving the built object with E. This set-up benefits from having a single source of truth, simplifying administrative and control procedures, and the possibility of semi-automated execution of the smart contract.

### **5. Preliminary Ontological Analysis of the IASB Conceptual Framework for Financial Reporting**

Based on the ontology defined above, we can make some preliminary analysis of the paragraphs of Conceptual Framework for Financial Reporting.

*Objective and Usefulness of FR. Elements of Financial Statements.* While FR is based on transactions and other events, the concepts (e.g., elements) and standards are defined about presentation and the disclosure of information in specific locations of financial reports. In a shared ledger, to improve the qualitative characteristics of FR, *financial meaning* for the transactions should be captured, reconciled and validated mostly automatically and at the time when this meaning is created, i.e., starting at intentional phases of reciprocity. Thus, the concepts should primarily concern offerings, contracts, transactions and other events (all qualitative characteristics of FR [1]), regarding aggregation and location as important for perception, but as secondary issues. If financial information is to be useful, it must be relevant and faithfully represent what it purports to represent. The usefulness of financial information is enhanced if it is comparable, verifiable, timely and understandable and will improve by having upper level explanations and a shared ledger. Digitized rights/obligations, smart contracts and transactions bear a qualitative difference for FR, similar to *audited* information, thus it is relevant to classify the shared ledger supported objects separately from each other.

*Economic resources, Assets and Liabilities.* CF, with whom we mostly agree for basic definitions of assets and liabilities, defines classification as the sorting of assets, liabilities, equity, income and expenses on the basis of shared characteristics. Such characteristics include the nature of the item, its role (function) within the business activities and how it is measured. These characteristics overlap with ours, but lack *timing, prospects, impairment, provision, quantity etc.* as a first-class concepts.

The *correlation* association in para 4.25 of [1] is expressed as: If one party has an obligation to transfer an economic resource (a liability), it follows that another party (or parties) have the right to receive that economic resource (an asset). but in para 4.26: A requirement for one party to recognise a liability (or asset) and measure it at a specified amount does not imply that the other party must recognise the corresponding asset (or liability) or measure it at the same amount.

In the context of a shared ledger, we will name the principle in para 4.25 as the *correlative principle*. Para 4.26 substantially undermines it, making it generally impossible to regard one economic relationship as common to all mediating parties. Having obtained consensus on relationships is an important feature which greatly improves FR faithfulness and reasoning and should be enforced by standards. Of course, after initial recognition and measurement, that relationship may gain specific internal features, however, the mutual relationship should be immutable. E.g., the new Lease standard IFRS 16 [2] states that “it is not essential that the lessee and lessor accounting models are symmetrical”, noting the costs of lessor’s accounting as the primary reason.

Such costs are not incurred when using a shared ledger, in which the lessor may see the lessee's information, while the benefits are in observing the state and changes of the lessor's resources.

FR requires individually valuating and classifying the economic relationships. If these relationships are initially in consensus in the shared ledger, the only way to ensure further consensus is for both parties to revalue based on the same market price, or to coordinate their reclassification based on a smart contract or regulation by IFRS.

*Example 4.* In a shared ledger environment, an EU resident E has a USD debt to the U.S. company C, the debt value in USD doesn't change, but E revalues it periodically to EUR for FR, using an E specific exchange rate from a bank. Furthermore, C may reclassify this debt as an *overdue* receivable, this fact should possibly be reconcilable with the E. Valuation is forward-looking while the transaction captures historical information.

*Executory contract.* The definitions of executory contracts in [1] roughly conform to our contract fulfillment model in Section 3, but lack concrete disclosure/recognition requirements, precise patterns and phases of contract and liability fulfillment that are common to several contract standards.

*Equity, income and expenses* in Conceptual Framework for FR are defined in paragraphs 4.44-49: *Equity claims* are claims on the residual interest in the assets of the entity after deducting all its liabilities. *Income* is increases in assets or decreases in liabilities that result in increases in equity, other than those relating to contributions from the holders of equity claims. *Expenses* are decreases in assets or increases in liabilities that result in decreases in equity, other than those relating to distributions to holders of equity claims. Per these definitions, each economic event that changes the value of an asset or a liability, simultaneously changes the value of another economic relationship - the equity claim of the owners. The equity claim, thus, is *an enduring* and has the disposition of exchanging the value of this claim against dividends, residual assets after liquidation etc. per articles of incorporation, or other agreements or instruments. These claims may form a base for further income tax obligations; or serve for intended purposes, e.g., some will be shared and some will be reinvested or reserved, i.e., different types of *endurants* may be specified for different types of dispositions to future exchanges.

