Me4DCAP V0.1

Technical Report

Mariana Curado Malta and Ana Alice Baptista
University of Minho, Algoritmi Center, Portugal
mariana.malta@algoritmi.uminho.pt
analice@dsi.uminho.pt

June 2013

Abstract

This report is framed in a research in progress project that has as goal the development of a method for the development of Dublin Core Application Profiles (Me4DCAP). The development of the first version of Me4DCAP has been published. This paper describes in detail Me4DCAP V0.1, showing the sources used to justify its design. Me4DCAP was based in a Design Science Research methodological approach. It has as starting point the Singapore framework for Dublin Core Application Profiles (DCAP) and the Rational Unified Process; and integrates also knowledge from: (i) software development processes and techniques, focusing on the early stages of the processes that deal with data modeling; and from (ii) the practices of the metadata community concerning DCAP development. Me4DCAP establishes the way through the DCAP development. It establishes when activities must take place, how they interconnect, and which deliverables they will bring about; it also suggests which techniques should be used to build these deliverables.

1 Introduction

The Semantic Web, or Web of Data, has technologies that "enable people to create data stores on the Web, build vocabularies, and write rules for handling data. Linked data are empowered by technologies" [W3C, 2012] that started to emerge in 1999. The Semantic Web is about common formats for integration and combination of data from different sources [W3C, 2012]. Metadata, both in its use and in its definition and description, are present at various levels. Metadata is data that describes resources with information [Press, 2004] and that follows well-defined rules of metadata schemes. A metadata scheme is a set of "metadata elements designed for a specific purpose, such as describing a particular type of information resource" [Press, 2004, p. 4]. The Dublin Core Metadata Initiative (DCMI) created new instruments so that those involved in the definition and use of metadata could speak a common language. These new instruments appeared with the aim to adapt the metadata community to the transformations the Semantic Web brought. The Dublin Core Abstract Model (DCAM) [Powell et al., 2007] appears with this purpose: it is a model developed by DCMI, for DCMI syntax specifications, that presents the components and constructs used in DCMI metadata. One of these constructs used in DCMI is the Dublin Core Application Profile (DCAP), "a generic construct for designing metadata records" [Baker and Coyle, 2009]. The Singapore Framework for Dublin Core Application Profiles" recommendation - [Baker et al., 2008] - defines the rules to build a DCAP. A DCAP is a very important construct to implement interoperability; therefore it is essential to have a method to be able to develop such a construct, in order to give DCAP developers a common ground of work. For the time being, the only guidelines available to develop a DCAP are stated in the Singapore Framework and the DCMI Guidelines for DCAP - c.f. [Baker et al., 2008] and [Baker and Coyle, 2009] - and they are too brief. In fact, a study that we have performed recently shows that show that there is no method to develop a DCAP [Curado Malta and Baptista, 2012]. The absence of guidelines showing life-cycle with standardised activities, as well as a set of well-defined design criteria, with defined techniques, make a DCAP development rather a craft than an engineering activity. Therefore, it is imperative to define a method for the development of DCAP.

Me4DCAP is a starting point for a method for the development of DCAP. Me4DCAP is under construction (c.f. [Curado Malta and Baptista, 2013]) in the scope of a research project that uses the Design Science Research (DSR) methodological approach.

This document is framed in this research in progress. It describes in detail Me4DCAP V0.1. This description establishes the way through the DCAP development, when activities must take place, how they interconnect, and which deliverables they will bring about; it also suggests which techniques should be used to build these deliverables. Me4DCAP defines a way for the construction of each Singapore Framework component.

This document proceeds as follows. The following section presents the DSR methodological approach application. The following section presents a detailed description of Me4DCAP V 0.1 design, showing the sources used to justify its design; throughout this description a case example is presented with the development of the Me4DCAP deliverables, in order to help on the understanding of the Me4DCAP use. We conclude by presenting closing conclusions and future work.

2 Work Methodology

Me4DCAP is framed in a Design Science methodological approach. Sub-section 2.1 presents Design Science and briefly defines our vision about Design Science Research (DSR), and sub-section 2.2 explains how we frame DSR in the Me4DCAP development.

2.1 Design Science

According to March and Smith (1995) natural science is descriptive and explanatory; it tries to understand reality, and is concerned with explaining *how* and *why* things are. “Design science attempts to create things that serve human purposes” [March and Smith, 1995, p. 253] and “aims the development of innovative artifacts that solve real-world problems” [Simon, 1996], thus ”design science is inherently a problem solving process“ [Hevner et al., 2004, p. 82]. An artifact is something that is artificial, constructed by humans [Hevner and Chatterjee, 2010]; artificial ”as opposed to something that occurs naturally“ [Gregor and Jones, 2007, p. 313] [Simon, 1996]. According to March and Smith (1995) design science ”offers prescriptions and creates artifacts that embody those prescriptions“. Design scientists ”rather than posing theories“ - like the natural scientists - they ”strive to create models, methods, and implementations that are innovative and valuable“ [March and Smith, 1995, p.254]. According to Hevner (2007) and March and Smith (1995) a research project with a Design Science methodological approach (DSR) produces artifacts that can be either *constructs*, *models*, *methods*, or *instantiations* of the last 3. An instantiation operationalises one of the 3 possible produced artifacts and ”demonstrate also the feasibility and effectiveness of the models and methods they contain“ [March and Smith, 1995, p.258]. All these artifacts ”are intended to affect phenomena in the real world“ [Purao, 2013, p. 18]. ”Artifacts must improve upon existing solutions to a problem or perhaps provide a first solution to an important problem“ [Hevner and Chatterjee, 2010, p. 6]. Purao (2002) and Rossi and Sein (2003), as cited in Vaishnavi and Kuechler (2012) refer to a 5th type of DSR result: *better theories.* The “construction phase of a design science research effort can be an experimental proof of method or an experimental exploration of method or both” [Vaishnavi and Kuechler, 2012].

Takeda et al. (1990), cited by [Vaishnavi and Kuechler, 2012] present a theory on the reasoning for the knowledge building process in design science. They define a design cycle that “generates understanding that could only be gained from the specific act of construction“ [Vaishnavi and Kuechler, 2012].

According to March and Smith (1995) DSR should live in the interaction of design and natural sciences. The 2 basic activities of natural sciences are: to theorize and to justify. The 2 basic activities of design science are: to build an artifact and to evaluate it. DSR “builds and evaluates artifacts, and it theorizes about these artifacts and justifies” [March and Smith, 1995, p. 256] the theories that the research output (the artifact itself) encapsulates. We need to understand *why* and *how* the artifact worked - those are the theories. We need to justify the theories, that is, gather evidence to test it [March and Smith, 1995].

Hevner (2007) proposes a framework for DSR projects with 3 cycles; these 3 cycles incorporate March and Smith’s view and the cyclic perspective of Takeda et a. (1990):

- the “Relevance Cycle“ that works in the ”Environment”;
- the ”Design Cycle“ that works in the core activities of building the artifact;
- the ”Rigor Cycle“ that works in the ”Knowledge Base” of scientific theories.

In **the Relevance Cycle** the “Environment“ supplies the research project with the needed requisites and the application context, and ”defines acceptance criteria for the ultimate evaluation for the research results“ [Hevner, 2007, p. 89]. In fact the evaluation of the produced artifact is a very important issue in DSR since it is a way to understand if the artifact developed is adequate to the need of the ”Environment”. According to Carvalho (2012), the DSR outcomes validation has 4 elements, that should be able to be checked during the process of evaluation, they are:

- the artifact success measured through its usefulness;
- the artifact generalisation, applicable to classes of situations or, using the term used by Hevner et al. (2004), instantiations;
- the artifact novelty - there is new knowledge when there is a new class of artifact, and finally;
- the explanation capability, that is, “the reasons for the success of the designed objects should be explained”[Carvalho, 2012, p. 5].

The activities of this cycle incorporate the ”build“ and “evaluate“ that March and Smith (1995) refer, the design sciences DSR activities. It incorporates the building activities since it is in this cycle that the design researcher finds the requisites of the application context - it is a part of the design process of the artifact. It incorporates the evaluating activities because it is in this cycle that the design researcher performs the ultimate evaluation of the research results. Among other issues mentioned, the generalisation issue of the artifact must be addressed through the implementation of several instantiations of the artifact in different domains of application in order to prove the artifact generalisation. It is with the generalisation that the design researcher will be able to elaborate new knowledge and extract new theories from it..

In the Design Cycle DSR "calls for multiple iterations (...) before contributions are output into the Relevance Cycle and the Rigor Cycle“ [Hevner, 2007, p. 91]. These iterations are cycles of construction and evaluation, and "these evaluation moments are performed in laboratory and experimental situations“ [Hevner, 2007, p. 91]. In this iterations we can see that Hevner (2007) is integrating in his framework the cyclic activities proposed by Takeda et al. (1990) in order to generate understanding. A DSR project uses as input for this Design Cycle the knowledge base "of scientific theories and engineering methods that provides the foundation" [Hevner, 2007, p. 89] from **the Rigor Cycle**, and the DSR project feeds back the knowledge base of the Rigor Cycle with new artifacts and "the experiences and expertise that define the state of the art in the application domain“ [Hevner, 2007, p. 89] of the DSR research project.

The activities of this Design Cycle incorporate the "build" and "evaluate“ research activities that March and Smith (1995) refer, the design sciences DSR activities. The Design Cycle is where the core activities of the artifact building are performed. And an experimental situation is used to evaluate the design of the artifact and feedback the construction cycle with the results of the experiment. In this cycle we also have a non-formal evaluation called by Vaishnavi (2012) as *micro-evaluation*: "a large number of “micro-evaluations” take place at every design detail decision. Each decision is followed by a “thought experiment” in which that part of the design is mentally exercised by the designer". A more formal and ultimate evaluation is performed, as already mentioned, in the Relevance Cycle.

The activities of the Rigor Cycle incorporate the “theorize“ and "justify“ research activities that March and Smith (1995) refer, the natural sciences DSR activities. In the DSR process new knowledge is incorporated in the artifact produced and in the process of design. This new knowledge has to be identified. The design researcher theorises to elaborate theories and after that he/she must “justify“ the theories produced.

2.2 Application of DSR

Our research project will contribute to produce a first solution to a problem: a method for the development of DCAP (Me4DCAP). Me4DCAP will provide the metadata community with a method that, as far as we know, does not exist. A method is a selection of techniques, the control of their usage and the integration of the obtained partial results [de Almeida and Pinto, 1995]. According to Nolan (1973), cited by March and Smith (1995) "a method is a set of steps (an algorithm or guideline) used to perform a task. Methods are based on a set of underlying constructs (language) and representation (model) of the solution space" [March and Smith, 1995, p. 257].

We will use Hevner's framework, together with March and Smith's and Takeda et al.'s view to describe our DSR methodological application.

In the Relevance Cycle the "Environment" in our DSR project is the metadata community, which supplies the application context and the requirements for the development of our artifact. These requisites were obtained:

- from a study that we developed in order to find out what were the techniques and methods that the metadata community used to build Application Profiles (c.f. Curado Malta & Baptista (2012)):
  - IMS Global Learning Consortium (2005)
  - Friesen et al. (2002)
  - DCMI (nd)
  - Buonazia and Masci (2007)
  - Chen and Chen (2005)
  - BSI (2005)
  - Onyancha et al. (2001)
  - Agostinho et al. (2004)
  - Marzal García-Quismondo et al. (2006)
  - Salokhe et al. (2008)
  - CWA (2006)
  - Eadie (2008)
  - de La Passadière and Jarraud (2004)
  - Nilsson (2008)
- from the analysis of semi-structured interviews conducted to developers of 3 of the only 5 DCAP existing so far (identified in a study to be published):
  - DRYAD [Carrier, 2008]
  - VMAP [Iglesias et al., 2009]
  - SWAP [Allinson and Powell, 2006]

- and from the analysis of the documentation of these 5 DCAP:
  - IAP [Eadie, 2008]
  - DRYAD [Carrier, 2008]
  - SWAP [Allinson and Powell, 2006]
  - TBM AP [Calverley and Johnston, 2009]
  - VMAP [Iglesias et al., 2009].

Concerning the evaluation of our artifact: the artifact is a method, the evaluation of such an intangible artifact can not provide objective metrics for a quantitative evaluation, like a software artifact e.g can. We need to use a subjectivist approach in order to use qualitative techniques. Our goal is to understand how well does our method works [March and Smith, 1995]; since it is a new method, we can not use the comparison approach as e.g Shovak and Even-Chaime (1987) used to compare 2 different methods for designing a data base schema. We need to:

- find evaluation techniques used in the methods engineering community, and adapt the chosen technique or techniques to our needs;

- develop particular metrics for the particular environment where the artifact is going to be evaluated.

This PhD project has time constraints; the DSR project will probably continue, framed in another context, a more deeper evaluation will be done in future work. The DSR outcomes validation with its 4 elements is still subject to some reflexion, but some thoughts can already be presented:

1. the artifact success measured through its usefulness - how can we prove our method usefulness?;

2. the artifact generalisation - we will look for ways to develop some instantiations of our method through field testing with other partners;

3. the artifact novelty - there is new knowledge when there is a new class of artifact, in our case, our new method is a new class of artifact in the metadata community, and finally;

4. the explanation capability - this validation element, which is directly connected to the element (1) validation, should be defined in the end of the DSR process when all the other elements are set.

