Noname manuscript No.
(will be inserted by the editor)

Using Ontologies to Capture the Semantics of a (Business) Process for Digital Preservation

Rudolf Mayer · Gonçalo Antunes · Artur Caetano · Marzieh Bakhshandeh · Andreas Rauber · José Borbinha

Received: date / Accepted: date

Abstract IT-supported business processes and computationally intensive science (called e-science) have become increasingly ubiquitous in the last decades. Along with this trend comes the need to make at least the most important of these processes available for the long term, to allow later analysis of their execution, or even a re-execution. As such, the preservation of scientific experiments and their results enables others to reproduce and verify the results as well as build on the result of earlier work. All but the simplest processes require to be described by a multitude of information objects, as well as their interconnections and relations, to be successfully preserved. To enable a semantic description of these objects in a structured manner, we developed a formal meta-model that can be utilised in the digital preservation of a process. The meta-model describes classes of elements and their relations, in the

Rudolf Mayer
Secure Business Austria, Favoritenstrasse 16, 1040 Vienna, Austria
E-mail: rmayer@sba-research.at

Gonçalo Antunes
INESC-ID, Lisbon, Portugal
E-mail: goncalo.antunes@ist.utl.pt

Artur Caetano
INESC-ID, Lisbon, Portugal
E-mail: artur.caetano@ist.utl.pt

Marzieh Bakhshandeh
INESC-ID, Lisbon, Portugal
E-mail: marzieh.bakhshandeh@ist.utl.pt

Andreas Rauber
Vienna University of Technology, 1040 Vienna, Austria
Secure Business Austria, Favoritenstrasse 16, 1040 Vienna, Austria
E-mail: rauber@ifs.tuwien.ac.at

José Borbinha
INESC-ID, Lisbon, Portugal
E-mail: jose.borbinha@ist.utl.pt

form of ontologies, with a core ontology describing the generic concepts, and extension mechanisms to map supplementary ontologies describing more specific aspects. In this paper, we present the overall architecture and individual ontologies, and motivate their usefulness via the application to use cases from different domains.

Keywords Digital Preservation · Process Preservation · Preservation Metadata · Context

1 Introduction

Digital preservation deals with ensuring long-term access to digital objects over a long period, when a change in user communities or the technological environment, e.g. file formats, software, operating systems or hardware platforms, would have rendered the document otherwise inaccessible.

So far, the main focus of research in this area has targeted digital objects that are static in their nature, such as text and multimedia documents. Also, the focus has often been on single items, such as documents, and not predominantly on digital objects that are interacting with and embedded in a context with other digital objects. There is however the need to extend the research towards dynamic objects (such as interactive art or video games), and beyond to whole processes and workflows.

The latter is an emerging topic especially in disciplines such as E-Science, where data-intensive experiments form a core of the research. These experiments and their results need to be verifiable to others in the community. They need to be preserved as researchers need to be able to reproduce and build on top of earlier experiments to verify and expand on the results. Besides E-Sciences, also processes executed in businesses or public organisations frequently need to be preserved, for reasons such as liability cases, where a company for example might need to prove that it executed its processes correctly, and faults did not occur because of their design or manufacturing.

This calls for a paradigm shift in Digital Preservation, from isolated and single files that are produced as the final outcome of a process, to the complete chain of data processing and decision making. In E-science, for example, the current approach is often to archive only the resulting publications, and at very most the data sets. However, in experimental and data intensive science there are normally a number of transformation steps applied to the data, and each of these steps might depend on a specific version of a computing environment, and a set of parameters to configure. Thus, such an approach is insufficient.

A process model for digitally preserving a process was introduced in [41], and is described below in Section 2. The process model includes a risk assessment step to identify processes worth preserving, an assessment of preservation approaches, and a risk treatment, which is the actual preservation and later redeployment of the process. The *context* of workflows and processes is formed of the environment they are executed in, and ranges from immediate and local aspects such as the software and hardware supporting the process, to

aspects such as the organisation the process is executed in, service providers, and laws and regulations. This process context is thus a crucial aspect of the aforementioned process preservation approach. On the one hand, it provides the semantics necessary to understand the process and identify and evaluate risks, and on the other hand, it supports the redeployment of a process into a suitable IT infrastructure (see Section 2).

The context of all but the simplest processes includes a multitude of elements, as well as their interconnections and relations. A meta-model that provides means for modelling process context and dependencies, and their semantics, in a structured manner so that all the information required for preserving and redeploying a process can be captured is necessary. To enable such a semantic description of these, during the course of the TIMBUS project[1], we developed a formal meta-model, the *TIMBUS context model*, that can be utilised in the digital preservation of a process.

The context model described in this paper is an evolution of an initial approach being presented in [30]. As it has proven difficult to develop a monolithic, comprehensive meta-model that can fit for all possible types of processes, the meta-model is therefore designed to be adaptable to requirements by different use cases. It is thus based on a set of smaller models that each describe semantics of specific concerns, and are integrated via mappings to a core model. Ontologies are a well-suited method to implement this architecture; specifically, we use OWL, the Web Ontology Language. Our meta-model thus consists of a set concern-specific ontologies that are integrated via mappings to a core (upper) ontology. Ontologies also enable flexible querying and reasoning of the models. Inference also allows as, for example, to assess the consistency of models against rules, verify the completeness of models, or produce reports based on the contents of the model. When designing the ontologies, the principles of concern-orientation, extensibility, modularity and viewpoint-orientation were utilised to constraint the meta-model.

The major contribution of our work are as follows:

- An extensive stakeholder requirement elicitation, where the concerns regarding digital preservation of real-world use cases were identified.
- An architecture that allows for integration of multiple languages describing specific aspects, allowing a flexible adaptation of the meta-model.
- Identification of a suitable core ontology that allows to integrate many specialised models.
- Identification of models that can describe the concepts identified by the use case stakeholders, and providing the integration to the core ontology.

The rest of this paper is organised as follows. Section 2 will provide an overview on related work in the area of Digital Preservation and specifically Preservation Metadata. Section 3 will then described the stakeholder requirements and guiding principles for the context model architecture. Section 4 and

[1] `http://timbusproject.net/`

Section 5 will then described the core and extension ontologies suitable for describing business processes. Section 6 will show examples of how the model has been applied to use case scenarios from different domains, before Section 8 will provide conclusions and an outlook on future work.

2 Related Work

In this section, we review and discuss related work in the areas of Digital Preservation, with a specific section dealing with Process Preservation and its relation to the context model, as well as Enterprise Architecture and Business Modelling.

2.1 Digital Preservation

The term *Digital Preservation* as defined in the UNESCO Guidelines for the Preservation of the Digital Heritage [48] is the process of preserving data of digital origin. The two main strategies for the preservation of digital heritage listed are migration ([27]) and emulation ([15]).

Although digital preservation has been traditionally driven by memory institutions and the cultural heritage sector [48], it is increasingly recognised that it is a problem affecting all organisations that manage information over time, and as such it affects most of contemporary organisations where information systems provide important support to the business.

The Open Archival Information System (OAIS) Reference Model [20] remains an important source of concepts to the field. It provides a framework for the understanding of archival concepts needed for long term digital information preservation and access. However, it lacks directives and guidelines to address complex preservation scenarios with multiple business support systems and complex digital objects in place. In such scenarios, digital preservation requires a holistic view, acting as a combination of organisational and business aspects with system and technological aspects, so that all the contextual aspects surrounding a complex digital object can be captured and the objective of rendering it in the future in the same or in similar conditions can be attained.

With this holistic concern in mind, digital information life cycle models have been designed, of which the DCC Curation Life Cycle Model [18] and the SHAMAN Information Life Cycle [9] are noticeable examples. The DCC Curation Life Cycle Model elongates the traditional scope of preservation to include curation. It addresses two phases: a *Curation* phase, which might involve the creation of new information or the access and reuse of already existing information and its appraisal and selection; and a *Preservation* phase, which involves the ingestion of the information into the archive, the application of preservation actions, and the storing of that information. During the two phases, community watch and participation and preservation planning

take place in order to keep descriptive metadata and representation information up to date. The SHAMAN Information Life Cycle, besides including the *Archival* phase already addressed by the OAIS model, suggests two additional pre-ingest phases and two additional post-access phases. The pre-ingest phases *Production* and *Assembly* aim at the capturing of the context of production of the object and its assembly into an information package, respectively. The post-access phases *Adoption* and *Use* concern the preparation of the retrieved package so that its information contents can be used.

This renewed understanding of preservation also creates the need for the development of new conceptual models that are able to synthesise this knowledge and make it re-applicable to different scenarios. The SHAMAN Reference Architecture [6] resulted from an infusion of knowledge in the digital preservation field and standards and best practices from the business and IT governance fields. It defines a set of preservation capabilities and their relationships and interaction with other organisational capabilities, so that its integration with the overall capabilities of an organisation is facilitated. The overall objective is to promote the alignment between the preservation objectives of the organisation, the organisation's processes, and the existing technological infrastructure. Additionally, the work done on the CASPAR project on Preservation Networks [13] is a relevant reference on the capturing of the dependencies of complex digital objects through the usage of entity-relationship-like models, although business and organisational aspects are left out of it.

In the terms of the OAIS[20], contextual information of digital objects can be denoted as *Representation Information*, i.e. the information needed so that Designated Communities can understand the digital object at a later point in time, as well as Preservation Description Information (PDI), i.e. is the additional metadata needed to manage the preservation of the objects. The context of information needed for preserving processes is considerably more complex than the objects the works described above had in mind, as it not only requires dealing with the structural properties of information, but also with the dynamic behaviour of processes. Thus, a meta-model to describe the Representation Information and PDI of business processes (in the form of their digital representation) along with the surrounding context needed for its long-term understandability is an innovative target being pursued by our approach.

For any digital preservation action, be it migration or emulation, the results of the action on the rendering have to be verified to see if the action in question is usable for the specific setting. Comparing different options is the challenge of preservation planning. In [7] a preservation planning workflow that allows for repeatable evaluation of preservation alternatives is described. When migrating files automatic characterisation can compare digital objects before and after migration to see if significant properties necessary to render the object correctly are still present. The context model shall also support these concerns.

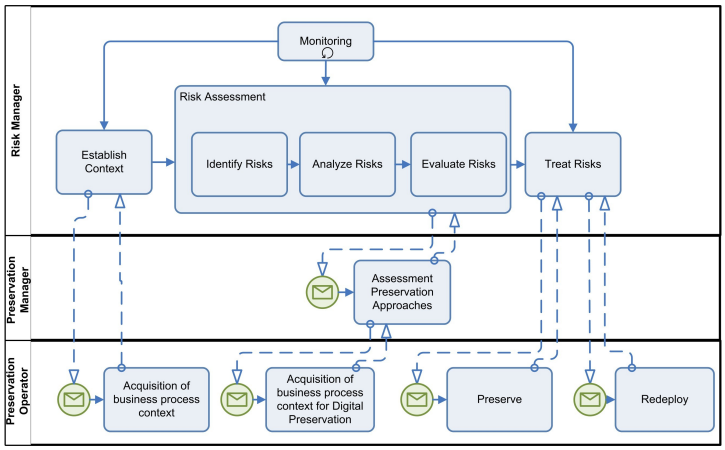


Fig. 1 TIMBUS framework for process preservation

2.2 Process Preservation Framework

A process model for digitally preserving a process was introduced in [41], and is depicted in Figure 1. Risk management, a well establish field with the goal of defined prevention and control mechanism to address risks related with assets and activities, forms a core activity in this preservation process. Preservation can be seen as a potential method to mitigate risks, derived from the potential loss of information over time. The risk management process used in the process preservation framework is based on the ISO 31000 standard [19]. The preservation process model includes several steps, which can be divided into three phases: plan, preserve and redeploy. These will be described below.