Meanwhile, equity claims are further classified by the type of economic event that caused the change of the asset or liability, e.g., "Fuel expenses", i.e., the event types are used as accounts. These classifications depict past performance of the enterprise and thus are *current life* or *final life events* as described in [16]. Thus, we argue that the Income and Expenses elements as changes in Equity and *endurants* are semantically overloaded by depicting economic performance events. Furthermore, economic event and disposition types are depicted in other Financial Statements [2] (Cash flow and Equity flow) and Statement Notes, but are not introduced as concepts (elements).

## 6. Conclusions

Financial reporting standard setting, implementation and the corresponding information system development, is at present a partially informal and long process and as exemplified by other domains, may be improved using ontological conceptual modeling approaches. That in turn may improve financial reporting information systems models.

Existing foundational and core ontologies, as exemplified by UFO ontology network usage, provide upper level patterns for representing FR concepts and

relationships. Existing domain ontologies such as the REA Accounting ontology do not provide sufficient and complying concepts for modeling FR. This paper has shown how COFRIS is addressing this research gap. Our analysis of economic relationships shows the benefits of this ontology, not only on the conceptual level but also with respect to the use of new (shared ledger) technology.

Contract economic relationship as a disposition of economic exchange events, creating new or progressing existing reciprocity lifecycle, is a fundamental and reusable pattern for capturing economic phenomena for FR. Ontological analysis allows for the explication of the core reciprocity phases and exchange types to capture full partition of the economic phenomena usable for FR. Introducing event reification per [16] should prevent income/expenses elements from semantic overloading and unify FR concepts for performance statements and statement notes. Aligning FR concepts with UFO allows for the understanding of FR concepts meaning and classification in the enterprise domain, as for instance, economic resource and assets definitions. Elaboration of correlative relations between enterprise and counterparty should lay a foundation for consensus based accounting in a shared ledger environment. As a further work, a full validation of COFRIS by modeling all IFRS standards is needed, including solving the ontology version transition problem.

## References

- [1] IASB Exposure Draft ED/2015/3. Conceptual Framework for Financial Reporting, IASB, 2015
- [2] IASB homepage, <http://www.ifrs.org/issued-standards/list-of-standards>, IASB, 2017.
- [3] G. Guizzardi, "Ontological foundations for structural conceptual models," University of Twente, 2005
- [4] Almeida, J.P.A., Guizzardi, G.: An Ontological Analysis of the Notion of Community in the RM-ODP Enterprise Language. *Computer Standards & Interfaces*, v. 34, p. 1, (2013).
- [5] Sales, T.P., et al: An Ontological Analysis of Value Propositions, EDOC 2017: 1-10.
- [6] M.C. Gerber, A. J. Gerber, A. van der Merwe, The Conceptual Framework for Financial Reporting as a Domain Ontology, Twenty-first Americas Conference on Information Systems, Puerto Rico, 2015.
- [7] A. Artale, et al., Formalising Temporal Constraints on Part-Whole Relations. *KR 2008: 673-683*
- [8] W.E. McCarthy, "The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment", *The Accounting Review*, (1982), 544-577.
- [9] ISO/IEC. Information Technology—Business Operational View—Part 4: Business Transactions Scenarios — Accounting and Economic Ontology, ISO/IEC FDIS 15944-4: 2015.
- [10] H. Weigand, et al., "Management Services-A Framework for Design", *CAiSE*, 2011: 582-596.
- [11] J. de Kruijff and H. Weigand, Understanding the Blockchain Using Enterprise Ontology. *CAiSE*, 2017: 29-43.
- [12] I. Blums and H. Weigand, Towards a Reference Ontology of Complex Economic Exchanges for Accounting Information Systems. EDOC 2016: 119-128
- [13] J. Nardi, et al., Towards a Commitment-based Reference Ontology for Services, EDOC, 2013.
- [14] C. Azevedo, et al., An Ontology-Based Well-Founded Proposal for Modeling Resources and Capabilities in ArchiMate, EDOC 2013, Vancouver, 2013.
- [15] J.Guerson, et al., OntoUML Lightweight Editor: A Model-Based Environment to Build, Evaluate and Implement Reference Ontologies, EDOC 2015, Demo Track, Adelaide, Australia, 2015.
- [16] G. Guizzardi, et al, Ontological Considerations about Events and Endurants in Business Models, *BPM 2016*
- [17] OMG. Financial Industry Business Ontology Foundations, v1.0, 2015.
- [18] C. Criffo, Almeida, J.P.A., Guizzardi, G., From an Ontology of Service Contracts to Contract Modeling in Enterprise Architecture, 21st IEEE Enterprise Computing Conference EDOC 2017.
- [19] F. Ruy, et al., "Ontology Engineering by Combining Ontology Patterns", *ER*, 2015.