For our PhD project, we will use as last resource in order to have some kind of evaluation in the project, the Focus Group (FG) technique. The first version (V0.1) of the artifact is going to be evaluated using 2 FG[1], one with a panel of metadata specialists, another with a panel of software development process specialists. The contributions from the FG will be used to tune a Me4DCAP V0.2. This work should be finished by December 2013.

In the Design Cycle:

1. (construction moments) the artifact is based on 3 types of knowledge (see Figure 1):
   (a) the Knowledge Base of the Rigor Cycle (see next point for details) on software development processes and techniques, focusing on the early stages of the processes that deal with data modeling: a literature review was done in order to identify the approaches that best serve our case;
   (b) the requisites of the Relevance Cycle (see previous point for details), that is, the information gathered in the study Curado Malta and Baptista (2012);
   (c) the analysis of the semi-structured interviews. In figure 22 we show in detail the sources that were used to define every component of the artifact (method).

   Our work is based on the Singapore Framework for Dublin Core Application Profiles (c.f. Baker et al. (2008)) and on the DCMI guidelines for Dublin Core Application Profiles (c.f. Baker and Coyle (2009)). The starting point in Knowledge Base is the Rational Unified Process (RUP) [Kruchten, 2004], as it is a method that goes deeply into all software development phases and is, to the best of our knowledge, the only one that exists in these patterns. RUP was developed 20 years ago (in the nineties) assimilating all the best software development practices, and presenting them in a way that makes it possible for them to adapt to a vast set of situations, projects or organisations [Kruchten, 2004]. RUP is used in more than 10000 organisations, in projects both big and small [Borges, 2008], thus showing to be a consistent mature method. RUP is tuned to our needs - which is perfectly in its goals.
2. (evaluation moments) We use an experimental situation to be able to evaluate our artifact. We have been working for 2 years with a group[2] of the world community of Social and Solidarity Economy (SSE). After a study of the requirements, the internal and external constraints and of the environment, we came to the conclusion that there was no

[1] According to Tremblay et al. (2012) this is the minimum number a DSR project can do.

[2] A group within RIPESS - http://www.ripess.org - accessed in 19 Feb 2013

DCAP that could be used for this community, therefore it is necessary to develop one [Curado Malta and Baptista, 2012]. We continue to work with the SSE group (they are our experimental situation) building with them a DCAP-SSE, and using our artifact (the method) in this process. We feed back the construction moments with the outputs of this experiment.

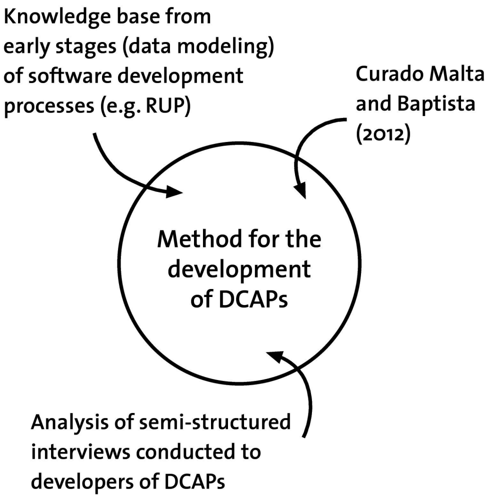
Knowledge base from early stages (data modeling) of software development processes (e.g. RUP)
Curado Malta and Baptista (2012)
Method for the development of DCAPs
Analysis of semi-structured interviews conducted to developers of DCAPs

Figure 1: The 3 inputs for the construction of the method in the Design Cycle

From the Rigor Cycle we have as input for our DSR project the knowledge base on software development processes and techniques:

- Methods:
  - Analysis and Design of Web based Information Systems [Takahashi and Liang, 1997]
  - Evolutionary, Object-oriented Software development [Hesse, 2003]
  - Relationship Management Methodology [Isakowitz et al., 1995]
  - Rational Unified Process [Kruchten, 2004]
  - Web Site Design Model [De Troyer and Leune, 1998]
  - Methontology [Fernández-López et al., 1997]
- Techniques:
  - Object Modeling Technique [Rumbaugh et al., 1990]
  - Object Role Modeling [Halpin, 1996]
  - Unified Model Language [Fowler, 2004]
  - Entity-relationship model [Chen, 1976]

This Rigor Cycle will have as input from the DSR Project the theories that come out of the DSR process. To elaborate the perspective theories that will be implicitly in this PhD work we will follow the framework defined by Gregor and Jones (2007). The new method - Me4DCAP - is also new Knowledge added to the Knowledge Base, so input for the Rigor Cycle. In the phase “Scope Definition” work planning initiates, its goal is to define DCAP application scope and to organise the work team. In this phase it is also where it is developed part of the Functional Requirements Component Stage 1. However, the development of the latter is not tight to this phase and overflows to the next one, Construction. In this phase, the Domain Model Component Stage 2 is developed; however, the development of this Component Stage is not, as happened before, tight to this phase and overflows to the next phase, the Development phase. In the Development phase the DSP Component Stage 3 is built. It the climax of all construction done until this moment, since the DSP Component Stage 3 development work is based on the Domain Model Component Stage 2 previously constructed and it is the Component Stage that defines the DCAP in its entirety. Finally in the Validation phase, the developed DCAP is validated. The 2 Guidelines Component Stage 4 and Component Stage 5 are developed throughout the Construction, Development and Validation phases.

3 Me4DCAP description

3.1 Me4DCAP approach

Me4DCAP has as starting point the Singapore Framework for Dublin Core Application Profiles (c.f. Nilsson et al. (2008)). According to the Singapore Framework, a DCAP is composed by:

- Functional Requirements (Component Stage 1);
- Domain Model (Component Stage 2);
- Description Set Profile (Component Stage 3);
- Usage guidelines (optional) (Component Stage 4), and;
- Syntax guidelines (optional) (Component Stage 5).

The starting point in the Knowledge Base is the Rational Unified Process (RUP) (c.f. Kruchen (2004)), as it is a method that goes deeply into all software development phases and it is widely used by the community of software development. RUP was developed 20 years ago (in the nineties) assimilating all the best software development practices, and presenting them in a way that makes it possible to adapt these development practices to a vast set of situations, projects or organisations [Kruchten, 2004]. According to Kroll and Kruchten (2003), cited by Borges (2008), RUP is used by 10 000 small and big companies all over the world. This number shows the importance of RUP software development process in the software engineering community. This fact together with the fact that exists a group of RUP specialists in the research lab this project is based on[3], made us to choose RUP as Me4DCAP definition starting point.

The development of a DCAP is an iterative process by stages, each stage being built on the results of the previous stage. This set of stages is the starting point for the definition of Me4DCAP. Me4DCAP defines a way for the construction of each component of each Singapore Framework stage. The components of each stage are called *Component Stage* and each one is identified by a number.

Based on the RUP approach (see figure 2), tuning it to our needs - which is perfectly in its goals - Me4DCAP has 4 phases (see figure 3): Scope Definition, Construction, Development and Validation. These phases are traversed along the project development as the Singapore Components Stage are being developed.

[3] See Algoritmi Center at University of Minho: `http://algoritmi.uminho.pt`

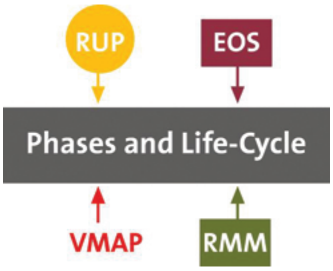

Figure 2: Sources for the Me4DCAP V0.1 phases and Life-Cycle definition

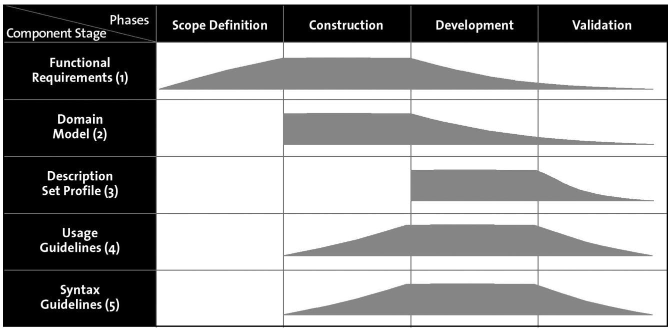

Figure 3: The Me4DCAP V0.1 phases

3.2 The DCAP development work team

Me4DCAP defines 4 types of stakeholders that interact in the DCAP development process: Managers, System Analysts, Metadata Programmers and Final Users. By Manager Me4DCAP means a manager of an organisation that has a Web system that has implemented or will be implementing the DCAP in development. By System Analyst Me4DCAP means a specialist that has technical skills in data modeling and in requirements elicitation, this person should also have some skills of group management. A Metadata Programmer is a specialist in metadata that can use the DSP [Nilsson, 2008] and RDF [W3C, 2010] languages, and understands the Semantic Web concepts. By Final User Me4DCAP means a user that works with the Web system that has implemented or will be implementing the DCAP in development. It should be noted that the multidisciplinary team is very important and should be respected for the success of the task of developing a DCAP.

3.3 Me4DCAP Life-Cycle development model

Throughout a DCAP development, deliverables are being produced to help to reach the aforementioned Component Stages. Figure 5 shows Me4DCAP life-cycle development model mentioning the artifacts that have to be produced and when they should be produced. The Me4DCAP life-cycle development model is iterative. As it is not possible to define all the requirements at the beginning of the DCAP development process, during its development one may feel the need to go back to the previous stage to add missing requirements. These iterations are at the end of Block 2 to Block 1; at the end of Block 4 to Block 2, or Block 1; at the end of Block 6 to Block 1, and at last at the end of Block 7 to Block 1, which presupposes a new iteration of the whole process. The number of iterations of the whole process depends on the dimension and complexity of the DCAP to be developed. Iterations will end when there is nothing new to "discover" or to add, as for requisites, depending on the results of the validation in laboratory and in production (see section 3.7).

The life-cycle iteration approach is based on RUP (see figure 2), as well as on the DCAP VMAP [Iglesias et al., 2009], on the methods EOS [Hesse, 2003] and RMM [Isakowitz et al., 1995]; they all used also iterative approaches on their life-cycle's development processes.

In order to develop a DCAP, the DCAP developers will need to follow the life-cycle of the development process, building the deliverables that will be used for the construction of the Component Stages (see figure 4). Some deliverables can be developed at the same time; that is the reason for them to be together in the same block. But some can't be built before others because they need information that comes out of the previous deliverables. Figure 5 shows the dependency between the deliverables.

3.4 Starting the DCAP development

The development of a DCAP is a collaborative building process, and the first step of such a process is to define the role of each member of the team, and agree on the rules of the group in order to respect everyone's view. If the work-team is geographically separated, collaborative tools should be defined as well as the rules to use them. This tools can be: audio/video conference (e.g. Skype), wiki pages (e.g. `http://dokuwiki.org`) or any other kind of tool (e.g. `http://www.pontaopad.me`) to register thoughts in a collaborative way.

Since the work-team is composed by persons from different backgrounds having different skills, it is very important to build a Glossary. This should be done from the beginning of the DCAP development process. A Glossary is a text document with the keywords (and its description) used in the DCAP. A Glossary is used to define important words commonly used by the work team while constructing the DCAP. In multi-disciplinary teams it is important that

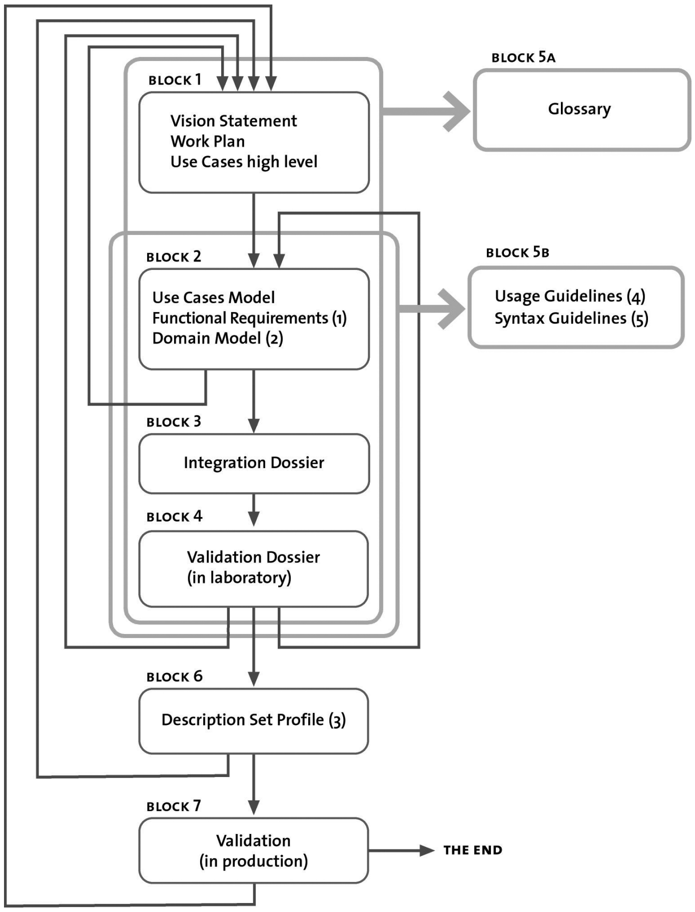

Figure 4: The Me4DCAP V0.1 life-cycle development model and the Me4DCAP V0.1 deliverables sequence

members of the work team speak all a common language, as it avoids misunderstandings and improves communication [Jacobson et al., 1999]. This document is build throughout the whole DCAP development process and is mandatory. Figure 6 shows the sources that also used a Glossary in their development processes: the Methontology [Fernández-López et al., 1997] method and the Rational Unified Process (RUP) [Kruchten, 2004].