The *planning* phase concerns the capturing of the process and its context. Triggered by the risk management, a first, coarse acquisition of the process context and the Assessment of Preservation Approaches are executed in the planning phase. The process context meta-model is essential to allow for a structured representation of these context aspects. Risks of the process are further identified by reviewing contractual, policy and legal obligations. Driven from the risk management perspective, where applicable, digital preservation is considered as a potential mitigation strategy for risks. At this stage, e.g. information on resources that might become obsolete, and how they impact the overall process execution, is an important bases for the analysis. The assessment of preservation strategies identifies and evaluates different approaches to make the process available in the future. To allow for a more detailed planning, in a subsequent step, more details of the process context are identified and described in the context model. A specific software module, the Preservation Alternative Identification module, can support the preservation expert in identifying potential preservation actions [31]. This module needs the context model as an input, to be able to reason on possible actions. If for example file

formats are identified for data used in the process, the module can propose format migration strategies.

Within the *preservation* phase, process data is captured from the original environment. The context model servers as descriptive a meta-data to this data, and relates single elements to each other. Optionally, preservation actions, such as migration or emulation, might be executed or prepared in this step, to ease a future redeployment. Finally, the data is prepared for archival storage.

During the *redeployment* phase, at some point in the future, a process is re-initiated in a new environment. The context model is essential in this step to understand all the dependencies of the original process, and to be able to make informed decisions on changes needed in the redeployed environment.

When redeploying a process, it is important to verify that the execution in the new environment is still according to expectations and requirements. To this end, a framework for process preservation validation has been introduced in [32]. This framework relies on the process context model to hold information on the initial and redeployed process contexts. It extends on the context model to allow for defining key characteristics (also referred to as significant properties) that need to be maintained, and how these can be measured and evaluated.

The process context is therefore a crucial aspect of the process preservation approach. On the one hand, it provides the semantics necessary to understand the process and identify and evaluate risks, and on the other hand, it supports the redeployment of a process into a suitable IT infrastructure.

2.3 Enterprise Architecture and Business Modelling

The capturing of context information is crucial for the effective preservation of business processes. Nonetheless, modelling of this context is a typical feature in enterprise architecture and in business process modelling. Enterprise architecture can be defined as "a coherent whole of principles, methods, and models that are used in the design and realization of an enterprise's organizational structure, business processes, information systems, and infrastructure" [25]. Enterprise architecture modelling frameworks and languages typically provide support for the capturing of information that addresses the needs of different stakeholders of the organization.

Several enterprise architecture frameworks have been developed through the years, providing methods and techniques for developing and organizing models [40]. Some approaches try to be as comprehensive as possible up to a certain level of abstraction, providing a meta-model that approaches the different levels of the organization. Such approaches are also referred as being "holistic" [14]. Notable references are ArchiMate [43], The Open Group Architecture Framework (TOGAF) [42], the U.S. Department of Defense Architecture Framework (DODAF) [46], and the U.K. Ministry of Defense Architecture Framework (MoDAF) [45]. All of the mentioned modelling frameworks allow

the capturing of business processes, application components, and infrastructure components, besides other relevant elements of the organization.

Other modelling languages focus the capturing with greater detail of business process information, focusing on aspects such as coordination and cooperation between the different participants of a process. The most notable example is that of the Business Process Model and Notation (BPMN) [34], which allows the formalisation of business process representations that can also be executed by determined workflow engines, providing in that way a bridge between process design and process implementation. However, extra-process details and dependencies are not captured by the language.

3 A Context Model for Process Preservation

In this section, we describe the architectural principles and requirements that drove the development of the TIMBUS Context Model. We also detail the architecture of the model that was derived from these principles and requirements.

3.1 Architecture Principles

Ontology engineering is a difficult task, especially within a setting such as the one faced for process preservation, where there is the need to arrive at a representation of knowledge not from a single domain, but deal with concepts that cut across several different domains. In order to ground our proposal in good practice and to allow for a structured and extensible approach to ontology engineering, we have followed an architecture-oriented approach to the elaboration of the context model.

According to the ISO 42010:2011 standard [21], an architecture description includes views that address the concerns of the system stakeholders. Those views are created according to the viewpoints of the stakeholders. Correspondence rules are created between the views to enforce composition, refinement, consistency, traceability, dependency, constraint and obligation relationships between the views. In this way, architecture can function as a communication tool between different stakeholders, as each is presented with its own consistent view on the system of interest, displayed in a manner that is understood by him, without any ambiguities or conflicts between concepts.

Based on the recommendations of the aforementioned standard and on the analysis of the problem, a set of architecture principles were defined. An architecture principle can be described as "a declarative statement that normatively prescribes a property of the design of an artefact, which is necessary to ensure that an artefact meets its essential requirements" [16]. The first two principles are derived directly from the standard:

– **Concern-orientation** The context model shall represent the concepts necessary and sufficient to address an explicit set of modelling concerns.

This means that the model shall be derived from the questions that need to be addressed and to provide answers to those questions. This also means that the model shall not support any concepts that are not explicitly derived from concern. The principle of concern-orientation and the principle of viewpoint-orientation (below) are described in detail in the ISO 42010:2011 standard [21], which defines requirements on the description of systems and enterprise architecture.

– **Expressiveness** The context model shall be able to represent the domain concepts without ambiguity. This entails defining the minimum set of types and relationships to describe a domain.

Although the need for multiple viewpoints and views on the system is recognized by the standard, the truth is that it is a challenge to maintain these relationships when multiple independent meta-models and models are involved [26]. As such, some architecture modelling approaches try to be as comprehensive as possible up to a certain level of abstraction, providing a meta-model that approaches the different aspects of systems and organizations [14], and from which multiple viewpoints are defined and views are derived.

But the fact is that, many times, the integration of many meta-models is imperative in order to provide project or domain-specific solutions to many problems [14],[51]. Therefore, there is a need for reconciling each individual view with each other, especially in the cases where multiple underlying meta-models are used, guaranteeing that the overall architecture is cohesive and consistent. Given this, the following architecture principle was defined.

– **Extensibility** The model must cope with extensions because context modelling entails using multiple concurrent perspectives on the same problem. This derives from being able to answer to multiple concerns. Therefore, domain-specific and domain-independent models must coexist and the overall context model must cope with multiple model transformation and integration. A specific concern is that the model is extensible to new application domains, beyond the ones that are the focus of the use cases in the TIMBUS project.

The integration of multiple meta-models involves determine possible relationships between the concepts belonging to different meta-models. If there is significant overlap between the meta-models, it means that a great number of mappings and/or transformations will have to be considered. If changes are made to a meta-model/model, changes also need to be propagated to a potentially large number of concepts/relationships belonging to other meta-models/models.

This problem can be facilitated with tool support. However, using multiple meta-models often requires using multiple specialized tools, making it especially problematic to ensure the consistency across models, especially when the meta-models evolve [24]. The automatic or semi-automatic validation of the conformance of the models to the meta-models might be available or not, depending on whether the models are fully computable (abstract syntax and

semantics) or not, or on the existing tool support. Hence, for the sake of the manageability of the updating process, it is important that the architecture complies with the principle of modularity, which is defined as follows.

– **Modularity** The models must follow the principles of high-cohesion and low-coupling. Observing these principles contributes to expressiveness and extensibility of the context models. It is especially important that adding new domain-specific aspects to the model does not interfere with the ontologies already present in the context model.

Architecture should address the information needs of stakeholders. Most notably, architecture should provide support to decision making [25], [22], functioning as an important information source for informed decisions. Given that the purpose of models is to answer questions about the modelled entities [22], the ability to analyse the models for retrieving answers to the stakeholders is also desirable [12]. Stakeholders should be able to obtain as much useful information as possible from the knowledge contained in the models, which might reach a great level of complexity when elaborated with detail [8]. Such analysis can of course be made without any automation, however, it can be difficult obtain useful information when dealing with complex scenarios [10].

Given this, creating computable representations for architecture models comes out as a relevant need [28]. The combination of the computable models along with the enforcement of the dependencies brings benefits for enterprise architecture, such as retrieval, management, and processing of information, allowing the specification of specialized viewpoints and the generation of the corresponding views. One example of such benefits is dependency analysis, which can be used for generating views depicting the dependencies existing between the different elements of the architecture. Hence, the following principle is defined.

– **Viewpoint-orientation** The model must support defining views over subsets of its concepts. This serves to facilitate the communication and the management of the models as viewpoints act as a separation of concerns mechanism. Viewpoints will facilitate addressing multiple concerns and managing the multiple extensions required to handle these concerns.

3.2 Stakeholder Requirements

In order to derive the requirements for the context model, it is necessary to determine the stakeholders' needs concerning the information they wish to obtain from the context model. During the course of the TIMBUS project, a number of use cases, from domains such as e-Science, civil engineering, or e-Health were specified in great detail, and became available to a detailed analysis. Thus, these formed the base for the context model. The approach for identifying aspects relevant for the digital preservation, and thus elements of the context model, was by asking the use case stakeholders to elaborate a set

of questions that they considered relevant to be answered if the process was to be preserved, as well as the expected outcomes. In total, more than 100 questions were elicited. Some of these are presented in Section 5, the complete list can be found in [4].

Those questions were then processed to find entities that should be in the context model, and classified into different groups. Some questions were independent of the use case, while some are rather specific to the domain the use case processes are embedded in. This is an indication that our context model should be able to provide generic elements that are used in a wide range of cases, and allow for extension for specific application domains. As such, its main requirements are:

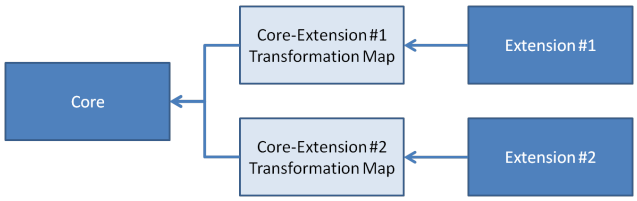
- Represent domain-specific business processes. The context model must support the generic description of business processes plus the domain-specific features of each approached scenario.
- Integrate multiple representations. Representing the context of a business process implies capturing the processes and their environment. This implies that the representation used for organising that information will have intersecting aspects that need to be integrated. These aspects may include:
  - Strategy (e.g. requirements, rules, drivers, principles, indicators),
  - Organisation (e.g. people, locations, roles),
  - Operations (e.g. processes, services, products),
  - Business support systems, including the application infrastructure (e.g. applications, software) and the technological infrastructure (e.g. hardware nodes, communication devices).
  - Domain-specific aspects related to the approached scenarios.
- Provide the means to analyse the representations of business processes. The context model representations are used to facilitate the assessment of business process preservation and redeployment from a conceptual and technical perspective. The verification and validation of both preservation and redeployment is important, and the context model shall provide a basis for performing such tasks.

Reasoning capabilities will eventually be used to retrieve answers to those stakeholder questions from a model of a use case.

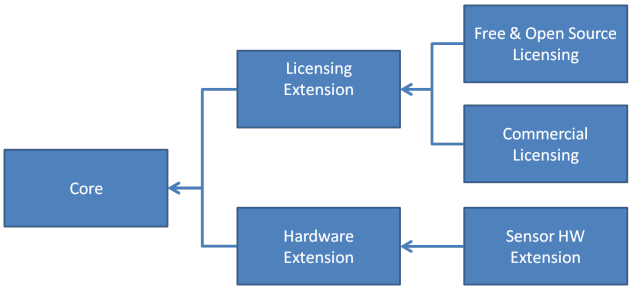
3.3 Context Model Architecture

Based on the requirements identified above, it was decided that ontologies would provide a suitable means to author the meta model. Specifically, we decided to adopt the Web Ontology Language (OWL) [47]. OWL is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. The architecture of the proposed context model established on the following concepts (cf. Figure 2(a)):

- A core ontology describing generic concepts regarding processes



(a) Mappings between Core (generic) and extension (specific) ontologies



(b) Hierarchical refinement of concepts

Fig. 2 Context Model architecture

- Extensions describing specific aspects regarding digital preservation of processes
- Extensions describing use case domain specific aspects
- Ontology Integration
- Model Transformation

The Core ontology represents a neutral, domain-independent language that is able to represent the core concepts of the context model. It is designated domain-independent since it does not address any specific domain-dependent concerns, but rather cuts across the whole organisation running the process, in a similar manner as for example enterprise architecture frameworks model businesses[25]. In ontology engineering, sometimes such an ontology is referred to as an "upper level ontology".

