D4.9 Refined Business Process Contexts

WP4 – Processes and Methods for Digitally Preserving Business Processes

Delivery Date: 31.03.2014

Dissemination Level: public

TIMBUS is supported by the European Union
under the 7th Framework Programme
for research and technological development and demonstration activities (FP7/2007-2013)
under grant agreement no. 269940

Deliverable Lead		
Name	**Organisation**	**e-mail**
Till Riedel	KIT	riedel@teco.edu

Contributors		
Name	**Organisation**	**e-mail**
Rudolf Mayer	SBA	rmayer@sba-research.org
Marzieh Bakhshandeh	INESC-ID	Marzieh.Bakhshandeh@ist.utl.pt
Barbara Kolany-Raiser	ITM	Barbara.Kolany@uni-muenster.de
Silviya-Aleksandrova Yankova	ITM	Silviya.Yankova@uni-muenster.de
Mykola Galushka	SAP	Mykola.Galushka@sap.com
Hossein Miri	KIT	miri@teco.edu

Internal Reviewer		
Name	**Organisation**	**e-mail**
Carlos Courtinho	CMS	carlos.coutinho@caixamagica.pt
Michael Nolan	Intel	michael.nolan@intel.com

Disclaimer

Table of Contents

List of Figures

List of Tables

List of Acronyms

Dw.n	Deliverable number n of work package w of TIMBUS
DIO	Domain Independent Ontology (in TIMBUS based on Archimate Metamodel)
DSO	Domain Specific Ontology
DP	Digital Preservation
WP	Work Package
ADE	Adverse Drug Event
ADR	Adverse Drug Reaction
BP	Business Process
BPD	Business Process Diagram
BPMN	Business Process Modelling Notation
CAD/CAM	Computer Aided Design/Manufacturing
CMRDP	Central Medical Repository for Drug Prescriptions
CUDF	Common Upgradeability Description Format
DIO	Domain Independent Ontology
DL	Description Logic
DP	Digital Preservation
DSO	Domain Specific Ontology
EA	Enterprise Architecture
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
HW	Hardware
IEEE	Institute of Electrical and Electronics Engineers (technology advancement organization)
IP	Internet Protocol
ISCO	Information System Construction Language
IT	Information Technology
JEE	Java for Enterprise Environments
LLM	Legal Lifecycle Management
MRD	Machine Readable Dictionary
NLP	Natural Language Processing
OWL	Web Ontology Language
PC	Personal Computer
PDT	Portable Data Terminal
PhC	Pharmaceutical Company
PREMIS	Preservation Metadata: Implementation Strategies (working group)
RCAAP	Repositório Científico de Acesso Aberto de Portugal (Science Database)
RDF	Resource Description Framework

RDFa	Annotation for the Resource Description Framework
SensorML	Sensor Markup Language
SOAP	Simple Object Access Protocol (XML protocol)
SPARQL	Simple Protocol and RDF (Resource Description Framework) Query Language
TIMBUS	EU FP7 Project dealing with “Timeless Business and Services”
W3C	World Wide Web Consortium
WAV	Windows Wave (audio format/file extension)
WP	Work Package
XML	ExTensible Markup Language

1 Executive Summary

Based on practical population of the model instances for specific scenarios we evaluate the capability of the re-designed TIMBUS Context Model to supporting a vast realm of context parameters, as well as being applied to these use-cases for further DP actions and reasoning. We further report on best practices and tools that can be used when adapting the context model to new use cases based on the experiences and developments made in the project. Based on this we describe the further actions needed to improve and sustain the TIMBUS Context Model beyond the scope of the project. This document reports on the refinement of the TIMBUS Context Model that captures context of business processes in order to enable risk-assessment and risk-driven preservation actions to be taken.

The document gives an overview about the refinement strategies applied. We applied a set of four different strategies to refining the Context Model:

- Documentation and Formalization of Best Practices and Tools
- Inclusion of new Ontologies
- Iterative Refinement
- Adaption and Mapping of existing Ontologies

For the documentation part, this document provides a good starting point for understanding the practical application of the context model. We especially present design patterns for the context model.

This deliverable reports on the creation process and shows the necessary steps for ontology engineering within the Context Model to include concepts needed by the use cases. Therefore LegalDSO was included as an ontology to the context model. With the validation of the LegalDSO in a real world scenario we demonstrate the applicability of the modelling approach.

Furthermore, the SensorDSO was updated as a showcase for iterative refinement. A report on the updates and the refined version of this DSO is reported in this deliverable. As we started from a very use-case-specific view, this version provides a higher semantic consistency and adds inline documentation to the ontology. The use-case driven approach is important because it helps to focus the work in D4.9 on the requirements and needs of the TIMBUS use cases.

All refinements are use-case-driven and based on evaluations within the use cases. We have included the internal evaluation results in this deliverable. We conducted three types of subjective evaluations. One technical, which gives an overview of the most pressing self-reported issues in the use cases; One risk-based evaluation, which gives a good check on what types of risks currently can be identified using the context model. Lastly, we present a subjective evaluation of the usability of the current context model (based on its state in M32).

As this document concludes the dedicated modelling efforts within Task 4.4 of the TIMBUS project, one of the most important aspects will be the sustaining of the of results. While other tasks are responsible for the dissemination activities of the project. The last sets out a first plan for the future.

2 Introduction

In TIMBUS, the terminology "context of a business process or service" refers to the physical, digital and social environment of an instance of the abstract concept of a business process or service. Similarly, the terminology "context model of a business process or service" refers to any modelling that can be used to document, abstract and characterize the physical and social situation of an instance of the abstract concept of a business process or service. The terms "Model" and "Ontology" are used in an exchangeable manner within this scope. The model or ontology general consists of a conceptual level (a "meta-model") and an instance level (the concrete context captured for a use case). This deliverable discusses the conceptual level of the context model that can be shared between use cases.

The main goals of this task are according to the Description of Work of the TIMBUS project:

- A "context model […] that describes the key parameters of context to be captured and reasoned about". "The context model will provide the main communication format for the context capture tool". Work on usage patterns and practical tool support is documented in this deliverable that address both automatic and manual processing of the context model.(see particularly section 4)
- "Define what needs to be documented with regard to business processes and supporting software/technology stack such that they can be redeployed in the future." This deliverable improves those documentation needs and addresses practical issues by adding best practices, new concepts, refined concepts and further mappings to existing formalizations. (see section 7)
- "Context parameters for the advanced dependency monitor (D6.2) corresponds to that of version numbering in traditional dependency analysis.". Here this work build upon T4.2 results and continues to integrate different via a common upper ontology. Particularly new mappings to existing context parameters in the preservation and business process domain are reported.
- "Capture physical aspects of business processes (sensor location, product movements, external processes such as cooling devices)." Further "the existing legal requirements to preserve business data will also be documented". More work was particularly done to map physical systems (SensorDSO, section 6) and legal systems (LegalDSO, section5) to such a context based dependency model.

The TIMBUS Context Model thus assumes central importance in the preservation of business processes, providing a mean of modelling context and dependencies so that all the information required for preserving and redeploying a process is captured. The first iteration of the TIMBUS Context Model served to provide a common understanding of the required concepts as well as identify areas of application. Thus, as was pointed out in D4.2, the TIMBUS Context Model needed to be refined, restructured and improved radically which was done as part of D4.3. The result of the restructuring efforts was a comprehensive model developed based on best-practices, standards, and industrial case stakeholder's requirements, with an extensible architecture and a governance method for evolving the model and adapting it for whatever preservation scenario. While the initial deliverable of this task (D4.5) provided a formal meta-model that captured those relevant aspects and key parameters in a Context Model, this deliverable addressing "Refined Business Process Contexts" was planned as "a refined and internal version of D4.5". The Context Model reported in D4.9, however, does not

follow the original modelling approach of D4.5 but builds upon the new integrated Context and Dependency model first introduced in the second iteration of D4.3.

This deliverable particularly addresses multiple review recommendation issued by the EC experts. First of all the ontology on legal aspects was developed and applied until M30 (recommendation c3), several refinements have been done by applying the so-called LegalDSO (section 5) to further use cases. In preparation to recommendation c8, an initial qualitative study (section 8) was conducted on the practical use of the context model. This deliverable reports on multiple lessons-learned from an application perspective . One primary topic for this deliverable was consolidation (c11) to allow for a stable basis for the tools that are being developed. Multiple aspects of consolidation are highlighted as “refinements” (see section 3)within this deliverable.

Particularly addressing WP4 review comments, we expanded our work towards more complex scenarios (e.g. importing more than 31 000 sensors as Nodes into the SensorDSO via relational database mappings and considering more that 30,000 packages for the SoftwareDSO via software extractors as reported in D6.5). During the reporting periods we further applied the refined context model to all use cases. In order to measure the effects and appropriateness of the current approach we conducted a formative study, that allowed us to optimize the context model with the remaining resources. While the approach generally seems to scale well, the study documents that further tool support and integration is needed.

This deliverable was concretely written as a direct result of incorporating all partner-specific use-case scenarios for digitally preserving business processes into a refinement process. This work was kicked off after all use cases implemented, i.e. populated, initial models of the relevant context of their business processes and services with the new Context Model based on the TIMBUS DIO as upper ontology based on Archimate.

The current extensible TIMBUS Context Model, as pictured in Figure 1, supports reasoning and inference, which can be used for checking inconsistencies on the model and inferring information that might be particularly useful for the TIMBUS preservation processes. The developed Context Model was applied to the industrial use-cases of WP7, WP8 and WP9. As mentioned before the work reported in this deliverable is a direct result of this application and build upon the lessons learned and the needs of the use cases.

As this document concludes the dedicated modelling efforts within the Task 4.4 of the TIMBUS project, one of the most important aspects will the sustaining of the of results. After a phase of repeated application this deliverable can be seen as a step towards sustaining the continuous development towards “defining” (in analogy to Capability Maturity Model [3] level 3) the development and maintenance process. While tasks on sustaining beyond the project end will be ongoing as part of the dissemination activities of the project section 9 documents a first plan for the future immediate future particularly for the use case pilots. As explained there more steps are needed to evolve towards a “managed” process.

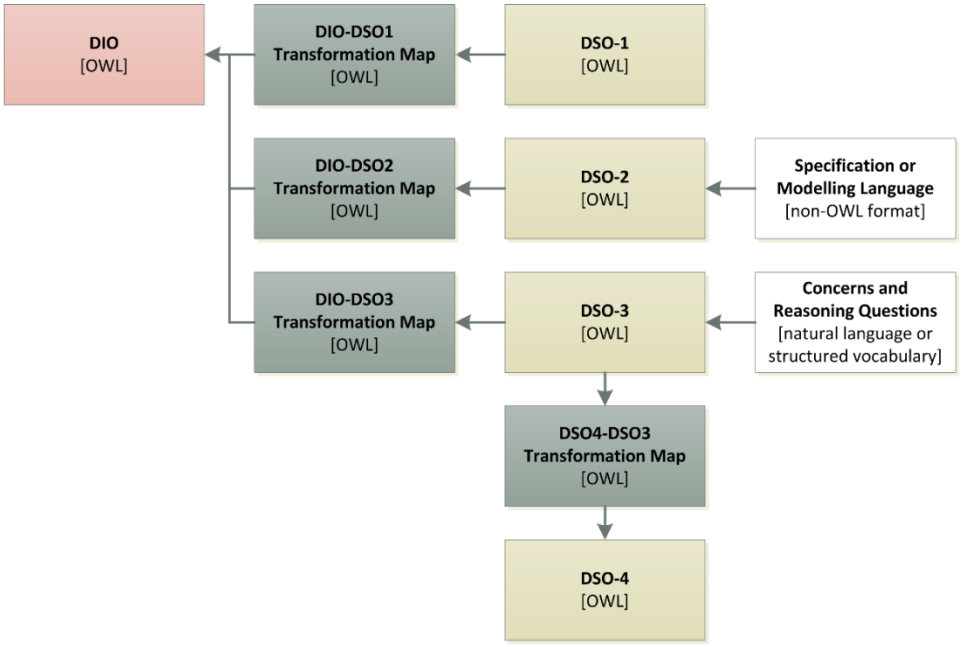

Figure 1: Relationship between DIO, DSOs, and transformation maps in the Context Model

The following section gives a documentation of the maintenance, i.e. refinement, strategies applied. Sections 4-6 provide examples and best-practices for concrete refinements of the context model in an attempt to document the development process. The first section provides a good starting point for understanding the practical application of the context model. The next section describes how to extend the context model. Section 6 show-cases changes made on findings based on the WP8 sensor use case and section 7 shows how to hook up the context model to other modelling efforts. Section 8, which is presenting peer and user review, was actually done in parallel to the work described in Sections 4 and 6 and describes the state of the context model. It provides a basis for judging the maturity of the current models provides formative insights for future refinements.

This deliverable is related to other deliverables as follows:

Although this document seeks to provide a overview of the Context Model from a user perspective more details on the theoretic background and structure of the Context Model can be found especially found in D4.3 which is publically available online [4]. It contains an in-depth description of the TIMBUS Context Model Architecture and of the Domain Independent Ontology (DIO) as well as many Domain Specific Ontologies (DSOs) and their mappings.

Of further interest for the application of the context model are also the deliverables of WP6 that describe the use of the Context Model within tools. Further the particularly following this deliverable further work on the Context Model was done within the use cases WP7-9.

3 General Refinement Strategies

Digital preservation (DP) of entire business processes and services has specific and challenging requirements. In businesses and services, very often processes are only partially or even not-at-all formalized, and change rapidly due to highly volatile business environments [14]. From this follows an increased need for autonomy: digital preservation of whole business processes and services has to happen automatically or at least semi-automatically to cope with the volatility and change pace. Preservation actions and target are not formalized in closed way but subject to digital preservation by an agreement between the stakeholders of a business process.

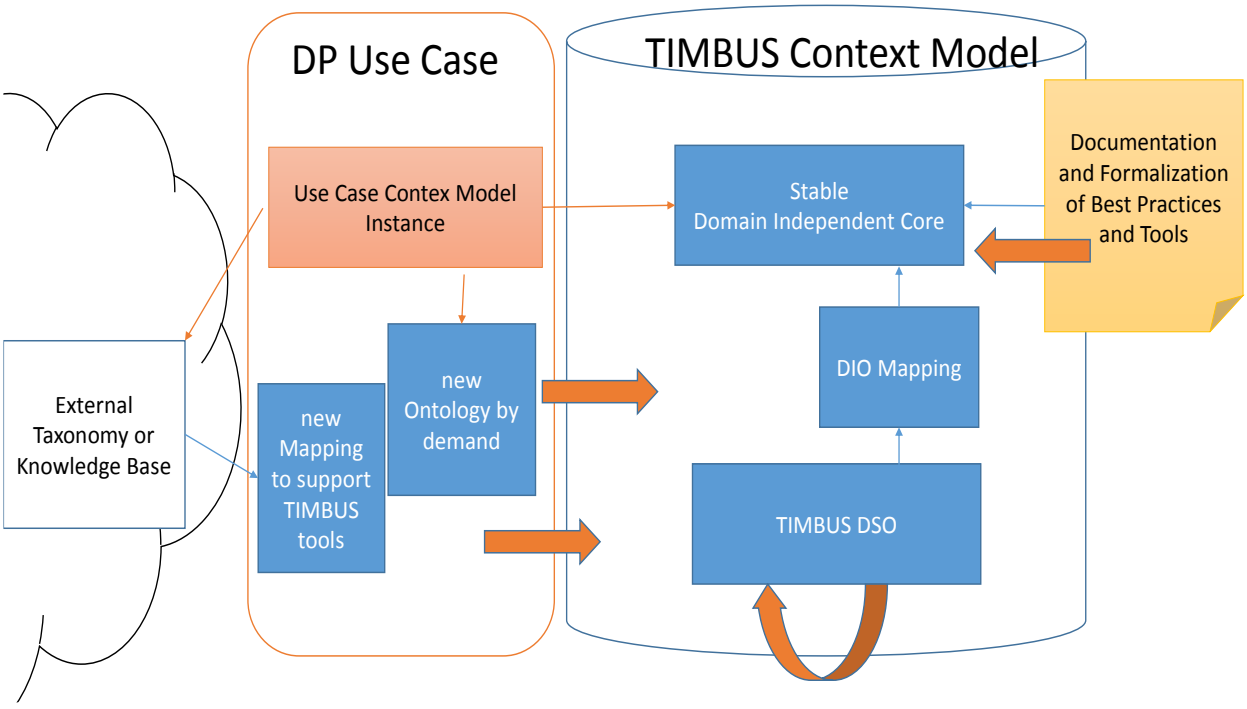

Figure 2: Context Models refinement process overview

As changes in scope of the digital preservation based on the evolving business scope and technological developments may become necessary, a basic feature of the context model is that in can be refined to the changing context of the business process. When considering this change, it becomes clear that also the list of concepts needed to express context information on the necessary granularity will undergo changes and challenges. As detailed in section 9, it is necessary for a sustainable development and to enable broad tool support to share those changes between use cases.

This iteration describes exactly the necessary feedback process. 4 different strategies are applied in this scope to refine the Context Model which are pictured in Figure 2:

- Documentation and Formalization of Best Practices and Tools

- Inclusion of new Ontologies
- Iterative Refinement
- Adaption and Mapping of existing Ontologies

In this document we describe each of those strategies in detail based on concrete developments and experiences made in the TIMBUS project. This document does not strive to capture all developments made on the context model as many of the developments are done in the scope of the use cases and the concrete tools development. We deliberately decided to use an open model based on the principles of description logic (DL), linked data and the semantic web. This also will allow external contributors to extend the current model based on their specific needs.

3.1 Formalization and Documentation using Best Practices and Tools

Building the TIMBUS core Domain Independent Ontology (DIO) on the Archimate standard [5] provides many opportunities by providing a low entrance barrier to DP from the architecture community, and allows getting agreement to concepts in an abstract level. The use cases go beyond modelling architectures towards analysing contextual risks and link resources for concrete preservation and redeployment actions, along with risk and preservation-relevant-context, internal and external to both the use case and the controllable domain of an entity. Furthermore, it is often unknown which aspects and detail will be relevant for future preservation actions.

It has become obvious that the practical use of the DIO needs to be documented in terms of examples and best practices for both common manual documentation and automatic extraction. Those best practices do not add new semantics to DIO but provide some pragmatics that helps doing the interfacing with other stakeholders and tools, now and in the future. We included in section 4 of this document examples of such design pattern and external resources, which are more thoroughly documented on the TIMBUS developer space in the opensourceprojects.eu portal.

3.2 Inclusion of new Ontologies

The LegalDSO is a perfect example of relevant legal concepts, that weren't captured in the first set of ontologies. The Legal DSO serves as a meta-model for legal aspects relevant aspects of digital preservation as published in [6]. The initial legal ontology was developed in the context of a drug prescription business process from WP9, which is highly dependent on the legal context and that could not be reasonably assessed without an in-depth understanding of those aspects. However, to be included as a TIMBUS DSO, these concepts were first validated on multiple other use cases and the concepts were mapped against the DIO. In section 5 we report the design process and validation of the LegalDSO, which serves as a good use case for future additions to the TIMBUS Context Model.

3.3 Iterative Refinement of Concepts

The SensorDSO was one of the first conceptual models done specifically to serve a use case. When the first ontology was designed, a number of relevant tools were still not available, so that the ontology was built upon existing pre-TIMBUS experiences. As an in-depth analysis of the context of the Dam Use Case (WP8) was done

and integration specifically with extractors and the overall TIMBUS architecture was progressing, it became evident that this ontology needed to be revised in parts to efficiently bridge the semantics of the domain with abstract reasoning on the DIO. As TIMBUS follows a business-process and risk-driven, these developments were motivated by the business-processes and risks of the use cases and led to refinements of tools and models.

It is important to note that this DSO evolved in a way to make the DIO view on the system stable and so that results analysis were not altered. The new DIO can better include and relate further contextual information and scales better when modelling real use cases. It can also be efficiently extracted automatically.

3.4 Adaption and Mapping of existing Ontologies

One of the major design principles underlying the TIMBUS context model is its extensibility and the possibility to include external ontologies. We have to acknowledge that TIMBUS is neither the only active project in the area of context and meta-data modelling nor in the area of digital preservation. Although it might seem tempting to design every bit from scratch, it is often better to bridge towards external efforts, especially since often a lot of context knowledge is already modelled and ready to be used. In the ontological frame of TIMBUS it often suffices to map existing concepts to the DIO or in some cases to DSOs. The information described via the existing ontologies can be harnessed on the level of the DIO. We will continue to build such bridges to other knowledge and context mining and modelling efforts for business processes their relevant artefacts.

The prominent examples that are presented in this document include the mapping and use of the PREMIS Ontology which provides another (resource oriented) view on digital preservation, and the mapping to the commonly used business modelling standard BPMN, which allows the interfacing of tools using the TIMBUS context model with a number of existing process models.

4 Improving the use of the Context Model

As discussed before, the focus of this deliverable is to improve the usability and applicability of the TIMBUS Context model. When looking at the core of the Context Model (which is the DIO) and the mapping towards the DSOs that enable a DIO view on use-case-specific artefacts, we are particularly considering two different opportunities for improvement for practical use. One is further formalization and documentation of best practices and the other is tool support. Without diminishing the importance of continuous and future documentation this report is a first starting point for understanding best practices for the use of the context model.

4.1 Common TIMBUS Best Practices

The available modelling concepts, especially the Archimate domain-independent concepts that are very broad, can be easily understood by different users in a different way and thus lead to rather differently looking models for the very same use-case.

When discussing and comparing or automatically analysing these models, their quality can be greatly increased if common design patterns are used and if the models adhere to them. Therefore, from the experiences made modelling the various use cases, and from the extensive comparison of the models created for them, we propose a set of design patterns that should be used for commonly recurring components in business processes (within one or between many), especially on the technology and application layer.