After defining the work team rules and how the work should be organised, work begins. Next section will guide us through the DCAP building process, showing us the Me4DCAP deliverables and the techniques that can be used to develop them. It will also shows in detail the sources used to justify Me4DCAP design (see appendix A – figure 22 for a full view of the sources); Me4DCAP follows the Singapore Framework Component Stages order; they are the center of all development.

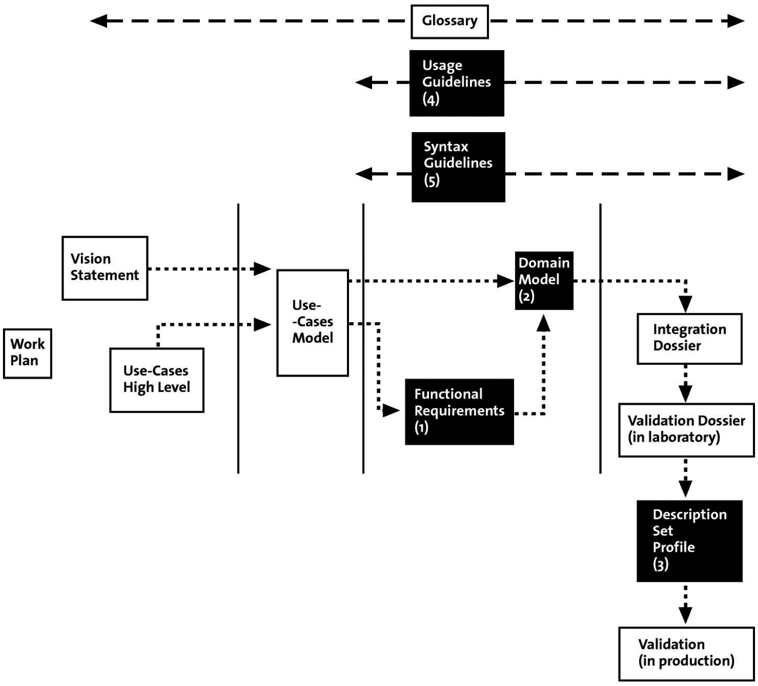

Figure 5: Me4DCAP V 0.1 deliverables dependency

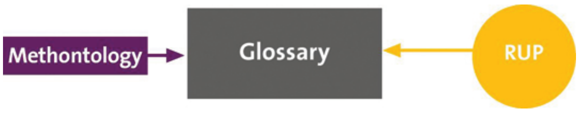

Figure 6: Sources for the Glossary artifact definition

3.5 Functional Requirements (Component Stage 1)

To build the mandatory Singapore Framework Component Stage 1 - Functional Requirements Me4DCAP defines the need to develop a set of 4 mandatory deliverables:

- the Work Plan;
- the Vision Statement;
- the Use Cases High Level;
- the Use-Cases Model.

These 4 deliverables are defined by RUP as a way to develop Functional Requirements. Also, the WSDM method [De Troyer and Leune, 1998], and the TBM AP DCAP [Calverley and Johnston, 2009] used also a Vision Statement artifact to define Functional Requirements (see figure 7).

Example: Craftsman's Catalog

Example Description

We are going to use a very simple example to explain the development of some deliverables of Me4DCAP. We will present a Vision Statement, a Use Case High Level, the detailed description of the Use-Cases of the Use Case Model, The Functional Requirements, the Domain Model, the ORM Diagram Data Model, the Integration Dossier and part of the Validation Dossier for this example. As we will see the other deliverables defined in Me4DCAP (Work Plan, the Glossary, the UML Use-Cases diagrams of the Use Case Model, parts of the Validation Dossier in laboratory and the Validation in production, the Usage Guidelines) will not be presented since they have to do with the dynamics of the working group and with very contextual situations that can not be simulated in the context of this document.

The example is as follows:

> A world community of craftsman (social and solidarity economy) has several different Web Based Information Systems (WIS) all over the world. This community wants to have the possibility of some of their WIS information to be available for the function of discovery on the Semantic Web: they want to share the information between them and have it also available for other WIS outside the community.
>
> They decided to develop a DCAP.

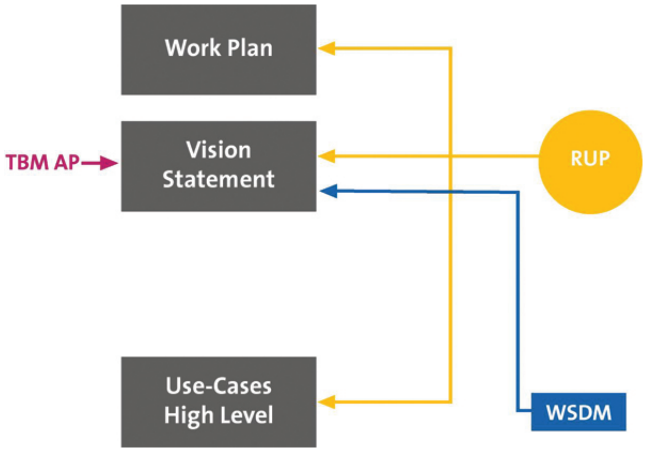

Figure 7: Sources for the Work Plan, Vision Statement and Use-Cases High Level deliverables definition

The first deliverables to be build are the Work Plan, the Vision Statement and the Use Cases High Level. After that, follows the Use-Case Model.

The Work Plan has as goal the time planning of the project activities; it is the follow up project timing and serves as a guide for the work team of the DCAP development project. The Work Plan refers the timings of each phase as the respective beginning and ending dates, and still the dates when each Component Stage should be ready. It will also be possible for the Work Plan to include information on the responsibilities of each element of the work team in the phase or deliverable in question. The Work Plan is a text document, a Gantt Chart or any other type of graph or scheme that the work team finds more convenient. The Work Plan should be built by all members of the work team, and negotiated among them, in order to fit all work team members time constraints. It is acceptable that the Work Plan has to be modified as the project evolves.

The Vision Statement is a document that shows what developers want to reach with the DCAP development. It defines the scope of the DCAP; it is a simple plain text document with no more than 200 words, describing the boundaries for the DCAP usage and what the work team wants to accomplish with the DCAP development. The technique used to develop the Vision Statement should be the brainstorming technique, where all the members of the team should feel free to write ideas on a board (physical board or web tool), followed by a discussion. In the end, the set of ideas chosen should be organized in simple sentences.

Follows the Use-Cases construction. Use Cases "offer a systematic and intuitive means of capturing functional requirements" [Jacobson et al., 1999, p. 37]. "Each use-case must include details about what has to be done to achieve its functionality" [Schneider and Winters, 2001, p. 27]. The Use-Cases will be used to develop the Functional Requirements and to understand the objects (and their properties) of the system to be studied. All the DCAP studied: TBM AP [Calverley and Johnston, 2009], [Allinson and Powell, 2006], VMAP [Iglesias et al., 2009], Images Application Profile (IAP) [Eadie, 2008] and DRYAD [Carrier, 2008], the method ADWIS [Takahashi and Liang, 1997], and the following literature [IMS Global Learning Consortium, 2005], [Friesen et al., 2002], [DCMI, nd] and [Buonazia and Masci, 2007] refer the need to build Use-Cases Models to help on the construction of Functional Requirements (see figure 8).

The Use-Cases Model is build in 2 steps:

1. A Use Cases High Level is built. The Use Cases High Level is a list of the Use-Cases to be described later in the Use-Case Model deliverable. This Use Cases High Level is a description, per Use-Case, with no more than 2 lines. The work team should discuss what kind of information they want to share; which are the actions a user needs to do to obtain

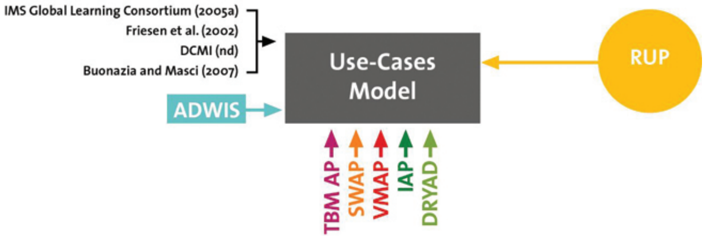

Figure 8: Sources for the Use-Cases Model definition

that information. Every kind of information is a Use-Case that must be detailed in the Use-Case Model, and that must be explained in 2 lines for every Use Case High Level. The technique used to develop this deliverable is the same as the Vision Statement deliverable;

2. A Use Cases Model is built.

The Use-Cases Model is built after the 3 deliverables (Work Plan, Vision Statement and Use Cases High Level) previously described are developed. But, having Me4DCAP an iterative life-cycle development model, the previous deliverables might have to be revisited more than once; there will be moments the work team will have to decide to release a draft version of the deliverables to follow the process, being aware that they will be working on those draft versions sometime later in the process.

The Use-Cases Model is composed of:

- the UML Use-Case diagram with the actors that interact in the Use-Cases, describing the functionality of the system [Kruchten, 2004];
- the set of all detailed Use-Cases.

For information on how to build an UML Use-Case diagram see Fowler (2004) or Jacobson et al. (1999).

Every Use-Case should be then documented in detail. This documentation should set the sequence of actions - a specific sequence of events that happen in the system - that a system performs to bring added value to a specific actor. An actor is somebody or something (automata) outside the system that interacts with it [Kruchten, 2004]. An Use Case detailed description is a flow of events description, and it should be developed using the template proposed by Schneider and Winters (2001). Every manager member of the work-group will know precisely which are the needs of the system in order to achieve certain objectives of functionality; they should be the persons to identify what are the actions that will bring value to the system.

So, the Use Case description should be developed by the managers, giving them the template of the flow-of-events defined by Schneider and Winters (2001) and a definition of Use Case (defined in the Glossary). The set of Use Case descriptions should be written on the board (physical board or web tool), and the work-team as a whole should revise them, with the System Analysts members of the work-team helping managers to clarify ideas.

After having the previously described 4 deliverables, the Functional Requirements can be built. As it can be seen in figure 9, again RUP is the source for Me4DCAP design in what concerns the need to build functional requirements in a software development process. Also all the studied DCAP have defined Functional Requirements (it could not be differently otherwise they would not have been called DCAP since it is a mandatory Singapore Framework deliverable). Also, other literature refers the need to build such a deliverable: [Chen and Chen, 2005], [BSI, 2005], [IMS Global Learning Consortium, 2005], [CWA, 2006], [de La Passadière and Jarraud, 2004] and [Eadie, 2008], and the DCMI Guidelines (c.f. Baker and Coyle (2009)).

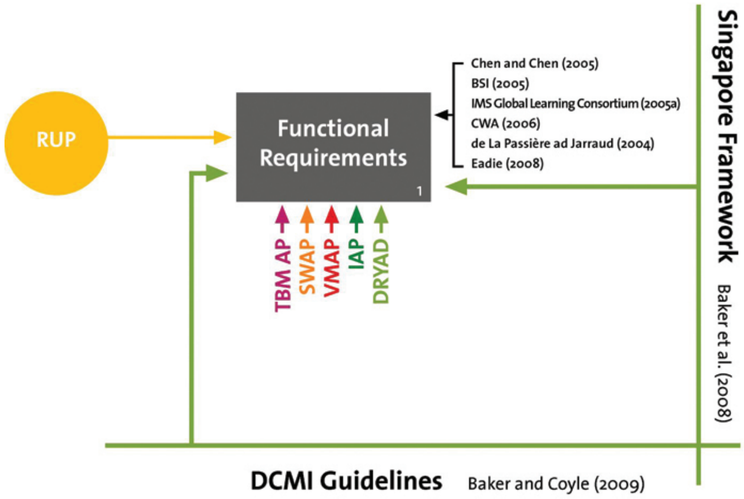

Figure 9: Sources for the Functional Requirements Component Stage 1 definition

Functional requirements "guide the development of the application profile by providing goals and boundaries and are an essential component of a successful application profile development process. This development is often a broad community task and may involve managers of services, experts in the materials being used, application developers, and potential end-users of the services" [Baker and Coyle, 2009]. The Functional Requirements Component Stage 1 is a text document, where general goals are mentioned as well as specific tasks [Baker and Coyle, 2009].

To develop the Functional Requirements the work-team should read, in group, the Use Cases detailed descriptions to identify which are the functional requirements that the use cases explicit. Short sentences should be used, and should be written on the board (physical board or on the working web tool). After that, the work-team should identify if there are no repeated ideas of functional requirements on the board. Each functional-requirement-idea should be developed more deeply; since certain ideas speak more to some work team members than to others each functional-requirement-idea should be distributed accordingly in order to be developed by the work team member more capable to do so. In the end of the process, the complete set of functional-requirements-ideas described deeply should be placed together on the board (physical board or web tool), and the whole group should discuss and review the final result.