An extension ontology represents a more specific language that addresses a particular set of concerns. These can on the one hand be generic digital preservation concerns, and on the other hand specific to the domain of the process to be preserved. For the former, preservation metadata models, or models on software licensing that would describe the concepts required to model the universe of licenses, and may include concepts that cover licensing models, licensing agreements, copyrights, license types (e.g. free software, open source), etc., are representatives.

Extensions might be arranged in hierarchies, as illustrated in Figure 2(b). For example, we might have an ontology that describes concepts of electronic devices (hardware), and then a refinement ontology that adds on top of these concepts to describe specific properties and types of sensors. Another example is software licenses, which might be described in more detail for both free and

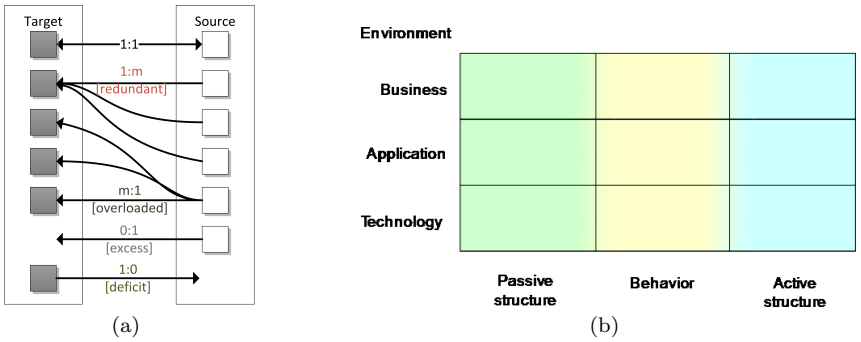


Fig. 3 (a) Types of representational deficiencies [49], (b) The ArchiMate Framework [43]

open source and commercial licenses, each having different sets of properties. This approach facilitates layering multiple extension ontologies according to the modelling needs.

The Context Model for a specific process to be described will comprise a number of extension ontologies. Some of these extensions will be already available, as they were developed for existing use cases, and some of will have to be developed if new specific concepts are needed. Each extension ontology should be designed with the minimum set of concepts required to describe a given domain. The Context Model has to be easily extendible, so that an additional extension ontology is added to the model without affecting the existing extensions. However, the number of extensions that will be part of the Context Model will depend on the actual domains needed to represent all the concerns of stakeholders of the addressed scenarios.

The ontology integration described above makes use of mapping transformations to relate an extension to the Core, or to relate multiple extensions to each other. Model transformation entails defining a mapping strategy from a source model to a destination model ([17], [38]), building a new ontology by finding common concepts between two (or more) different ontologies [35]. Other integration techniques exist, such as ontology alignment and ontology merging [35]. However, given the architecture of the context model, ontology integration is the most adequate technique.

Depending on the extension ontology to be integrated, the mapping might create different types of representational deficiencies, which are of course expected since the extension might address very specific concepts not present in the Core. Any deviation from a 1:1 mapping should be considered such a deficiency. Two aspects might be analysed: ontological completeness and ontological clarity. The Bunge-Wand-Weber (BWW) representation model ([11]) can be used as an inspiration in the study of ontological completeness by analysing the extent to which a source modelling language has a deficit of entities mapping to the set of entities proposed in target modelling language. Ontological clarity might be analysed by determining the extent to which the source modelling language constructs are overloaded (i.e. they map to two or

more constructs in the target), redundant (i.e. two or more language constructs map to the same construct in the target model), or excess (i.e. they map to none of the constructs in the target model) (cf. Figure 3(a)).

4 Context Model: Core Ontology

To ground our approach to context modelling and address the aforementioned principles, we decided to use the ArchiMate 2.0 language ([43]) as Core ontology. ArchiMate is an international standard that covers the domain of enterprise architecture. The ArchiMate modelling language includes a minimum set of concepts and relationships and the framework includes a minimum set of layers and aspects to enable modelling of the majority of cases ([43]). Therefore, it can be considered a domain-independent language in the setting of enterprise architecture. The motivation to select the ArchiMate language as the core is that its design principles largely overlap with those of the context model. Namely, ArchiMate is a language that provides a high-level of abstraction, is concern-oriented and viewpoint-oriented and was designed with extensibility in mind – which proves to be important as ArchiMate does not address domain-specific concerns that were identified as stakeholder requirements, such as licenses, patents, legal requirements, sensors, and so on.

The framework organises the modelling language concepts in a 3×3 matrix: the rows capture the enterprise layers, i.e., business, application, and technology, and the columns capture cross layer aspects, i.e., active structure, behaviour and passive structure. Figure 3(b) depicts this organisation of the framework. Figure 4 depicts the main concepts provided by ArchiMate, the colours of the elements corresponding to the categorisation into active structure, behaviour and passive structure. The active structure contains entities capable of performing behaviour; the behaviour, contains elements defined as units of activity performed by one or more active structure elements; and the passive structure contains objects on which behaviour is performed. Some of these elements are described below, for a complete and detailed description, please refer to [43].

The business layer is concerned with products and services offered to external customers, realised by the business processes of the organisation, which are performed by business cases. The *Process* concepts allows to model activities, which can be performed by *Actors* and *Roles*. Each activity can consume or produce *(Business) Objects*, thus realising inputs and outputs, and utilise *(Business) Services*, which are accessed via *(Business) Interfaces*. *Events* as well as *Junctions* (and/or) complement the business layer concepts, which therefore allow to model the details of a business process. Additional concepts provided, such as roles, actors and services, go beyond the expressiveness of other process modelling frameworks, such as Business Process Model and Notation (BPMN) [34].

The application layer is concerned with the application services, which support the business layer and are realised by software applications. *Compo-*

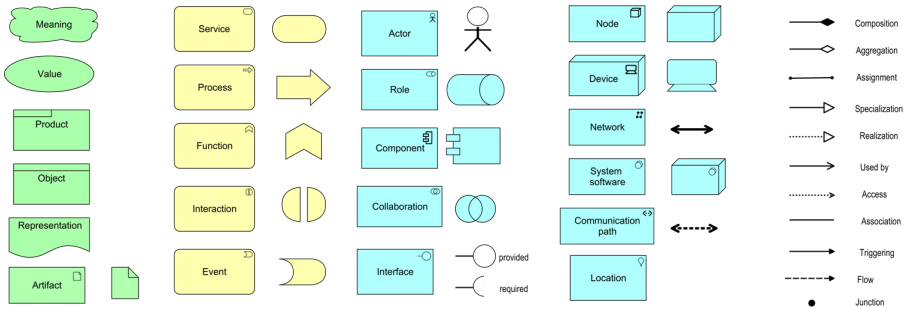


Fig. 4 Concepts and Relations in the ArchiMate meta-model [43]

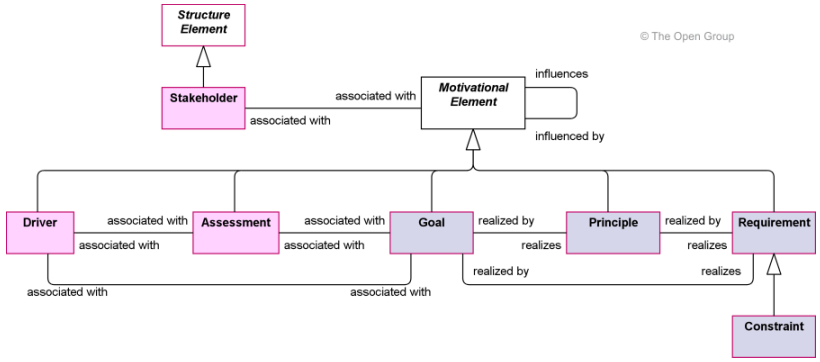


Fig. 5 Concepts in the ArchiMate Motivation extension [43]

nents form the basic building blocks to model a software environment, where Components might *Collaborate*, use *(Application) Services* via a defined *(Application)Interface*, and exchange *(Data) Objects*.

The technology layer is concerned with the infrastructure services offered to applications, realised by hardware and system software. A *Device* represents hardware, and can itself be composed of several devices, such as a Server computer might consists of processors, RAM and hard disks. A *Node* is an abstraction and represents a computing resource, such as a *Database server*. *SystemSoftware*, such as an operating system, a database management system, or a web server, is usually deployed on a node. *Artifacts* are pyhsical pieces of data, such as a file. Analogous to the other layers, infrastructure *Services* and *Interface* can be described. Finally, a *Network* realizes the *Communication Path* between different Nodes.

ArchiMate also defines the possible relationships between elements, both as intra- and inter-layer dependencies. Inter-layer dependencies between two layers are usually fulfilled by the “Used By” relationship, where the lower-level layer usually provides a service which is used by elements at the higher level layer. Other types of inter-layer dependencies can also occur, such as when an element at a higher layer is realised by an element at a lower layer, or

when a lower layer element is assigned to a higher layer element (for instance, when a business process, function, or interaction is fully automated, an assign relation is used in conjunction with the respective application component; the same also happens between business service and application interface). As an example for intra-layer relations, the Business layer provides the *triggering* relation for modelling the execution sequence of the process steps.

ArchiMate also defines the *Motivation extension*, which adds more concepts to the core ArchiMate language. These are shown in Figure 5. This extension includes concepts to describe the actual motivations or intentions, that is the goals, principles, requirements, and constraints, and the sources of these intentions, namely stakeholders, drivers, and assessments.

Besides providing a framework and a modelling language, and in line with the recommended practice on architecture descriptions described in ISO 42010, ArchiMate also provides a set of viewpoints that can be used to accommodate different concerns. The viewpoints act as filters on the model and are used to specify different views upon the model, highlight different aspects that matter to different stakeholders. Some viewpoints display intra-layer concepts and dependencies, while others display cross layer concepts and relationships.

In general, it can be noted that ArchiMate, coming from the domain of Enterprise Architecture, is specifically powerful and well defined in regards of the business layer, and also the application layer. The technology layer is, in contrast, where most of the extensions presented below will refine the concepts.

4.1 OWL Representation of the Context Model

To allow an efficient, machine-processable representation, the ArchiMate language meta-model was converted to an OWL representation. This allows to apply inference on the models, and an easy integration with other (existing) models expressed as ontologies, e.g. the ones used to extend the core ontology by more specific aspects, and thus enabling reuse and integration of knowledge. ArchiMate itself is grounded in the entity-relation paradigm, providing specialisation of these generic concepts into enterprise architecture concepts. As such, the creation of an ontological representation of the ArchiMate meta-model involves mapping between the concepts and relations of ArchiMate into constructs available in OWL, as shown in Table 1. Concepts were mapped into OWL classes, relations were mapped into OWL ObjectProperties, and instantiations of the meta-model concepts in a concrete model are via OWL Individuals.