4.1.1 (External) Web Service call design pattern

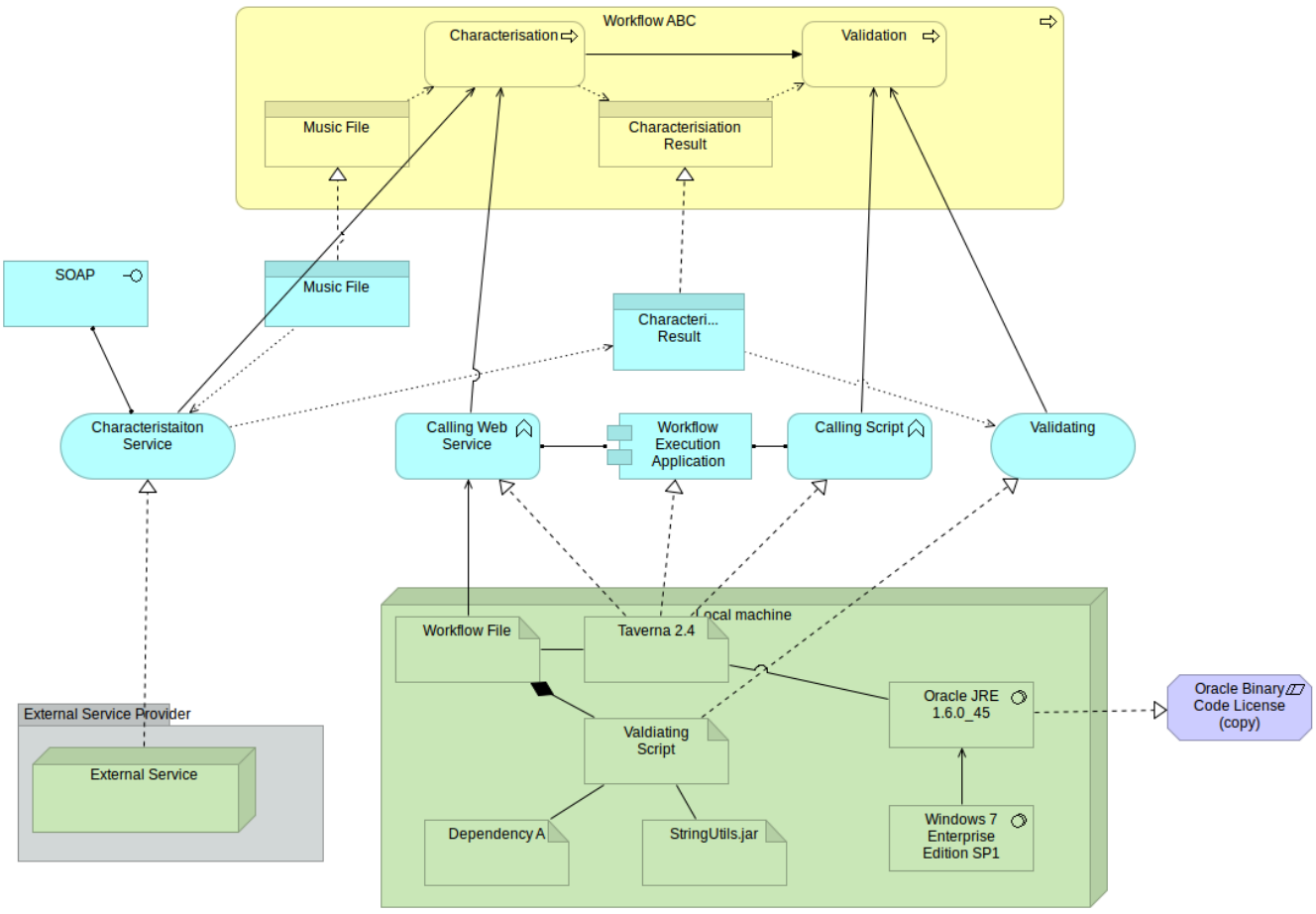

Figure 3: Example use of an external web service

Figure 3 shows a typical model of a business process implemented by different local and external services (invoked by a workflow engine). We suggest to always model the following elements:

- A web service is provided by an application service, in the example: ***Characterisation Service;***
- It needs to define an interface, example: ***SOAP;***
- It has input and output parameters – data objects (example: ***Music file***, ***Characterisation Result***);
- It might be connected to a Node, which can be detailed (if internal) or not (if external).

4.1.2 Use of Prototypes and Instance Specialization

Often the DIO is manually modelled using abstract object, however, extractors can extract thousand of elements that can be automatically mined. Instead of creating many single objects, we suggest the use of the *specializes* relation in analogy to *prototype* or *instance-based* programming. The semantics should be understood that all relations should be copied to the related object. An example is e.g. a Node extensometer

at a concrete damn that can exist 1000x times. Instead of creating relations into the different DIO layers from 1000 individual nodes we use one prototype node that is specialized 1000.

4.1.3 Data Object vs Artifact Usage

Particularly mapping DSO concepts to the DIO different types of data entities are important, particularly to digital preservation. Data fed into Application Function is typically modelled as Data Object in Archimate. Commonly when modelling DP problems in DSO, however, the particularly the "physical" data is of importance. Therefore we suggest the use of **Artifact** wherever any digital preservation can be directly conducted on the object in question.

4.1.4 Artifact vs System Software Usage

The Archimate specification [1] is not very precise on Artifact vs. System Software

- "An artifact is defined as a physical piece of data that is used or produced in a software development process, or by deployment and operation of a system. An artifact represents a concrete element in the physical world. It is typically used to model (software) products such as source files, executables, scripts, database tables, messages, documents, specifications, and model files. An instance (copy) of an artifact can be deployed on a node. An artifact could be used to represent a physical data component that realizes a data object."

- "System software represents a software environment for specific types of components and objects that are deployed on it in the form of artifacts. System software is a specialization of a node that is used to model the software environment in which artifacts run. This can be, for example, an operating system, a JEE application server, a database system, or a workflow engine. Also, system software can be used to represent, for example, communication middleware. Usually, system software is combined with a device representing the hardware nvironment to form a general node."

In our experience it was often debatable, which element would e.g., be "Microsoft Word", where some sources suggest to use an Artifact. Also the type of a program library (a JAR or a .so) vs. a configuration file, which would both be Artifacts, is not very logical. We finally suggest to stick to the examples of the Archimate definition by doing induction on the provided examples in the specification:

- System Software in the example: the ***operating system***, and the ***Java Virtual Machine***

- Artifacts: Java Libraries (*StringUtils.jar*, ...), Taverna (executable Application), Workflow definition (XML File)

4.1.5 Design Pattern Executable Application

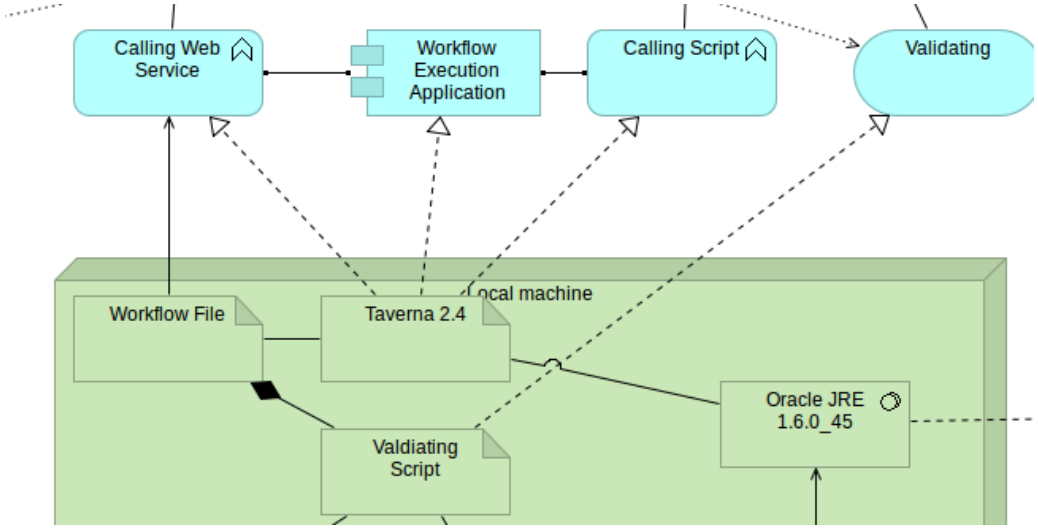

Figure 4: Example showing Taverna as Workflow Execution Application

As shown in Figure 4:

- If an Artifact is in fact an executable application, provide an application component (**Workflow Execution Application**, *realizedBy* **Taverna 2.4**)
- Specific functions provided by the Application component can be modelled, especially if they are called by a process tep (***Calling Web Service***, ***Calling Script*** are two functions provided by Taverna)

4.2 Typical Tools for manually working with the Context Model

Typically, modelling a use case in TIMBUS starts with the DIO, which can be done via a native Archimate editor. The model is then converted to OWL using the Archi2OWL Conversion Tools. From then on, standard semantic web tool chains can be used. In the following sections, we shortly describe the tools typically used when working with the Context Model. They provide a best practice for modelling.

4.2.1 Archimate Editor and Archi2Owl Conversion

TIMBUS build upon the Archimate standard as upper ontology, i.e. the TIMBUS DIO. *Archi* is a free, open source, cross-platform tool and editor to create such Archimate models used throughout the TIMBUS project. It is commonly used to manually create DIO models for business processes within TIMBUS. *Archi* is java-based and available for most operating systems (http://archi.cetis.ac.uk/download.html). In order to work with the Archimate Model as a DIO Instance, a converter tool from Archimate to OWL was developed for TIMBUS, in the scope of deliverable D6.2, which processes the instances of the concepts of the model and transforms them to an ontology containing OWL Individuals.

The converter is available online via:

https://opensourceprojects.eu/p/timbus/context-model/converters/archi2owl.

In order to use the converter inside *Archi*, the user has to copy the generated jar file dist/net.timbusproject.context.archi2owl-plugin.jar to the "plugins" directory in his *Archi* installation folder.

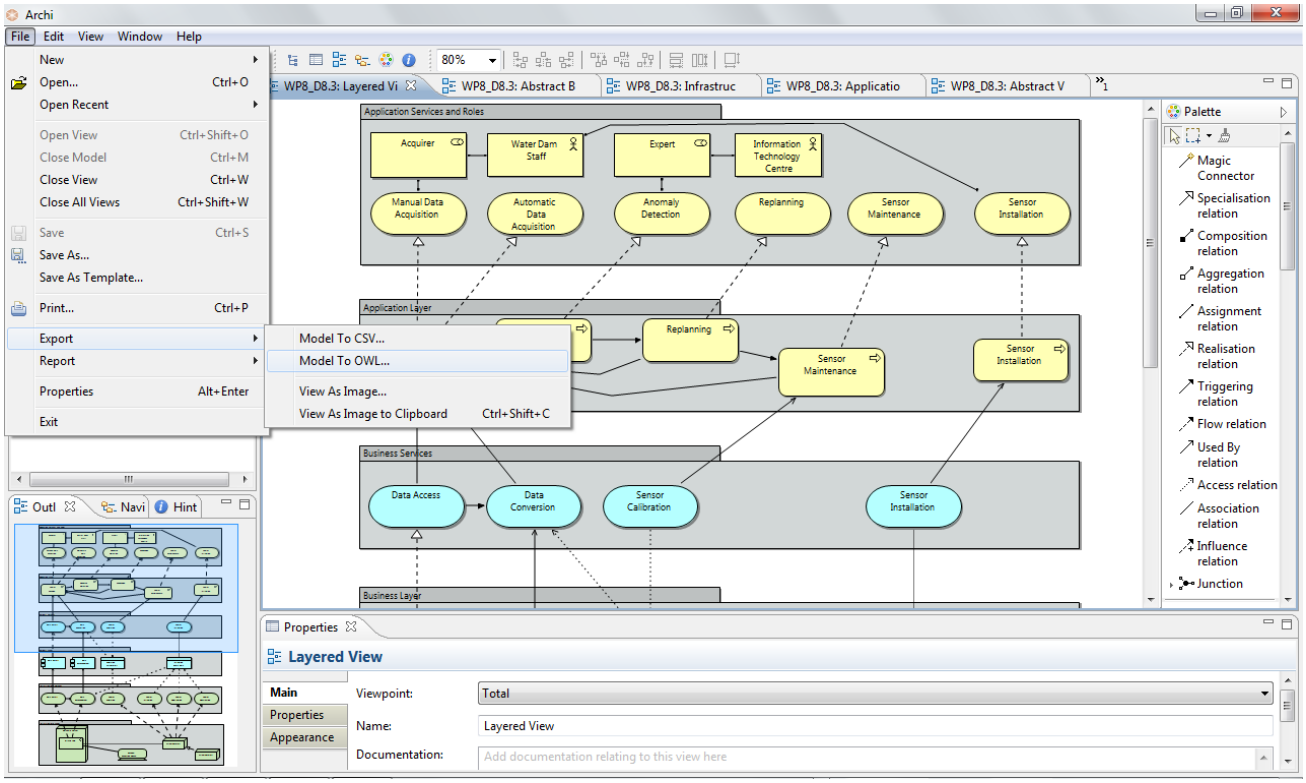

Figure 5: The conversion functionality available via the Menu Entry "File -> Export -> TIMBUS Owl"

Alternatively a user can run

```
java -jar dist/net.timbusproject.context.archi2owl-standalone.jar -i <archi file> -o <dio file>
```

from the command line.

4.2.2 Editing, Visualization and Querying with Protégé

Protégé is a free, open-source platform with an extensible architecture that provides a growing user community with a suite of tools to construct domain models and knowledge-based applications with ontologies [2]. It is developed by the SMI (Stanford Medical Informatics) group at Stanford University. Protégé has a community of thousands of users. The core of this environment is the ontology editor, and it holds a library of plugins that add more functionality to the environment. An OWL representation of the legal ontology was built in protégé. In order to edit, query and visualize ontology instances manually, we mostly use the Protégé Desktop tool.

The TIMBUS Projects developed a visualization plugin in the scope of task T6.2 that retains the layouting of the original Archimate Source files. To install the plugin get it from: https://opensourceprojects.eu/p/timbus/context-model/ontology-visualisation/

In order to run it, follow the build instructions in the README documents.

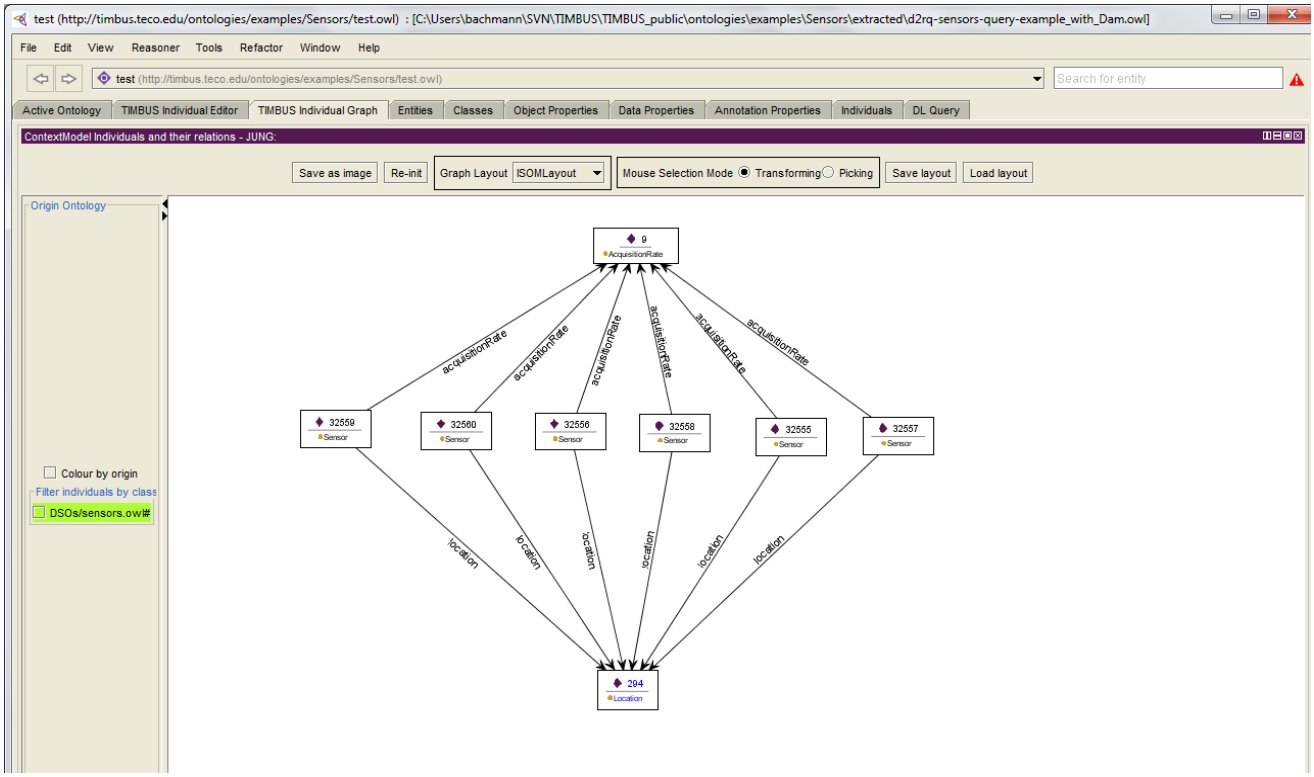

Figure 6: TIMBUS plugins for editing and visualization are available via the "Windows->Tabs" menu entry.

4.2.3 Querying via a Graph Database and SPARQL Endpoint

Jena Fuseki provides a SPARQL interface over HTTP that can be used by numerous command-line and web-based tools, and is an easy entry point for trying TIMBUS ontologies in your own tools. Because the configuration of the triple store is not trivial, we are providing a Makefile that imports all relevant manually created RDF Graphs (i.e., DIO, DSO instances) into the triple store within one service that provides a so called union graph that behaves similar to the Protégé interface: The Makefile parses all *owl:import* statements automatically and loads those ontologies.

The following commands should work on any system that has a GNU make, wget and java binaries installed:

```
wget -R -np https://timbus.teco.edu/ontologies/

cd ontologies

make all

make start
```

The Fuseki server provides a webpage on http://localhost:3030. By default, all examples covered in this deliverable (with the exception of PHAIDRA and RCAAP) will be available as SPARQL Services. The Makefile

enables the JENA owl-micro reasoned by default, which covers most constructs used by DIO/DSO axioms and has a reasonable performance for use on standard PC systems.

All necessary files including the Jena and Fuseki runtime are automatically installed in the local directory by the Makefile

```
<#service_{NAME}> rdf:type fuseki:Service ;
    rdfs:label                          "{NAME}" ;
    fuseki:name                         "{NAME}" ;
    fuseki:serviceQuery                 "query" ;
    fuseki:serviceQuery                 "sparql" ;
    fuseki:dataset                      <#{NAME}_infdataset> ;
    .

<#{NAME}_infdataset> rdf:type ja:RDFDataset ;
    ja:defaultGraph <#{NAME}_infmodel> ;
    .

<#{NAME}_graph> rdf:type tdb:GraphTDB ;
    tdb:dataset <#{NAME}_dataset> ;
    tdb:graphName <urn:x-arq:UnionGraph> ;
    .

<#{NAME}_infmodel> a ja:InfModel ;
     ja:baseModel <#{NAME}_graph>;
     ja:reasoner
          [ja:reasonerURL <{REASONER_URL}>].

<#{NAME}_dataset> rdf:type tdb:DatasetTDB ;
    tdb:location "db/{NAME}" ;
```

Figure 7: Template assembly to instantiate Fuseki services

4.3 Typical Reasoning on the Context Model

By using the reasoning capabilities, a question such as the following can be answered for validating the correctness of the ontology:

1. What BusinessServices are used by the Customer BusinessRole?
2. What BusinessProcesses are used by the Customer BusinessRole?
3. What Archimate concepts belong to the ApplicationLayer?
4. What Archimate concepts are BehaviouralAspect?

In order to obtain an answer to the example questions above, they need to be converted into a description logic statements so that it can be processed by the Protégé DL reasoned.

1. BusinessService and usedBy value Customer
2. BusinessProcess and realizes some(BusinessService and usedBy value Customer)
3. hasLayer some ApplicationLayer

4. hasAspect some BehavioralAspect

TIMBUS takes from the computational inference features of ontologies for performing business process analysis. The proposed architecture makes possible the usage of reasoning for performing analysis of the models, providing the required views for stakeholders. Four possible analysis configurations are described below:

- *Intra-DIO reasoning*, when inference is limited to the concepts of the DIO. If the elements of the DIO are organized in different layers, then *inter-layer DIO reasoning* or *intra-layer DIO reasoning* is possible.
- *Intra-DSO reasoning*, when inference is limited to the concepts of the DSO;
- *Cross-DSO reasoning*, whenever a mapping transformation between different DSOs is available, inference can use concepts from different DSOs;
- *Cross DIO-DSO reasoning*, when inference uses concepts from both the DIO and one or more DSOs, requiring a mapping transformation between each DIO-DSO pair.

By using a Description Logic reasoner, features such as consistency checking, dependency inferring, and completeness checking are available, thus allowing the checking of the correctness of the EA models expressed on the DIO and DSOs. Querying facilities are also available for some ontology languages, which makes possible the retrieval of data stored in the ontologies and even simple computation by the application of filters.

The different reasoning configurations described can be used for the purpose of identifying the dependencies between elements more easily. For instance, the question "*what are the technological entities supporting the process acquisition of readings?*" can be translated into the descriptive logic query

Thing and hasLayer some TechnologyLayer and hasAspect some BehavioralAspect or hasAspect some ActiveStructuralAspect and dependsDown value Acquisition_of_readings

which uses elements belonging exclusively to the DIO, thus being an example of intra-DIO reasoning, more precisely, inter-layer-DIO reasoning.