Once developed the Singapore Framework Component Stage 1 - Functional Requirements, Me4DCAP follows to the Singapore Framework Component Stage 2 - Domain Model development.

Example: Craftsman's Catalog

Vision Statement

Our community wants a system that will allows us:

- To share the catalog information among the world community of craftsmen;
- To share the catalog information with other e-commerce web sites;
- To provide aggregate data about our worldwide dimension.

The DCAP will give more strength to the commercial bonds of our community. It also will:

- support the sharing of information;
- open doors to the open world of the Semantic Web;
- provide information about the world dimension (statistics) for lobbying in the Governmental Institutions;
- to show that the craftsmen community sales politic is based on the labor and not on the profit by providing the profit of every product.

Use Cases High Level

Case 1

to look for a product by type or category, price, region (city or country)

Case 2

to have access to statistical information and statistical aggregated information.

Case 3

to locate craftsmen (their product's information and shop's location) in a specific city, region or country.

Example: Craftsman's Catalog

Use Cases Model: detailed Use-Cases
Case 1
A user[a] looks for a product. The user:

1. enters the type of product (or category of product);
2. from the result obtained, wants to see the products in a range of prices;
3. from the result obtained, wants to look in a region or city.

Note: the search order does not need to be in this specific order.
Case 2
A user looks for information:

1. on the percentage of products in a category in the the world community (region or city or country).
2. the average age of the world community of craftsman (region or city or country);
3. the % of men and % of women working in the world community of craftsman (region or city or country);
4. the annual income of the world community of craftsmen (region or city or country)

Note: the search order does not need to be in this specific order.
Case 3

1. A user wants to locate craftsmen in a specific location: region or city or country.
2. for every craftsman, the user wants to know:
    - which types of products he sells;
    - the prices of the products and the decomposition of the price (price; profit);
    - location of the shop where the products are sold (a craftsman can have more than one shop, and the product prices change depending on the shop they are sold).

Functional Requirements

- Facilitate the creation and sharing of consistent metadata
- Support search of any, or all, elements, particularly of type of Product, Category of Product and Location of Craftsman (use case 1 and 3).
- Browse by any element (use case 1 and 3).
- Provide metadata for statistical information building (use case 2).

[a] By "user" we mean an entity that looks for information in metadata (any kind of automata searching for information).

3.6 Domain Model (Component Stage 2)

The Singapore Framework Component Stage 2 - Domain Model is mandatory. The domain model "captures the most important types of objects in the context of the system." [Jacobson et al., 1999, p. 119]. According to Baker and Coyle (2009) "a domain model is a description of what things your metadata will describe, and the relationships between those things. The domain model is the basic blueprint for the construction of the application profile". The Domain Model is the mandatory Singapore Component Stage 2, and is also developed in all the DCAP analysed. The RUP process [Kruchten, 2004] and the Web Site Design Model [De Troyer and Leune, 1998], and the DCMI Guidelines [Baker and Coyle, 2009] referred the need for such an deliverable development (see figure 10).

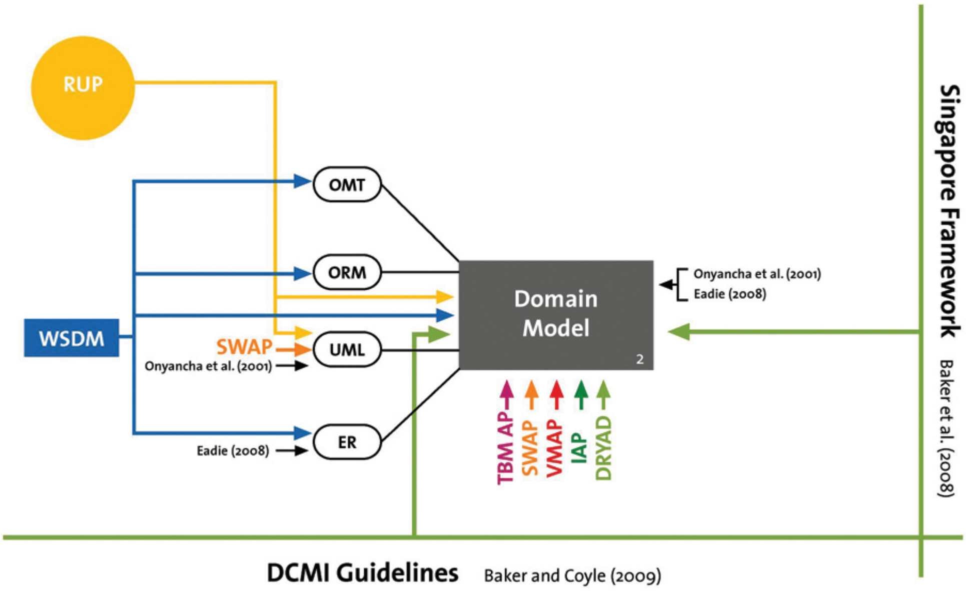

Figure 10: Sources for the Domain Model Component Stage 2 definition

The domain model is build based on the Functional Requirements Component Stage 1 and on the Use-Cases Model deliverable, both described in Section 3.5. The domain model development can also use the help of other information depending on the DCAP development context. In cases where access to documentation or to the information system database is possible, to resort to the Document Analysis technique to define the domain model is a must; the more information the DCAP development process has, the better.

Figure 10 shows that various techniques were used to develop the Domain Model. The DCAP SWAP used an Unified Model Language (UML) class diagram [Fowler, 2004] with details suppressed, as well as the work reported by [Onyancha et al., 2001]. RUP [Kruchten, 2004], being an UML centered process, also uses UML. The WSDM [De Troyer and Leune, 1998] suggests that various techniques can be used: UML, the Object Modeling Technique (OMT) [Rumbaugh et al., 1990], the Object Role Modeling (ORM) [Halpin, 2006], the Entity-relationship model (ER) [Chen, 1976]. Eadie (2008) also reports in its work the use of the ER technique.

Me4DCAP suggests that the Domain Model should be developed using an UML class diagram with details suppressed. The UML diagram identifies the classes of objects and the relationships among them but the classes' methods and the objects' attributes are omitted since the methods' definition is not in the frame of a DCAP development, and the objects' attributes will be defined later in the DCAP development process (see section 3.7 - the ORM Diagram data model). The Entity-Relationship diagram [Chen, 1976] (with attributes and cardinalities omitted as well), showing the entities and the relationships among them can be an alternative to the UML class diagram technique.

Once developed the Singapore Framework Component Stage 2 - Domain Model with one of the techniques suggested, Me4DCAP follows to the Singapore Framework Component Stage 3 - Description Set Profile development.

Example: Craftsman's Catalog

Domain Model
Figure 11 shows our example Domain Model. We need 3 classes of objects: the "Product", the "Craftsman" - that produces those products - and the "Shop" - where the products are sold. Probably later we will need more classes of objects, depending on the ORM Diagram Data Model we will built from this Domain Model and from the Use Cases and Functional Requirements we have defined.

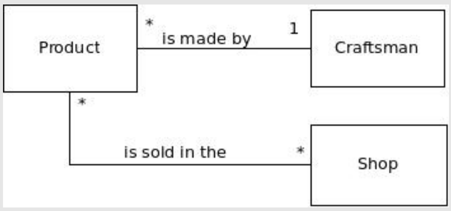

Figure 11: The Domain Model for our catalog example

3.7 Description Set Profile (Component Stage 3)

To develop the mandatory Singapore Component Stage 3 - Description Set Profile (DSP) Me4DCAP defines the need to develop a set of 2 mandatory dossiers:

1. The Integration Dossier;
2. The Validation Dossier (in laboratory).

The Integration dossier comprises 3 deliverables:

1. an Object Role Modeling (ORM/NIAM) [Halpin, 2006] diagram data model;

2. a State Of The Art report;

3. a Document of Integration.

All these deliverables are mandatory.

Figure 12 shows that the Document of Integration was referred in the method Methontoloy [Fernández-López et al., 1997] and by the literature [CWA, 2006]; also the DCMI Guidelines [Baker and Coyle, 2009] present such a document. Figure 12 also shows that all the analysed DCAP performed a state of the art; and all the following literature: [Chen and Chen, 2005], [BSI, 2005], [Onyancha et al., 2001], [Agostinho et al., 2004], [Marzal García-Quismondo et al., 2 [Buonazia and Masci, 2007] and [Salokhe et al., 2008].

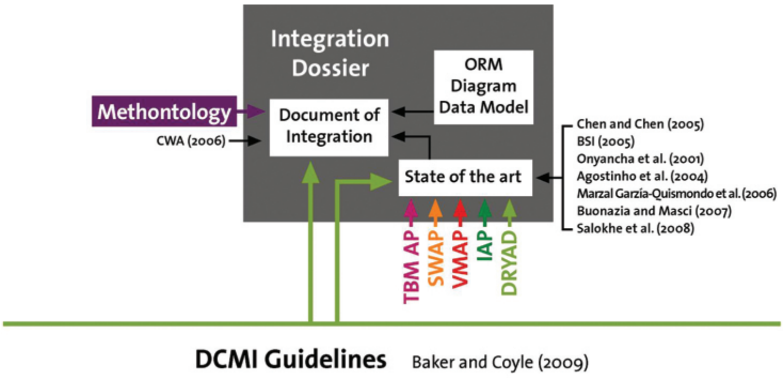

Figure 12: Sources for the Integration Dossier deliverable definition

Based on the Singapore Framework Component Stage 2 - Domain Model, Me4DCAP next goal is to develop the DCAP Data Model using the ORM diagram technique. The DCAP ORM diagram data model is a DCAP representation of its:

- classes of objects (defined in the Domain Model);
- attributes of the classes of objects;
- attributes' constraints, such as their repeatability, domain and multi-language option.

The ORM is a very interesting technique for the modeling of a data structure of a DCAP because it is property centric, unlike UML (which is object centric). ORM has also a natural approach to semantic modeling [Nijssen and Halpin, 1989] since "ORM models the world in terms of just objects and roles and hence has only one data structure – the relationship type" [Halpin, 1998, p. 7]. One can extract short sentences from an ORM diagram with facts: subject, predicate and an object. This fact has a similar approach to RDF triples. The use of this kind of technique is also a simple way to validate domain modeling work to non domain modeling specialists. This simplicity comes from the natural language information analysis method "where the application is described in terms readily understood by users, rather than being recast in terms of implementation data-structures" [Halpin, 1996, p. 1]. Another feature of the ORM Diagram technique is that it allows users to do population checking in the diagram itself, through the building of populated tables near the relationships.

ORM modeling is a technique that has already been explored in the past by the research community using it in "semantic contexts" e.g., XML data structures building (c.f. Spyns et al. (2002)) and Ontology modeling in the Semantic Web (c.f. Bird et al. (2000)).

There are several types of software that can draw ORM diagrams - c.f `http://www.orm.net/`.

The ORM diagram development should start by the drawing of the objects defined in Domain Model and the relationships among them. After that, every object should be defined in detail, defining the object properties, their obligatoriness, repeatability, codification and multi-language possibility. In the end of the ORM diagram construction process, a population checking should be done to assure that all information is incorporated in the diagram. The ORM diagram should be developed by the system analysts and the metadata developers of the work team. The population checking can be done by the whole work team. In this part of the DCAP development process, every class of object and attribute should have been already described in plain text, in the Usage Guidelines Component Stage 4 (see section 3.8 for details about this deliverable). This work should be done by the whole group, specially by the managers since they know best the DCAP domain of application.

Example: Craftsman Catalog

Integration Dossier: ORM diagram Data Model

The ORM diagram Data Model (only main object types) for the craftsman's catalog is presented in Figure 13. The application domain defined that a Product has a different price depending on the shop it is sold. Because of that we need to define a new object type called "Price". A "Product" costs a "Price" at "Shop": a ternary relationship between this objects type is defined. We can see that a "Product" can not be sold at a different "Price" in the same shop, but can be sold at a different price in a different shop (the combination of "Product" and "Shop" must be unique - stated by the pink line over the roles "Product" and "Shop"). The diagram also shows that a "Craftsman" makes "Product" or that a Product is made by "Craftsman". A Product must have a Craftsman defined (stated by the pink dot in the Product object type). "Craftman" can have many "Product" (stated by the pink line over the role). The definition of the "Price" of a Product and the "Shop" it is sold are not mandatory.