To facilitate reasoning and verification of model correctness, restrictions were added into the object properties: InverseObjectProperties and SuperObjectProperties axioms were added to the OWL ontology, so that derived relationships can be extracted through the use of reasoners. Cardinalities were also added to reinforce the coherence of the ontology and its compliance to the ArchiMate meta-model. For instance, since each concept in the core ArchiMate meta-model is part of exactly one layer and one structure, such coherence was

Table 1 Mapping of Archimate Elements to OWL Elements

Archimate Element	OWL Element
Concept	Class
Relation	ObjectProperty
Concept Instances	Individuals

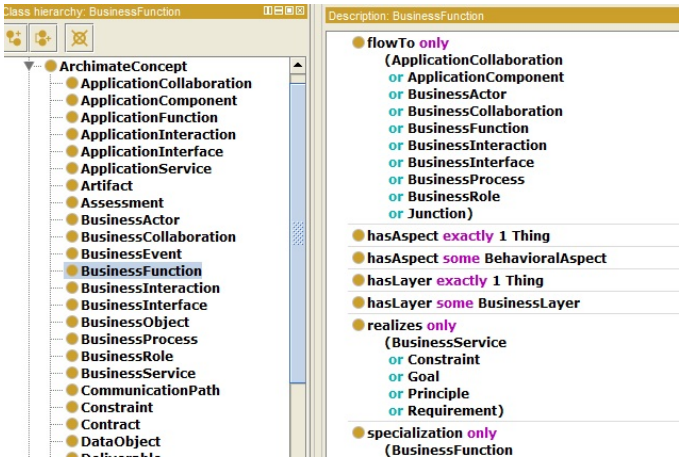


Fig. 6 OWL representation of the ArchiMate concepts and relations, specifically the Business Function Class and respective Object Properties

enforced through cardinality restrictions. Inference (reasoning) will be used, for example, to assess the consistency of models against rules, verify the completeness of models, or produce reports based on the contents of the model.

Figure 4.1 depicts an excerpt the OWL representation of ArchiMate in the Protégé ontology editor[2], with the Business Function class highlighted on the left pane and respective properties, including restrictions on the right pane. The OWL representation of the ArchiMate meta-model, as well as the extensions mentioned below, can be found at `http://timbus.teco.edu/ontologies/`. All ontologies are represented as OWL 2.

5 Extension Ontologies

The core ontology of the Context Model described above provides a basis for describing a process context with coarse concepts. While the business layer is relatively detailed, on the application and technology layer, rather few high-level concepts such as “Device” are provided, and many aspects are not touched at all. Therefore, it is necessary to refine and augment the concepts of the core ontology by models covering specific aspects relevant to the preservation of the process. In this section, we thus present a set of specific extension

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

ontologies, each of which tailored to address explicit modelling concerns, to complement the core ontology.

As with the core ontology, the requirements for these extensions came from the use case scenarios. We will exemplify how these concepts can be used to model specific aspects in the discussion of the use case application in Section 6. Through the requirements elicitation, the following categories of concerns emerged.

- **Legal**. This category includes all legal requirements imposed on the processes and surrounding context. For example, this can be regulation on how long certain parts of a system need to be preserved, and whether certain elements are eligible for preservation.
- **License**. This category includes all aspects related to licenses, and concentrates initially on software licenses. Relevant aspects are e.g. the types of licenses under which software was made available, and the clauses they contain. These license clauses then pose restrictions on what can be performed with the software. Licenses are to a certain point a specialisation of legal requirements.
- **Software Dependencies** This category contains information on dependencies of software to specific other software components, including information on the exact versions, if applicable.
- **Patents**. This category contains aspects on patents, e.g. who is the owner of a specific patent, what the patent covers, or when it was granted. Patents also imply a restriction on how a software, hardware or method can be used. Again, these are to a certain part a specialisation of legal requirements.
- **Data**. This category includes information on data used in the process (created, read or modified, by users or software), such as information on the input data of an application. This also includes metadata on the data, describing e.g. if the data contains personalised information.
- **Data Formats**. This category includes information on which data format objects used in the process adhere too. This type of information is the main concern of traditional digital preservation activities, and thus a tight interlinking to previous results is aimed for.
- **Hardware**. This category includes aspects related to hardware, from desktop systems, computational and storage server infrastructure, to customised devices such as handheld devices employed in the civil engineering use case.
- **Sensors**. A specialisation of hardware, mainly dealing with sensors employed in the civil engineering use case. Sensors may differ in their appearance, from basic systems that need to be read via special instruments, to complex devices that have embedded software for processing.

Wherever possible, the extension ontologies are based on already existing languages, for which then the ontology mapping to the core ontology was provided. However in some cases, new ontologies had to be developed, based on existing vocabularies. It is important to note that the mapping between the extensions and the core ontology is not 1:1. Instead, as described in Section 3.3 and depicted in Figure 3(a), in most cases there is an excess of concepts in

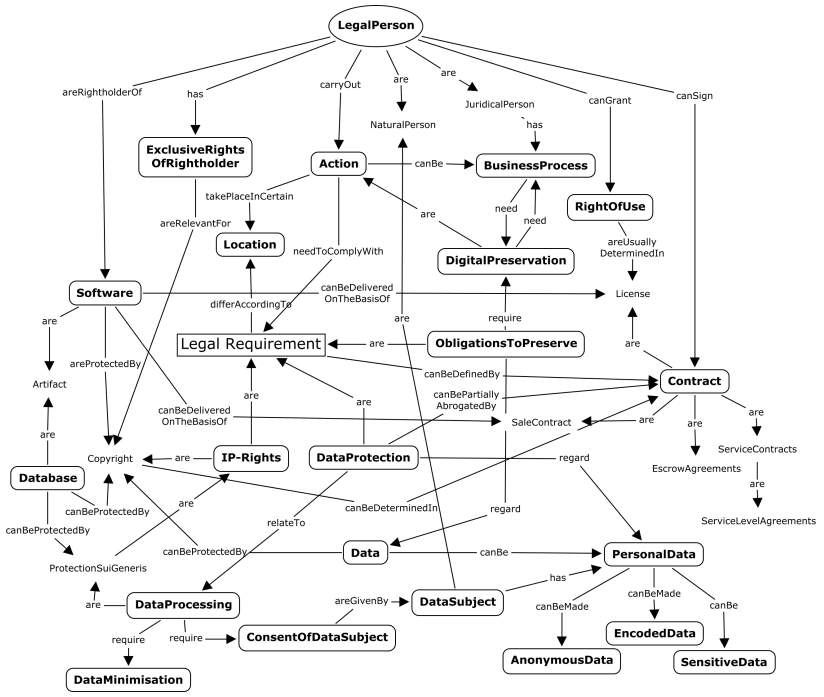


Fig. 7 Conceptual map of the legal aspects ontology

the extension ontology, i.e. some concepts map to none of the concepts in the target model. These concepts are mostly of such nature as that they describe constructs that are mapped to the core ontology in more detail, and thus this excess of concepts is desired.

The set of extension ontologies currently available is not complete for all possible scenarios, but the extension mechanism is easily applicable to provide other domain specific ontologies that might be needed for capturing the context of other use cases. At the moment, the project has integrated the ontologies described below.

5.1 Legal Aspects

In digital preservation activities, the reproduction of the stored data and information is inevitable. This includes reproduction of copyright-protected data and software, as well as intellectual-property materials, e.g. software. It is, therefore, crucial that legal aspects are considered when performing digital preservation of processes, which include software and data alike. Some of the questions identified by use-case the stakeholders regarding legal aspects are *Which database is protected by sui generis protection*, *Is there personal data included in the process*, *Which software is protected by copyright*. As mentioned

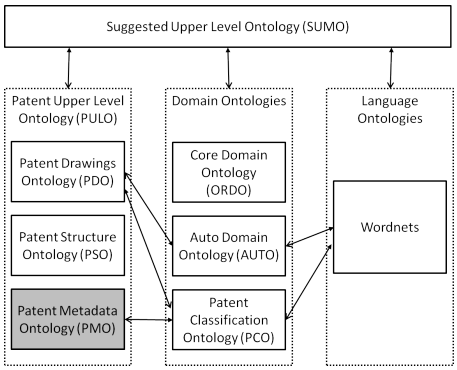


Fig. 8 Structure of the PATExpert ontologies, and their integration with SUMO and other external ontologies

earlier, reasoning capabilities will finally be used to retrieve answers to those questions from the model of a use case.

There is some work regarding ontologies of the legal aspects, e.g. the EU DALOS project (Drafting Legislation with Ontology-Based Support) [39], which created a domain ontology representing the consumer law. Along this example, most legal ontologies, try to model the law, rather than the restrictions and effects it might have on certain actions such as digital preservation. Thus, a dedicated ontology describing these aspects is currently being developed during the TIMBUS project [23]. A graphical overview showing the constructs and relations with other ontologies in the context model is given in Figure 7. It contains aspects such as data protection, intellectual property rights, actors, etc. This is currently being transformed into a formal ontology.

5.2 Patents

This ontology describes information on patents (or more general, intellectual copyright) that are relevant to the preservation of business processes. The question is thus whether specific algorithms, software solutions, or hardware components are affected by patents. If this is the case, it could have implications on whether, or to what level of completeness, the preservation of the processes could be performed. Some of the questions identified regarding patents are *Which patents are required for a certain component*, *What patents are used when executing process*, or *How long is the patent valid for*.

The most suitable candidate we identified for this domain is a result of the EU-funded PATExpert project[3]. PATExpert defined a suite of ontologies that describe patent documents, covering aspects such as the structure of documents and content they provide. It is mapped and integrated against the "Suggested Upper Merged Ontology" (SUMO) [33], in an architecture similar

[3] `http://cordis.europa.eu/ist/kct/patexpert_synopsis.htm`

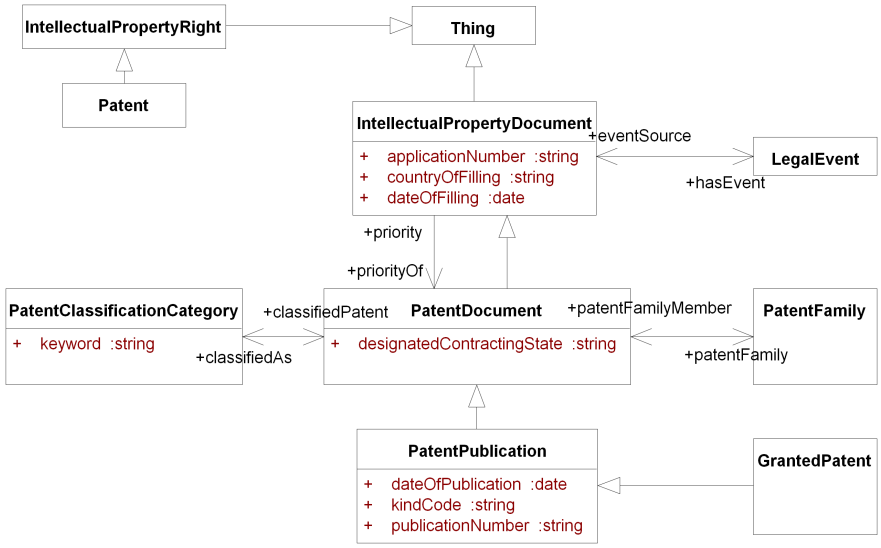


Fig. 9 Detailed view on the PMO ontology

to the one employed for the TIMBUS Context Model. An overview on the modules of the suite is given in Figure 8. Of these, the Patent Metadata Ontology (PMO), shown in Figure 9, contains the concepts of PatentDocument, with the subclasses PatentPublication and GrantedPatent. These are described in detail by IntellectualPropertyDocument, which adds information on publication date and authors. Thus, the PMO is capable of answering the questions identified above. The mapping we opted for is to consider the GrantedPatentDocument to be a specialised version of a Constraint in the core ontology. Mapping this concept is sufficient, as the GrantedPatentDocument and the related IntellectualPropertyDocument contains information on the owner and the publication date of the patent.