Another example is the question "*which ApplicationComponents were responsible for performing the acquisition and transformation of the readings for SensorType Drain?*". For being able to answer this question, we need elements either from the DIO (i.e., ApplicationComponents) and the sensor DSO (i.e., SensorType). The descriptive logic query

ApplicationComponent and dependsUp some (Sensor and hasSensorType value Drain)

provides the results that can be seen Figure 8 thus being an example of cross-DIO-DSO reasoning.

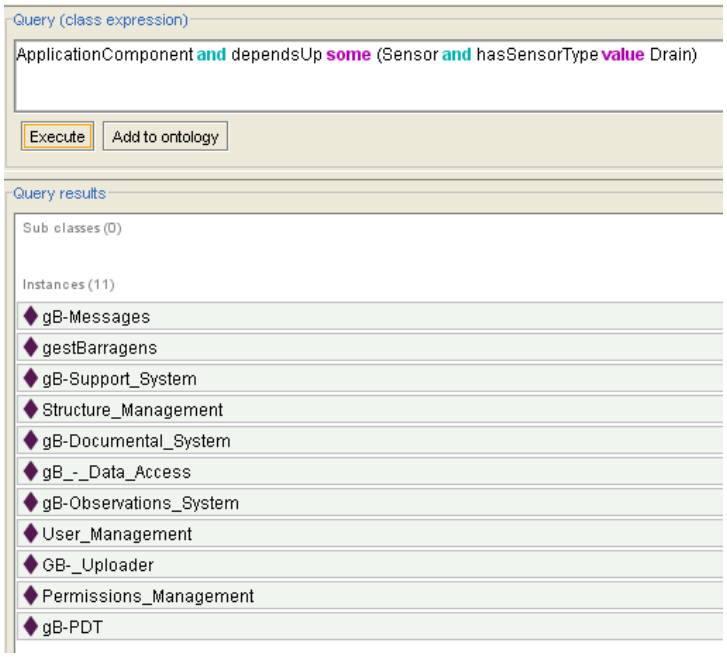

Figure 8: Cross-DIO-DSO reasoning example.

The execution of these queries will thus highlight the dependencies between the different elements of the infrastructure, and can be used as a valuable tool for decision making. By identifying the dependencies, it is possible to trace the propagation of the changes throughout the architecture. That information can then be used by the organization for decision making. Moreover, since the architecture can be enriched with the addition of new DSOs, other types of semantics might be included on the models, making possible that other kind of decision-making analysis can be performed. For instance, if a DSO that includes runtime data automatically captured from the environment, the impact might even be automatically quantified.

All examples can be tried out using the Protégé Desktop downloading the example files via http://timbus.teco.edu/ontologies.

5 Extending the Context Model: A Legal DSO

When modelling the context of a business system, users can always come to the point where the expressiveness of the existing concepts does not suffice anymore. Using Semantic Web Technology like OWL a user can dynamically add and refine concepts by extensions. However, often it is expected that concepts are reused and have a high relevance to future preservation tasks; thus it makes sense to extend the Context Model for that purpose.

When modelling the risk of the TIMBUS use cases, it became obvious that several preservation risks had to do with the legal context. While the Context Model was firstly only targeting physical surrounding and software dependencies apart as contexts apart from the core enterprise architecture, fitting concepts were missing. This has been overcome by designing a completely new DSO for the legal context. The steps described below present a best practice description for designing a new DSO from scratch, and linking it to the Context Model.

The theoretic work was partially published in [15]. The paper returns particularly contains a good overview of the related work in the field of Legal modelling in general

5.1 Introduction

Law is becoming an essential application domain for technology developments in many domains such as digital preservation domain. For example, in the case of digital preservation, all kinds of copyright protected data are an exclusive right of the copyright holder. Every process of a digital preservation system may violate this right if it stores copyright protected material. In simple words, the problems with legal taxonomies arise when the creators and the users don't share the same perspective. It usually happens when the creators of the taxonomy are lawyers and the users are not lawyers. Legal taxonomies for digital preservation can be represented with ontologies which are an explicit account of a shared understanding in any domain, and can improve communication which, in turn, can give rise to greater reuse, sharing, transparency, and interoperability. An inherent element of every DP activity is ensuring the authenticity and legitimacy of the performed actions and processes. One solution could be building a Legal Ontology for the DP domain, in order to integrate different legal perspectives and perform reasoning and inference over legal knowledge and information. Ontologies play an important role in knowledge sharing in the field of knowledge representation and reasoning. In simpler terms, Ontologies provide a common description of a conceptualization. This section presents a Legal Ontology that provides a hierarchical overview of how legal constraints and obligations (e.g., IP rights and licensing issues) could be implemented in an automated process of a DP system. The correctness of our legal ontology is validated with a set of competency questions defined in a specific case study. The aim is to obtain a clearer taxonomical view of the necessary legal knowledge that will address the concerns of industrial use-case DP stakeholders. Finally, a mapping between the Legal DSO and TIMBUS Context model (DIO) concepts was made using ontology matching techniques, which will be validated in the future work.

In section 5.2, the methodology for building the legal ontology is described in detailed. Also, the legal conceptual map that was made as the first phase of the methodology is described in detail along with a real scenario example. In section 5.3, the E-health case study that has been used for the legal ontology validation is described. Next, in section 5.4 the validation and reasoning questions applied to the legal ontology is

presented. A brief summary of the mapping of the legal ontology to the current TIMBUD context model (DIO) has been presented.

This approach to create a new DSO is one of the major elements of TIMBUS Context Modelling. We first develop standardized terminology in order to describe the various types and concepts existing in the legal domains along with the relations among them. Nevertheless, there is not in the scope of TIMBUS or any DP use case to build a universal legal ontology. Taking this into account, the objective of this first draft is to gain more detailed knowledge of the legal information required to overcome the weaknesses of the TIMBUS DIO. That information is to serve as extension to the core Context Model ontology.

5.2 Legal Conceptual Map

In the following we describe the conceptual modelling process on the basis of legal concepts relevant to digital preservation. We use bold font type for **concepts** and italic font type for *relations*. In **Fehler! Verweisquelle konnte nicht gefunden werden.** we can see a conceptual map of the legal perspective. As we live in a society where there are legal rules for the conduct of **Legal Person**s, their **Action**s *NeedToComplyWith* the **Legal Requirement**s imposed by the law. A **Legal Requirement** means generally everything that is demanded or imposed as an obligation by law (the description of the further concepts can be found in **Fehler! Verweisquelle konnte nicht gefunden werden.**). As a matter of course, **Legal Requirement**s *DifferAccordingTo* the **Location** where **Legal Person**s *carryOut* their **Action**s because the legal rules in each country are different and depend on the national legislation

The **Action** of **DigitalPreservation** *NeedToComplyWith* **Legal Requirement**s as well. Regarding **DigitalPreservation,** the most relevant **Legal Requirement**s *are* **Data Protection**, **IP-Rights**, **ObligationsToPreserve** and **Contracting**. In order to lawfully preserve **BusinessProcess**es each **Legal Person** has to be aware of legal restrictions, conduct law-abiding and fulfil its legal obligation.

For example, there are legal **ObligationsToPreservem** which *require* **DigitalPreservation**. Such **ObligationsToPreserve** can be generally found especially in the areas of tax law (annual balances, invoices, etc) or medical law (the health records of patients) where it appears essential that specific **Data** files need to be archived for a long period of time.

Artifacts (in the Legal namespace not the DIO) like **Software**, **Database**s or other types of **Data** *CanBeProtectedBy* **Copyright**. In order to be able to digitally preserve them without any infringement of **IP-Rights** a **Legal Person** has to be aware how far the protection of these **Artifact**s reaches and whether preservation **Action**s/Methods like Migration or Porting are allowed. While **Software** is usually a subject of **Copyright** protection, **Data** and **Database**s need to fulfil more specific criteria to be protected by **IP-Rights**. **Database**s for example *CanBeProtectedBy* either **Copyright** if they constitute the author's own intellectual creation; or if that is not the case, they have simply **ProtectionSuiGeneris** if their maker has made a substantial investment. According to the differing scope of protection different methods and technics for **DigitalPreservation** are permissible.

The scope of **IP-Right** protection *CanBeDefinedBy* not only national law or European directives but by **Contract**s as well. Due to the fact that **Legal Person**s *AreRightholderOf* **Software**, they *CanGrant* **RightOfUse** to other **Legal Person**s by signing (*CanSign*) a **Contract**. These **Contract**s *are* usually **License**s or **Sale Contract**s and **Software** *canBeDeliveredOnTheBasisOf* of these **Contract** types. Thus, not only the author and original

rightholder of the **Software** but other **Legal Person**s as well can be authorised to use the **Software** and obtain **RightofUse**. In this sense, some aspects of **Copyright** like the **RightofUse** *canBeDeterminedIn* **Contract**s. For example, the licensor *CanGrant* the licensee the right to freely modify or migrate the **Software** in a **License Contract** and thus make **Action**s necessary for the execution of **DigitalPreservation** legally feasible. In case that one **JuridicalPerson** like a company offers **DigitalPreservation** as a service for other **JuridicalPerson**s, they *CanSign* a **ServiceContract** and define the particular parameters of appointed service level in an annex to the **Contract** called **ServiceLevelAgreements.**

Data *CanBe* related to an identified or identifiable **NaturalPerson** and therefore *CanBe* **PersonalData** or even **SensitiveData**. Such **Data** needs legal protection from any acts of **DataProcessing** which are unwelcomed by the **NaturalPerson** to whom the **PersonalData** belongs. This due to the principle that every **NaturalPerson** has the right of informational self-determination and the right of privacy. Therefore, privacy security and **Data Protection** *are* essential **Legal Requirement**s and the compliance with them is monitored by public authorities.

Thus, if **Data** is digitally preserved it has to be guaranteed that the **Action**s necessary for **DigitalPreservation** are compliant with (*NeedToComplyWith*) the rules of **Data Protection**. One basic concept of **DataProtection** is that **DataProcessing** *requires* the **ConsentOfDataSubject**. The **NaturalPerson** to whom the **PersonalData** belongs is called in this sense **Datasubject.** The **ConsentOfDataSubject** has to be given in advance regarding the specific **DataProcessing** process and cannot be generic. A way to be compliant with the rules of **DataProtection** can be to "hide" the personal component of **Data** as well as the connection between the certain **Datasubject** and its **PersonalData** by transforming the **Data** to **AnonymousData** or **EncodedData**. Table 1 in Appendix B Legal Conceptual Map shows summary of Classes and relationships in the Legal conceptual map.

5.3 Reality Scenario of the legal conceptual map

By using a reality scenario the different legal concepts and their relations in the conceptual map are described and explained how the different concepts are connected and related with one another. We consider Pharmaceutical Company (**Juridical Person**) wants to use a Software which was created by the Software Development Company (**Juridical Person**). Therefore the Pharmaceutical Company signs a License Contract with the Software Development Company. As the Pharmaceutical Company wants to digitally preserve this Software in their DP System. So they have to take a look in the License Contract were the **right of use** is determined. In the License Contract actions like Reproduction and Migration which are indispensable for digital preservation are not explicitly established. Therefore it is necessary that the Pharmaceutical Company negotiates an amendment of the existing License Contract. The **Software** is **copyright** (are **IP-Rights**) protected and the Software Development Company as the **rightholder** has the **exclusive rights** of the **software** and therefore can grant the additionally needed **rights of use** to the Pharmaceutical Company. They have also signed a **Service** and Maintenance **Contract** and in the **Service Level Agreement** all the details regarding e.g., the correction time are established. As the Software is crucial for the Pharmaceutical Company they have also signed an **Escrow Agreement** to be sure that in case of bankruptcy of the Software Development Company or the non-maintenance of the Software the Pharmaceutical Company get all the relevant material like the Source Code and the Development Documentation to be able to maintain the Software or to contract another Company (**Juridical Person**) to do the relevant corrections for them. Since the Pharmaceutical Company has

to fulfil their Legal **Obligations to preserve** all their relevant **data** for Tax Law as e.g., the payroll information. This payroll information contains the **personal data** of the employees (**Natural Person**) of the Pharmaceutical Company. Therefore the Pharmaceutical Company has to fulfil the **Data Protection** requirements. For storing the payroll information with the **personal data** of the employees in the DP System the Pharmaceutical Company needs the **Consent of** the employees who are the **data subjects** or a legal permission. Because a general **Consent** of the **Data Subject** cannot be given, the Pharmaceutical Company needs the **Consent** for each **Data Processing** Operation. It often happens that work in companies is outsourced. In case the Pharmaceutical Company wants to contract another Company (**Juridical Person**) to do the Digital Preservation for them. Hence the Pharmaceutical Company has to think where this other Company will have the DP System Storage because depending on the **Location** of the Storage of the DP System the **Data Protection** Regulation will differ. Because it makes a difference if the DP Storage is in another European Country or may be in the United States were other **Legal Requirements** of **Data Protection** have to be fulfilled to allow the Transfer of the **Personal Data** (**Data Processing**) to this other Company (**Juridical Person**). The Pharmaceutical Company has to fulfil sector specific **Obligations to Preserve** e.g., **Data** like manufacturing formulae or testing results because the Pharmaceutical Company is obliged to keep records about the batch documentation of the medical products. If this batch documentation exists in electronically form this documents must be stored in the DP System. This is very important because the competent national (**Juridical Person**) or European authorities (**Juridical Person**) can audit or control the Pharmaceutical Company and will require the provision of the stored **Data.** As mentioned above the Pharmaceutical Company has also to fulfil non-sector-specific **Obligations to Preserve** the relevant information e.g., for Tax law or Commercial Law. Hence it is necessary to preserve all relevant **Data** e.g., which contains in information relevant for taxes. This would not only be the already mentioned payroll information, but also e-invoices or e-mails which contain relevant information what can be significant in case of a tax-examination.

5.4 Legal Concept Description

The description of all the concepts is explained in Table 1.

Table 1: Summary of Legal Concept Descriptions

Concept	Relation
Legal Requirement	*Something that is demanded or imposed as an obligation by law.*
Data Protection	*A legal requirement/obligation that ensures personal data is not corrupted, is accessible for authorized purposes only, and is in compliance with applicable legal constraints.*
IP-Rights	*Rights based on the principle that a particular type of intellectual property is protected and allocating exclusive rights to the author. The author can grant rights of use to a third person, e.g., by licensing.*
Copyright	*A legal concept giving the creator of an original work exclusive rights to it.*
ProtectionSuiGeneris	*Protection of qualitatively and/or quantitatively substantial investment in either the obtaining, verification or presentation of contents*

ObligationsToPreserve	*Requirements to preserve/store data in order to fulfil legal obligations, e.g., with regard to tax law, commercial law, etc.*
Contract	*Agreement by two or more parties to create legal obligations, establish rights between them.*
SaleContract	*Permanent allocation of a product connected with the transfer of ownership; e.g., regarding software no obligation to return or delete it from the application system after a certain time.*
ServiceContract	*Contract in which parties oblige themselves to take action and deliver a particular result*
ServiceLevelAgreement	*Part of a service contract which formally defines the owed level of service, establishing further rights and duties.*
License	*A contract between a right holder and an end-user of the licensed application, to authorize its use and/or distribution.*
Escrow Agreement	*Escrow (Software Escrow) is a contract that offers a mitigation strategy by placing a trustable third-party between the developer and customer.*
Location	*An area, which is characterized by particular geographical, social and/or cultural features*
LegalPerson	*Each entity that is able to perfom legal actions.*
NaturalPerson	*Human being with legal capacity to act*
JuridicalPerson	*Entity created by law having distinct identity, legal personality, duties and rights considered to be acting as single fictional individual, e.g., a corporation, company*
Artifact	*An object that has been intentionally made or produced for a certain purpose, such as a picture, text, database, etc. If certain requirements depending on national law are fulfilled, the objects could be protected with regard to IP rights.*
Software	*is any set of machine-readable instructions that directs a computer's processor to perform specific operations*
Database	*A collection of independent works, data or other materials arranged in a systematic or methodical way and individually accessible by electronic or other means*
ExclusiveRightOfRightholder	*Rights granted solely to the author by law.*
RightOfUse	*Entitlement to use the protected work, e.g., exploitation rights*
Action	*Activity of a legal person*
BusinessProcess	*A collection of related and structured activities or tasks to achieve a particular objective, e.g., the production of a specific business service, business function or product*
DigitalPreservation	*The series of managed activities necessary to ensure continued access to digital materials for as long as necessary, involving the planning, resource allocation, and application of preservation methods and technologies to ensure that digital information of continuing value*

	remains accessible and usable. It combines policies, strategies and actions to ensure access to reformatted and born digital content regardless of the challenges of media failure and technological change. The goal of digital preservation is the accurate rendering of authenticated content over time.
DataProcessing	*Any operation or set of operations which is performed upon data, especially personal data, whether or not by automatic means*
DataMinimisation	*policy of gathering the least amount of personal information necessary to perform a given function*
PersonalData	*Any information relating to an identified or identifiable natural person*
SensitiveData	*Data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, trade-union membership, and data concerning health or sex life.*
Datasubject	*Person affected by transfer and processing of data*
ConsentOfDataSubject	*Informed and specific agreement which must be given freely, clearly, and unambiguously by a data subject before transfer of data*
EncodedData	*Information which is being put into code*
AnonymousData	*Information which cannot be related to an identifiable person*

5.4.1 Legal Ontology

Once the legal conceptual map was built and validated. It was used as the input to build the ontology in Protégé (see section 4.2.2). The World Wide Web Consortium (W3C) has proposed several languages such as XML, RDF, OWL to model the information. OWL is the latest and the most complex ontology language presented by W3C. OWL is a Semantic Web language designed to represent rich and complex knowledge about things, groups of things, and relations between things. OWL tries to cover the weakness in RDF language. OWL has three different languages: OWL Lite, OWL DL and OWL Full, which vary in complexity and can be used by specific communities of implementers and users.

From the legal conceptual map concepts were mapped into OWL classes, relations were mapped into OWL ObjectProperties, and restrictions were added into those properties: InverseObjectProperties and SuperObjectProperties axioms were added to the OWL ontology. Cardinalities were also added to some of the concepts and relations. Furthermore, some DataProperty were defined. Also, the concept descriptions in Table 1 were added as annotation (rdfs:comment) for each concept in the ontology.

Figure 9 shows the OWL representation of the legal ontology with the Software class highlighted on the left pane and the annotation description and restriction in the middle pane and respective object properties and data properties in the right pane.

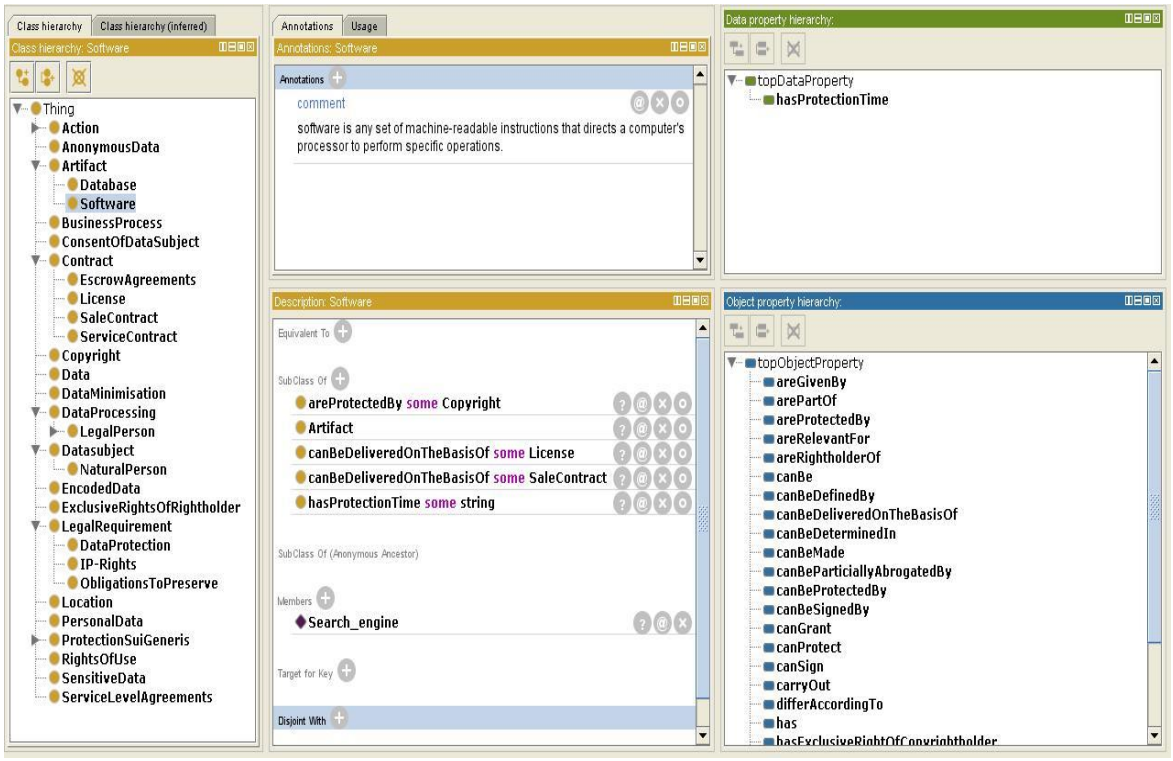

Figure 9: OWL representation of the legal ontology

The following Figure 10 up to Figure 13 showcase some general questions validated in the Legal OWL representation.

Figure 10: "Which are the basic legal requirements regarding digital preservation?"