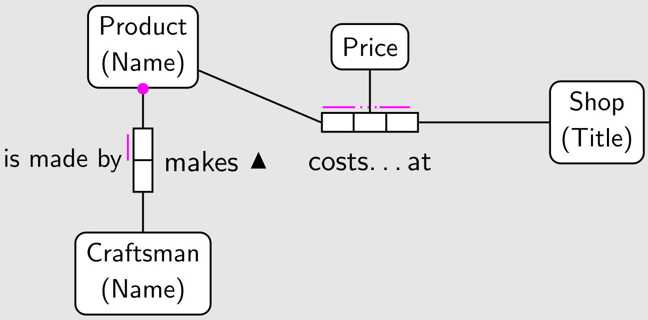

Figure 13: The ORM diagram: the main object types and their roles

Figure 14 shows the object type "Craftsman" in detail. The application domain defined that a craftsman sells products in one country and in no other (that is the reason for Country property to be in the Craftsman object and not in the Shop object - see Figure 16. A "Craftsman" is identified by a "Name", lives in a "Country" - Country Code - defined by a Vocabulary Encoding Scheme (VES) URI. The "Country" uses a "Currency" - Currency Code defined by a VES URI. A "Craftsman" can have several "Other Name", was born in a "Date" - format defined by a Syntax Encoding Scheme (SES) URI; has one gender - defined by a VES URI and one email address. All these attributes are mandatory except "Email" and "Other Name". Finally, a Craftsman has, per year (property "Date" of the Object type "Annual Income") - with a SES URI defining the format, an "Annual income". The unit of the annual income is defined by the "Coin" attribute.

Figure 15 shows the object type "Product" in detail. A "Product" is identified by a "Name" (which has different language versions), a Type and a "Description" (which has different language versions). A "Product" can be classified in several "Categories" - defined by a VES URI. The "Category", "Type" and "Description" properties are mandatory. Since the price of a product depends on the shop it is sold, "Price" is not defined as a property of a "Product" in the ORM diagram of the object type "Product" but as an object type by itself defined in Figure 17.

Example: Craftsman Catalog

Integration Dossier: ORM diagram Data Model (cont.)

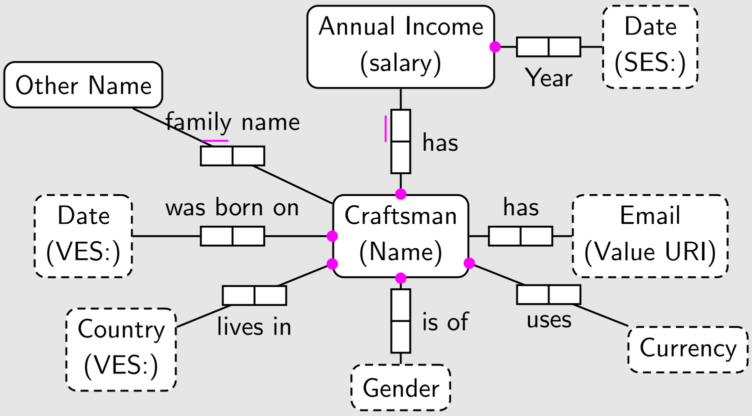

Figure 14: The ORM diagram for the object type "Craftsman"

Figure 16 shows the object type "Shop" in detail. A shop is the place where the craftsman sells its own products.A craftsman can have more than one shop, one of them has to be defined as the headquarters (head is a boolean data-type). A shop has a Location (mandatory) - it can be the name of a City or a Village or of a place depending on the Country; and a phone (not mandatory).

Figure 17 shows the object type "Price" in detail. A price has a "Currency Value" (the cost - e.g. 2; 3; 20), a "Quantity Eligible" (e.g. kilo, gram, liter, unit), a "Quantity Value" (e.g. 1, 0.2, 200) and a "Profit" - % of the cost that is earned - without the material costs and other costs). All "Price" properties are mandatory.

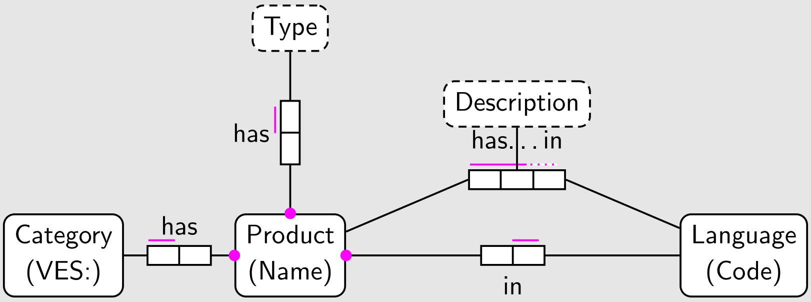

Figure 15: The ORM diagram for the object type "Product"

Example: Craftsman Catalog

Integration Dossier: ORM diagram Data Model (Cont.)

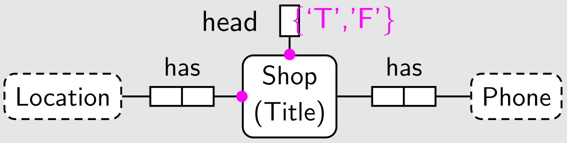

Figure 16: The ORM diagram for the object type "Shop"

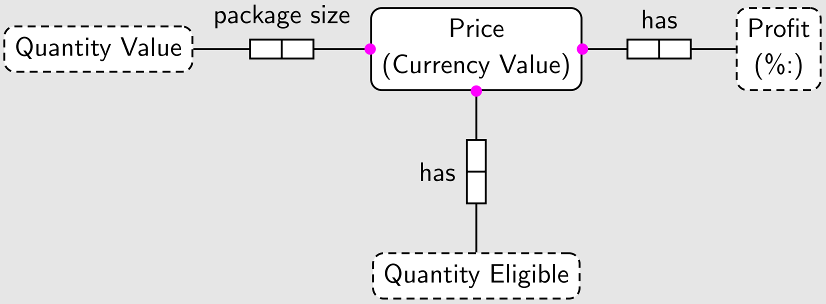

Figure 17: The ORM diagram for the object type "Price"

After the ORM Diagram data model construction, Me4DCAP defines as next step the application of a metadata scheme property to every property of the objects of the Data Model. The attributes are described, each and everyone, by the existing properties of the metadata schemes of the metadata community. In case of not being able to describe some of the object properties with the existing metadata schemes, those properties should be described with new created properties metadata schemes. According to Baker and Coyle (2009), this process is done in 2 steps:

- To perform a State of the art to existing metadata schemes - that are described in RDF - to find out from the existing schemes which ones can describe the identified attributes. This work should be done by the metadata programmers of the work-team;
- Creation of new properties: in case there are no properties on the metadata schemes of the state of the art to describe some of the identified Data Model object properties. This work should be done by the metadata programmers of the work-team.

The existing information on the state of the art and on the ORM diagram Data Model will be used to build a Document of Integration; this work should be done by the metadata programmers of the work-team. The Document of Integration shows, in a matrix, per line, every property or class and its constraints, described by the properties of the metadata schemes and encoding schemes chosen. The template of the Document of Integration can be downloaded from the repository of University of Minho accessible through the URL http://hdl.handle.net/1822/24379; it is based on the table presented by the DCMI Guidelines [Baker and Coyle, 2009]. The Document of Integration has 2 tables:

- The first table defines de Namespaces used, it has the following items:

  Title The title of the namespace (e.g. Dublin Core Metadata Initiative Terms)

  Full Namespace URI The Uniform Resource Identifier that identifies the namespace (e.g. http://www.purl.org/dcterms)

  Prefix The prefix used (e.g dcterms)

- The second table defines the property and its constraints, described by the properties of the metadata schemes (or vocabularies) and encoding schemes chosen. It has the following items:

  Label The label given to the property or class (let's use an example, the Date of Birth of a person. In this case Label can be "Date of Birth")

  Class or Property To identify if the line describes a property or a class (e.g. property).

  Definition The namespace and property identified (e.g. foaf:birthday).

  Range The range of the property. It can be a "literal" or a "non-literal" value. A "literal" is a value that " by definition may consist of just one value string, optionally augmented with a language tag (in a plain value string) or a data type identifier (in typed value string)" [Baker and Coyle, 2009]. A "non-literal" value is anything other that it is not a "literal". In our case example (date of birth) it should be "literal".

  Type of Range The specification of the range. Use as many columns as needed. It can be a "Value String", a "Vocabulary Encoding String", a "Syntax Encoding String" and a "Value URI". If it is a "Value String" it is possible that the property can have several language versions or can be of a certain type (e.g boolean); that should be referred also in the Type of Range (e.g. Boolean Data

Type). More than one column should be use when there is more that one information to provide, e.g. a certain property can be a “Value String“ and can have a “SES“ in order to follow a predefined format. In case that a VES or a SES is needed, the URI should be presented. In our case example the Type of Range has 2 columns with “Value String” in the first column and ”SES (http://purl.org/dc/terms/W3CDTF)“ in the latter.

Related Description If the property or class has a description with other properties (e.g. an author might have definitions as email, name, surname...etc.). It can be a ”Yes” or a ”No”. In our case example ”No“.

Min The minimum number of times the property can be used (e.g. 1 - we are only able to be born once!)

Max The maximum number of times the property can be used (e.g. 1 - for the same reason!)

Me4DCAP defines as next step the execution of the validation of the work done until the present moment of the development process. In order to do that, a validation in laboratory is executed; Me4DCAP calls it the “Validation Dossier”. The Validation Dossier comprises 3 mandatory deliverables:

- a Validation Report;
- a Document of Validation;
- a Questionnaire.

Figure 18 shows that the SWAP [Allinson and Powell, 2006] and VMAP [Iglesias et al., 2009] DCAP have also used a Document of Validation to perform laboratory validations of the DCAP. Also Agostinho et al. (2004) have used such a technique.

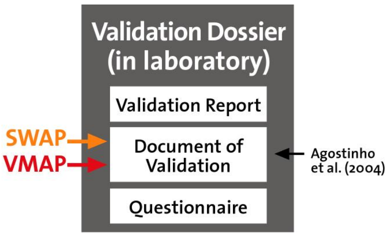

Figure 18: Sources for the Validation Dossier (in laboratory) definition

Example: Craftsman Catalog

Integration Dossier: State Of The Art report
A State Of The Art was done and is reported in the link `http://hdl.handle.net/1822/23412` - Matrix II and III. To describe the information on the Craftsman's Catalog Data Model we have chosen the following:

Name	URI
Metadata Schemes or Vocabularies	
DCTerms	http://purl.org/dc/terms/
FOAF	http://xmlns.com/foaf/0.1/
Good Relations	http://purl.org/goodrelations/v1#
RDF	http://www.w3.org/1999/02/22-rdf-syntax-ns#
Syntax Encondig Schemes	
W3C-DTF	http://purl.org/dc/terms/W3CDTF
Vocabulary Encoding Schemes	
MARC Country	http://id.loc.gov/vocabulary/countries

Example: Craftsman Catalog

Integration Dossier: Document of Integration
The document of integration is presented in Appendix B.
The work team did not find any metadata schemes (or vocabularies) on the State Of The Art to describe some properties and classes. We have defined a fictitious URI cc=http://example.com/craftsman_catalog/v1/; the properties are as follows:

- properties:

  cc:salary
  the annual salary of a craftsman

  cc:head
  the boolean property to describe if a shop is a headquarter

  cc:profit
  the profit on a product

- classes:

  cc:shop
  to identify the "Shop" class

  cc:annual_income
  to identify the "Annual Income" class

Example: Craftsman Catalog

Integration Dossier: Document of Integration (Cont.)
The work team decided to create 2 Vocabulary Encoding Schemes (VES):

- for the property “Category“ of object type ”Product“ since the categories of the craftsmen’ community are too social solidarity community specific with some categories not existent in the market economy, to whom these VES found where developed for. The VES CAT URI is defined as http://example.com/craftsman_catalog/CAT/. As an example, a ”Indigenous Art“ category is referred with the http://example.com/craftsman_catalog/CAT/indigenous_art URI.
- for the property ”Gender“ of the object type ”Craftsman“. This VES is composed by the terms ”Male“, Female”,“Transgender”,“Bigender’ and “Transvestite“. The Craftsman community decided to create this VES since there was none with a classification out of the classical binary concept, with two rigidly fixed options: male or female. The VES GENDER URI is defined as http://example.com/craftsman_catalog/GENDER/. As an example, a ”Male“ Gender is referred with the http://example.com/craftsman_catalog/GENDER/Male URI.

A laboratory validation should take place, so as to check:

- its adequacy to what has been defined in the “Vision Statement” artifact: a meeting of the work-team should take place to evaluate the answer to the defined vision (see Vision Statement in section 3.5). The work-team should make a report (text document) with the conclusions of the meeting and recommendations.

- DCAP adaptation to the resources that are going to be described by the DCAP: the validation is done through the application of the DCAP to a resource sample. This validation work is done in 2 stages:

  1. Application of the DCAP to a resource sample. The work-team should identify a set of resources that constitutes a trustworthy sample of the application domain of the developing DCAP, and from there, final users, chosen by the stakeholders members of the work-team, and the metadata programmers as resource persons, should complete the Document of Validation with data referring to each resource. The Document of Validation should be simple to fill in, where each element of the metadata is populated with the data that corresponds to the resource. This Document of Validation should be accompanied by the 2 Guidelines Component Stage 4 and Component Stage 5. A template of a Document of Validation can be downloaded from the repository of University of Minho accessible through the URL http://hdl.handle.net/1822/24379;

2. Answer to a set of questions: the final users chosen in the previous step, together with the metadata programmers of the work-team, should answer to a set of questions to assess the difficulties of the validation process. The goal is to assess if there is data for which the DCAP has no description, or if there are DCAP elements, defined as compulsory, that could not be fulfilled with the information existing in a given resource, or any other type of difficulty or ambiguity. The questions to be asked to the DCAP validators could be like:
    - Could you describe all your data with the available elements? If not, please refer the difficulties;
    - Were there any DCAP metadata elements left that you could not fulfill? Which? Did this happen for lack of data or because you did not know how to do it?;
    - Did you have any difficulty in particular to describe your data? Were there any ambiguities?;
    - Is there anything else you want to add?