5.3 Software Licences

Companies often acquire some of the software components they use to support their processes from third parties, either as so-called commercial off-the-shelf software, or customised software. Software licenses concern the rights and obligations a party has regarding these acquired software applications and components. The license in this case is a specific kind of contract that grants certain rights to the license taker regarding the usage of the software, e.g. as a component his own applications use. It defines for example whether the customer can get access to the source code, modify it, redistribute the software, etc.

Some of the questions identified regarding patents are *Which licenses are open-source*, *What are the licenses required to execute a software application*, or

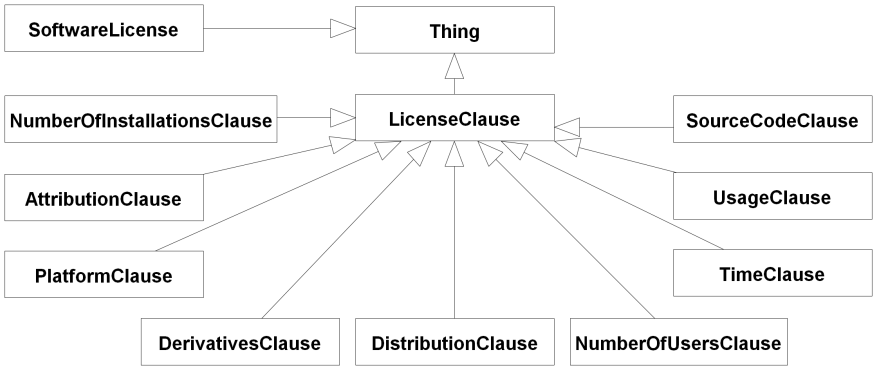
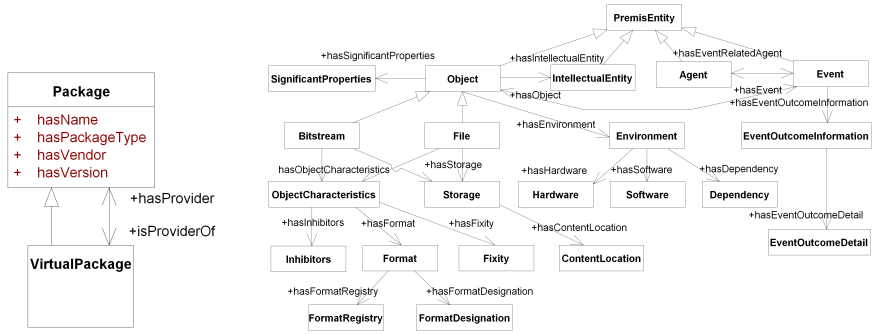


Fig. 10 Software license ontology, part of "The Software Ontology"

What restrictions on preservation actions are allowed according to the license. A suitable candidate for this domain was identified in a part of "The Software Ontology"[4] (SWO), which is an ontology for describing software tools, their types, tasks, versions, provenance and associated data. SWO has originated in a project between the European Bioinformatics Institute and the University of Manchester, and has thus a focus on this domain, with many of its classes tailored to it. The ontology is structured in many different components, concerning e.g., versions, organisations, algorithms, or interfaces. One of these components is dedicated to licenses, and suits the needs of the reasoning questions outlined above. The ontology models two important concepts: Software licenses, and License clauses. License clauses define properties and restrictions on what can be done with the software, e.g. whether redistribution is allowed, and in what form (with or without notice), or whether there is a restriction on the number of users that can use the software. Software licenses are a composition of clauses. Some abstract classes exist, e.g. the abstract class "Open source licenses" defines that the source code is available. Specific licenses are subclasses of a software license. The ontology pre-defines a set of these, but is not complete on commonly used free open source software licenses. The ontology is illustrated in Figure 10.

The ontology mapping is relative straightforward, and allows both a Software license and a License clause to be specified as a subclass of a constraint. This way, one can profit from the pre-defined standard licenses in case such a license is used, but can easily combine a custom license from the clauses, without having to modify the domain-specific ontology.



(a) (b)

Fig. 11 (a) Structure of the software dependencies (CUDF) and (b) a subset of the PREMIS ontologies

5.4 Software Dependencies

Regarding software applications, important questions to answer are *What other components, libraries or applications does a certain piece of software depend on*, and the inverse, *What other components etc. depend on a certain piece of software*. This is important when considering preservation actions on specific parts of the software stack utilised in the process, where also the question *What are alternative software application that are equivalent* might become important.

Technical dependencies on software and operating systems can be captured and described via the Common Upgradeability Description Format (CUDF) [44]. This format was originally developed for operating systems which are based on packages, i.e. where there is a set of package repositories ("universe"), and a package manager application that is responsible for installing new packages, and in turn also all the packages needed by these new packages. This is the case for most Linux operating systems.

The CUDF framework, shown in Figure 11(a), defines two concepts (*package* and *virtual package*), and a number of relations between these – among others *depends*, *recommends*, *conflicts*, and *provides*. Further, a number of data properties are provided. This allows to describe in detail the current software setup on a system, and to analyse the impact of potential changes to elements of the software stack. Although being developed for package-based systems, CUDF can be applied also to other other operating systems. For example, dependencies of Windows software applications to Dynamic Link Libraries (DLLs) can be described using the concepts and relations from CUDF. CUDF is integrated into the Context Model by defining a package to be a specialisation of an *Artifact* and a *System Software*.

[4] `http://theswo.sourceforge.net/`

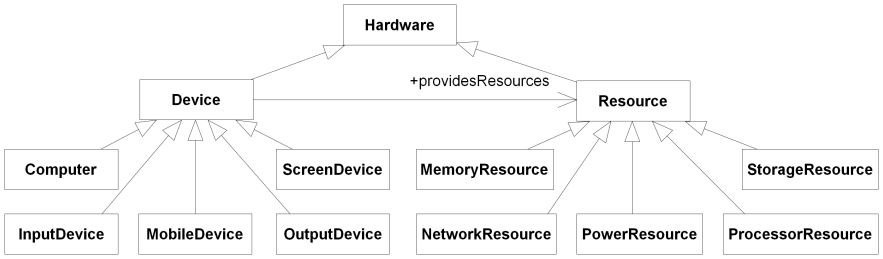


Fig. 12 Structure of the first hierarchy levels of the Hardware ontology

5.5 Preservation Metadata: File Formats and Storage

In a business or scientific process, a number of digital objects are created, modified or read. Information on the format of these objects is crucial for any preservation action to be carried out, as e.g. migration to a different format might require changes in the rest of the process. Besides having more impact on subsequent processing steps, in regards of formats, the scenario of a business process is not much more complex than in traditional digital preservation settings.

File formats are among the main concerns of traditional digital preservation activities, and thus it is easy to identify suitable, existing ontologies. We adopted the PREMIS Data Dictionary [36], which is also available in the form of an ontology. The data dictionary defines five types of entities: Intellectual, Object, Event, Agent, and Rights. It then defines 45 concepts belonging to these types, as well as relations and data properties. A part of this ontology is depicted in Figure 11(b).

To integrate PREMIS in our meta-model, we map the *File* entity to an ArchiMate *Artifact*, and can then utilise the PREMIS elements of *Format* and *FormatRegistry* to further describe them. Also, *Storage*, *ContentLocation* and *Software*, as well as *Agents* are mapped to the core ontology. Further, we are considering mappings of *Rights* to both the core and the legal extension ontologies.

5.6 Hardware

Even though in many processes the hardware employed to host the software applications might be standard commodity hardware, the exact specifications of which can still influence the run-time behaviour of a process that might be critical in certain circumstances, such as execution speed. Further, certain processes might rely on special devices for human-computer interaction, or utilise certain hardware capabilities for computation, such as using graphical processing units (GPUs) for large-scale experiments in scientific processes.

We identified an ontology developed with the aim of describing persuasive and context-aware computing in [37], and found it to be rather complete for

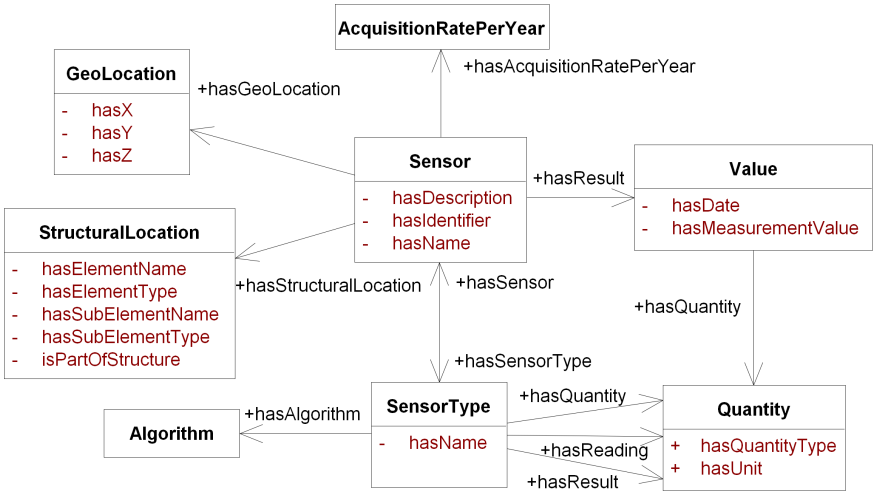


Fig. 13 Classes and object properties of the Sensor Ontology

our purposes. The first hierarchy levels of the ontology structure are given in Figure 12. The main concepts are *Resource*, which describes mostly hardware components, and *Device*, which rather concerns complete computer devices, such as a notebook or a PDA. The latter is also relevant for the civil engineering use case that will be described in Section 6, where some data acquisition is performed via hand-held devices.

The ontology mapping is primarily to map the *Resource* in the hardware ontology to the *Device* concept in the core, and the *Device* in the hardware ontology to the *Node* concept in the core ontology.

5.7 Sensors

Sensors are a very important element in civil engineering structural monitoring and safety, and thus one of our main concerns in that use case. Sensors measure values that can be then processed and analysed, so that the structural behaviour is predicted, and safety measures are taken, if needed. Different types of sensors measure different types of quantities in different ways, and also the interface with the sensor can vary a lot, from manual reading to electronic and network data transmission. Some sensor value readings might need to be post-processed and converted. Some of the questions identified regarding sensors are *Which sensor types can measure the physical quantity Y*, *What is the reading interface to the sensor*, *What are the measurement units for the sensor*, *Which sensor data is post-processed.*

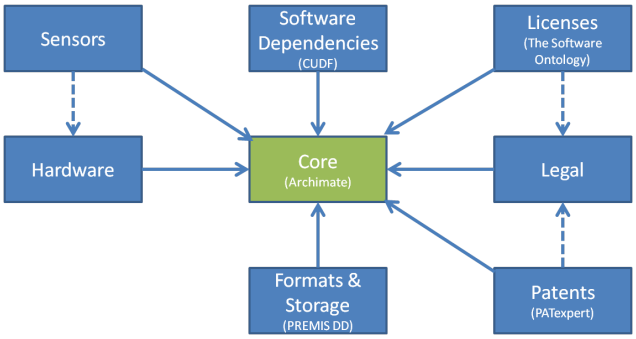


Fig. 14 Overview on available extensions and their relation to the core ontology

Different alternatives available for modelling sensors were analysed, such as SensorML[5] and TransducerML[6]. However, these models and schemas are mostly focused on supporting sensor data (readings), and don't primarily aim at describing the characteristics of the sensors, which is the main concern identified by the stakeholders. Therefore, as a first step towards a well-grounded Sensors extension, we developed an ontology that is expressing the stakeholder requirements, the structure of which is depicted in Figure 13. The mapping to the core ontology is on the concepts of value (*Artifact*), geo and structural location (*Location*), and sensor (*Node*).