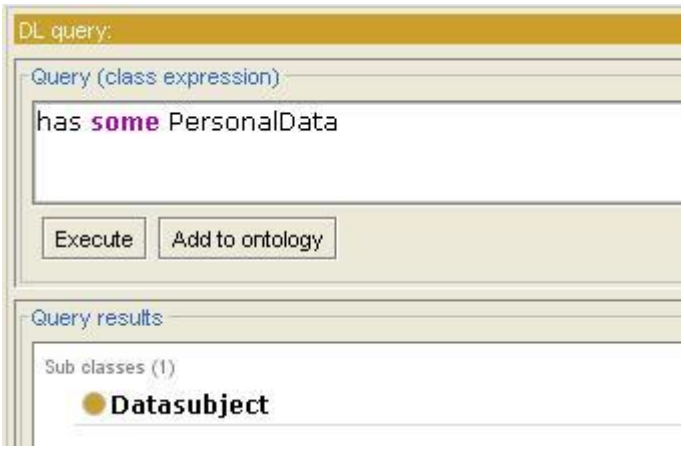

Figure 11: "Who has personal data?"

Figure 12: "Which type of contracts exist in the case of digital preservation?"

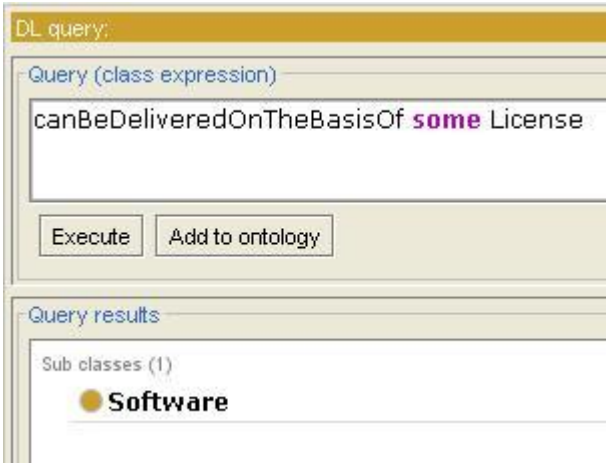

Figure 13: "How can the right to use software be granted?"

5.4.2 Case Study

The basis for the identification of the necessary legal entities of our ontology were the reasoning queries (competency questions) extracted from the industrial use-case stakeholders. Our specific case-study for the validation phase is an eHealth scenario.

Each prescription drug package sold in Europe must contain information about how it works and what the intended effect is. It also has to contain a description of side effects, instructions, and cautions for its use; including warnings about possible allergies. During a patient's visit, the medical practitioner tries to identify the best treatment strategy, which may include a prescription of one or more drugs. If a medical practitioner prescribes several drugs, the taking of various drugs may lead to Adverse Drug Reaction (ADR) [7,8]. ADR refers to physical or psychological reactions caused by medications that were taken at normal dosage but combined with other drugs. In contrast, the term Adverse Drug Event (ADE) refers to any harm caused by drugs, irrespective of the use or the dosage that was taken [9,10].

This case study examines an eHealth scenario in which problems related to ADR are tackled by means of a web-based solution that allows users to research ADE rules used by physicians and pharmacists.

It is obvious that, in this scenario, errors can seriously affect patients' health. This may in turn lead to lawsuits. In such court proceedings it would be necessary to re-construct the entire research process. This re-run would have to be conducted using the same soft- and hardware as well as the same input data. This is only possible if constant tracking and long-term conservation of discovery business processes is guaranteed.

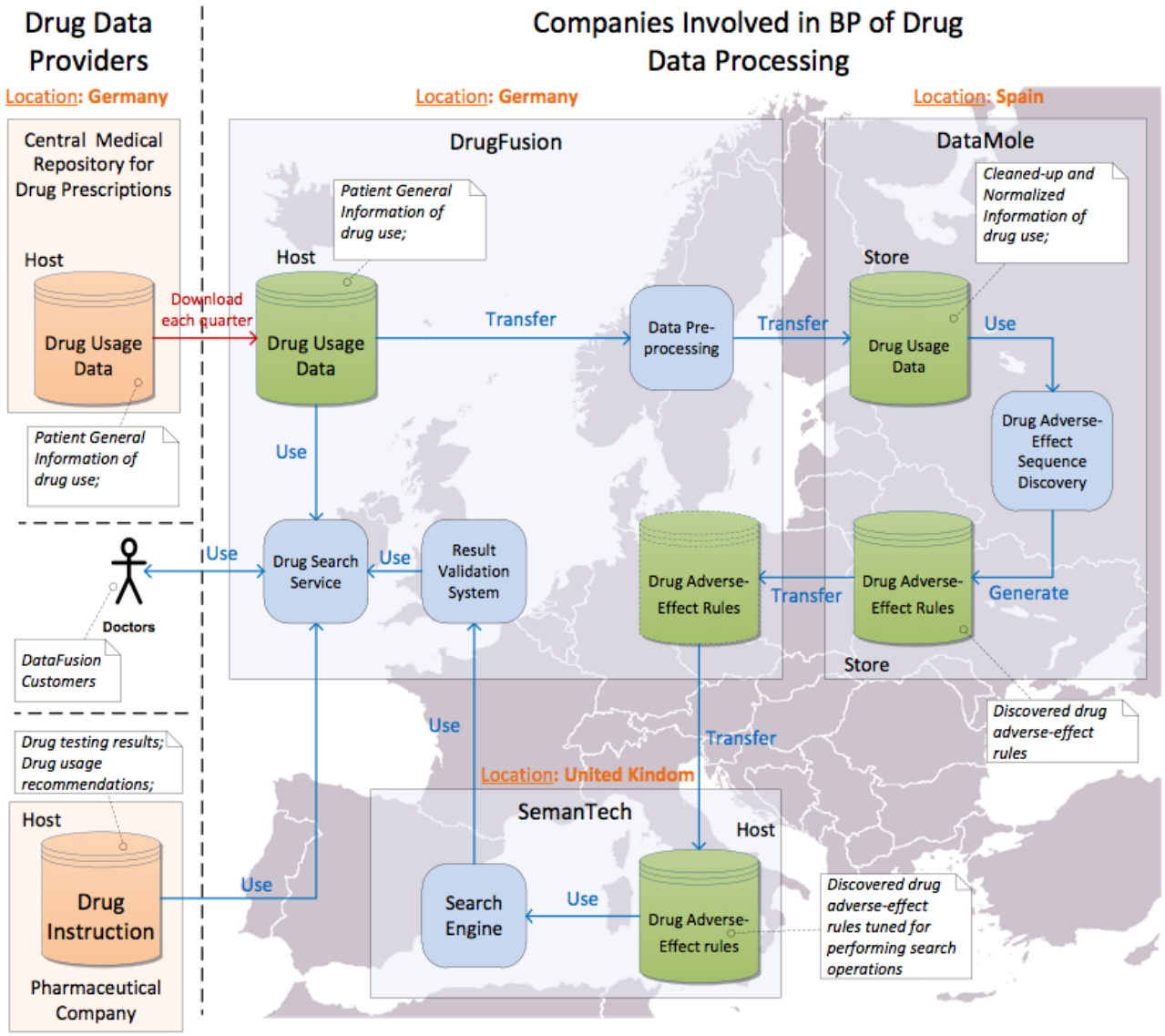

Figure 14: WP9 eHealth Use-Case Schematic

A high-level overview of the WP9 eHealth case-study (see also D9.3), which provides doctors and pharmacists with information about potential ADEs, is shown in Figure 13. The eHealth business processes include three companies: DrugFusion, DataMole and SemanTech. They also utilize two external Web-services: Central Medical Repository for Drug Prescriptions (CMRDP) and Pharmaceutical Company (PhC).

Long-term digital preservation enables corporations to comply with legal obligations and to provide for the safety of drug prescription within the Common European Market. The enterprises referred to in this use-case are real; their names were, however, modified for reasons of confidentiality. The discovery of business process can be distinguished from the search for business process. The function of the discovery process is to analyse data relating to drug usage collected quarterly and to generate ADE rules. The search for business process, in

contrast, indexes and retrieves relevant ADE rules that apply for patients in a certain physical or mental condition.

Legal risks in this case study include violations of data protection laws, infringements of Licenses and contractual rules (which exist between the companies involved in the business processes) and liability for incorrect information or damages. Liability for damages may consist of contractual liability, but also to tort liability as well as criminal liability. This scenario instances was defined in the legal ontology as individuals of different concepts. For example, from Figure 14, DrugFusion, DataMole, SemanTech, Pharmaceutical companied are all instances of the juridical person concept. Drug Pre-Processing, Drug Adverse Effect Sequence Discovery, Search Engine, Result Validation are all instances of the software concept. The Drug Adverse Effect Discovery and Drug Adverse Effect search are all instances of the business process concept. Also ADE Rules Database and Drug Usage Databases are all instances of the database concept. Drug Usage Data and Drug Adverse Effect rules are all instance of the data concept. Germany, UK, and Spain are instances of the location concept. Finally the contract between the DrugFusion&DataMole, DrugFusion&SemanTech, Doctors&DrugFusion are all instances of the service contract concept.

5.4.3 Validation Outcomes

After implementing the scenario in the legal ontology we had to validate the legal ontology by performing reasoning and inference over legal knowledge and information, we have applied reasoning queries (competency questions) to our Legal Ontology. The goal here is to ensure consistency/conformity and attain specialized legal information for the DP of whole business processes and services. A set of predefined competency questions were used in order to validate ontology. According to [11] one of the ways to determine the scope of the ontology is to sketch a list of questions that a knowledge base based on the ontology should be able to answer. The set of competency questions defined to validate the legal ontology is composed by the following questions:

1. What kind of data is stored in the repository (Anonymous data)?
2. Which database is protected by ProtectionSuiGeneris?
3. Which software has 70 years time protection by copyright?
4. Which database has 15 years time protection by SuiGeneris?
5. Who has the exclusive right of the copyrightholder for the Drug Instruction database?
6. What is the business process that exists between the drugfusion&datamole company?
7. What is the business process that exists between the drugfusion&semantech company?

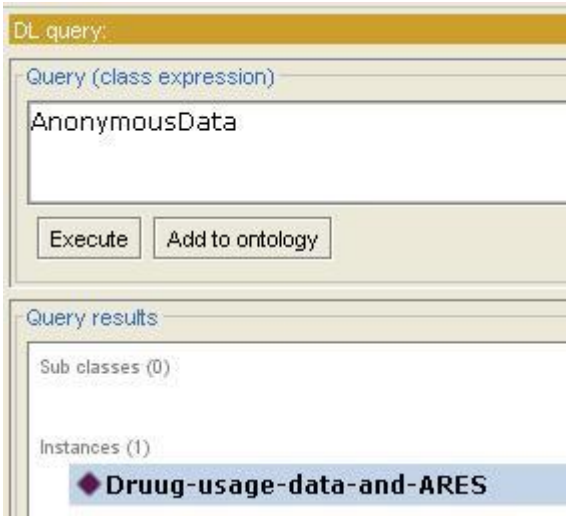

Figure 15: "What kind of data is stored in the repository (Anonymous data)?"

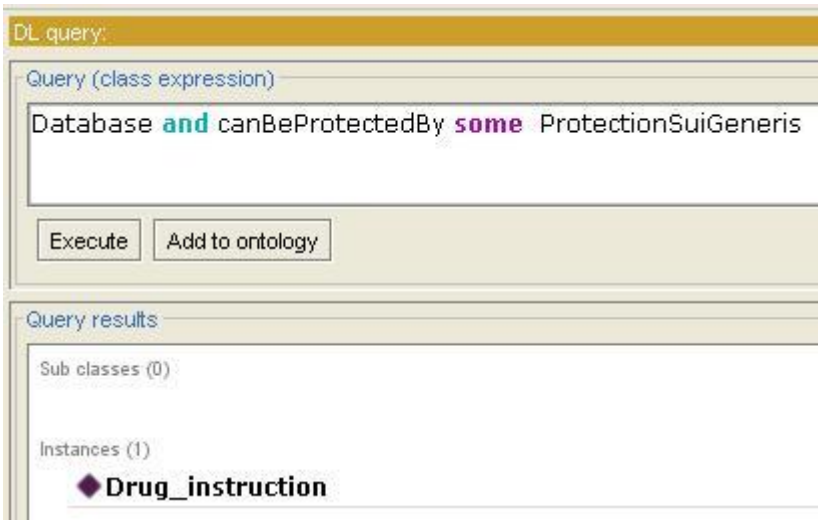

Figure 16: "Which database is protected by ProtectionSuigeneirs?"

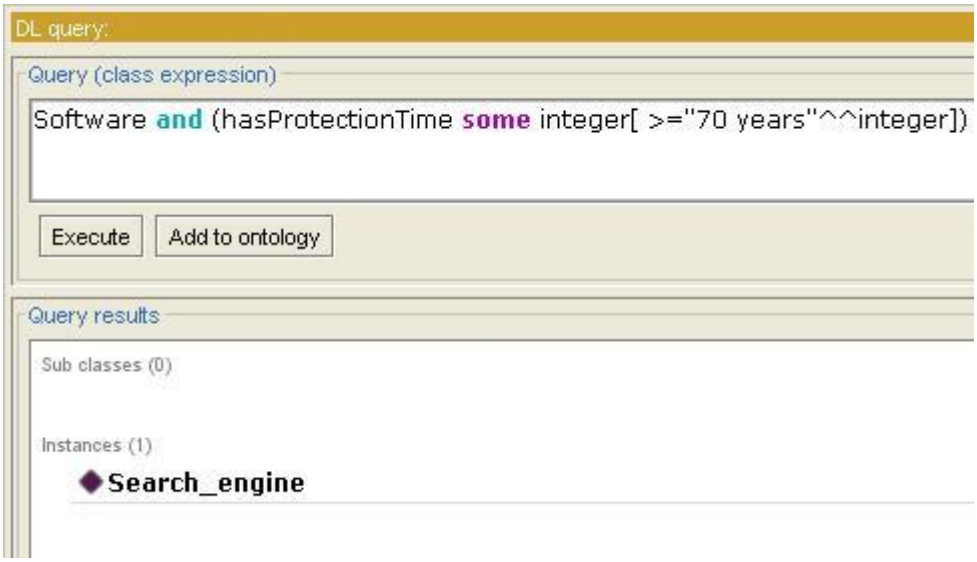

Figure 17: "Which software has 70 years time protection by copyright?"

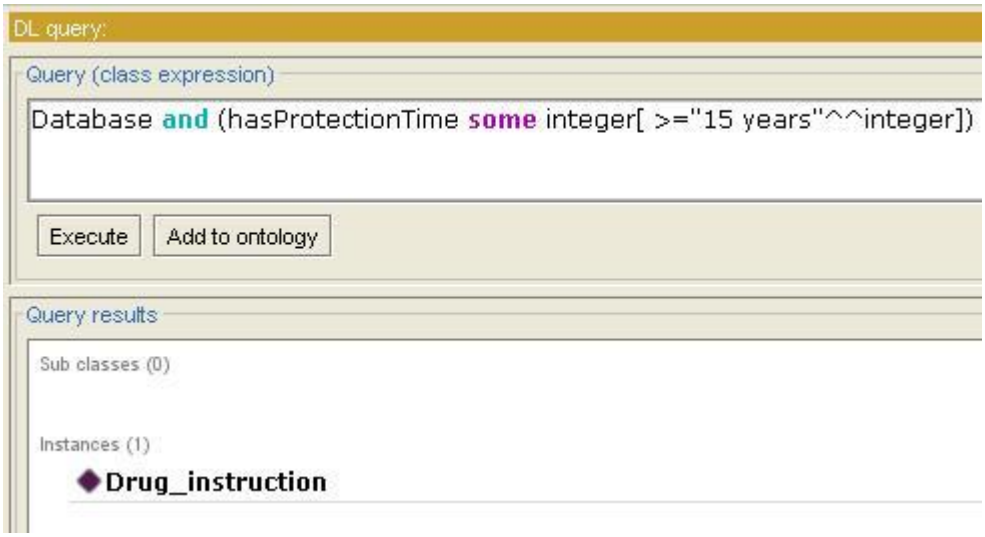

Figure 18: "Which database has 15 years time protection by suigeneris?"

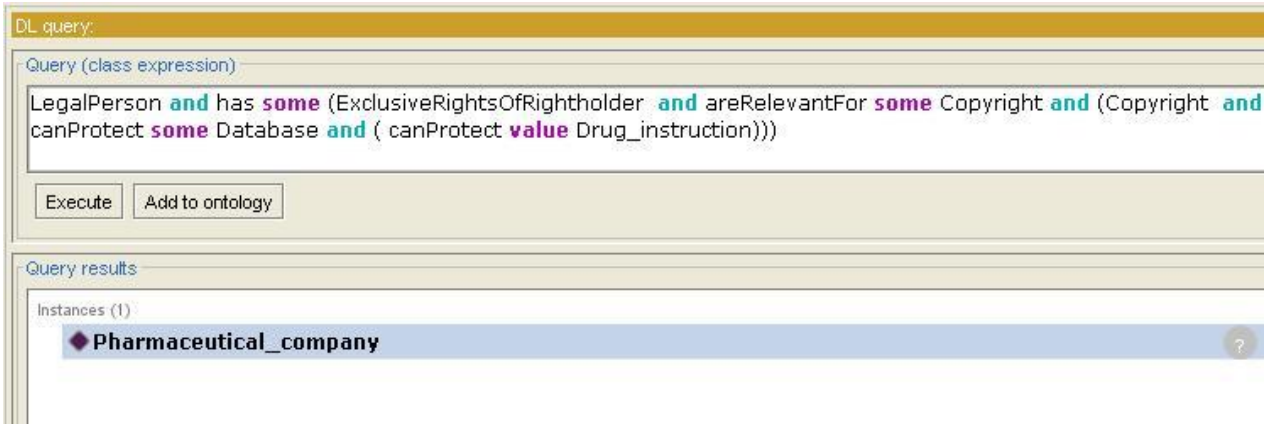

Figure 19: "Who has the exclusive right of the copyrightholder for the Drug Instruction database?"

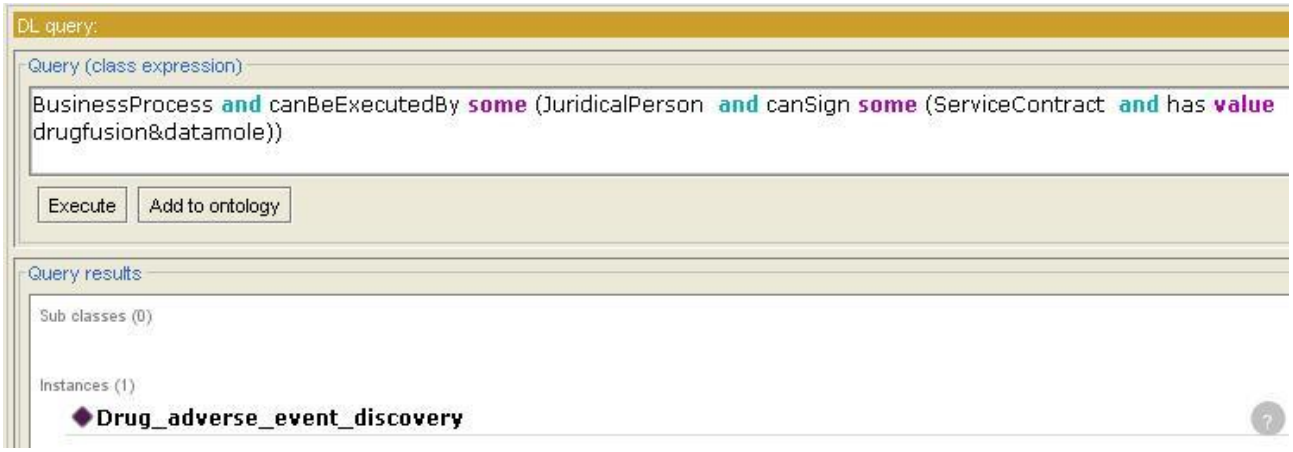

Figure 20: "What is the business process that exists between the drugfusion&datamole company?"

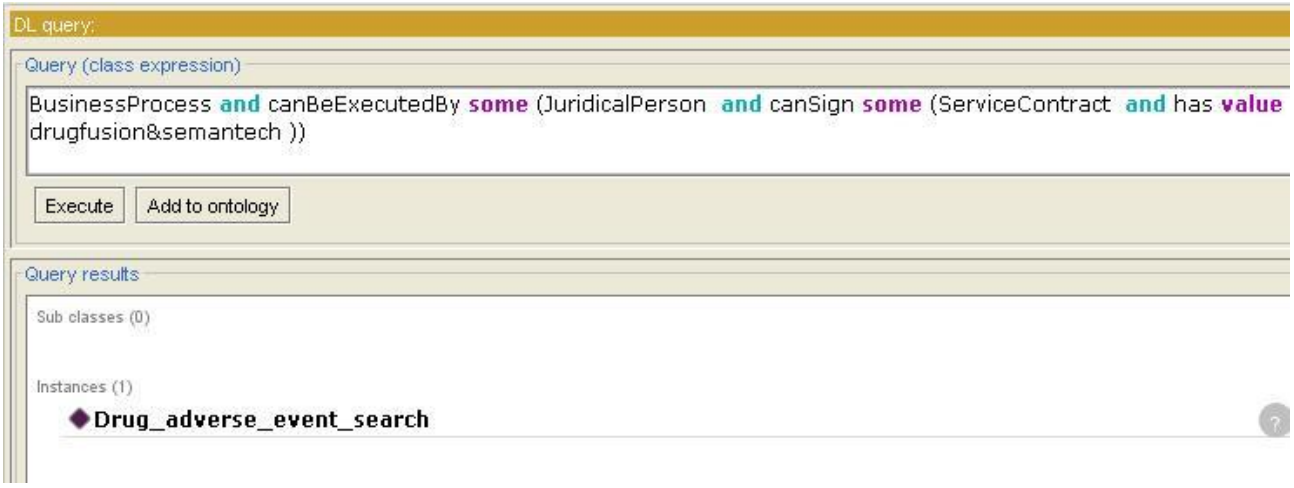

Figure 21: "What is the business process that exists between the drugfusion&semantech company?"