According to the results of the questionnaire, the process iterates or follows to the DSP development (see figure 4).

Next step is the development of DSP deliverable. This deliverable is the Singapore Component Stage 3, and it is mandatory.

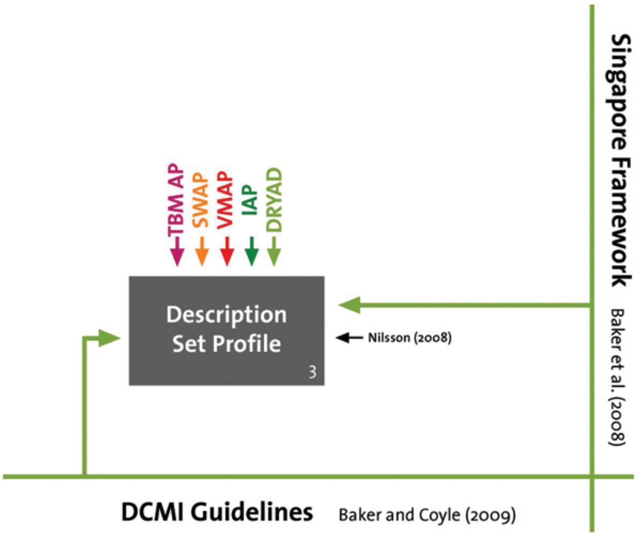

Figure 19: Sources for the DSP Component Stage 3 definition

This task consists in detailing the metadata elements and constraints in the DSP language defined by Nilsson (2008); the work team should have as reference the Integration Dossier as source of information.

Further information, including implementation examples, can be found in Baker and Coyle (2009). This task should be performed by the metadata programmers of the work-team.

Example: Craftsman Catalog

Validation Dossier (in laboratory): Document of Validation
We are going to perform the validation of the document of integration with 2 resources:

Resource 1
João Curado Silva, Portugal. João was born in 4 December 1969, his email is joao.silva@example.com. The annual income of João was 12000 euro in 2011 and 12150 euro in 2011. João has 2 shops: one in Porto (the headquarters) with the phone 220137789, another in Guimarães with the phone 2532345678. He sells only one product in these 2 shops: a special pillow for people with allergies, it costs 75 euro in Porto and 70 euro in Guimarães. The profit in both shops is 5%.

Resource 2
Maria Alba Mejia, Spain. Maria was born in 12 May 1980, her email is maria.mejia@yahoo.es. The annual income of Maria Alba was 15000 euro in 2010 and 2011 and 13000 euro in 2012. She has 3 shops: one in Seville (the headquarters) with the phone (95) 423 05 05, one in Barcelona (93) 451 19 14 and one in Madrid (91) 772 20 87. She sells two products:

- Lavender soap: made out of the lavender she grows in her properties, with no chemicals, only natural products. This product is sold in the Seville shop by 2 euro, in Barcelona by 2.3 euro and in Madrid by 4 euro. From the shops outside Madrid she gets 7% of profit and from the shop in Madrid she gets 9% of the profit.
- Lavender perfume: made out of the lavender she grows in her properties, with no chemicals, only natural products. This product is sold in Seville by 15 euro and in Madrid in packages of 3 soaps by 100 euro. She gets 5% of profit in Seville and 7% in Madrid.

The document of validation for these two resource samples are presented in Appendix C. If the RDF description of each resource is developed in one file, all together, there is no need to use connection properties between classes. But, if RDF descriptions of the resources are developed in different files, than we will have to link classes in order not to loose the connections between them. In this moment of the DCAP development process programmers probably don't know yet the real URI links they will set, so we decided to name the links and number them like ”Link1”. We can see, e.g., that the Product class ”Lavender soap” has the link ”Link2” and the Shop class ”Barcelona“ has the “Link6“. So the Soap sold in Barcelona has ”Link2” for the class Product and “Link6“ for the class Shop in its description.

Example: Craftsman Catalog

Description Set Profile
The DSP codification in XML is presented in Appendix D.

3.8 Guidelines (Component Stage 4 and Component Stage 5)

The Singapore Framework Components Stage 4 and 5 - Usage Guidelines and Syntax Guidelines are not mandatory in the Singapore Framework. Me4DCAP does not make them mandatory but recommends that they should be developed since it helps the final users of the DCAP application to apply correctly the properties and constraints. Figure 20 shows that these guidelines (very specific to a DCAP development) are referred naturally by the Singapore Framework and by DCMI Guidelines. Also RUP refers the use of guidelines as an important tool for the good implementation of the developed systems. so it encourages its development.

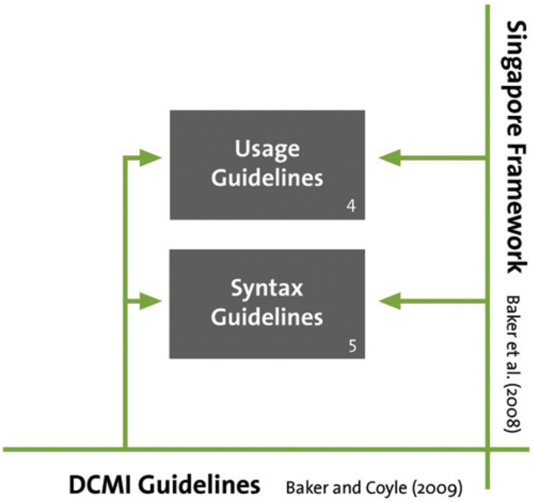

Figure 20: Sources for the Usage Guidelines and Syntax Guidelines Component Stage 4 & 5 definition

DCMI Guidelines explains: "Description Set Profile defines the -what-' of the application profile; usage guidelines provide the -how- and -why-. Usage guidelines offer instructions to those who will create the metadata records. Ideally, they explain each property and anticipate the decisions that must be made in the course of creating a metadata record" [Baker and Coyle, 2009].

For detailed information see Baker and Coyle (2009) and Nilsson et al. (2008).

Me4DCAP defines that the developing of the Usage Guidelines starts at the same time as the Domain Model (see figure 4 - Block2) and is developed throughout the next 2 blocks of the DCAP development (figure 4 - Block3 and Block4).

The development of the Syntax Guidelines needs that the Integration Dossier has been already developed in a certain stage in order to have already some maturity. This deliverable describes ”any application profile-specific syntaxes and/or syntax guidelines, if any“ [Baker and Coyle, 2009]. For detailed information see Baker and Coyle (2009) and Nilsson et al. (2008).

The Usage Guidelines can be developed by both types of members of the work team, the managers and the metadata programmers, since the description of the attributes and classes of objects is information that will have to be filled in by the domain experts.

The Syntax Guidelines have to be developed by the metadata programmers since it is a very technical document.

3.9 Finishing the DCAP development

A validation in production of the DCAP should be performed. RUP integrates such an artifact (see figure 21). This process of validation can be done using a log registration technique or observing final-users working with the system that has implemented the DCAP developed. The results of this validation in production should be reported to the work-team in order to review and access the DCAP definitions. If there is new information to introduce in the process, the whole DCAP development process should start from Block 1 (see figure 4), and every deliverable should be checked against this new information.

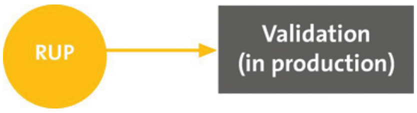

Figure 21: Sources for the Validation (in production) definition

3.10 Conclusions and Future Work

This paper is framed in a research in progress PhD project based in a Design Science Research (DSR) approach. The goal of this PhD project is to develop a method for the development of Dublin Core Application Profiles (Me4DCAP); in fact a study that we have performed recently (c.f. Curado Malta and Baptista (2012)) have shown that there is none. This paper describes in detail the first version of Me4DCAP. Me4DCAP has as starting point the Singapore framework for Dublin Core Application Profiles (DCAP) and the Rational Unified Process; and integrates also knowledge from:

- software development processes and techniques, focusing on the early stages of the processes that deal with data modeling;
- the practices of the metadata community concerning DCAP development.

Me4DCAP detailed description establishes the way through the DCAP development. It establishes when activities must take place, how they interconnect, and which deliverables they will bring about; it also suggests which techniques could be used to build these deliverables. Me4DCAP defines a way for the construction of each Singapore Framework component. Me4DCAP description follows the order of these Singapore components, showing, in detail, which are the knowledge sources that justify Me4DCAP design.

The work presented is still in progress; the experimental situation of the DSR Design Cycle is still in place and this first version of Me4DCAP will be tuned with the outputs from this work.

Me4DCAP V.01 validation will be subject in the immediate future to two Confirmatory Focus Groups, and we expect to find projects in developing DCAP using Me4DCAP V0.1 for feedback and fine-tuning. We expect to integrate all the outputs of the whole validation process of Me4DCAP V0.1 in order to tune a new version of the method – Me4DCAP V0.2.

Acknowledgements

This work is sponsored by FEDER funds through the Competitivity Factors Operational Programme - COMPETE and by National funds through FCT - Foundation for Science and Technology within the scope of the project: FCOMP-01-0124-FFEDER-022674.
We would like to thank the RIPESS group for accepting the DCAP-SEE development challenge as an experimental situation for our research project; without them the work described in this article was not possible.
We would like to thank Ida Cruz for the drawings of this paper.

References

[Agostinho et al., 2004] Agostinho, S., Bennett, S., Lockyer, L., and Harper, B. (2004). Developing a learning object metadata application profile based on lom suitable for the australian higher education context. *Australasian Journal of Educational Technology*, 20(2):191–208.

[Allinson and Powell, 2006] Allinson, J. and Powell, A. (2006). SWAP application profile. `http://www.ukoln.ac.uk/repositories/digirep/index/Eprints_Application_Profile`. Access in 23 Oct 2012.

[Baker and Coyle, 2009] Baker, T. and Coyle, K. (2009). Guidelines for dublin core application profiles. http://dublincore.org/documents/profile-guidelines/. Accessed in 26 Jun 2011. Confirmed 1 Feb 2013.

[Baker et al., 2008] Baker, T., Nilsson, M., and Johnston, P. (2008). The singapore framework for dublin core application profiles. `http://dublincore.org/documents/singapore-framework/`. Accessed in 26 Jun 2011. Confirmed 9 Feb 2013.

[Bird et al., 2000] Bird, L., Goodchild, A., and Halpin, T. (2000). Object role modelling and XML-Schema. *Conceptual Modeling-ER 2000*, pages 661–705.

[Borges, 2008] Borges, P. (2008). Configuração do rup com vista à simplificação dos elencos processuais em pmes de desenvolvimento de software. Master's thesis, University of Minho.

[BSI, 2005] BSI (2005). *BS 8419-1:2005 Interoperability between metadata systems used for learning, education and training. Code of practice for the development of application profiles.* BSI.

[Buonazia and Masci, 2007] Buonazia, I. and Masci, M. (2007). Il PICO application profile. un dublin core application profile per il portale della cultura italiana.

[Calverley and Johnston, 2009] Calverley, G. and Johnston, P. (2009). Time-based media application profile: Definition phase report. Final version: V3.0, JISC.

[Carrier, 2008] Carrier, S. (2008). The dryad repository application profile: Process, development, and refinement. Master's thesis.

[Carvalho, 2012] Carvalho, J. A. (2012). Validation criteria for the outcomes of design research. In *IT Artefact & Workpractice Intervention, a Pre-ECIS and AIS SIG Prag Workshop*.

[Chen, 1976] Chen, P. (1976). The entity-relationship model—toward a unified view of data. *ACM Transactions on Database Systems (TODS)*, 1(1):9–36.

[Chen and Chen, 2005] Chen, Y. and Chen, S. (2005). Metadata lifecycle model and metadata interoperability. In *5th International Conference on Conception of Library and Information Science*.

[Curado Malta and Baptista, 2012] Curado Malta, M. and Baptista, A. A. (2012). State of the art on methodologies for the development of a metadata application profile. In J. M. Dodero, M. Palomo-Duarte, P. K., editor, *Comunications in Computer and Information Sciences*, volume 343, page 61–73. Springer-Verlag Berlin Heidelberg.

[Curado Malta and Baptista, 2013] Curado Malta, M. and Baptista, A. A. (2013). Me4DCAP V0.1: a method for the development of dublin core application profiles. *ElPub2013 Proceedings*. Fourthcoming.

[CWA, 2006] CWA (2006). *Guidelines and support for building application profiles in e-learning (CEN Workshop Agreement, CWA 15555:2006 E)*. CEN, European Committee for Standardization, ftp://ftp.cenorm.be/PUBLIC/CWAs/e-Europe/WS-LT/cwa15555-00-2006-Jun.pdf. Acedido em 2 de Julho de 2012.

[DCMI, nd] DCMI (n/d). Dublin core education application profile (working draft of v0.4). https://docs.google.com/Doc?id=dn8z3gs_38cgwkvv. Acedido em 2 de Junho de 2012.