As sensors are a special form of hardware, thus an integration with the hardware ontology mentioned above is currently ongoing. We are further working towards integrating this sensor ontology with the Semantic Sensor Net Ontology published by the W3C[7].

An overview on the current extensions available and their mapping to the core ontology and other extension ontologies can be seen in Figure 14. Different alternatives were analysed for creating the mappings between the extensions and the core ontology, including OWL *equivalentTo* statements, which intuitively would be a clear candidate. However, in most cases the extensions define only one possible *specialisation* of a core concept, and thus *subClass* statements were employed on the mappings.

6 Use Case Application

The TIMBUS project has developed a number of use case scenarios, among them an e-Science application, where a machine learning experiment is performed, an e-Health scenario where doctors use an expert system regarding drug usage, and an civil engineering use case, dealing with the process of mon-

[5] `http://www.ogcnetwork.net/SensorML`
[6] `http://www.ogcnetwork.net/infomodels/tml`
[7] `http://www.w3.org/2005/Incubator/ssn/wiki/Semantic_Sensor_Net_Ontology`

itoring a large structure such as a dam. These are presented and discussed in detail below.

6.1 e-Science Experiment – Machine Learning

We will first present the e-Science experiment. The case itself has some particularities that can be used to show the potential of the approach taken to the context model. The process deals with conducting an experiment in the domain of machine learning. Specifically, it tests the usefulness of a method for automatically classifying items in a music collection into a set of predefined categories corresponding to music genres. The automatic classification is performed by learning a machine learning model from music data that has been labelled manually by experts into genres. To this end, the music data is first represented in a numerical form, by extracting representative features from the sound signal. The test of the model is on the classification accuracy, which is the count of how many genres the algorithm can detect correctly. The experiment is performed by a researcher which aims to collect performance metrics for classification and make comparisons to the state of the art. The motivation for performing the preservation of such a process is related to any possible challenges to the results that can be made by members of the research community, e.g. that the results were not correctly obtained, that parameter settings for the machine learning were not chosen soundly, etc. Thus, by preserving such process, the provenance and authenticity of the results can be proven, and the process can easily be repeated on different data, or with altering the parameters, at a later stage, as well.

In detail, the process exists of the following steps. First, the music data and the ground truth ("gold standard") labels of the genre assignment are acquired from external providers, e.g. a content provider such as the Free Music Archive[8], and music websites such as MusicBrainz[9].

Then features are extracted from the music files, using an external service, for example the feature extraction service provided by The Echonest[10]. Next, the features and the genre assignments are combined into a file, following e.g. the WEKA Attribute-Relation File Format (ARFF) [50]. This feature file, and a set of parameters form the basis for learning a machine learning model, using a third-party library such as WEKA [50]. Finally, the performance of that model is evaluated as described above. This process is described in much more detail in [29] and [30].The process is implemented and executed using the Taverna Workflow engine[11].

The music classification process was in the first step modelled using the core ontology concepts of the meta model. Figure 15 depicts a graphical repre-

[8] `http://freemusicarchive.org/`

[9] `http://Musicbrainz.org`

[10] `http://the.echonest.com`

[11] `www.taverna.org.uk`

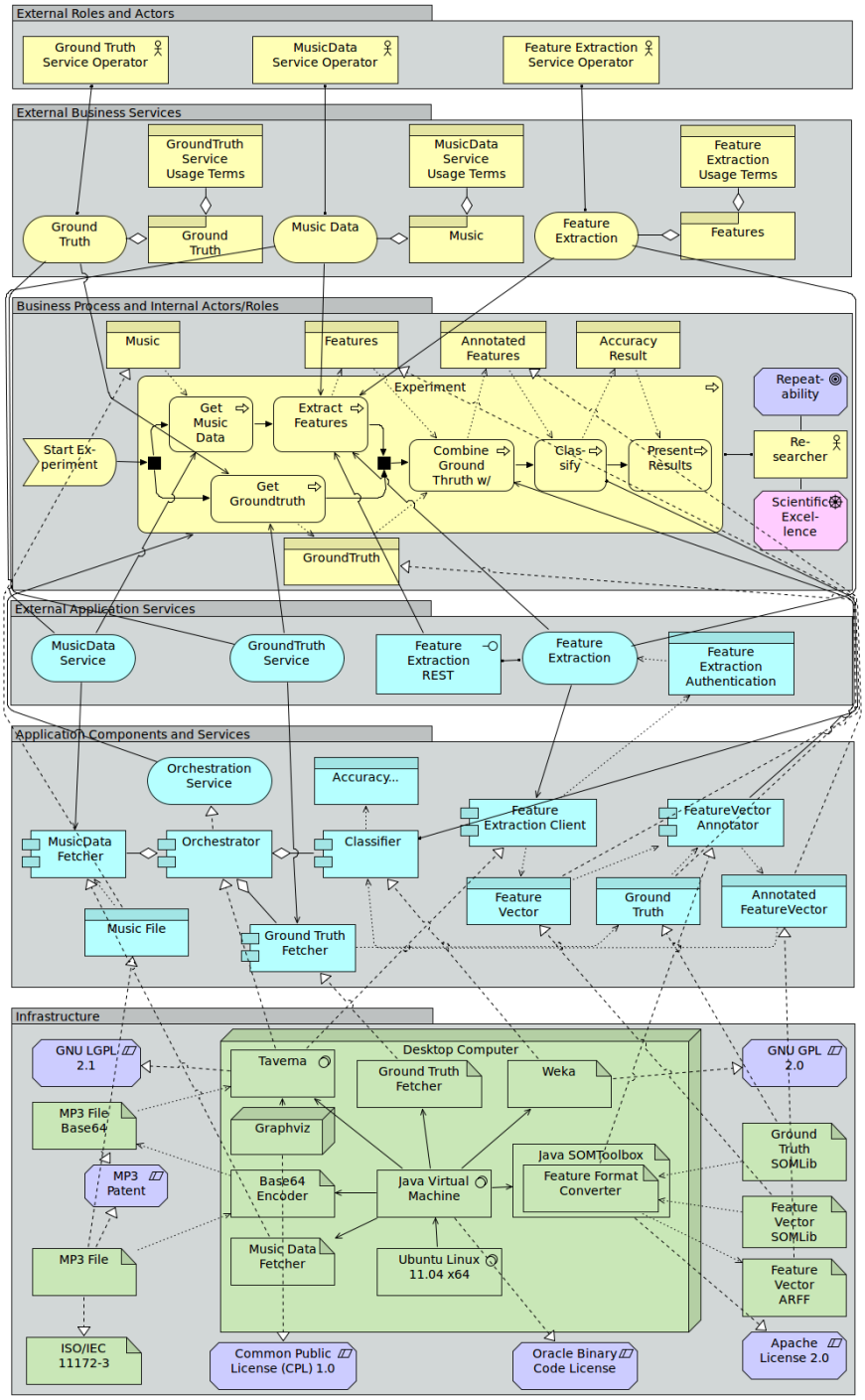


Fig. 15 e-Science experiment, core ontology concepts (excerpt)

sentation of this model in an ArchiMate editor[12]. It has to be noted that not all parts are shown in all detail, especially on the technological layer, many details on the software setup are omitted in the graphical representation, for clarity reasons. The business layer on the top represents the process execution and flow, including the services and people involved, and the objects (data) exchanged between the steps. As the main actor, the researcher is executing the process. Other actors include the operators of the external services. On the technology layer, we can observe the main software components identified – the machine learning toolkit WEKA, various helpers to fetch the ground truth and the music data, the format converter, and the Taverna workflow engine. All of these depend on the Java Virtual Machine, and were executed in an Ubuntu Linux platform; further software dependencies have been omitted for this graphical representation. We can also observe the artifacts that are exchanged between the process steps.

Using just the core ontology, the context model depicts the process dynamics and the dependencies between each step of the process, and the associated applications and technology supporting it. It captures the service agreements which are associated with services and product offerings, as well as the software licences, which are mainly associated with elements at the level of the technology layer, and patents, which in this case is associated with the MP3 format of the music files and is also depicted at the level of the technology layer. These are, in the core ontology, depicted as constraints. This is only a coarse representation of the context, and to be able to infer more on the implications of certain software licenses, or to describe the file formats utilised in the use case in more detail, we need to apply the concepts provided by the extension ontologies.

These specific aspects are then created involving the usage of different extension ontologies integrated with the core ontology in the second step of modelling. Figure 16 shows a segment of the context model instance, specifically elements on the technology layer, similar to Figure 15. It illustrates the usage of the Patent extension, the Software Licenses extension, and the PREMIS extension. Elements that are described in more detail using extension ontology concepts are marked in colours[13]. These might be newly added elements, such as "MP3Format", or elements already described by core ontology elements, but refined by extension ontology aspects, such as "FeatureVectorARFF", which is an *Artifact*, and a *File* in the PREMIS domain.

In particular, the MP3 files are described in more detail on the one hand by using the PREMIS ontology to designate the *Artifact* as a *File*, and then describe it in more detail with a file format, which is linked to a file format registry. Specifically, the link is to the PRONOM registry, which is a centrally maintained database of around 1,100 file formats. As mentioned earlier, this

[12] `http://www.archimatetool.com/`

[13] The visualisation was created with a custom Protégé plugin, developed by the TIMBUS project, available at `http://opensourceprojects.eu/p/timbus/context-model/ontology-visualisation`

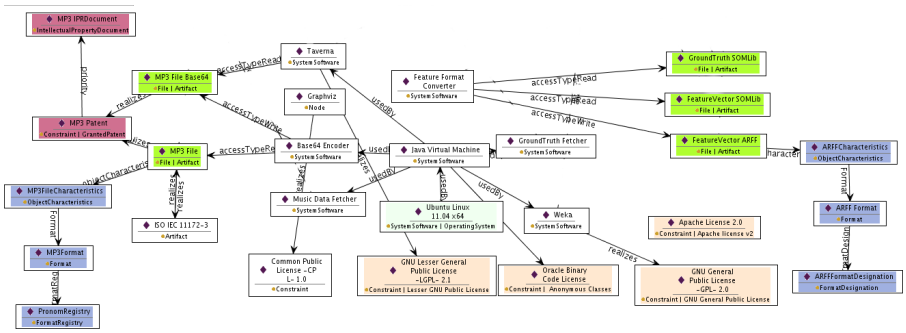


Fig. 16 e-Science experiment, core and extension ontology concepts

information can then be used by the Preservation Alternative Identification module [31], which can propose file format migrations as preservation actions.

Moreover, the MP3 files are put into context to patents involving the MP3 algorithm for data compression in music (the codec). Also, the software license constraints are specified using the available concepts from the Software License ontology, which, as mentioned above, is part of "The Software Ontology". This might have a constraining influence on what preservation actions can be performed on the software, e.g. if the software can not be redistributed. Finally, there is a specific file format, "ARFF", that is not available in a file format registry at the moment. The format is therefore further described in the context model via the *FormatDesignation* concept from PREMIS, e.g. by a specification document. This implies that this documentation document should also be part of the archival submission package that is prepared for the process, as otherwise the format might not be well understood in the future.

Reasoning on the context model instance of a specific process, using e.g. DL queries, helps the preservation expert to infer information that is relevant for the risk analysis and preservation of the process. For example, if a certain software component is at risk and needs to be modified or replaced, the impact of this on other software components, and the overall the execution of the process is of importance. In discovering this, identifying which process activities are actually depending on a certain software is an important aspect, and can be answered by a DL query as seen in Figure 17. When there are several business processes that are using partly overlapping infrastructure, then the questions which process is depending on a certain element is even more interesting. Figure 18 shows the listing of the technology entities supporting the *Experiment*.