Figure 19, Figure 20 and Figure 21 depict the results of these reasoning processes. As you can see, our first version of the legal ontology was able to successfully answer all the queries. We will proceed in the future by performing more queries and also in different use cases.

5.4.4 Mapping to TIMBUS Context Model

Ontology mapping is the process of identifying common concepts and resources shared between ontologies, so when different ontologies contain facts about the same activities and resources, we can find new and interesting relationships between other activities and resources in those ontologies [12].

The legal ontology was made as the domain specific ontology for legal perspective in order to be mapped to the TIMBUS context model (DIO). The legal DSO was designed to be highly cohesive, meaning that it is limited to describing the concepts, relationships and rules pertaining to a single domain.

The current legal DSO version is related to six of the high-level concepts of the DIO. This transformation process was straightforward when there is a one-to-one relationship between the concepts of the DSO and the DIO.

Table 2 shows the initial version of the mapping, between the DIO and DSO concepts (Figure 22 shows the technical when imported to Protégé). We will validate the mapping in the future work.

Table 2: Legal DSO Mapping

Legal DSO Concept	Context model (DIO) Concept	Mapping Rational
Business Process	Business Process	The concepts were defined as syntactically and semantically equal
Location	Location	The concepts were defined as syntactically and semantically equal
Contract	Contract	There is a minor contract: a legal contract is the subclass of contracts that creates legal obligations
Artefact	Artefact	There is a potential coverage mismatch arising from the definitions that cannot fully be resolved. We are however considering legal artefacts equivalent as we do not see any critical inconsistencies arising from treating them equally.

Data	Data Object	In the legal ontology data is mostly used to distinguish different qualities of data. This can roughly be translated to data on the application layer of Archimate. Thus we define those concepts equivalent.
Legal Requirement	Requirement	Legal Requirements are a special type of requirements in terms of the DIO

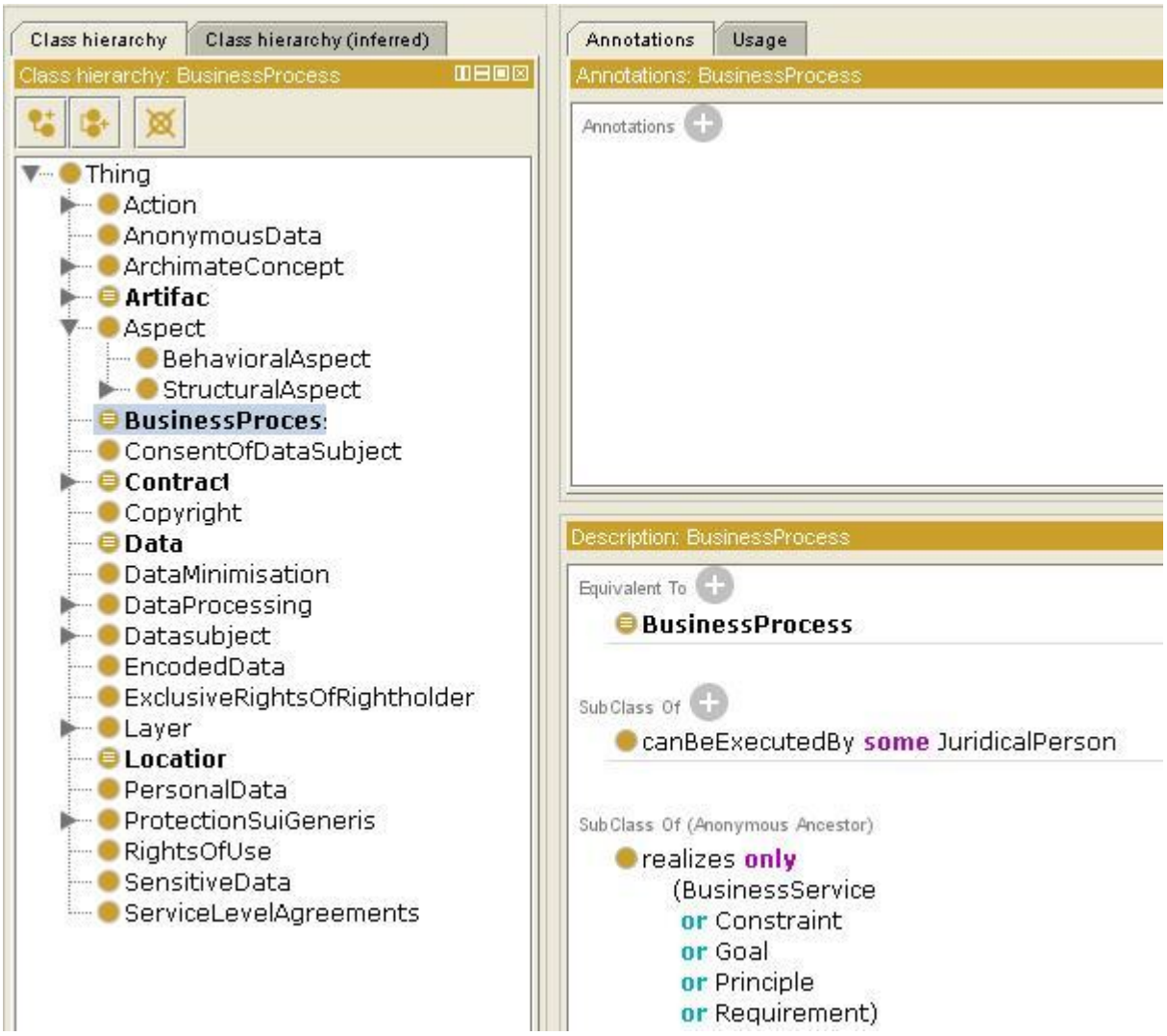

Figure 22: Mapping legal DSO and the DIO in Protégé

6 Refining Concepts: A Sensor DSO

6.1 Introduction

Sensors are the core components of the civil engineering use case (D8.1). The associated processes and information are crucial and important for ensuring the digital preservation and an intended redeployment.

Relevant processes are the gathering of sensor measurements, their validation and, associated, an anomaly detection. Other processes are the conversion of raw readings into physical quantities and the sensor calibration (both require the preservation of the Function and the constants). Additional to the process information, sensor meta-information like location, acquisition rate and the sensor type and the sensor readings themselves are also stored.

6.2 Overview

This domain-specific ontology describes **Sensors** of a certain **SensorType** (ontology concepts are set in bold type, while relations are italic). Each **Sensor** *hasLocation* which can be defined in more detail by its **GeoLocation** or **StructuralLocation**. Each **Sensor** is part of a data acquisition process that takes place in a frequent basis related to the **AcquisitionRate**. Within a **Campaign** the **Readings** of a **Sensor** are gathered. These are raw reading **Quantities** that are converted into result **Quantities** using a conversion **Function** and predefined **Parameters**. Each **Quantity** has a defined **QuantityType** and **Unit**. From time to time, each sensor needs to be recalibrated using Constants and an associated calibration **Function**. **Parameters**, **Constants** and **Functions** do not have to be defined within the ontology itself, the can also be part of a file within the file system which is referenced using the **Source** concept. For a later re-deployment a **HistoricalReadingValueModel** of a **Sensor** can be created.

6.3 Refinements

In contrast to the first version of the domain-specific ontology for the sensors scenario, several concepts and properties were added, renamed or modified. First of all, some concepts had an incorrect name which initiated a renaming.

- The concept **Algorithm** was renamed in Function as it is not a step-by-step procedure for calculations but a mathematical function.
- The concept **Coefficient** is now named Parameters as these values represent inputs for mathematical functions and not multipliers or coefficients for these functions.

Moreover, some concepts where refined.

- The **AcquisitionRatePerYear** was renamed into an **AcquisitionRate** to be more general and applicable to a wider range of sensors. In case there is a sensor that samples several times a second or more, the **AcquisitionRatePerYear** value would be confusing and intangible.

Additional to all concept changes mentioned above, there were also some new concepts introduced in the refined version of the ontology.

- Previously, the unit and type of a **Quantity** were referenced using *hasUnit* and *hasQuantityType* and represented by a **String**. Now, there are concepts for both of them, **Unit** and **QuantityType** to formalize them more properly.

- To enable references between a reading and the campaign in which it was acquired, the concept **Campaign** was added.

- The concept Source was added to enable referencing an entity, e.g., a file on the file system containing relevant data.

- The concept **HistoricalReadingValueModel** was added due to the objective of emulating and replaying sensors to bind a Sensor to the model that learned from the sensor's historical reading values and that is able to replay these.

But not only concepts but also properties were refined.

- In the course of renaming the concepts **AcquisitionRatePerYear**, **Algorithm** and **Coefficients** also the associated properties *hasAcquisitionRatePerYear*, has **Algorithm** and *hasCoefficients* were renamed into *hasAcquisitionRate*, *hasFunction* and *hasParameters*.

- The **isPartOfStructure** property was renamed to **isSubLocationOf** as it better captures the relation to the dam.

- The properties *hasGeoLocation* and *hasStructuralLocation* were replaced by the more general property **hasLocation** which refers to the concept Location which is a superclass of **GeoLocation** and **StructuralLocation**.

- The properties *hasX***,** *hasY* and *hasZ*, all three related to the **GeoLocation**, where renamed into *hasXCoordinate*, *hasYCoordinate* and *hasZCoordinate* to emphasize that they are coordinates relative to a predefined origin point within the structure.

- The properties *hasElement* and *hasSubElement* are now captured by the more general property *hasSubLocation*.

- In the previous version of the ontology, the raw readings of an acquisition were referenced using *hasReading* and the physical quantities (results of the conversion process) using *hasResult*. To avoid confusions by having the relations "**Sensor** *hasReading* **Reading**" and "**Reading** *hasReading* Quantity" and thereby using *hasReading* to refer to a **Reading** and **Quantity**, the property

hasRawReading were added to avoid the confusion and to emphasize that "Reading hasRawReading Quantity".

In contrast to the previous sensors ontology, some new properties were added.

- To address the newly introduced concepts the properties *hasCampaign***,** *hasHistoricalReadingValueModel* and *hasSource* were added.
- In the context of an anomaly detection, **Readings** are checked if they match predicted values. To capture the results of this procedure, the properties *isAbnormalReading* and *isNormalReading* were added.

6.4 Refined Ontology Structure and its documentation

For readability we have put the updated structure in Annex A. The latest version of the documentation including the OWL Sources is available online via (http://timbus.teco.edu/ontologies/DSOs/sensors.html).

We extended the SensorDSO using RDFa labels and comment to allow an inline documentation of the ontology. In earlier version the documentation was done separately from the ontology, which especially during refinements had the potential of creating inconsistencies. We used the *specgen* tool (www.github.com/specgen/specgen) to automatically generate human-readable documentation for the ontology. Furthermore by using ontologies also for the documentation, it is available online and linked to the concepts when practically working with the ontology during modelling and analysis.

6.5 Mapping to the DIO

Concepts from the domain-specific ontology are mapped to concepts in the domain-independent ontology.

Table 3: SensorDSO Mapping

DSO	DIO concept	Mapping
Sensor	Node	A Sensor is a DIO Node with the DataProperty hasType=Sensor
Function	Application Function	Functions in the SensorDSO are very specific types of Application Function
StructuralLocation	Location	Structural location is equivalent to the broad location term used in the DIO. No specific assumptions are

		made. Subclasses like GeoLocation contain specific information
Source	Artifact	The Source of a Model is a concrete Physical Data Artifact that can be stored and preserved.
HistoricalReadingValueModel	Data Object	The Historical Data Model is the input for Application Functions like Calibration, Emulation, etc.
Reading	Data Object	A Sensor Reading is a Common sub-type of Data Object in Sensing Oriented Use Cases. Commonly they are implemented via Databases

7 Relating the Context Model

7.1 Business Process Model and Notation Mapping

Business Process Model and Notation (BPMN) is a widely adopted graphical language for business processes modelling. A business process is described in BPMN in a Business Process Diagram (BPD), i.e., an annotated graph whose nodes explicitly represent activities, control flows, data, and auxiliary information about the process.

Examples of BPMN elements are: Event, Activity Gateway and Sequence Flow. Properties of basic elements concern both the usage of the BPMN elements to compose the business process diagrams, and the behaviour of the elements during the execution of a process.

Archimate and thus the TIMBUS DIO are structured languages that provide a holistic representation of enterprise architectures over different levels of abstraction from technology via applications to the business level. Because of its holistic nature, models produced in Archimate usually have to be re-drawn, in some other language in a detailed stage of the process such as a BPMN. This leads to unnecessary delays and costs during a transformation process.

The coherence between different models is difficult due to the different meta-model each language has (a meta-model is defined as an explicit model of the concepts and relationships between those concepts required to develop models) and correspondingly due to the lack of formal connection between these meta-models. Though, a better integration of the meta-models would make such transformations easier. Linkage between BPMN and EA meta-models would allow viewing the business processes at different layers of the enterprise in detail. Furthermore, transformations between Archimate and BPMN models, will guarantee the portability of these transformations between different modelling tools in use by different organisations. This would help to have a coherent modelling landscape between the models of Archimate and BPMN requires these models to be interrelated, and ultimately, the meta-models of the underlying modelling languages. Hence, it would improve extensibility and expressiveness of the two meta-models.

The most practical way to proceed with the mapping of Archimate and BPMN is to use ontologies and ontology mapping techniques as depicted in Figure 23.

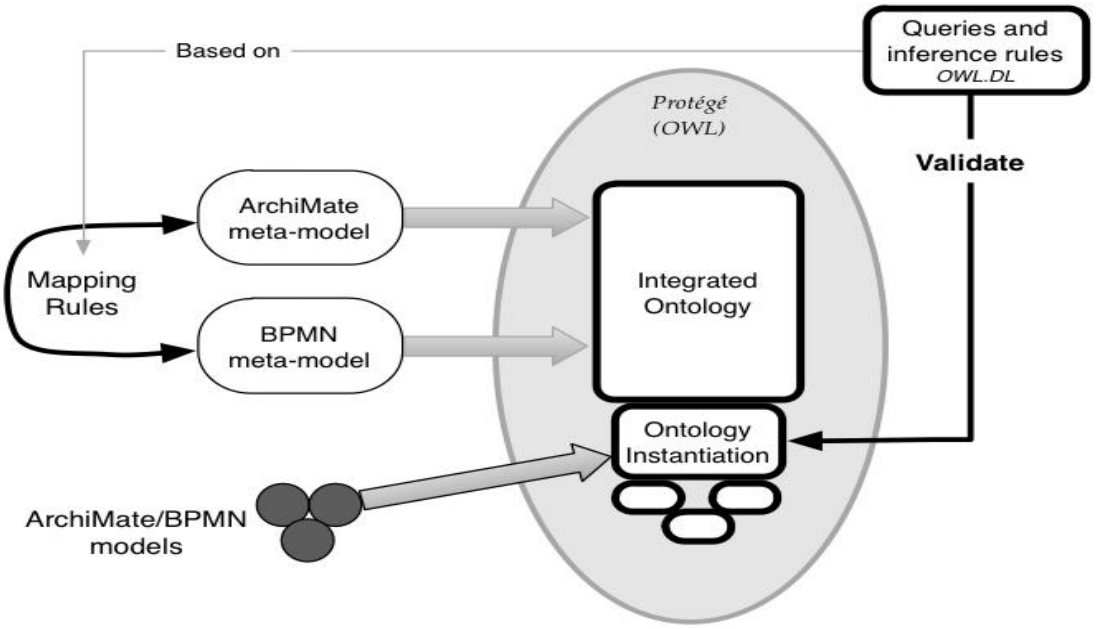

Figure 23: Ontology mapping scheme

As you can see from Figure 23, the goal is to integrate Archimate and BPMN meta-models by the use of the TIMBUS Context Model. For this purpose it was required to first transform the Archimate (see D4.3) and BPMN meta-models into ontologies (OWL), and then make an integrated ontology by mapping the common concepts and relationships of them by the use of ontology mapping techniques.

In the following table that shows the concept based integration we had to resolve different types of mismatches, between the ontologies.

- Syntactic mismatches: Two ontologies are syntactically heterogeneous if they are represented by different representation languages. To resolve this type of mismatches, simply transform the representation language of one ontology to the representation language of the other ontology. Although many times, translation is difficult and even impossible and may lead to source information omission.

- Lexical mismatches: Describes the heterogeneities between the names of entities, instances, properties or relations.

- Semantic mismatches: The mismatches identified at this level are related to the content of the input ontologies. The mismatches are classified into three abstract forms of mismatches

a) Coverage. Two ontologies are different from each other in that they cover different (possibly overlapping) portions of the world (or even of a single domain).

b) Granularity. Two ontologies are different from each other in that one provides a more/less detailed description of the same entity.

c) Perspective. Two ontologies are different from each other in that one may provide a viewpoint on some domain which is different from the viewpoint adopted in another ontology.

In the mapping between the TIMBUS DIO and BPMN, there were two kinds of mismatches. First, the lexical mismatch synonyms type, i.e., the same entity is represented by two different names, such as, Business Interface from Archimate and Interface from BPMN. Second, we have granularity mismatch, i.e., the same section of the domain is described but the depth of details is not equal. In Archimate and BPMN the concept Business Process is included in both ontologies but in the BPMN ontology it is described in a more detailed way, i.e., it has more attributes. Most of the ontology mapping approaches focus on automating the discovery of a mapping. But in our case, requires a precise mapping, so the mapping was done manually 1:1 mapping:

Table 4: BPMN Mapping

DIO	BPMN	Mapping rationale
Business Process	Activity(Sub-process, Task)	The business process definition in ArchiMate is more high level than BPMN. A BPMN model, indeed, provides a more detailed view of the actual process (sub-process, and task), than what a business process in ArchiMate model would do. Therefore, the business process in BPMN corresponds to a specialisation of (part of) the ArchiMate business process.
Business Interface	Interface	In ArchiMate a Business Interface is defined as a point of access where a business service is made available to the environment. The definition which corresponds to the definition of BPMN interface. Therefore, the interface in BPMN corresponds to a specialisation of the ArchiMate business interface.
Business Event	Event	In ArchiMate a business event is something that happens externally or internally and is similar to the concept of Event in BPMN. Therefore, the event in BPMN corresponds to a specialisation of the ArchiMate business event.
Business Collaboration	Collaboration	A Business collaboration is a possibly temporary collection of business roles performing a collaborative behaviour (interactions) within an organization. The notion is similar to the collaboration in BPMN, which is a collection of participants, shown as pools and of their interactions, shown by message flows. Therefore, the collaboration in BPMN corresponds to a specialisation of the ArchiMate business collaboration.
Business Interaction	Choreography Activities	Business interactions are external behaviours from the perspective of the roles participating in the business collaboration. Also in BPMN, choreography formalizes the way business Participants coordinate their interactions (extended type of Collaboration). Therefore, the choreography in BPMN corresponds to a specialisation of the ArchiMate business interactions.
Business Object	Data Object	A business object in ArchiMate is defined as a passive element that has relevance from a business perspective. In simple terms, a business object shows the data passed between business processes. This construct corresponds to the BPMN data input and data output, the primary constructs for modelling data within the Process flow .

Business Role, Business Actor	Pool, Participant	Business actors and business roles represent constructs performing behaviours, such as business processes. These entities correspond to the BPMN participant, a performer (of a process).
Application Function	Service Task, Script Task	An application function in ArchiMate describes the internal behaviour of an application component. Correspondingly in BPMN there exist activity types representing application functions (e.g., service task, script task).
Artefact	Data Store	In ArchiMate an artefact is defined as a physical piece of data that is used or produced in a software development process. This concept and, in particular the permanent artefact in ArchiMate can be mapped to the BPMN data store concept, which represent data that exists independently of a process and that are permanently stored.

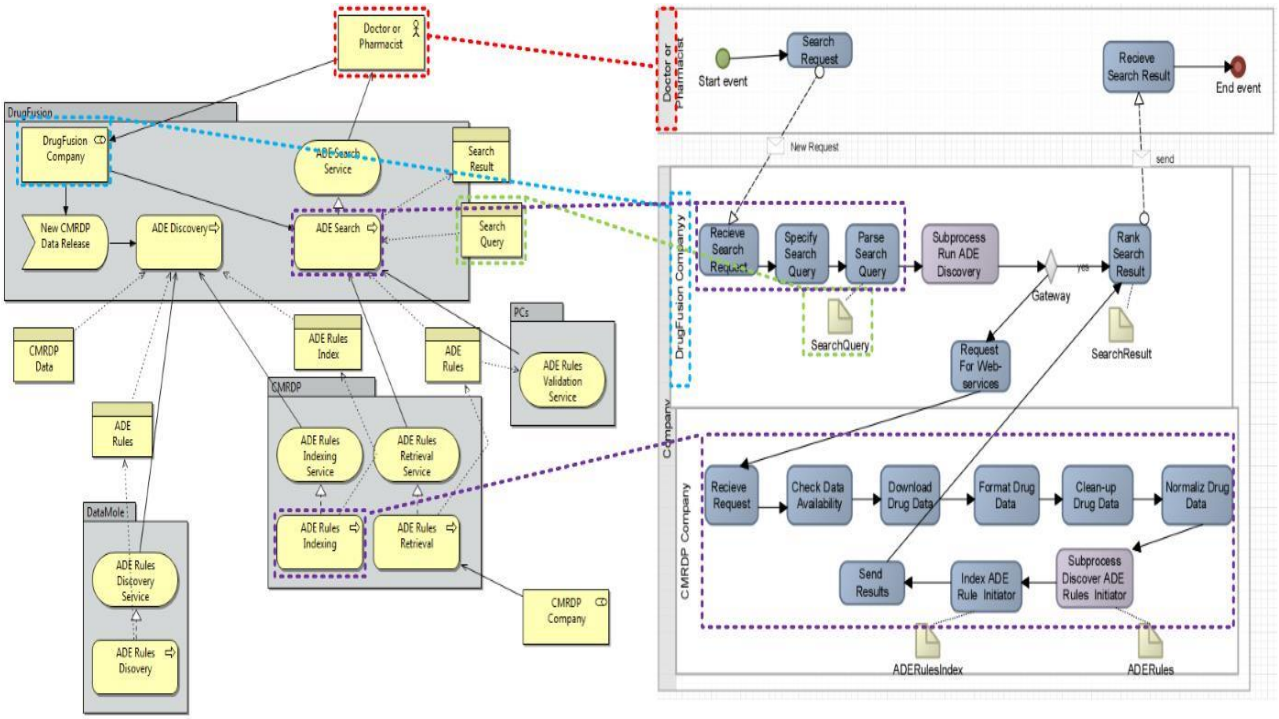

Figure 24: Illustration of Archimate BPMN Integration

The final integration of BPMN with the TIMBUS DIO is shown in Figure 24 based on a concrete example. The dotted lines represent semantically equivalent part in the DIO (left) and the BPMN model (right). The example clearly shows the different view points of both graphical model and their targeted granularity. While the TIMBUS DIO clearly focusses on the architectural context of the process the Business Modelling Notation focusses on the steps necessary to execute the model at runtime.