[de Almeida and Pinto, 1995] de Almeida, J. and Pinto, J. (1995). *A investigação nas ciências sociais*. Presença.

[de La Passadière and Jarraud, 2004] de La Passadière, B. and Jarraud, P. (2004). Manuel, a lom application profile for c@mpusciences®. *Sciences et Technologies de l Information et de la Communication pour l Éducation et la Formation*, 11.

[De Troyer and Leune, 1998] De Troyer, O. and Leune, C. (1998). Wsdm: a user centered design method for web sites. *Computer Networks and ISDN Systems*, 30(1):85–94.

[Eadie, 2008] Eadie, M. (2008). Towards an application profile for images. *Ariadne*, 55.

[Fernández-López et al., 1997] Fernández-López, M., Gómez-Pérez, A., and Juristo, N. (1997). Methontology: from ontological art towards ontological engineering.

[Fowler, 2004] Fowler, M. (2004). *UML distilled*. Addison-Wesley Professional.

[Friesen et al., 2002] Friesen, N., Mason, J., and Ward, N. (2002). Building educational metadata application profiles. In *Proceedings of the International Conference on Dublin Core and Metadata for e-Communities*, volume 2002, pages 63–69.

[Gregor and Jones, 2007] Gregor, S. and Jones, D. (2007). The anatomy of a design theory. *Journal of the Association for Information Systems*, 8(5):312–335.

[Halpin, 1996] Halpin, T. (1996). Business rules and object-role modeling. *Database Programming and Design*, 9:66–72.

[Halpin, 1998] Halpin, T. (1998). UML Data Models From An ORM Perspective: Part 1. *Journal of Conceptual Modeling*.

[Halpin, 2006] Halpin, T. (2006). Object-role modeling (orm/niam). *Handbook on architectures of information systems*, pages 81–103.

[Hesse, 2003] Hesse, W. (2003). Dinosaur meets archaeopteryx? or: Is there an alternative for rational's unified process? *Software and Systems Modeling*, 2(4):240–247.

[Hevner, 2007] Hevner, A. (2007). The three cycle view of design science research. *Scandinavian Journal of Information Systems*, 19(2):87.

[Hevner and Chatterjee, 2010] Hevner, A. and Chatterjee, S. (2010). *Design Research in Information Systems - Theory and Practice*, volume 22 of *Integrated Series in Information Systems*. Springer.

[Hevner et al., 2004] Hevner, A., March, S., Park, J., and Ram, S. (2004). Design science in information systems research. *MIS quarterly*, 28(1):75–105.

[Iglesias et al., 2009] Iglesias, C. A., Garijo, M., Molina, D., and de Juan, P. (2009). VMAP. A Dublin Core Application Profile for Musical Resources. In Fabio Satori, Miguel A. Sicilia, N. M., editor, *Metadata and Semantic Research: Third International Conference, MTSR 2009*, number 46 in CCIS, pages 1–12. MTSR, Springer Verlag.

[IMS Global Learning Consortium, 2005] IMS Global Learning Consortium, I. (2005). *IMS Application Profile Guidelines: Part 1 - Management Overview*. IMS Global Learning Consortium, Inc., version 1 edition.

[Isakowitz et al., 1995] Isakowitz, T., Stohr, E., and Balasubramanian, P. (1995). Rmm: a methodology for structured hypermedia design. *Communications of the ACM*, 38(8):34–44.

[Jacobson et al., 1999] Jacobson, I., Booch, G., and Rumbaugh, J. (1999). *The Unified Software Development Process*. Addison Wesley.

[Kroll and Kruchten, 2003] Kroll, P. and Kruchten, P. (2003). *The rational unified process made easy: a practitioner's guide to the RUP*. Addison-Wesley Professional.

[Kruchten, 2004] Kruchten, P. (2004). *The rational unified process: an introduction*. Addison-Wesley Professional.

[March and Smith, 1995] March, S. and Smith, G. (1995). Design and natural science research on information technology. *Decision support systems*, 15(4):251–266.

[Marzal García-Quismondo et al., 2006] Marzal García-Quismondo, M., Calzada Prado, J., and Cuevas Cerveró, A. (2006). Desarrollo de un esquema de metadatos para la descripción de recursos educativos: el perfil de aplicación mimeta. *Revista española de documentación científica*, 29(4):551–571.

[Nijssen and Halpin, 1989] Nijssen, G. and Halpin, T. (1989). *Conceptual Schema and Relational Database Design: a fact oriented approach*. Prentice Hall.

[Nilsson, 2008] Nilsson, M. (2008). Description set profiles: A constraint language for dublin core application profiles. `http://dublincore.org/documents/2008/03/31/dc-dsp/`. This is a DCMI Working Draft.

[Nolan, 1973] Nolan, R. L. (1973). Managing the computer resource: a stage hypothesis. *Communications of the ACM*, 16(7):399–405.

[Onyancha et al., 2001] Onyancha, I., Keizer, J., and Katz, S. (2001). A dublin core application profile in the agricultural domain. In *International Conference on Dublin Core and Metadata Applications*, number 0, pages pp–185.

[Powell et al., 2007] Powell, A., Nilsson, M., Naeve, A., Baker, T., and Johnston, P. (2007). DCMI abstract model. `http://dublincore.org/documents/2007/06/04/abstract-model/`. Accessed in 26.06.2011.

[Press, 2004] Press, N. (2004). Understanding metadata. *NISO Press Booklets, US, ISBN*, 1:880124–62.

[Purao, 2002] Purao, S. (2002). Design research in the technology of information systems: Truth or dare. *online], Pennsylvania State University, http://purao. ist. psu. edu/working-papers/dare-purao. pdf.*

[Purao, 2013] Purao, S. (2013). Truth or dare: The ontology question in design science research. *Journal of Database Management*. Forthcoming.

[Rossi and Sein, 2003] Rossi, M. and Sein, M. K. (2003). Design research workshop: a proactive research approach. *Presentation delivered at IRIS*, 26:9–12.

[Rumbaugh et al., 1990] Rumbaugh, J. R., Blaha, M. R., Lorensen, W., Eddy, F., and Premerlani, W. (1990). Object-oriented modeling and design.

[Salokhe et al., 2008] Salokhe, G., Pesce, V., and Liesthout, J. (2008). Organization metadata application profile. Technical report, FAO.

[Schneider and Winters, 2001] Schneider, G. and Winters, J. (2001). *Applying use cases: a practical guide*. Addison-Wesley.

[Shoval and Even-Chaime, 1987] Shoval, P. and Even-Chaime, M. (1987). Database schema design: an experimental comparison between normalization and information analysis. *ACM SIGMIS Database*, 18(3):30–39.

[Simon, 1996] Simon, H. A. (1996). *The sciences of the artificial*. MIT press.

[Spyns et al., 2002] Spyns, P., Meersman, R., and Jarrar, M. (2002). Data modelling versus ontology engineering. *ACM SIGMOD Record*, 31(4):12–17.

[Takahashi and Liang, 1997] Takahashi, K. and Liang, E. (1997). Analysis and design of web-based information systems. *Computer Networks and ISDN Systems*, 29(8):1167–1180.

[Takeda et al., 1990] Takeda, H., Veerkamp, P., and Yoshikawa, H. (1990). Modeling design process. *AI magazine*, 11(4):37.

[Tremblay et al., 2010] Tremblay, M., Hevner, A., and Berndt, D. (2010). The use of focus groups in design science research. *Design Research in Information Systems*, pages 121–143.

[Vaishnavi and Kuechler, 2012] Vaishnavi, V. and Kuechler, B. (2012). Design science research in information systems. `http://desrist.org/design-research-in-information-systems/`. Accessed in 25 April 2013.

[W3C, 2010] W3C (2010). RDF - W3C standards. `http://www.w3.org/RDF/`. Acessed in 14 Jan 2012. Confirmed 9 Feb 2013.

[W3C, 2012] W3C (2012). W3C semantic web activity. `http://www.w3.org/2001/sw/`. Accessed 30 Jul 2012.

A Overall Me4DCAP sources of information

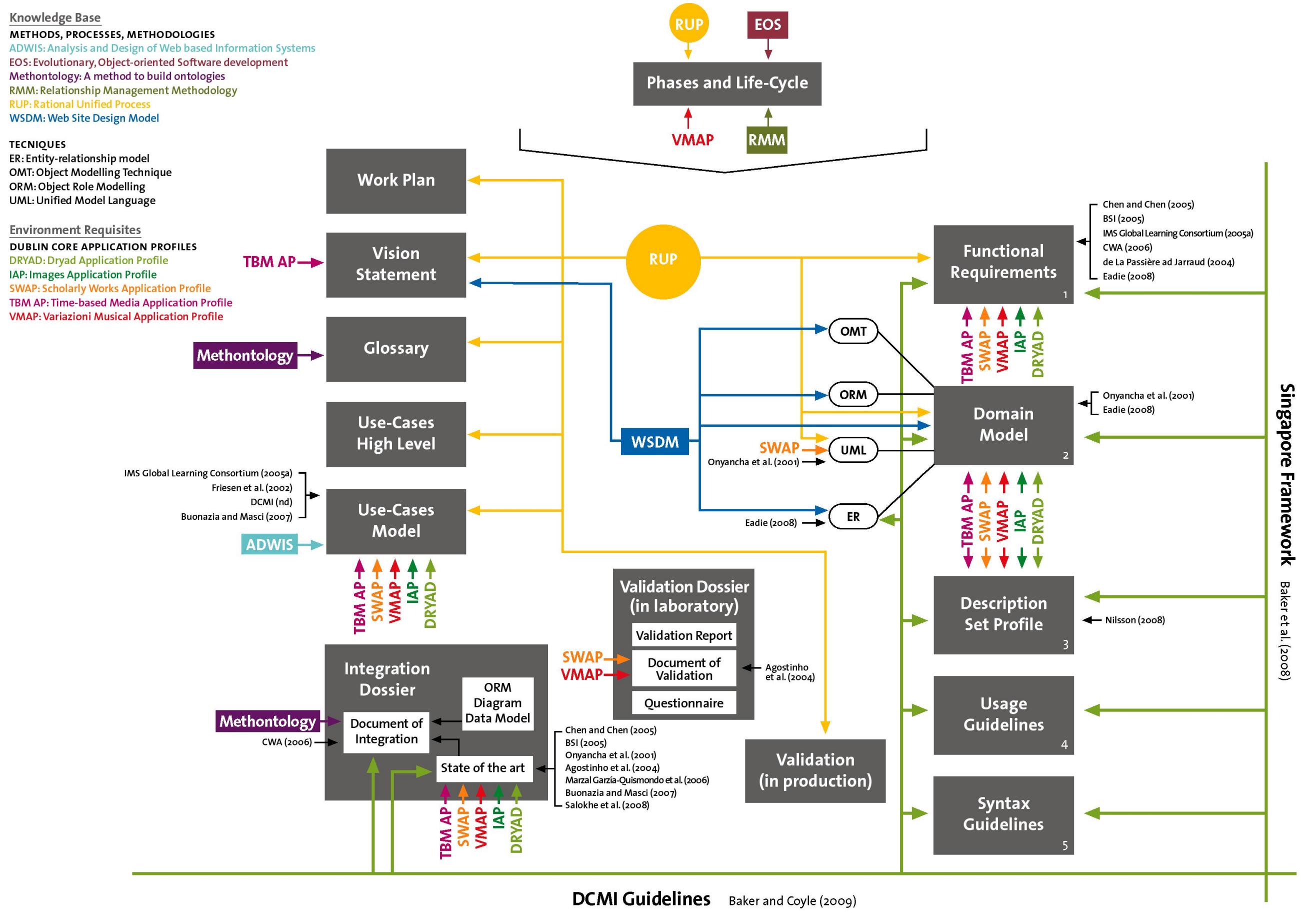

Figure 22: Overall view of the inputs in the Me4DCAP design from: (i) the knowledge base sources and; (ii) the requisites of the evironment

B Document of Integration

Table 1: Definition of Namespaces used

Title	Full Namespace URI	Prefix
Dublin Core Metadata Initiative Terms	http://www.purl.org/dcterms/	dcterms
Craftsmen's Catalgue	http://purl.org/craftsmen_catalogue/v1/	cc
Friend of a friend	http://xmlns.com/foaf/0.1/	foaf
Good Relations	http://purl.org/goodrelations/v1#	gr
Resource Description Framework	http://www.w3.org/1999/02/22-rdf-syntax-ns#	RDF

Table 2: Definition of Properties

Label	Class or Property	Definition	Range	Type of Range		Related description	Min	Max
Craftsman	**Class**	**foaf:agent**	n/a	n/a	n/a	**YES**	1	1
Name	Property	foaf:firstName	literal	Value String		no	1	1
Other Name	Property	foaf:family_name	literal	Value String		no	0	Unlimited
Gender	Property	foaf:gender	non-literal	VES [1]		no	1	1
Date of Birth	Property	foaf:birthday	literal	Value String	SES [2]	no	1	1
Country	Property	gr:location	non-literal	VES[3]		no	1	1
Currency	Property	gr:hasCurrency	non-literal	VES [6]		no	1	1
Email	Property	foaf:mbox	non-literal	Value URI		no	0	1
Annual Income	**Class**	**cc:annual_income**	n/a	n/a	n/a	**YES**	0	unlimited
Salary	Property	cc:salary	literal	Value String		no	1	1
Date	Property	dcterms:Date	literal	Value String	SES [2]	no	1	1