6.2 Civil Engineering – Structural Safety Monitoring

The civil engineering use case deals with sensor data acquisition, by an organisation mandated by law to monitor the structural behaviour of large civil engineering structures. In particular, this use case deals with dams, where the

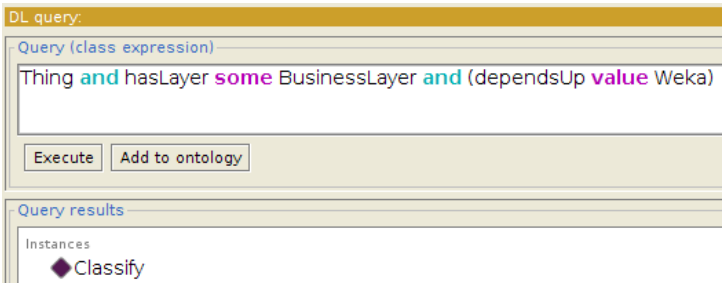


Fig. 17 "Which business layer elements are depending on the *WEKA* System Software?" DL Query results

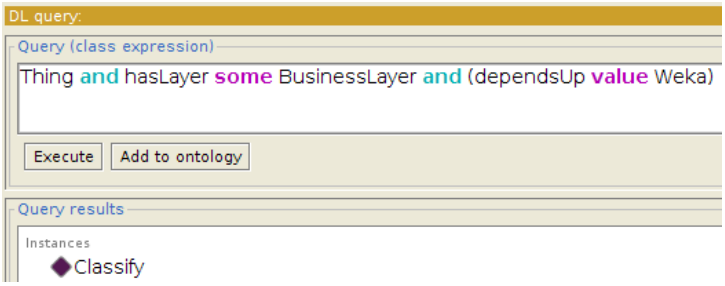


Fig. 18 "What are the technological entities supporting business process Experiment?" DL Query results

organisation in a first step is gathering observation data on the structure, such as pressure, temperature, displacement, and water levels. Subsequently, there are a number of data analysis and model building steps performed, in order to detect and predict any problems with the structural safety of the dam, and thus to prevent accidents. Process preservation is an important aspect for this organisation especially in case of failures, when they need to investigate why their monitoring activity failed. Reasons for not being able to predict the failure can have various causes, such as a incorrect data gathering, faulty processing and transforming of the data, or wrong methods when analysing the data. In addition, preserving the data analysis processes is interesting for later re-examination e.g. with different data or different methods.

The process to which the context model was applied deals with the acquisition of sensor data, which is required for analysing the behaviour of a dam and its structural safety, from different sensors installed along the structure. The data acquisition is either manual or automatic, depending on the type of sensors available. This process is illustrated in Figure 19. Once the data is acquired by the sensors, it is uploaded to an information system *GestBarragens*, developed in-house by the organisation responsible for the monitoring. The upload can by directly from the sensors, using web services, or through portable devices which are operated by employees in the dam. Once that data enters the system, it needs to be validated and then transformed from raw to

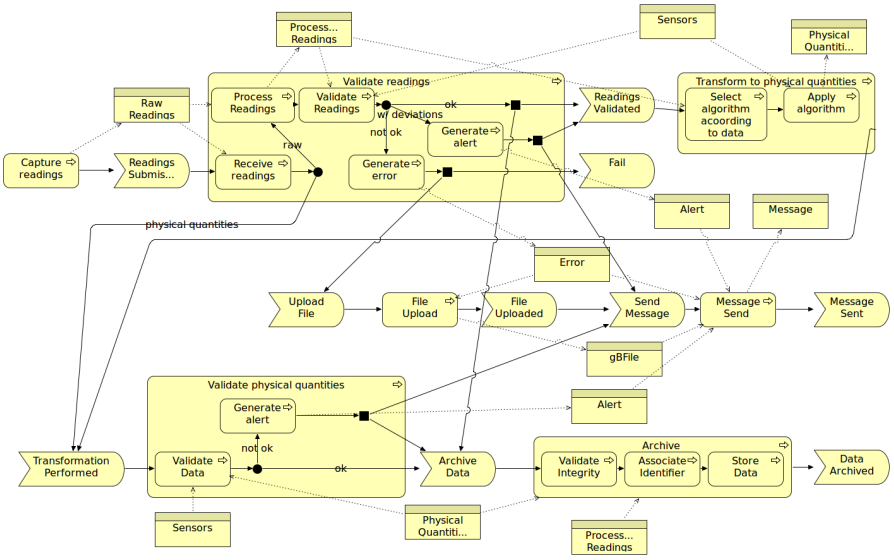


Fig. 19 Civil engineering use case: sensor data acquisition process

engineering quantities that can be analysed by the civil engineers. After being transformed, the data is again validated and archived. More details on this process and on the motivations surrounding it can be consulted in [2].

A section of the model using the core ontology, namely the technology layer aspects, is presented in Figure 20. Again it should be noted that not all detail is given in this graphical representation. The model reveals that two server systems are in use, with different tasks (database vs. application and data interface) and a very different software stack, with a different operating system (Linux and Windows, respectively). The application server hosts the GestBarragens applications, running on the Microsoft Internet Information Services (IIS) application server and .NET Framework; it communicates with the database server via the Oracle Client and the Net8 protocol. Also, a number of client applications are providing data. This data comes from sensors, and the interface used to ingest it into the system, and the format and processing state it is in varies greatly. On the one hand, a user-initiated upload via a Web Client is possible. On the other hand, there are interfaces that allow direct uploading from the data producer. This can be an automatic sensor that communicates via SOAP/HTTP with GestBarragens. Other data producers collect the data manually via a Portable data terminal (PDT). In this case, there is a communication via TCP/IP, supported by the *MC Gateway* application.

The civil engineering use case also employs the previously mentioned extension ontologies, such as hardware, software dependencies, licenses or file formats (PREMIS). In addition, it requires the sensor extension to describe these important aspects of the process in more detail. Figure 21 shows a sample

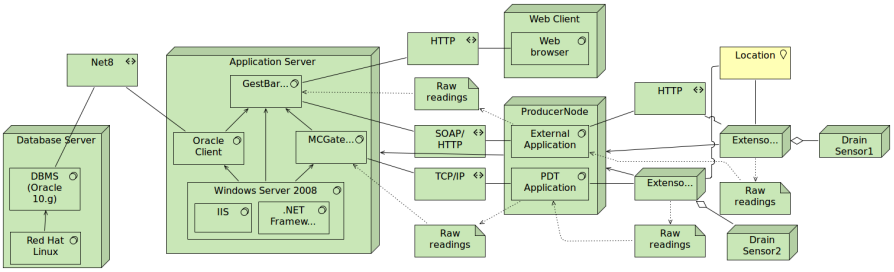


Fig. 20 Civil engineering use case, core ontology concepts of the technology layer

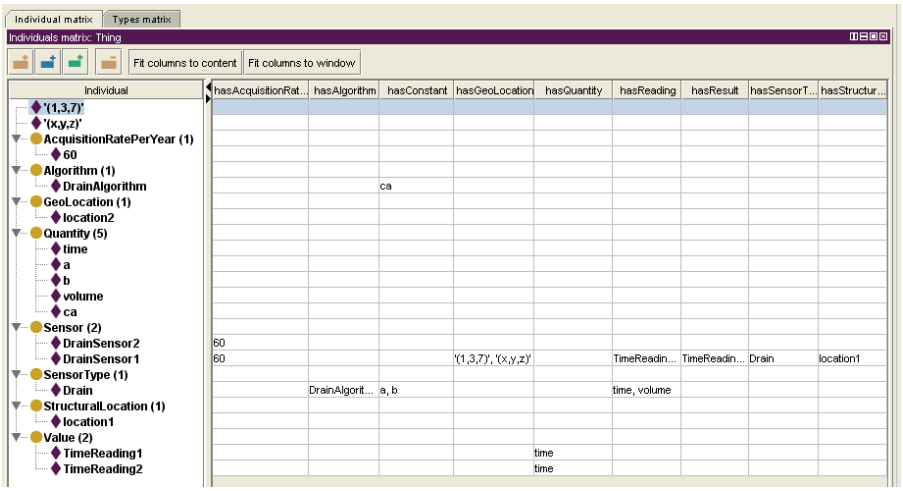


Fig. 21 Civil engineering use case, sensor ontology concepts in Protégé

instance, depicting sensor types, format of readings, the quantities employed (time, pressure, ...), the algorithms used, the frequency of the acquisition, the location in the structure, among others. Reasoning queries specific to this DSO can help to identify for example which sensors are depending on a certain transformation algorithm, to analyse impacts of a change in that algorithm. Also the software setup on a specific Portable Data Terminal is of interest, as this can influence the way data is processed.

The applicability of the context model, with both the core and extension ontologies, for Enterprise Architecture driven applications, has been discussed in [5], [1], [3], where we also provide an analysis of the suitability of certain aspects of the context model for the use cases developed in TIMBUS. The interested reader is thus referred to those publications for other use case examples. Also the aspects of reasoning on the context model, using e.g. OWL description logics, are elaborated in detail there. Section 2.2 detailed how the context model is used in the process preservation framework.

7 Automatic Capturing of Context

Manually populating a model of the size as in the use cases presented in Section 6 is a tedious, and potentially error-prone task. An important aspect for the practical feasibility of the proposed model is thus the support for automatic capturing of the process context, which can also help in achieving a more complete model. While it has to be noted that it is likely that some sections of the model will remain to be filled in manually, especially when it comes to elements on the business layer, in particular aspects on the technical level are well suited for being automatically captured with software tools.

Currently, the following automatic context extraction tools are available:

- Software packages and dependencies, for operating systems that are based on the APT package manager (Debian and derivatives such as Ubuntu) and RPM (Red Hat Linux and derivatives such as Scientific Linux)
- Software dependencies of an application towards Windows .NET dynamic link libraries (DLLs)
- Software dependencies towards other components, e.g. Java libraries (JAR) for Maven based projects, and Perl modules (.pm)
- Licenses associated with a software package, mainly for open-source licenses, based on
- Hardware in Desktop and Server environments, using operating-system specific hardware monitoring tools
- Discovery of external dependencies, via monitoring of system processes and network connections
- Process mining based on process logs, to establish a model of the process being executed

All these software tools provide a detailed view on a specific aspect of the context model. They are then normally manually verified and adjusted by experts. Thus, this semi-automatic process of deriving a process context model for a specific use case is particularly promising.

8 Conclusions and Future Work

Process preservation deals with capturing, archiving and later analysing and redeploying a process, which may be a business process, or scientific process, or some other IT-supported process. Process preservation has a different scope from that of traditional digital preservation methods. The digital content to be stored encompasses many more aspects and is more complex in its nature. Instead of single (or compound) static files, the digital preservation experts have to deal with a potentially large network of interrelated aspects relevant for understanding the process. The type of information required to be preserved for a process can vary a lot, depending on the domain of the process.

In this paper, we thus presented an extensible meta-model for capturing the context of a process, to enable the documentation, preservation and later

analysis and redeployment of the process in a new environment. The meta-model is based on ontologies, and allows for a structured description of these aspects. It is based on a modular and extensible architecture, which allows for customisation to specific domains. When the current meta-model is not capable of representing some domain knowledge, ontologies (existing or created for that purposed) can easily be mapped to the core ontology.

The context model provides support in the various stages of the framework for process preservation. It forms the basis for the risk analysis, and is the major component for the preservation planning stage, where digital assets at risk, and the impact of them failing on the process execution, are identified. Based on the information formally described in the context model, and strategies to alleviate their impact by preservation actions can be automatically proposed and assessed, thus greatly reducing the human effort and expert knowledge required in this phase. The context model serves as an important meta-data element to describe all the digital objects employed in the process, and thus is an integral part of the archived process. During redeployment of the process, the context model is employed to recreate the process in an environment compatible with the original one. It further is utilised in the verification step, where we assess whether the process is still performing to the defined expectations. The context model thus allows the domain expert to formalise knowledge on the business process. This information can then be utilised by archivist, digital preservation experts and semi-automatic tools to provide preservation services.