7.2 PREMIS Mapping

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 [13], 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 25.

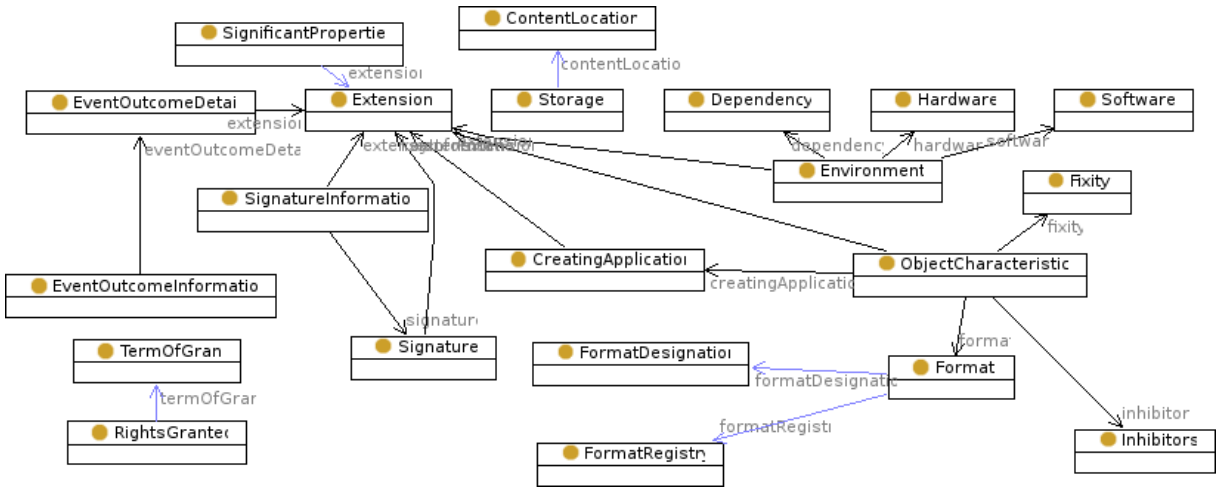

Figure 25: Part of the PREMIS ontology

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 for example, *Storage*, *ContentLocation* and *Software*, as well as *Agents* are mapped to the core ontology. Further, we are currently considering mappings of Rights to both the core and the legal extension ontologies. An overview on the mapping is given in Table 5.

Table 5: PREMIS Mapping

PREMIS Element	DIO Element	Comment
Agent	BusinessActor	
ContentLocation	Location	
CopyrightInformation	Constraint	*Will also be considered for mapping to legal DSO*
Dependency	Artifact	

File	Artifact	
Hardware	Device	
IntellectualEntity	BusinessObject	
LicenseInformation	Constraint	*Will also be considered for mapping to legal DSO and Software DSO*
Object	DataObject	
Software	SystemSoftware	*Could also be to Artifact, depending on the type of Software*
Storage	Node	

7.2.1 Relevant Design Patterns

The main uses of the PREMIS DSO are towards File formats. For this, the following is suggested

- Declare an Artifact Individual to be also of class “File” in PREMIS
- Relate it to an ObjectCharacteristics individual
- Relate the ObjectCharacteristics to a “Format” individual.
- Depending on whether the Format is well known and in a registry, use either a FormatRegistry or a FormatDesignitation to describe the Format.

8 Use Case driven Evaluation

The evaluation presented in this section provides a good snapshot of the TIMBUS context model development. It was done after all use cases were finished modelling the context of the relevant business processes with the updated set of ontologies available after Month 30 (this includes the LegalDSO that was applied already to some use cases).

The goal of the evaluation was not to judge the quality of the instantiations but to identify and prioritize aspects of TIMBUS context model refinements and to validate the overall approach of the "new" context model. We deliberately provide this evaluation "as is", without drawing statistically questionable conclusion as we only were able to analyse a very small sample of implementations.

This evaluation, however, is an important step to give an intuitive, subjective impression on the maturity of the context model that is needed for the sustaining of the ontologies, to plan further developments that need to be done in the scope of the tool development and the use cases and most certainly for the pilots planned in the use cases.

The context model instantiation and the context model ontologies subject to the evaluations are mostly openly available on https://timbus.teco.edu/ ontologies/?r=236 (Versions from 2013-12-01).

Since then further developments have taken place that are not captured by this evaluation, but are partially a direct result of the evaluation or at least backed by the evaluation.

We performed a multidimensional evaluation of the context model. As we did not consider any performance measures and the number of implementations subject to this study is small, this evaluation is purely subjective. We analyzed the Context Model regarding the three in our view most relevant dimension:

1. The technical possibility to capture all relevant preservation context for business processes
2. The ability to represent preservation relevant risks and to analyze them
3. The subjective usability of the approach by preservation experts

In order to analyze the first we used self-reporting by the use case owners. For the second point we analyzed all identified risks in the use cases regarding their representation in the populated context models. For the last point we employed guided interview.

8.1 Subjective Technical Evaluation

In a first questionnaire we ask tool experts to name and describe technical problems occurred while populating instances of the Context Model. For this we considered the use cases of the TIMBUS project from WP7-9. This exercise was done to assess what are the subjectively pressing technical problems.

While some of the problems are "only" training and documentation issues, some other problems, especially concerning missing parts, were explicitly addressed by this deliverable. Some other problems relate to fundamental design decisions of the Context Model and can most probably not be "fixed" without sacrificing

other properties of the Context Model. The list also does not claim to be complete in any sense. Because they are all self-reported in an open fashion the following issue are not an objective or problem oriented evaluation, but overview gives a good insight of practical development problems that exist.

The following list is a complete list of problems reported that were independently reported by the different use cases, edited only for readability and clustered by similarities for this deliverable.

8.1.1 Modelling Tools

Some issues were reported in relation not to the concepts and structures, but the tool support for modelling. While the first aspect can be addressed by hosting a web version of the modelling tools and the third was trivial to fix, the second issue is targeting the appropriateness of using structured meta-data in general.

8.1.1.1 Tool availability in IT Infrastructures

The tools used for modelling are non-standard open source tools that need to be installed locally on a computer. In the case of the modelling of legal ontologies this was restricted by local IT policy.

8.1.1.2 Ontology modelling is non-intuitive for non-experts

The way to model information as graphs or ontology instances is not usual in many domains. In many domains context is captured by written natural language. To transfer this knowledge either someone is needed who understands both concepts, or long discussions are needed.

8.1.1.3 Tools did not work out of the box

Archi-Plugin for converting Archimate model to OWL didn't work when first tried.

8.1.2 Issues related particularly to scaling of manual editing

All issues reported regarding manual modelling target problems related the scaling of the approach when having many entities. The need for tool support in population is mentioned explicitedly.

8.1.2.1 Level of Complexity and Detail in DSpace

In the RCAAP scenario, the infrastructure is quite complex, containing tens of instances of DSpace running in different nodes, which becomes quite troublesome to capture using the Archi tool.

This is not good because it is difficult to approach complex cases starting with the Archi tool. Hopefully, part of the DIO population could be made automatically through extractors.

8.1.2.2 Details in CAD/CAM

In the CAD/CAM scenario, business processes are too detailed, which becomes quite troublesome to capture using the Archi tool. And the context model does not cover the concepts of business process in detail.

This is not good because practically we are not capturing all the details of a business process in CAD CAM scenario.

8.1.2.3 Too many sensors for manual modelling

When trying to populate sensors as nodes: There are a lot of sensors within the LNEC database but not all of them are relevant for the scenario in T8.3 and T8.4. Hence, only some sensors were populated so far.

8.1.2.1 Missing Extractors for eHealth

Archi-model represents BP, infrastructure and application layers. After converting it to OWL, the next step would be populating the OWL with information collected by extractors. For eHealth use case, we need identify: used software packages (and associated dependencies), the hardware landscape (servers, routers, storages …) and licenses.

8.1.3 Issues related to the use of the DIO and OWL

Many issues were targeting general problems of the chosen underlying formalisms. While there is a trade-off in using such formalisms, the problems reported especially regarding OWL need to be taken under serious considerations. The ability to formalize and express the problem domains efficiently needs to be particularly evaluated in the use cases. The comments regarding the Archimate model are debatable and can in our impression be addressed by more documentation and training.

8.1.3.1 General Expressiveness of OWL

In the legal domain much concept is described via axiomatic statements over the whole model, by answering specific questions (true/false). This is essentially different from the open world ontology approach TIMBUS follows (true/do not know).

8.1.3.1 Difficulty to formalize any automatic reasoning

The OWL model represent BP instance and unify BP flow, Software, Hardware, Legal information. However, to make any reasoning queries we need a better human understanding how the complete landscape is looks like.

8.1.3.2 Efficiency of using DIO Concepts vs. Data types

In the CAD/CAM scenario, we have some instance that it would be great to cover them as data properties in DIO instead of instances of concepts. Which we could take advantage of giving value to them just by data properties.

8.1.3.3 Using understanding element relations in Archimate

On a relation level problems occurred choosing the connection to use to connect specific elements within Archimate.

When creating a DIO for the scenario in T8.3/T8.4, it was difficult to understand how to link elements from different Archimate layers with each other

8.1.4 Issues related to the interlinking with DSOs

The concepts of DSOs and DIOs and their interlinking is seemingly not as well understood by the use case owners as hoped. This indicates that probably further tools are needed to assist this process. The last issue should be addressable by open world reasoning as employed by TIMBUS. More experiences will need to be made in including external ontologies.

8.1.4.1 Interlinking Business Processes with Dependencies

We have separate OWL describing BP + DSO instances of (Software, Hardware, and Licenses), it is unclear how to map our dependencies to the business process.

8.1.4.2 Interlinking Sensors with DIO Model

When mapping sensorDSO and WP8 DIO Instances problems occurred figuring out how to map a DSO and DIO with each other. When modelling the scenario with a sensorDSO Instance and a DIO Instance, there is no way to automatically map different ontologies with each other

8.1.4.3 Including the HwDSO

The data properties currently available in the HwDSO use non-conventional data types defined in a separate ontology by the same author. While this couldn't be a problem in terms of knowledge representation, the reasoners available with Protégé are not able to reason on top of this.

This is not good because we will not be able to use commonly available reasoners.

8.1.5 Issues related to the completeness DSOs

As mentioned in the beginning of the document. Refinement is probably a constant process. Concrete ontologies expectedly failed to express necessary aspects in some scenarios.

8.1.5.1 HwDSO Concepts

While being applied to the use case, the Hardware DSO demonstrated to be fairly incomplete in terms of the properties it can capture. Currently supported properties for HW resources are currAvailable, currUsed, and maxCapacity. The Hardware DSO being used is published at http://davy.preuveneers.be/ontologies/2008/01/Hardware.owl

8.1.5.2 SensorDSOs Concepts

Concepts capturing the observation plan, replanning/maintenance/calibration, anomaly detection and model learning on the necessary abstraction level could not be identified

8.2 Risk-driven Evaluation

In a second step for the evaluation we asked all use case owners of the TIMBUS project to assess the applicability of the context model to the risk analysis. For this we presented them with a set of questions that they should answer for every risk identified earlier. Unfortunately the risk analysis tasks of WP7 were still running at this point in time, so that the evaluation presented here is only a snapshot of the work in progress.

For each of the common risks we ask the use case owner to specify:

Relevant Ontology Concepts

The idea is to ensure that the relevant concepts are really present in the ontology and if the ontologies are generally balanced and scoped well. As this is only an early snapshot we expected a clear tendency towards DIO concepts, especially since many of the DSO concepts need to be automatically extracted from tools.

Relevant Ontology Examples

This question most done to double check the presence of those concepts in the concrete instantiation. Some use cases did not report on which could indicate that the reporting is based on future plans rather on the concrete instantiation.

Can be retrieved from Model

For this question we provided multiple predefined answer possibilities that should express the explicitness of the information found in the model:

1. Very Explicit (SPARQL query)
2. Inferable (Algorithm definable)
3. Implicit (no further context needed for human expert)
4. Contextually Implicit (w/ general domain knowledge)
5. Model generally helpful
6. Not Addressed
7. Not sure if addressed

Not all use cases answered using those dimensions also he choice “Very Explicit (SPARQL query)” was surprisingly only answered once (which can indicate the need for analysis and reasoning tools). The participants also did not distinguish between degrees of implicitness. The questionnaire was also made on the assumption that all relevant information could be to some extend linked to the context model, however became clear that “the” context model in this evaluation can only be seen as a snapshot, so that the participants extended the choice of answers in many cases by “not applicable”.

This is why we grouped the answers into only three categories:

- **Risk related resources inferable from Context Model:** contains all explicit and implicit answers above where all information is in the model (1-3)
- **Risk addressed by the Context Model:** where some relevant information or concepts are in the context model (4-5) or 6/7, if a specific concept was mentioned
- **Risk addressed currently not applicable to the Context Model Instantiation:** all other answers

Sensitivity and Specificity

We further asked to the use case owners to judge the accuracy of the information that can be retrieved from the context model with respect to precision and recall. Because we do not have any objective ground truth, we asked for a subjective answer

- **Many instances with low relevance:** Typically high sensitivity, high hit rate but low specificity
- **Perfect subset for decisions:** Both sensitivity and specificity high
- **Few but not Complete:** Good specificity but not ideal hit rate
- **Not yet determinable:** Ground truth not clear
- **Not yet tested**

We have omitted the results of this part of the evaluation entirely because if the question was answered at all it was answered with either “perfect subset” or “not yet tested/determinable”. This seems clear since all information evaluated in this iteration only was populated and no inference or analysis was done.

It seems reasonable to repeat such an evaluation after the refinements done within the scope of this deliverable especially when the set of tools is complete. This, however, would require further iterations by the use cases and can thus not be done within the scope of this task anymore.

8.2.1 WP7 Phaidra

These are the results for the Phaidra use case. This is the only use case analysed here for which no public context model is available. The analysis for this use case was done on a preliminary risk analysis as the risk analysis task was ongoing in this use case. Also the LegalDSO was not applied to this use case yet which is reflected by the results.

Table 6: Risks Inferable from Context Model shows the risk that can be directly related to the context model. In the Phaidra use case those risks is typically related to software/hardware environments. Further risks which currently are not capture but potentially could be captured are related to legal concerns (Table 7). The applicability of the concrete LegalDSO concepts still needs to be validated for the use case. Further concepts that are more related to classical preservation rather than to business process related risks, like data and data model changes are also currently not captured. Socio-economic risks are currently not addressed by the context model instantiation (see Table 8).

Table 6: Risks Inferable from Context Model

Risks	Relevant Ontology Concepts	Relevant Instance Examples

Changes in Service due to change of business model	BusinessService/ApplicationService /InfrastrutureService/Node	FedoraMysl19
Functionality fault due to accidental system failure	BusinessService / ApplicationService / InfrastrutureService / Node / SystemSoftware / Artifact	Debian 7.0
Functionality fault due to hardware unavailability	Node / Device	PhaidraServer
Functionality fault due to software unavailability	Artifact / Systemsoftware	ImageMagick
Functionality fault due to unavailability of core utilities	Network (external Nodes)	
Functionality faults due to environment changes	(Artifact) / Systemsoftware / Node / Device	Debian 7.0
Functionality faults due to system change	Artifact	Fedora Commons; Book Viewer
Hardware unavailability due to local environmental phenomenon	Node / Device (in combination with Location potentially)	PhaidraServer
Loss of data authenticity due to internal or external attack	InfrastructureService, ApplicationService, Artifacts, SystemSoftware	
Software faults due to software obsolescence	Artifact / Systemsoftware	ImageMagick
Software unavailability due to hardware unavailability	Device	PhaidraServer
Software unavailability due to software faults	Artifact / Systemsoftware	ImageMagick

Table 7: Risks currently not captured by the Context Model Instantiation

Risk	Relevant DSO or DIO Concept
Legal liability due to modification using administration rights	LegalDSO

Legal liability due to illicit use of the repository from Phaidra team	LegalDSO
Legal liability due to non-compliance with claims of cancelation	LegalDSO
Legal liability due to non-compliance with retention period	LegalDSO
Non-compliance with retention period due to loss of data	LegalDSO
Reputation loss due to legal liability	LegalDSO
Loss of data due to internal or external attacks	Artifact
Loss of data due to software faults	Artifact
Loss of data integrity due to accidental system failure	Artifact
Loss of data integrity due to internal or external attacks	Artifact
Loss of data integrity due to changes in data model	Artifact
Functionality faults due to changes in data model	Artifact
Reputation loss due to Loss of metadata	Artifact
Shortcomings in semantic understandability due to changes to data model	Artifact
Shortcomings in semantic understandability due to Loss of metadata	Artifact
Functionality faults due to loss of expert knowledge	Artifact
Changes in organizational structure due to change of business model	BusinessRole or BusinessActor

Table 8: Risks currently not applicable to the Context Model

Risk
Change of Business Model due to financial loss
Financial loss due to change of business model
Functionality Fault due to changes in services
Functionality fault due to financial loss

Functionality fault due to internal or external attack
Functionality unavailable due to functionality fault
Increased workload on personnel
Loss of data integrity due to software fault
Loss of Expert Knowledge due to Changes in Organizational Structure
Reputation Loss
Reputation loss
Reputation Loss due to illicit use of repository from user
Reputation loss due to illicit use of the repository from Phaidra Team
Reputation loss due to internal or external attacks
Reputation loss due to loss of data

8.2.2 WP7 Open Source Workflows

For the open source worflows the situation looks similar. Notably the dependency on external systems (not directly modelled by the context model of the use case) is much higher.

Table 9: Risk related resources inferable from Context Model

Risk	Concept	Example
Workflow execution failure due to application dependency fault	Artifact / Systemsoftware	imagemagick
Workflow execution failure due to application or library dependency incompatibility	Artifact / Systemsoftware	Stringutils.jar
Workflow execution failure due to application dependency unavailability	Artifact / Systemsoftware	imagemagick
Workflow execution failure due to library dependency unavailability	Artifact / Systemsoftware	Stringutils.jar
Workflow execution failure due to unavailability of data dependencies	Artifact	WAV File
Workflow execution failure due to supporting hardware faults	Device	Desktop PC x64
Workflow execution failure due to supporting hardware obsolescence	Device	Desktop PC x64
Workflow execution failure due to library dependency faults	Artifact / Systemsoftware	Stringutils.jar
Workflow execution failure due to web service dependency fault	Application Service	http://fue.onb.ac.at/lscapeservices/scape-jhove2200-service
Workflow execution failure due to web service dependency unavailability	Application Interface	http://fue.onb.ac.at/lscapeservices/scape-jhove2200-service
Workflow execution failure due to workflow engine fault	Artifact / Systemsoftware	Taverna 2.4
Workflow execution failure due to workflow engine unavailability	Artifact / Systemsoftware	Taverna 2.4
Workflow execution failure due to unsupported operating system	SystemSoftware	Ubuntu Linux 12.04

Application dependency unavailability due to application license expiration	*Constraint / License*	*Oracle Binary Code License*

Table 10: Risks addressed by Context Model

Risk	Concept
Web service unavailability due to web service dependency authentication failure	*Constraint*
Shortcomings in semantic understandability due to loss or lack of documentation	

8.2.3 WP8 Sensors Use Case

In contrast to the above mentioned use cases the WP8 Sensor/Dam Use Case associates most risks with domain specific concepts. Many of those concepts are however mapped directly to the DIO and are direct subclasses. Still from looking at the evaluation results, it seems that the use case has a much more domain specific view on risks. Notably also risks associated with human resources are modelled in the context model. Here the DIO is used directly (see e.g. Table 11, OR01). Notably none of the legal and licence risks was seen as applicable by the use case owner although specific ontologies exist (see Table 13). Here more training and documentation might be needed. Further most socio-economic risks where not modelled. Also risk like the System Environment Changes that are a core target for TIMBUS where strangely reported as not applicable. Those issues would need to be resolved and investigated in the context of the further exploration and evaluation of the use case.