Label	Class or Property	Definition	Range	Type of Range		Related description	Min	Max
Product	**Class**	**gr:ProductOrService**	n/a	n/a	n/a	**YES**	0	unlimited
Name	Property	dcterms: title	literal	Value String	Language versions	no	1	1
Type	Property	rdf:type	non-literal	VES[4]		no	1	1
Description	Property	gr: description	literal	Value String	Language versions	no	1	1
Category	Property	gr:category	non-literal	VES[5]		no	1	unlimited
Shop	**Class**	**cc:shop**	n/a	n/a	n/a	**YES**	0	unlimited
Title	Property	dcterms: title	literal	Value String		no	1	1
Location	Property	gr:location	literal	Value String		no	1	1
Phone	Property	foaf:phone	literal	Value String		no	0	1
head	Property	cc:head	literal	Value String	Boolean DataType	no	1	1
Price	**Class**	**gr:UnitPriceSpecification**	n/a	n/a	n/a	**YES**	0	unlimited
Currency Value	Property	gr:hasCurrencyValue	literal	Value String		no	1	1
Quantity Eligible	Property	gr:hasEligibleQuantity	literal	Value String		no	1	1
Quantity Value	Property	gr:QuantitativeValue	literal	Value String		no	1	1
Profit	Property	cc:profit	literal	Value String		no	1	1

[1] http://example.com/craftsman_catalog/GENDER
[2] http://purl.org/dc/terms/W3CDTF
[3] http://id.loc.gov/vocabulary/countries
[4] http://www.productontology.org/
[5] http://example.com/craftsman_catalog/CAT/
[6] http://telegraphis.net/data/currencies/

C Document of Validation

Label	Value
Craftsman	***Link 1***
Name	João
Other Name	Curado
Other Name	Silva
Gender	http://example.com/craftsman_catalog/GENDER/Transgender
Date of Birth	1969-12-04
Country	http://id.loc.gov/vocabulary/countries/po
Currency	http://telegraphis.net/data/currencies/EUR#EUR
Email	mailto: joao.silva@example.com
Annual Income	
Salary	12000
Date	2011
Craftsman Link	Link1
Annual Income	
Salary	12150
Date	2012
Craftsman Link	Link1
Product	***Link2***
Type	http://www.productontology.org/id/Pillow
Description (en)	A special pillow for people with allergies
Description (po)	Uma almofada especial para pessoas alérgicas
Category	http://purl.org/craftsmen_catalogue/CAT/bed
Category	http://purl.org/craftsmen_catalogue/CAT/bedroom
Craftsman Link	Link1
Shop	***Link3***
Location	Porto
Phone	220137789
head	T
Shop	***Link4***
Location	Guimarães
Phone	2532345678
head	N
Price	
Currency Value	15
Quantity Value	1
Quantity Eligible	unit
Profit	5
Shop Link	Link3
Product Link	Link2
Price	
Currency Value	13
Quantity Value	1
Quantity Eligible	unit
Profit	5
Shop Link	Link4
Product Link	Link2

Resource 2

Label	Value
Craftsman	***Link1***
Name	Maria Alba
Other Name	Mejia
Gender	http://example.com/craftsman_catalog/GENDER/F
Date of Birth	1983-05-12
Country	http://id.loc.gov/vocabulary/countries/sp
Currency	http://telegraphis.net/data/currencies/EUR#EUR
Email	mailto: maria.mejia@yahoo.es
Annual Income	
Salary	15000
Date	2010
Craftsman Link	Link1
Annual Income	
Salary	13000
Date	2011
Craftsman Link	Link1
Annual Income	
Salary	13000
Date	2011
Craftsman Link	Link1
Product	***Link2***
Type	http://www.productontology.org/id/Soap
Description (en)	A soap made of lavender growing in my properties, with no chemicals.
Description (po)	Um sabonete especial, com alfazema que cresce nos meus campos. Sem químicos.
Category	http://purl.org/craftsmen_catalogue/CAT/personal-hygiene
Craftsman Link	Link1
Product	***Link3***
Type	http://www.productontology.org/id/Parfume
Description (en)	A perfume made of lavender growing in my properties, with no chemicals.
Description (po)	Um perfume especial, com alfazema que cresce nos meus campos. Sem químicos.
Category	http://purl.org/craftsmen_catalogue/CAT/personal-hygiene
Craftsman Link	Link1
Shop	***Link4***
Location	Sevilla
Phone	(95) 423 05 05
head	T
Shop	***Link5***
Location	Madrid
Phone	(91) 772 20 87
head	F
Shop	***Link6***
Location	Barcelona
Phone	(93) 451 19 14
head	F
Price	
Currency Value	2

Quantity Value	1
Quantity Eligible	unit
Profit	7
Shop Link	Link4
Product Link	Link2
Price	
Currency Value	2.3
Quantity Value	1
Quantity Eligible	unit
Profit	7
Shop Link	Link6
Product Link	Link2
Price	
Currency Value	4
Quantity Value	1
Quantity Eligible	unit
Profit	9
Shop Link	Link5
Product Link	Link2
Price	
Currency Value	15
Quantity Value	1
Quantity Eligible	unit
Profit	5
Shop Link	Link4
Product Link	Link3
Price	
Currency Value	100
Quantity Value	3
Quantity Eligible	unit
Profit	7
Shop Link	Link5
Product Link	Link3

D Description Set Profile. XML codification

```
<?xml version=”1.0” ?>
<DescriptionSetTemplate xmlns=”http://dublincore.org/xml/dc-dsp/2008/03/31” >

  <DescriptionTemplate ID=”Craftsman” minOccur=”1” maxOccur=”1”
standalone=”yes”>
    <ResourceClass>http://xmlns.com/foaf/0.1/agent</ResourceClass>

      <StatementTemplate ID=”Name”  minOccurs=”1” maxOccurs=”1” type=”literal”>
        <Property>http://xmlns.com/foaf/0.1/firstName</Property>
      </StatementTemplate>

      <StatementTemplate ID=”Other_Name”  minOccurs=”1” maxOccurs=”infinite”
type=”literal”>
        <Property>http://xmlns.com/foaf/0.1/family_name</Property>
      </StatementTemplate>

      <StatementTemplate ID=”Gender”  minOccurs=”1” maxOccurs=”1”
type=”nonliteral”>
        <Property>http://xmlns.com/foaf/0.1/gender</Property>
        <NonLiteralConstraint>

<VocabularyEncodingSchemeURI>http://example.com/craftsman_catalog/GENDER</Vocabu
laryEncodingSchemeURI>
        </NonLiteralConstraint>
      </StatementTemplate>

      <StatementTemplate ID=”Birthday” minOccurs=”1” maxOccurs=”1”
type=”literal”>
        <Property>http://xmlns.com/foaf/0.1/birthday</Property>
        <LiteralConstraint>

<SyntaxEncodingScheme>http://purl.org/dc/terms/W3CDTF</SyntaxEncodingScheme>
        </LiteralConstraint>
      </StatementTemplate>

      <StatementTemplate ID=”Country” minOccurs=”1” maxOccurs=”1”
type=”nonliteral”>
```

```
            <Property>http://purl.org/goodrelations/v1#location</
                Property>
            <NonLiteralConstraint>

<VocabularyEncodingSchemeURI>http://id.loc.gov/vocabulary/
    countries</VocabularyE
ncodingSchemeURI>
            </NonLiteralConstraint>
        </StatementTemplate>

        <StatementTemplate ID=”Currency” minOccurs=”1” maxOccurs=”
            1”
type=”nonliteral”>
            <Property>http://purl.org/goodrelations/v1#hasCurrency</
                Property>
            <NonLiteralConstraint>

<VocabularyEncodingSchemeURI>http://telegraphis.net/data/
    currencies/</Vocabulary
EncodingSchemeURI>
            </NonLiteralConstraint>
        </StatementTemplate>

        <StatementTemplate ID=”Email” minOccurs=”0” maxOccurs=”1”
type=”nonliteral”>
            <Property>http://xmlns.com/foaf/0.1/mbox</Property>
            <NonLiteralConstraint>
              <ValueURIOccurrence>mandatory</ValueURIOccurrence>
            </NonLiteralConstraint>
        </StatementTemplate>
    </DescriptionTemplate>

    <DescriptionTemplate ID=”Annual_Income” minOccur=”0” maxOccur=
        ”infinite”
standalone=”no”>

<ResourceClass>http://example.com/craftsman_catalog/
    annual_income</ResourceClass
>

        <StatementTemplate ID=”Other_Name” minOccurs=”1”
            maxOccurs=”1”
type=”literal”>
            <Property>http://example.com/craftsman_catalog/salary</
                Property>
        </StatementTemplate>

        <StatementTemplate ID=”Date” minOccurs=”1” maxOccurs=”1”
            type=”literal”>
```

```
          <Property>http://www.purl.org/dcterms/Date</Property>
          <LiteralConstraint>

<SyntaxEncodingScheme>http://purl.org/dc/terms/W3CDTF</
    SyntaxEncodingScheme>
          </LiteralConstraint>
        </StatementTemplate>
   </DescriptionTemplate>

   <DescriptionTemplate ID=”Product” minOccur=”0” maxOccur=”
       infinite”
standalone=”no”>

<ResourceClass>http://purl.org/goodrelations/v1#ProductOrService
    </ResourceClass>

        <StatementTemplate ID=”Name”  minOccurs=”1”  maxOccurs=”1”
            type=”literal”>
          <Property>http://www.purl.org/dcterms/title</Property>
          <LiteralConstraint>
            <LanguageOccurrence>optional</LanguageOccurrence>
          </LiteralConstraint>
        </StatementTemplate>

        <StatementTemplate ID=”Type” minOccurs=”1” maxOccurs=”1”
type=”nonliteral”>
          <Property>http://www.w3.org/1999/02/22-rdf-syntax-ns#
              type</Property>
          <NonLiteralConstraint>

<VocabularyEncodingSchemeURI>http://www.productontology.org/</
    VocabularyEncoding
SchemeURI>
          </NonLiteralConstraint>
        </StatementTemplate>

        <StatementTemplate ID=”Description”  minOccurs=”1”
            maxOccurs=”1”
type=”literal”>
          <Property>http://purl.org/goodrelations/v1#description</
              Property>
          <LiteralConstraint>
            <LanguageOccurrence>optional</LanguageOccurrence>
          </LiteralConstraint>
        </StatementTemplate>

        <StatementTemplate ID=”Category” minOccurs=”1” maxOccurs=”
            infinite”
type=”nonliteral”>
```

```
        <Property>http://purl.org/goodrelations/v1#category</Property>
        <NonLiteralConstraint>

<VocabularyEncodingSchemeURI>http://example.com/craftsman_catalog/CAT/</Vocabula
ryEncodingSchemeURI>
        </NonLiteralConstraint>
      </StatementTemplate>
  </DescriptionTemplate>

  <DescriptionTemplate ID="Shop" minOccur="0" maxOccur="infinite"
standalone="no">
      <ResourceClass>http://example.com/craftsman_catalog/shop</ResourceClass>

      <StatementTemplate ID="Title"  minOccurs="1" maxOccurs="1" type="literal">
        <Property>http://www.purl.org/dcterms/title</Property>
      </StatementTemplate>

      <StatementTemplate ID="Location"  minOccurs="1" maxOccurs="1"
type="literal">
        <Property>http://purl.org/goodrelations/v1#location</Property>
      </StatementTemplate>

      <StatementTemplate ID="Phone"  minOccurs="1" maxOccurs="1" type="literal">
        <Property>http://xmlns.com/foaf/0.1/phone</Property>
      </StatementTemplate>

      <StatementTemplate ID="head"  minOccurs="1" maxOccurs="1" type="literal">
        <Property>http://example.com/craftsman_catalog/head</Property>
      </StatementTemplate>
  </DescriptionTemplate>

  <DescriptionTemplate ID="Price" minOccur="0" maxOccur="infinite"
standalone="no">

<ResourceClass>http://purl.org/goodrelations/v1#UnitPriceSpecification</Resource
Class>

```

```
        <StatementTemplate ID=”Currency␣Value”  minOccurs=”1” maxOccurs=”1”
type=”literal”>
          <Property>http://purl.org/goodrelations/v1#hasCurrencyValue</Property>
        </StatementTemplate>

        <StatementTemplate ID=”Quantity␣Eligible”  minOccurs=”1” maxOccurs=”1”
type=”literal”>
          <Property>http://purl.org/goodrelations/v1#hasEligibleQuantity</Property>


        </StatementTemplate>

        <StatementTemplate ID=”Quantity␣Value”  minOccurs=”1” maxOccurs=”1”
type=”literal”>
          <Property>http://purl.org/goodrelations/v1#QuantityValue</Property>
        </StatementTemplate>

        <StatementTemplate ID=”Profit”  minOccurs=”1” maxOccurs=”1”
type=”literal”>
          <Property>http://example.com/craftsman_catalog/profit</Property>
        </StatementTemplate>
    </DescriptionTemplate>
<\DescriptionSetTemplate>
```