Future work will focus on refining the currently existing extension ontologies, and providing ontologies for the concerns already identified but not yet addressed until this point. We are also extending the number of application use cases for the context model. As such, we are working on a use case of a recommender system in the e-Health domain, and on the preservation of Digital Library systems, specifically customised installations of a Fedora-Commons and a DSpace system. We are also working with modelling data-intensive research processes in other domains.

In parallel, work focuses on developing and improving the software modules that are automatically populating some parts of the context model. Finally, the context model will be employed in the preservation and redeployment phases of the TIMBUS model for process preservation. This will happen in the upcoming year, when the software developed for this task is provided.

9 Acknowledgements

Part of this work was supported by the project TIMBUS, co-funded by the EU under the FP7 contract 269940, by national funds through FCT (Fundaçãao para a Ciência e a Tecnologia), under project PEstOE/EEI/LA0021/2013, the grant (SFRH/BD/69121/2010) to Gonçalo Antunes, and COMET K1, FFG - Austrian Research Promotion Agency.

References

1. Antunes, G., Bakhshandeh, M., Mayer, R., Borbinha, J., Caetano, A.: Using ontologies for enterprise architecture analysis. In: Proceedings of the 8th Trends in Enterprise Architecture Research Workshop (TEAR 2013), in conjunction with the 17th IEEE International EDOC Conference (EDOC 2013). Vancouver, British Columbia, Canada (2013)
2. Antunes, G., Barateiro, J., Proença, D., Silva, A., Caetano, A., Borbinha, J., Vieira, R., Freitas, R., Hecheltjen, M., Kolany, B., Yankova, S.: Deliverable Use Case Definition and Requirements - Civil Enginieering Infrastructures. Tech. rep. (2012). `http://timbusproject.net/resources/publications/public-project-deliverables`
3. Antunes, G., Caetano, A., Bakhshandeh, M., Mayer, R., Borbinha, J.: Using ontologies for enterprise architecture model alignment. In: Proceedings of the 4th Workshop on Business and IT Alignment (BITA 2013). Poznan, Poland (2013)
4. Antunes, G., Caetano, A., Bakhshandeh, M., Mayer, R., Miri, H., Galushka, M., Draws, D., Coutinho, C.: Deliverable Dependency Models Iter. 2. Tech. rep. (2013). `http://timbusproject.net/resources/publications/public-project-deliverables`
5. Bakhshandeh, M., Antunes, G., Mayer, R., Borbinha, J., Caetano, A.: On the use of ontologies to specify and integrate enterprise architecture models. In: Proceedings of the 8th International Workshop on Vocabularies, Ontologies and Rules for the Enterprise and Beyond (VORTE 2013), in conjunction with the 17th IEEE International EDOC Conference (EDOC 2013). Vancouver, British Columbia, Canada (2013)
6. Becker, C., Antunes, G., Barateiro, J., Vieira, R.: A capability model for digital preservation - analyzing concerns, drivers, constraints, capabilities and maturities. In: Proceedings of the 8th International Conference on Preservation of Digital Objects (iPRES 2011) (2011)
7. Becker, C., Kulovits, H., Guttenbrunner, M., Strodl, S., Rauber, A., Hofman, H.: Systematic planning for digital preservation: Evaluating potential strategies and building preservation plans. IJDL **10**(4), 133–157 (2009). DOI 10.1007/s00799-009-0057-1
8. Binz, T., Leymann, F., Nowak, A., Schumm, D.: Improving the manageability of enterprise topologies through segmentation, graph transformation, and analysis strategies. In: 2012 16th IEEE International Enterprise Distributed Object Computing Conference (EDOC) (2012)
9. Brocks, H., Kranstedt, A., Jaschke, G., Hemmje, M.: Smart Information and Knowledge Management, chap. Modeling Context for Digital Preservation, pp. 197–226. Springer Berlin/Heidelberg (2010)
10. Buckl, S., Buschle, M., Johnson, P., Matthes, F., Schweda, C.M.: A meta-language for enterprise architecture analysis. In: 16th International Conference on Exploring Modeling Methods for Systems Analysis and Design (EMMSAD 2011), London, United Kingdom (2011)
11. Bunge, M.: Treatise on Basic Philosophy: Volume 3: Ontology I: The Furniture of the World. Reidel (1977)
12. Buschle, M., Ullberg, J., Franke, U., Lagerström, R., Sommestad, T.: A tool for enterprise architecture analysis using the prm formalism. In: Information Systems Evolution, *Lecture Notes in Business Information Processing*, vol. 72, pp. 108–121. Springer Berlin Heidelberg (2011). DOI 10.1007/978-3-642-17722-4_8
13. Conway, E., Matthews, B., Giaretta, D., Lambert, S., Wilson, M.: Managing risks in the preservation of research data with preservation network. The International Journal of Digital Curation **7**, 3–15 (2012)
14. Fischer, R., Aier, S., , Winter, R.: A federated approach to enterprise architecture model maintenance. Enterprise Modeling and Information Systems Architectures **2**, 14–22 (2007)
15. Granger, S.: Emulation as a digital preservation strategy. D-Lib Magazine **Vol. 6 (10)** (2000). `http://www.dlib.org/dlib/october00/granger/10granger.html`
16. Greefhorst, D., Proper, E.: Architecture Principles: The Cornerstones of Enterprise Architecture. The Enterprise Engineering Series. Springer (2011)
17. Guizzardi, G.: Ontological foundations for structural conceptual models. CTIT, Centre for Telematics and Information Technology (2005)

18. Higgins, S.: The DCC curation lifecycle model. The International Journal of Digital Curation **3**, 134–140 (2008)
19. ISO: ISO 31000: 2009 Risk management – Principles and Guidelines
20. ISO: Space data and information transfer systems – Open archival information system – Reference model (ISO 14721:2003) (2003)
21. ISO, IEC and IEEE: ISO/IEC/IEEE 42010:2011 – Systems and Software Engineering – Architecture Description. Tech. rep. (2011)
22. Johnson, P., Ekstedt, M.: Enterprise Architecture: Models and Analyses for Information Systems Decision Making. Lightning Source Incorporated (2007)
23. Kolany-Raiser, B., Yankova, S.A., Bakhshandeh, M., Miri, H., Galushka, M., Caetano, A., Borbinha, J.: Towards a legal ontology for the digital preservation domain. In: Proceedings of the International Conference on Information and Communication Technology, Law, Protection, and Access Rights (2013)
24. Kramler, G., Kappel, G., Reiter, T., Kapsammer, E., Retschitzegger, W., Schwinger, W.: Towards a semantic infrastructure supporting model-based tool integration. In: International Workshop on Global Integrated Model Management (GaMMa '06) (2006)
25. Lankhorst, M.: Enterprise architecture at work: modelling, communication, and analysis. Springer (2005)
26. Lankhorst, M.: Enterprise architecture modeling - the issue of integration. Advanced Engineering Informations **18**, 205–216 (2006)
27. Marcum, D.B.: The preservation of digital information. The Journal of Academic Librarianship **22**(6), 451 – 454 (1996). DOI 10.1016/S0099-1333(96)90006-3
28. Martin, R.A., Robertson, E.L., Springer, J.A.: Architectural principles for enterprise frameworks: Guidance for interoperability. In: International Conference on Enterprise Integration Modelling and Technology 2004 (ICEIMT 2004), Toronto, Canada (2004)
29. Mayer, R., Rauber, A.: Towards Time-resilient MIR Processes. In: Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR 2012), pp. 337–342. Porto, Portugal (2012)
30. Mayer, R., Rauber, A., Neumann, M.A., Thomson, J., Antunes, G.: Preserving scientific processes from design to publication. In: Proceedings of the 16th International Conference on Theory and Practice of Digital Libraries (TPDL 2012), *Lecture Notes in Computer Science*, vol. 7489, pp. 113–124. Springer, Cyprus (2012). DOI 10.1007/978-3-642-33290-6_13
31. Mayer, R., Strodl, J.B.S., Rauber, A.: Automatic discovery of preservation alternatives supported by community maintained knowledge bases. In: Proceedings of the 11th International Conference on Digital Preservation (iPres 2014). Melbourne, Australia (2014)
32. Miksa, T., Proell, S., Mayer, R., Strodl, S., Vieira, R., Barateiro, J., Rauber, A.: Framework for verification of preserved and redeployed processes. In: Proceedings of the 10th International Conference on Preservation of Digital Objects (IPRES2013). Lisbon, Portugal (2013)
33. Niles, I., Pease, A.: Toward a Standard Upper Ontology. In: C. Welty, B. Smith (eds.) Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001), pp. 2–9 (2001)
34. OMG: Business Process Model and Notation (BPMN), Version 2.0, vol. OMG Standard, formal/2011-01-03, 2011. Object Management Group (2011)
35. Pinto, H.S., Gomez-Perez, A., Martins, J.P.: Some issues on ontology integration. In: Proceedings of IJCAI99's Workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends. Stockholm, Sweden (1999)
36. PREMIS Editorial Committee: Premis data dictionary for preservation metadata. Tech. rep. (2008). http://www.loc.gov/standards/premis/v2/premis-2-0.pdf
37. Preuveneers, D., Van den Bergh, J., Wagelaar, D., Georges, A., Rigole, P., Clerckx, T., Berbers, Y., Coninx, K., Jonckers, V., De Bosschere, K.: Towards an extensible context ontology for ambient intelligence. In: Ambient intelligence, pp. 148–159. Springer (2004)
38. Rosemann, M., Green, P., Indulska, M.: A reference methodology for conducting ontological analyses. In: Conceptual Modeling–ER 2004, *Lecture Notes in Computer Science*, vol. 3288, pp. 110–121. Springer Berlin Heidelberg (2004). DOI 10.1007/978-3-540-30464-7_10

39. Sartor, G., Pompeu, C., Biasiotti, M.A., Fernández-Barrera, M. (eds.): Approaches to legal ontologies: Theories, Domains, Methodologies, vol. 1. Springer (2011)
40. Schekkerman, J.: How to Survive in the Jungle of Enterprise Architecture Frameworks: Creating or Choosing an Enterprise Architecture Framework. Trafford Publishing (2006)
41. Strodl, S., Mayer, R., Antunes, G., Draws, D., Rauber, A.: Digital preservation of a process and its application to e-science experiments. In: Proceedings of the 10th International Conference on Preservation of Digital Objects (IPRES2013), pp. 117–125. Lisbon, Portugal (2013)
42. The Open Group: TOGAF version 9.1. Van Haren Publishing (2011)
43. The Open Group: ArchiMate 2.0 Specification. Van Haren Publishing (2012)
44. Treinen, R., Zacchiroli, S.: Description of the CUDF Format. Tech. rep. (2008). http://arxiv.org/abs/0811.3621
45. U. K. Ministry of Defence: MOD Architecture Framework, Version 1.2.004. U. K. Ministry of Defence (2010). http://www.mod.uk/DefenceInternet/AboutDefence/WhatWeDo/Information Management/MODAF
46. U.S. Department of Defense: DoD Architecture Framework, Version 2.02. U.S. Department of Defense (2010). http://dodcio.defense.gov/sites/dodaf20
47. W3C: OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition). Tech. rep. (2012). W3C Recommendation
48. Webb, C.: Guidelines for the Preservation of Digital Heritage. National Library of Australia (2005)
49. Weber, R.: Ontological Foundations of Information Systems. Coopers & Lybrand (1997)
50. Witten, I., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
51. Zivkovic, S., Kuhn, H., Karagiannis, D.: Facilitate modeling using method integration: An approach uisng mapping and integration. In: 15th European Conference in Information Systems (2007)