Table 11: Risks related Resources Inferable from Context Model

Risk	Concept
Shortcomings in Technical Understandability (OR10)	Sensor: Parameters, Sensor:Function
Loss of Archived Information/Data (OR02)	Sensor:Reading, Sensor:HistoricalValueModel,
Loss of Authenticity of Information (OR06)	Sensor:HistoricalValueModel
Loss of Integrity of Information (OR07)	Sensor:HistoricalValueModel
Accidental or Deliberate System Failure (OR12)	Sensor:AcquisitionRate, Sensor:HistoricalValueModel
Destruction or Non-availability of Physical Facilities (OR13)	Location, Sensor:HistoricalValueModel

Software Faults (OR17)	SystemSoftware, Sensor:HistoricalValueModel
Hardware Faults (OR20)	Node, Sensor:Sensor, Sensor:HistoricalValueModel
Hardware Obsolescence (OR21)	Sensor:HistoricalValueModel
Loss or Lack of Metadata (OR03)	Sensor::SensorType, Sensor:AcquisitionDevice, Acquirer

Table 12: Risks addressed by the Context Model

Risk	**Concept**	**Example**
Plan-specific Non-compliance with Obligations of LNEC towards Dam Owner (LR05)	Artifact	Observation plan
Non-compliance with Data Protection Obligations (LR02)	Actor	Acquirer, Expert
Loss of Expert Knowledge (OR01)	Actor, Sensor:Function	Expert
Shortcomings in Semantic Understandability (OR09)	General, All Concepts of the SensorModel beyond the Reading	

Table 13: Risks Currently not Applicable to Context Model Instantiation

Risk
Reputation Loss/Loss of Trust (SR01)
Financial Loss (SR02)
Competitive Risk (SR03)
Non-compliance with General Legal Obligations (LG01)
Non-compliance with Dam Safety Legislation in Relation to the Dam Owner (LG03)
Non-compliance with Dam Safety Legislation in Relation to the Authority (LR04)
Software Licence Expiration (LR06)
Loss of Provenance/Origin (OR04)
Loss of Confidentiality of Information (OR05)
Loss of Reliability (OR08)
Local Destructive or Disruptive Environmental Phenomenon (OR11)
Non-availability of Core Utilities (OR14)
Internal Service Unavailability (OR15)
System Environment Changes (OR16)
Software Obsolescence (OR18)

Software Unavailability (OR19)
Unavailability of Third Party Services (OR22)
Lack or Loss of Processes for New or Modified Sensor Types (OR23)
Lack or Loss of Support for New or Updated Representation Formats (OR24)
Lack or Loss of Support to New or Updated Data Acquisition Systems (OR25)

8.2.4 WP9 Health Scenario

The Health use case also classifies competitive risks as not applicable by the context model. Interestingly many of the legal risks were not mentioned in this early evaluation but has been addressed within the scope of this deliverable (see section 5.4.2). Interestingly the Risk was mostly mapped either to Business Processes or Data Objects (application and business layer). This shows that reasoning on the model will be needed to infer technological entities associated with concrete risks that can lead to potential automatic preservation actions.

Table 14: Risks related to resources inferable from Context Model

Risk	**Context Model Concepts**	**Examples**
Financial Loss	BusinessProcess	DrugFusion
System unavailability	Device, Network, SystemSoftware	DataMole Discovery server
Accidental or Deliberate System Failure	Device, Network	DataMole Discovery server
Software Obsolescence	SystemSoftware	
Hardware Obsolescence	Device	
Software Faults	SystemSoftware	Rules Discovery Engine
Hardware Faults	Device	DataMole FTP server

Table 15: Risks Addressed by Context Model

Risk	**Concept**	**Instance**
Reputation Loss/Loss of Trust	BusinessProcess,	
Non-Compliance with Data Protection Obligations	BusinessProcess,	DrugFusion
Non-Compliance with Licenses and Contracts which exists between the Companies involved in the Business Process	BusinessProcess,	SemanTech, DrugFusion

Liability for Incorrect Information	BusinessProcess,	DrugFusion , DataMole
Liability for Damages	BusinessProcess	DrugFusion
Loss of Expert Knowledge)	BusinessObject	DataMole
Loss of Archived Information/Data	DataObject	
Loss or Lack of Metadata	DataObject	
Loss of Confidentiality of Information	DataObject	
Loss of Authenticity of Information	DataObject	
Loss of Integrity of Information	DataObject	

8.2.5 Discussion

Overall the majority of the risks initially considered can already be represented by the Context Model. However, there is a considerable number of risks that at least the users of the Context Model did not associate with the content they populated. This can be for multiple reasons that need to be evaluated. Either e.g. the tools and documentation are not sufficient so that the usability of the Context Model in this regard was not given. Another possible explanation is the focus of the project: some of the Risk can probably be covered better with other business continuity or preservation tools or manually using non-formalized models (e.g. provenance, economic risks). Thus further integration with existing domain tools seems a good strategy (compare section 7)

The need for DIO and inter-DIO-DSO reasoning can be motivated by this evaluation as in WP7 risks are often associated with the technology layer, while in WP9 concepts from the application layer and business layer are referred. Wp8 uses mostly DSO concepts to describe risks. Generally however, the possibility to automatically infer relevant parts is seen critical for many risks. The evaluation supports that a mix of automatic and manual methods is needed to identify entities that require preservation action.

8.3 Usability Evaluation

The issues with the not-anonymized questionnaire based assessment using open questions were that individual subjective assessments of the usability of the context models are implicitly contained in many of the answers. The quality of the use of the context model is highly dependent on the effort needed, the intuitive understandability and the attractiveness of the overall methodology for the individual user. It is thus important to better understand user experience for the use cases and gather important background knowledge.

This was done in anonymous fashion using structured and then transcribed interviews.

Before we started we identified a number of potentially relevant aspects using multidimensional analysis partially based on assumptions partially based on the self-reported issues:

- **Perceived Effort (Efficiency)**
- **Perspicuity**
  - of the Context Model Concept
  - of the produced outcome
- **Immediate Benefit from the Result (Stimulation)**
- **Perceived Quality (& Dependability)**
  - of the Context Model Concept
  - of the produced outcome
- **Attractiveness**

We thus asked experts from the use cases the following question in an open interview style. In asking the questions we made a trade-off between the accuracy, the understandability and biasing of the questions. It was most important for us that the questions were understood intuitively and that the interviewee could freely express his own view on the question. We also tried not to interrupt the interviewee where possible and just made sure that all questions where addressed when possible. Our expectation coming from other evaluations was that users are typically much more willing to share their views orally especially when treated anonymously that in written questionnaires.

Because the set of people actively working with the context model in the use cases is quite low (with some overlaps between uses cases), the number of interviews was limited. We ended up with 8 interviews covering all use cases of the TIMBUS project (2 of them were not used as the interviewees did not feel suitable to answer most of the questions). 4 of the interviews was done in German language and translated to English for this report. 4 interviews were done in English. Each interview lasted approximately 15-30 minutes. All interviews were conducted remotely and recorded via teleconferencing after informing the interviewee about the fact and the anonymization procedure. The basis for the analysis is the anonymized literal transcription of the interviews.

8.3.1 "How and where did you use the Context Model"

This question was primarily asked to have the interviewee reflect his own work and to give the interviewee a chance to present his background, which was important for an efficient answering of the following evaluation relevant questions. Any answers here are not reported as they cannot be anonymized. We confirmed that the Context Model was used in all use cases. Mostly with few exceptions only the DIO was used by the interviewees.

After asking for reasons why DSOs weren't used comments included:

"Let's say we still had our issues with Archimate"

8.3.2 "How long did it take you to model a typical use case"

It was obvious to us that any answered would not be in any sense comparable in an inter-subjective way. However we deliberately chose the question to give the interviewee a chance express his subjective effort.

"The initial plan you get done within an hour or a half hour. The refinement takes long. However, it is unclear to me if that can be accounted as use case [effort], because we discussed the concepts among us".

Specifically multiple people explicitly mentioned the iterative character in modelling: "The first model had 15 to 20 elements, and then it was extended to 50-60 (although there were no guidelines on what we really needed in there yet). "It was definitively very iterative", another expert states, "I normally do things an iterative fashion, adding more and more things as I go." Most experts agree on rough time frame: "The initial skeleton is done in half an hour but then we do not know much [more than before]."

Especially, it was suggested that the process could be done quicker after an initial learning curve. One person stated that the first model needed to be refined over and over the course of nearly one year. "The [the next anonymized] model was then done in 1-2 hours, because I knew what I should exactly model and what [concepts] were present in [the DIO]."

Another expert specifically commented on the relation of processes and infrastructure and DSO and DIO in terms of effort: "I could say that I used more time spending more time capturing information for the DIO that the DSOs. [The Use Case was] from the process perspective really simple the complexity is at the level of the infrastructure. It took 75% for the DIO and 35% for the DSOs"

Another issue specific effort mentioned in one case was that of communication: "Because we were working with a group of [experts], that they have different terminologies [from ours]: so it was very hard to find examples and individuals out of the use cases so we spent like one month just working on a table [for populating individuals]"

8.3.3 "Did you find the needed concepts and was it clear for you how to use them"

While the technical evaluation already pin-points some potential shortcomings it is difficult to judge the aspects of completeness and expressiveness on a global level (as people try to identify single shortcomings if you explicitly asked for them). As we cannot apply any formal definition in our case and an end-to-end validation was not yet done we rely on the subjective assessment of experts, which we try to address with this simple question.

In this context most people positively commented that it was a parallel process actually refining the context model and populating the ontology.

"With time [I knew what to use]: Especially since I was in involved in [the development of the new context model]. Everything was present in the old model […] it was rather a search for fitting equivalences in cases something was either too specific before or not unambiguous."

“I put much work in myself defining DSOs, because I saw [new concepts were needed] in the use case.” Asked if those new concepts were use case specific: “They were surely not use case specific: we looked for generic concepts that were fitting”.

Explicitly asked again if everything could be mapped: “I do think this is theoretically possible, but we are not done defining DSOs […] for [one DSO] we should be OK by now, but we have no validation step: The step is missing that says this is sufficient for automated processing.” Another expert in the field of ontology engineering comments similarly “actually even until now there is not a certain algorithm or strategy that you can prove, that an ontology is complete or not. And that is a problem.”

“[We made] a prove of concept, that this concept is enough for our case, it's enough for the digital preservation perspective. […] we felt ok, it's more than enough. But for the other use case now [the experts say] some concepts are missing and we should […] add more concepts. So I cannot say that it's complete, because we are seeing in the new use cases that it's not complete.”

Other expert weren’t involved in DSO creation and encountered an other problems arising from their complexity: “As I wasn't involved in it creation, I had to study them to see if they had enough expressivity and after proceeded to populating. “ “For example in hardware DSO there are some classes I didn't find any use for those and there were lots of other classes that was brought in by the imports, that I didn't find use for those”.

Regarding the clarity in using DIO Concepts a few comments were made.

“We are using Archimate for something it was not originally intended for: for technical representations […] There is a gap between intended use and actual use. From a technical perspective an infrastructure function hardly makes sense.”

One problem a person reported was that “we need to still approach those cases and really elaborate some conventions in term of what can I capture using the DIO and what can I capture using some DSOs”.

”We hit some inaccuracies: we discussed for a long time, when something should be concept X and when concept Y matching it against the specification, only to discover after a few month that it is against the specification [how we finally did it]. […] Sometimes it was not clear internally when 3 people were working on the same thing” Someone else pointed out when asked about the ability to model everything (next question): “The problem is rather that Archi allows too much. We can model a database in 15 different ways and degrees of detail […]”

For another it looked different: “In the case of DIO it was pretty simple as I am used to the concept and elements.”

8.3.4 “Were you able to model all relevant parts of the scenario and was it clear what to include and exclude?”

One potential problem of ontologies especially when dealing with such a broad concept like context is scoping. The risk driven analysis was meant to identify potential problems regarding the risk analysis. In order to

understand if some parts of context were left out or included by deliberate decision by the technical expert, we included this question regarding the expressiveness and the completeness of the instantiation.

One expert particularly commented on the scoping of one of the ontologies: “I start with papers and put boundaries, [write down] the concepts […] then connecting them together and then stop. Because if you don't stop and you don't have boundary, it's just expanding and this tree increases and increases. This was another problem we had […] this tree was very, very huge.” “[DP experts were] complaining to me that this tree should be shrinked, but domain experts wouldn't accept, because they would say, […] the deeper the tree goes, the better.”

Comments were targeted towards that the individual scope is also dependent on the use case and domain. “In our initial use case rather yes [regarding when to stop]. In [another use case] this was different because the setting is just broad. The difference was that [both] had initially a very different definition [quality]. The first one was described much more focused from the start.”

Another expert states “I felt that the level of detail that was already there was not enough, […] but of course: what is the depth of information that we need? I couldn't really tell because the reality is that we don't have requirements concerning the depth of information.” The expert explains: “If we already knew […] what kind/what detail of information do we need […] in the future, that would make my work more easy […] otherwise it is only a guess that I need more information or [it] is sufficient. So we are really guessing”.

Others had a more intuitive approach: “I don’t know if it was really complete, but in [the] use case this border became relatively clear, because I just knew there wasn’t anywhere further to I could go.”

8.3.5 “Did you use the instances you created for anything yourself?”

When creating something new one important factor is that it is of immediate use for yourself. Thus it was important for us to understand if there was an immediate benefit or feedback for the person who did the instantiation of the context model.

The interviewees that were only concerned with modelling mostly answered, simply “no”. In three cases the model was used directly in the tools the interviewees were involved the tool development and used their own models to develop and test their tools. In two cases the model was only tool output (no manual modelling was involved at all). None of the interviewees actually yet used other tools on the instances, but clearly had the motivation to do so.

8.3.6 “How happy are you with overall quality of the concepts you used?”

This question and the following questions were meant to express the subjective attitude towards the context model. This subjective impression is typically highly relevant for the output that is created using a tool. Most experts were generally satisfied: “These concepts clearly have their justification, if we clearly specify them: This makes all sense!”

Comments were made regarding the origin of the DIO: “You clearly notice that it originates in business architecture: the business layer is very good. There is nearly nothing, where you say, there are more layers

needed. The technology layer is definitively very scarce […] The application layer contains few information, it is only the interface between business and infrastructure." When asked if that means that the concepts are unbalanced: "you can clearly see that by the fact that [the DIO] has 32 [Concepts] and 16 on the Business Layer"

8.3.7 "If you would need to judge your own outcome: how happy are you?"

This question about the perceived quality of the own created output was not meant to judge different use cases but to assess the satisfaction achieved by user and to identify potential problems regarding finding the appropriate person for creating the model, shortcomings in terms of training. All interviewees answered the question relatively positively. One interviewee responded: "How satisfied? Naturally very much! [Laughs]"

One comment was related to the quality of the documented knowledge: "I want to document an external service but the only thing I know is an interface, what is behind can change all the time". The person commented he really missed mechanisms to capture that fact."

One concern was that "currently our own instantiations are partially somewhat inconsistently modelled"

8.3.8 "Would you recommend the use of the Context Model to a colleague?"

"I do yes and no; [The context model is] an ingenious concept and a major outcome of TIMBUS. It is very sound; [however] we need to define the rules of the game [otherwise] we will have problems if we do not use it consistently." Another expert states "A procedure model is missing, that says where to start and that one can iteratively refine." This expert also has a differentiated opinion "the ontologies, yes." but same time "[Overall] no but not because not because the ontologies, but the whole package still a little immature". The expert gives the following explanations. "What is difficult is the tool support: with more than 60 individuals I don't get further with Protégé."

Another expert confirms the same thing about the Archi editor: "The problem modelling that with Archi becomes really tough as you have to create lots of elements and relationships between those elements and graphically it becomes very cluttered and very complex." Multiple statements were directed towards automatic extraction: "The weak point of this idea is, that we are doing all of this manually. "

Comments were made especially regarding documentation: "Definitively examples are missing, so that you could say: that are the 3 representative examples. Otherwise one is confronted with a whole set of concepts and is overwhelmed."

Generally the use of the DIO based on Archimate was well received and perceived mature: "is very complete with these three layers, business, application and technology, and it would be very helpful, because it's very comprehensive"

9 Sustaining the Context Model

This deliverable concludes Task 4.4 and describes a further step in the development of the concepts/structure, semantics and pragmatics of the context model. The evaluation section shows that there are many dimensions to modelling a Context Model and with each application to use cases, also outside the TIMBUS project; this model can be refined over time as described in this deliverable. Even with more applications to use cases, beyond the scope of the TIMBUS project, there will be always possibilities for improvement. Organizing those improvements along with further applications (which depends on each other) will be the bases for any sustainment strategy. In this section we sketch a first strategy that will be reviewed and adapted in the context of a broader strategy for exploitation (WP2) and dissemination (WP3).

9.1 Distributed Development Strategy

The refinement methods, discussed in this deliverable, along with the structural decisions made in D4.3 are at the core of a distributed development strategy, which decreases many risks associated with ending this task.

One of the most important factors is the assurance of responsibilities and stakeholders. Based on the feedback we collected in this deliverable we have created an initial table that especially identifies the use cases that should take the lead in refinement of the concepts of the Context Model components (and thus will have a high influence on their development). One of the main goals will be the further maturing of the components.

9.2 Standardization

It has been the strategy of TIMBUS to hook up to existing standards like the Open Group Archimate Standard, the CUDF Standard and the PREMIS Standard where ever this makes sense. One challenge for the sustainment will be to continuously maintain the alignment in those components of the Context Model and to communicate improvements towards the standardization activities. With INESC-ID, CMS and DPC we have identified perfect stakeholders for performing those activities.

The LegalDSO is in turn part of a very complicated Domain where standardization is part of the legislative process. Any progress influencing legal standards here will be depending on the ability to influence national and European legislation, which is one of ITM exploitation goals and will be further driven by them.

For other parts like the SensorDSO it was decided to actually perform a very domain specific modelling that fits the use case. TIMBUS deliberately decided against using existing heavy weight standards like SensorML or very technical driven standards like IEEE 1451. For this DSO and other DSOs that have been created on this basis, the benefits of standardization needs be reassessed based on the driving use case (in this cases WP8).

An important activity in this regard is also the further scientific dissemination that was successfully done in multiple parts especially for the DIO and that that is also followed up upon in WP3.

9.3 Maturity

Table 16: Identification of stakeholders and responsibilities

	Maturity*	Short-Term/WP	Long-Term
Archimate Alignment	Defined	ALL	INESCID
SensorDSO	Repeatable	WP8	KIT
LegalDSO	Repeatable	WP9	ITM/INESCID
LicenseDSO	Repeatable	WP9	ITM/SBA
PREMIS Alignment	Repeatable	WP7	DPC/SBA
CUDF Alignment	Initial	WP7	CMS
HwDSO	Initial	WP8	SBA

We have tried to grade the different components regarding the capability maturity model [3] in:

- Initial (chaotic, ad hoc, individual heroics) - a new or undocumented repeat process.
- Repeatable - at least documented sufficiently repeating is feasible.
- Defined - a standard business process using Procedures, Processes, and Work Instructions
- Managed - the process is quantitatively managed in accordance with agreed-upon metrics.
- Optimizing - process optimization/improvement becomes focus of process management

Only the Archimate Alignment has reached a defined state. One of the most important steps will be the evaluation of the use cases which will actually establish the metrics that are needed to reach a managed state.

For all other DSOs it is partially the responsibility of the tool development to establish a defined process around the integration. This is ongoing and could not be assessed in this deliverable. To reach this defined stage will be a precondition for evaluating pilots and thus the context model in a non-subjective fashion.

The usability evaluation suggests that the transition from defined to managed will be the hardest as it will be very difficult in the eye of the experts to objectively assess the completeness and expressiveness of both the

concepts and instances. The end-to-end evaluation done in the use cases hopefully can give important insights that can be generalized.

9.4 Economic Scope of Sustaining

One of the most important aspects of sustaining is the economic interest in improving the models and to ensure interoperability. This will in our view mostly depend on the integration of the context model in commercially or freely available but value-generating tools. Therefore the integration of the context model and the interoperation of the context model as part of these tools is very important for the sustainability and evolution of the concepts. As depicted Figure 26 the context model is the central part of the TIMBUS architecture. All cases where a common exploitation of TIMBUS tools are planned will help the sustainment of the Context Model. This aspect is further presented D2.3 as part of the exploitation plan.

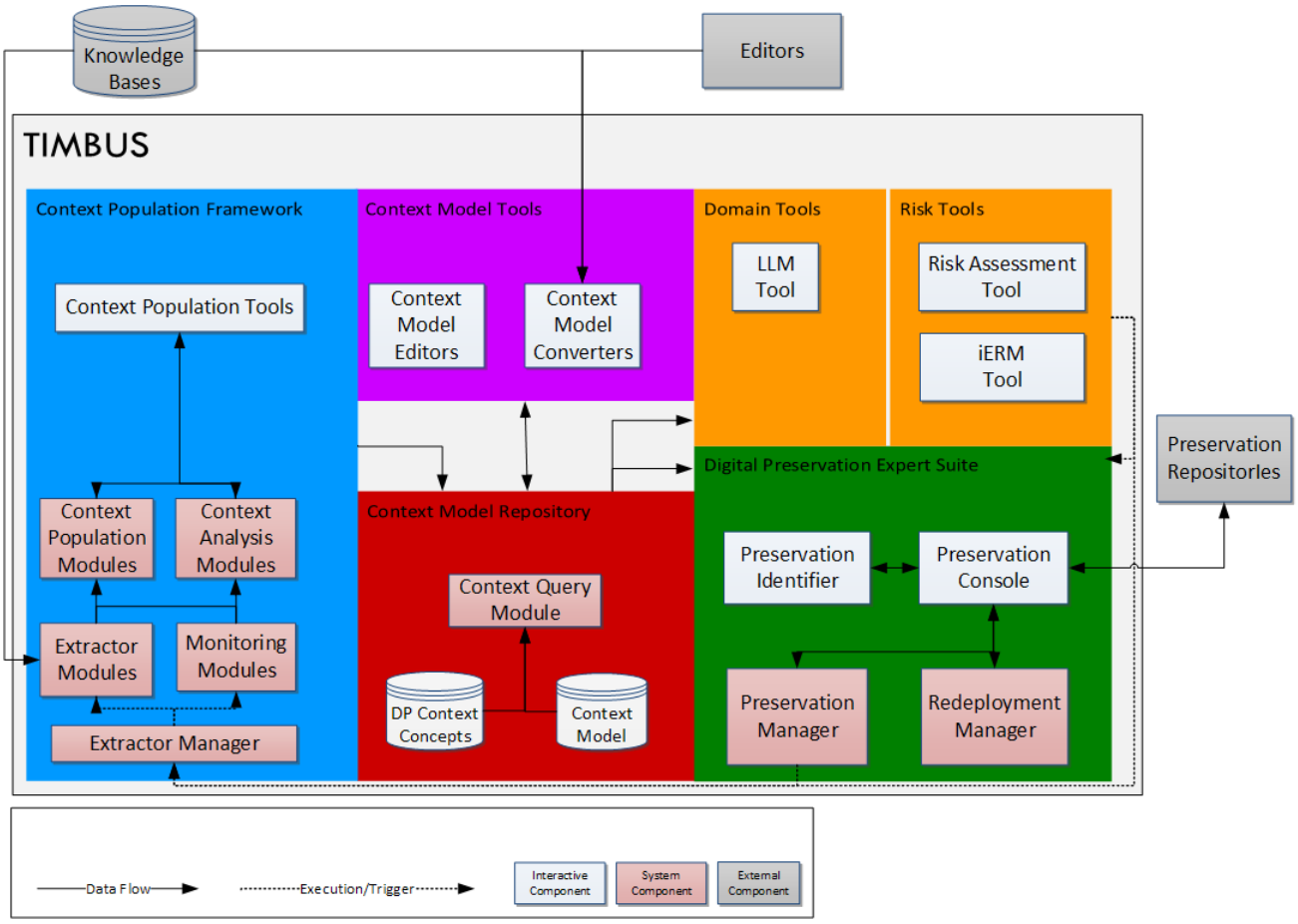

Figure 26: The Context Model as central part of the TIMBUS architecture and tools (simplified)

10 Conclusion and Outlook

This deliverable provides an overview of practical consideration when using and refining the context model. It documents the state of development as of March 2014. As such we have presented refinement strategies, concrete usage examples, a subjective evaluation based on the use cases as well as a sustaining strategy that can build upon the aforementioned.

The evaluation has shown that the DIO build upon the Archimate meta-model which evolved from the IEEE 1471 definition for “software-intensive systems” gives the TIMBUS context model a strong foundations. In places like hardware, sensors or licence we contextually enhance this model using state of the art ontological methods based on OWL semantics.

As TIMBUS is dealing with the whole scale and complexity of enterprise business processes today and in the future it will be impossible to develop a final model to precisely capture all business context. We believe that with the second generation TIMBUS context model we have built a scalable and sustainable basis for a continuous evolution process. The DSOs we have developed until this point cover most parts of relevant context.

However, formative evaluations have clearly shown that for a productive use of the context model, further exploration is necessary within the use cases. Economic and cost aspects are not covered by our research. Here we need to count on external contribution. Many of the further refinements needed will come directly from the TIMBUS use cases as they are evolving. The decision to build upon extensibility by the DSO mechanism proves helpful for continuing the development in a distributed fashion, which can extend beyond the TIMBUS project. The next step that will be the responsibility of each partner, that will be sustaining the developments. After the completion of this task and the project up to date versions of all ontologies and examples will be made centrally available via:

http://opensourceprojects.eu/p/timbus/context-model

References

1. Haren, V. "Archimate 2.1 Specification." (2013). Van Haren Publishing, ISBN: 9401800030

2. *Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., ... & Tu, S. W. (2003). The evolution of Protégé: an environment for knowledge-based systems development. International Journal of Human-computer studies, 58(1), 89-123.*

3. Paulk, M. (1993). *Capability maturity model for software*. John Wiley & Sons, Inc..

4. Antunes, G., et. al. (2012), Timbus Deliverable 4.3, Second Iteration, http://timbusproject.net/component/docman/doc_download/142-43m24dependencymodelsiter2pdf

5. Lankhorst, M. M., Proper, H. A., & Jonkers, H. (2009). The architecture of the archimate language. In *Enterprise, Business-Process and Information Systems Modeling* (pp. 367-380). Springer Berlin Heidelberg.

6. Hoeren, T., Kolany-Raiser, B., Yankova, S., & Hecheltjen, M. (Eds.). (2013). *Legal Aspects of Digital Preservation*. Edward Elgar Publishing.

7. Jin, H., Chen, J., He, H., Williams, G.J., Kelman, C., O'Keefe, C.M. (2008) Mining Unexpected Temporal Associations: Applications in Detecting Adverse Drug Reactions. *IEEE Transactions on Information Technology in Biomedicine, 12(4): 488-500*.

8. Ji, Y., Ying, H., Dews, P., Mansour, A., Tran, J., Miller, R.E., Massanari, R.M. (2011) A Potential Causal Association Mining Algorithm for Screening Adverse Drug Reactions in Post Marketing Surveillance. *IEEE Transactions on Information Technology in Biomedicine 15(3): 428-437*.

9. Huang, Y.T., Lin, S.F., Chiu, C.C., Yeh, H.Y. and Soo, V.W. (2007) Probability Analysis on Associations of Adverse Drug Events with Drug-Drug Interactions. *BIBE: 1308-1312*.

10. Koutkias, V., Kilintzis, V., Stalidis, G., Lazou, K., Niès, J., Durand-Texte, L., McNair, P., Beuscart, R., and Maglaveras, N. (2012) Knowledge Engineering for Adverse Drug Event Prevention: On the Design and Development of a Uniform, Contextualized, and Sustainable Knowledge-Based Framework. *Journal of Biomedical Informatics 45(3): 495-506*.

11. M.S. Fox and M. Gruninger. Enterprise modeling. AI magazine, 19(3):109, 1998

12. Pinto, H. Sofia, Asunción Gómez-Pérez, and João P. Martins. "Some issues on ontology integration." IJCAI and the Scandinavian AI Societies. CEUR Workshop Proceedings, 1999.

13. PREMIS Editorial Committee: Premis data dictionary for preservation metadata. Technical report (2008). http://www.loc.gov/standards/premis/v2/premis-2-0.pdf

14. Moitra, D., & Ganesh, J. (2005). Web services and flexible business processes: towards the adaptive enterprise. *Information & Management*, *42*(7), 921-933.

15. Kolany-Raiser, B. Yankova, S. ,Bakhshandeh, M., Miri, H. Galushka, H. Caetano, A. , and Borbinha J.. (2013) *Towards a Legal Ontology for the Digital Preservation Domain*. Information and Communication Technology and Law - Protection and Access Rights

Appendix A: Revised Version on the SensorDSO

Latest version:

http://timbus.teco.edu/ontologies/DSOs/sensors.html (OWL)

10.1 Abstract

The *Sensor Ontology Specification* provides basic concepts and properties for describing Sensors as a semantic graph. This document contains an RDFa description of the *Sensors Ontology* and some additional information and examples.

10.2 The Sensors Ontology cross-reference: Listing the Sensors Ontology Classes and Properties

The *Sensors Ontology* introduces the following classes and properties.

Classes: | AcquisitionRate | Campaign | Constant | Function | GeoLocation | HistoricalReadingValueModel | Location | Parameter | Quantity | QuantityType | Reading | Sensor | SensorType | Source | StructuralLocation | Unit |

Properties: | hasAcquisitionRate | hasCampaign | hasConstant | hasDate | hasDescription | hasElementName | hasElementType | hasFunction | hasHistoricalReadingValueModel | hasIdentifier | hasLocation | hasName | hasParameters | hasQuantityType | hasRawReading | hasReading | hasResult | hasSource | hasSubElementName | hasSubElementType | hasSubLocation | hasType | hasUnit | hasValue | hasXCoordinate | hasYCoordinate | hasZCoordinate | isAbnormalReading | isNormalReading | isRawReadingOf | isResultOf | isSubLocationOf |

10.3 Classes

10.3.1 Class: sensors:AcquisitionRate

Acquisition Rate - Frequency with which sensor measurements are acquired.

10.3.2 Class: sensors:Campaign

Campaign - A logical set of consecutive Readings defined by an acquisition process / observation plan.

10.3.3 Class: sensors:Constant

Constant - A constant is a fixed Parameter that was defined at a certain point in time.

Sub class of Parameter

Restriction(s):
The property sensors:hasValue must be set http://www.w3.org/2001/XMLSchema#double time(s) and 1 time(s)
The property sensors:hasDate must be set http://www.w3.org/2001/XMLSchema#dateTime time(s) and 1 time(s)

10.3.4 Class: sensors:Function

Function - An abstract mathematical function.

10.3.5 Class: sensors:GeoLocation

Geographical Location - Location relative to defined Structural Location.

Sub class of Location

Restriction(s):
The property sensors:hasYCoordinate must be set http://www.w3.org/2001/XMLSchema#double time(s) and 1 time(s)
The property sensors:hasZCoordinate must be set http://www.w3.org/2001/XMLSchema#double time(s) and 1 time(s)
The property sensors:hasXCoordinate must be set http://www.w3.org/2001/XMLSchema#double time(s) and 1 time(s)

10.3.6 Class: sensors:HistoricalReadingValueModel

Historical Reading Value Model - The (statistical) model of a Sensor aggregated from historical sensor Readings.

Restriction(s):
The property sensors:hasSource must be set 1 time(s) and http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Source time(s)

10.3.7 Class: sensors:Location

Location - An abstract location of a Sensor.

Has sub class Structural Location Geographical Location sensors:StructuralLocation sensors:GeoLocation

10.3.8 Class: sensors:Parameter

Parameter - Any input value for a Function that is not a sensor Reading.

Restriction(s): The property sensors:hasSource must be set 1 time(s) and http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Source time(s)

Has sub class Constant sensors:Constant

10.3.9 Class: sensors:Quantity

Quantity - A measurement quantity of a defined QuantityType, consisting of a value and a Unit.

Restriction(s):
The property sensors:hasUnit must be set 1 time(s) and http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Unit time(s)
The property sensors:hasQuantityType must be set 1 time(s) and http://timbus.teco.edu/ontologies/DSOs/sensors.owl#QuantityType time(s)
The property sensors:hasValue must be set http://www.w3.org/2001/XMLSchema#double time(s) and 1 time(s)

10.3.10 Class: sensors:QuantityType

QuantityType - A type of a Quantity.

10.3.11 Class: sensors:Reading

Reading - An acquisition event of a Sensor measurement.

Restriction(s):
The property sensors:isNormalReading must be set http://www.w3.org/2001/XMLSchema#boolean time(s) and 1 time(s)
The property sensors:hasRawReading must be set http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Quantity time(s)
The property sensors:hasResult must be set http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Quantity time(s)
The property sensors:hasCampaign must be set 1 time(s) and http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Campaign time(s)
The property sensors:isAbnormalReading must be set http://www.w3.org/2001/XMLSchema#boolean time(s) and 1 time(s)
The property sensors:hasDate must be set http://www.w3.org/2001/XMLSchema#dateTime time(s) and 1 time(s)

[#] [http://timbus.teco.edu/ontologies/DSOs/sensors.html - glance]

10.3.12 Class: sensors:Sensor

Sensor - A transducer converting physical phenomena into electrical signals that can be sampled by Readings.

Restriction(s):
The property sensors:hasReading must be set http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Reading time(s)
The property sensors:hasLocation must be set http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Location time(s)
The property sensors:hasType must be set 1 time(s) and http://timbus.teco.edu/ontologies/DSOs/sensors.owl#SensorType time(s)
The property sensors:hasHistoricalReadingValueModel must be set http://timbus.teco.edu/ontologies/DSOs/sensors.owl#HistoricalReadingValueModel time(s)
The property sensors:hasParameters must be set http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Parameter time(s)
The property sensors:hasAcquisitionRate must be set http://timbus.teco.edu/ontologies/DSOs/sensors.owl#AcquisitionRate time(s)

10.3.13 Class: sensors:SensorType

Sensor Type - The type of Sensor.

Restriction(s): The property sensors:hasFunction must be set http://timbus.teco.edu/ontologies/DSOs/sensors.owl#Function time(s)

10.3.14 Class: sensors:Source

Source - A piece of data that is the serialization of Parameters or HistoricalReadingValueModels.

10.3.15 Class: sensors:StructuralLocation

Structural Location - The Structural Location of a Sensor within a dam.

Sub class of Location

10.3.16 Class: sensors:Unit

Unit - The unit of a Quantity.

10.4 Properties

10.4.1 Property: sensors:hasAcquisitionRate

has Acquisition Rate - Associates a Sensor and its Acquisition Rate.

10.4.2 Property: sensors:hasCampaign

has Campaign - Associates a Reading and its Campaign.

10.4.3 Property: sensors:hasConstant

has Constant - Associates an entity and its constants.

10.4.4 Property: sensors:hasDate

has Date - Associates a Reading and its date.

Range: xsd:dateTime

10.4.5 Property: sensors:hasDescription

has Description - Associates an entity and its description.

Range: xsd:string

10.4.6 Property: sensors:hasElementName

has Element Name - Indicates the name of the element of a Structural Location.

10.4.7 Property: sensors:hasElementType

has Element Type - Indicates the type of the element of a Structural Location.

10.4.8 Property: sensors:hasFunction

has Function - Associates an entity and a Function.

10.4.9 Property: sensors:hasHistoricalReadingValueModel

has Historical Reading Value Model - Associates a Sensor and its Historical Reading Value Model.

10.4.10 Property: sensors:hasIdentifier

has Identifier - Associates an entity and its identifier.

Range: xsd:int

10.4.11 Property: sensors:hasLocation

has Location - Associates an entity and its Location.

10.4.12 Property: sensors:hasName

has Name - Associates an entity and its name.

Range: xsd:string

10.4.13 Property: sensors:hasParameters

has Parameters - Associates an entity and its Parameters.

10.4.14 Property: sensors:hasQuantityType

has Acquisition Rate - Associates a Quantity and its Quantity Type.

10.4.15 Property: sensors:hasRawReading

has Raw Reading - Associates a Reading and its raw reading.

Has inverse property is Raw Reading Of

10.4.16 Property: sensors:hasReading

has Reading - Associates a Sensor and its Reading.

10.4.17 Property: sensors:hasResult

has Result - Associates a Reading and its result.

Has inverse property is Result Of

10.4.18 Property: sensors:hasSource

has Source - Associates and entity and its Source.

10.4.19 Property: sensors:hasSubElementName

has Sub Element Name - Indicates the name of the Sub Element of a Structural Location

10.4.20 Property: sensors:hasSubElementType

has Sub Element Type - Indicates the type of the sub element of a Structural Location.

10.4.21 Property: sensors:hasSubLocation

has Sub Location - Associates the Location and its sub location.

Has inverse property is Sub Location Of

10.4.22 Property: sensors:hasType

has Type - Associates a Sensor and its type.

10.4.23 Property: sensors:hasUnit

has Unit - Associates the Quantity and its Unit.

10.4.24 Property: sensors:hasValue

has Value - Associates the Quantity and its value.

Range: xsd:double

10.4.25 Property: sensors:hasXCoordinate

has X Coordinate - Associates the Sensor and the X coordinate of the Structural Location.

10.4.26 Property: sensors:hasYCoordinate

has Y Coordinate - Associates the Sensor and the Y coordinate of the Structural Location.

10.4.27 Property: sensors:hasZCoordinate

has Z Coordinate - Associates the Sensor and the Z coordinate of the Structural Location.

10.4.28 Property: sensors:isAbnormalReading

is Abnormal Reading - Indicates if a Reading is abnormal.

10.4.29 Property: sensors:isNormalReading

is Normal Reading - Indicates if a Reading is normal.

10.4.30 Property: sensors:isRawReadingOf

is Raw Reading Of - Indicates that a quantity is a raw reading of a Reading.

Inverse property of has Raw Reading

10.4.31 Property: sensors:isResultOf

is Result Of - Indicates that a quantity is a result of a Reading.

Inverse property of has Result

10.4.32 Property: sensors:isSubLocationOf

is Sub Location Of - Indicates that an entity is a sub location of another entity.

Inverse property of has Sub Location

Appendix B Legal Conceptual Map

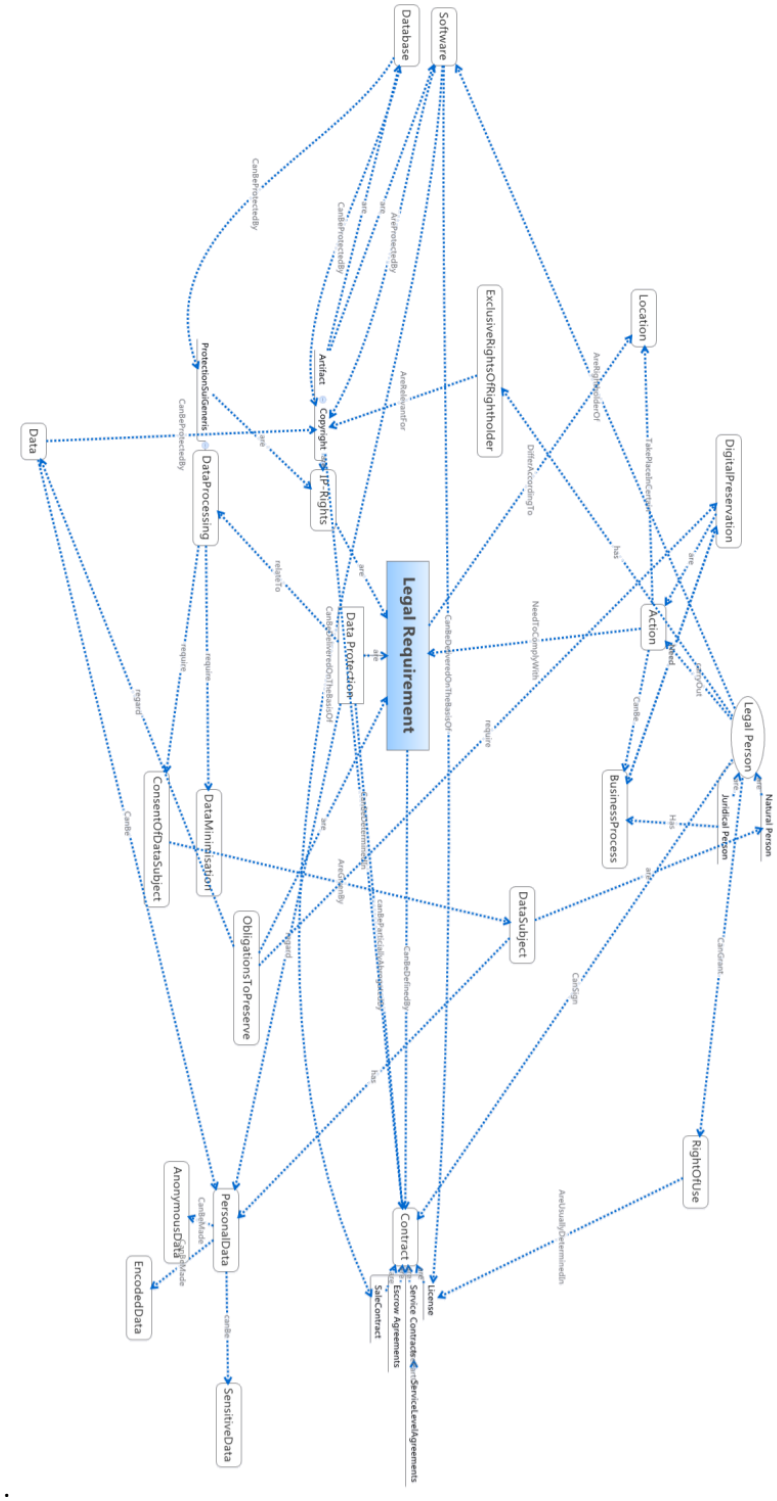

Figure 27: Conceptual map of Legal Ontology Concept and Relations

Table 17: Summary of Relationships between Legal Concepts

Concept	Relation	Concept2
Action	*CanBe*	**BusinessProcess**
Action	*NeedToComplyWith*	**Legal Requirement**
Action	*TakePlaceInCertain*	**Location**
BusinessProcess	*Need*	**DigitalPreservation**
ConsentOfDataSubject	*AreGivenBy*	**Datasubject**
Contract	*CanBeDefinedBy*	**Legal Requirement**
Data	*CanBe*	**PersonalData**
Data	*CanBeProtectedBy*	**Copyright**
Data Protection	*Are*	**Legal Requirement**
Data Protection	*canBeParticiallyAbrogatedBy*	**Contract**
Data Protection	*Regard*	**PersonalData**
DataProcessing	*relateTo*	**Data Protection**
DataProcessing	*Require*	**DataMinimisation**
DataProcessing	*Require*	**ConsentOfDataSubject**
DataProcessing	*Require*	**DataMinimisation**
DataProcessing	*Require*	**ConsentOfDataSubject**
Datasubject	*Has*	**PersonalData**
DigitalPreservation	*Are*	**Action**
DigitalPreservation	*Need*	**BusinessProcess**
Escrow Agreements	*Are*	**Contract**
IP-Rights	*Are*	**Legal Requirement**
IP-Rights	*Are*	**Copyright**
IP-Rights	*Are*	**ProtectionSuiGeneris**
JuridicalPerson	*Are*	**LegalPerson**
JuridicalPerson	*Has*	**BusinessProcess**
LegalPerson	*AreRightholderOf*	**Software**
LegalPerson	*CanGrant*	**RightOfUse**
LegalPerson	*CanSign*	**Contract**
LegalPerson	*carryout*	**Action**
LegalPerson	*Has*	**ExclusiveRightsOfRightholder**
Location	*DifferAccordingTo*	**Legal Requirement**

NaturalPerson	*Are*	**LegalPerson**
NaturalPerson	*Are*	**Datasubject**
ObligationsToPreserve	*Are*	**Legal Requirement**
ObligationsToPreserve	*Regard*	**Data**
ObligationsToPreserve	*Require*	**DigitalPreservation**
PersonalData	*CanBe*	**SensitiveData**
PersonalData	*CanBeMade*	**AnonymousData**
PersonalData	*CanBeMade*	**EncodedData**
PersonalData	*CanBeMade*	**AnonymousData**
PersonalData	*CanBeMade*	**EncodedData**
Sale Contract	*Are*	**Contract**
Service Contract	*Are*	**Contract**