Me4DCAP V0.1: a method for the development of Dublin Core Application Profiles

Mariana CURADO MALTA [a,1], Ana Alice BAPTISTA [a]

[a] *University of Minho, Algoritmi Center, Portugal*

Abstract. Recent studies show that there is no method to develop a Dublin Core Application Profiles (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. This paper presents the first version of a method to develop Dublin Core Application Profiles (Me4DACP V0.1) that has been developed in a PhD project with a Design Science Research (DSR) approach. Me4DCAP was built having as starting point the Singapore Framework for DCAP and shows the way through the DCAP development. It encompasses a group of pre-defined interconnected activities, explicitly states when they should take place, what techniques coul be used to execute them and what artifacts should result from their execution.

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"[1] that started to emerge in 1999. The Semantic Web is about common formats for integration and combination of data from different sources [1]. Metadata, both in its use and in its definition and description, is present at various levels. Metadata is data that describes resources with information [2] 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" [2, p. 4].

The Dublin Core Metadata Initiative[2] (DCMI) created new instruments so that those involved in the definition of metadata descriptions 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) [3] 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 is the Dublin Core Application Profile (DCAP), "a generic construct for

---

designing metadata records" [4]. The Singapore Framework for Dublin Core Application Profiles" recommendation - c.f. [5] - 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[3] 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. [5] and [4] - and they are too brief. In fact, recent studies show that there is no formal method to develop a DCAP [7]. 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 have a method for the development of DCAP.

The work detailed in this article is framed in a PhD project that has as goal to develop such a method. In this article we present a condensed description of the first version (V0.1) of the method for the development of DCAP (Me4DCAP). A report with a detailed description of Me4DCAP is available at `http://hdl.handle.net/1822/23537`. This research is still in progress. This article is organised in 4 sections. In Section 2 we present the Design Science Research methodological approach. In Section 3 we present the first version of our results, i.e., Me4DCAP V0.1. Finally, conclusions and future work are drawn in section 4.

2. Design Science Research methodological approach

Our work is based on a design science research (DSR) approach. Design Science aims at the development of innovative artifacts that solve real-world problems [8], thus "Design Science is inherently a problem solving process" [9, p. 82]. An artifact is something that is artificial, constructed by humans [10], a project using a DSR methodology produces artifacts that can be either constructs, models, methods, or instantiations of one of these 3 artifacts [11]. "Artifacts must be improved upon existing solutions to a problem or perhaps provide a first solution to an important problem" [10, p.6]. Our research project will develop a first solution to a problem: a method for the development of DCAP (Me4DCAP) to provide the metadata community with a method that, to the best of our knowledge, does not exist.

According to A. Hevner [11] DSR has 3 cycles: (i) the "Relevance Cycle" that works in the "Environment"; (ii) the "Design Cycle" that works in the core activities of building the artifact, and; (iii) 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" [11, p. 89].

In the Design Cycle DSR "calls for multiple iterations (...) before contributions are output into the Relevance Cycle and the Rigor Cycle" [11, p. 91]. These iterations are cycles of construction and evaluation, and "these evaluation moments are performed in laboratory and experimental situations" [11, p. 91].

---

[3]A method is a selection of techniques, the control of their usage and the integration of the obtained partial results [6].

In the Rigor Cycle DSR uses as input for the Design Cycle the knowledge base "of scientific theories and engineering methods that provides the foundation" [11, p. 89] and the DSR project feeds back the knowledge base with new artifacts and "the experiences and expertise that define the state of the art in the application domain" of the DSR research project.

3. Description of Me4DCAP V0.1

3.1. 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 [12] and RDF 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.

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 multidisciplinary teams it is important that the members of the work team speak all a common language, as it avoids misunderstandings and improves communication [13].

3.2. Me4DCAP approach

Me4DCAP was built having 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);
- Syntax guidelines (optional) (Component Stage 5).

The starting point in the Knowledge Base is the Rational Unified Process (RUP) (c.f. Kruchten (2004)). Me4DCAP establishes the way through the DCAP development: it establishes when activities must take place, how they interconnect, and which artifacts they will bring about; it also suggests which techniques could be used to build these artifacts.

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 our 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.

Me4DCAP has 4 phases (see figure 2): Scope Definition, Construction, Development and Validation. These phases are traversed along the project development as the Singapore Components Stage are being developed.
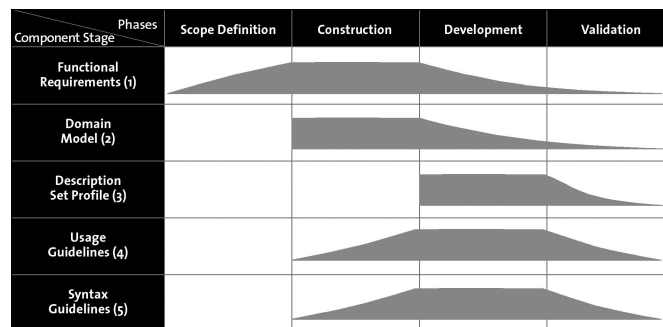
Phases / Component Stage	Scope Definition	Construction	Development	Validation
Functional Requirements (1)				
Domain Model (2)				
Description Set Profile (3)				
Usage Guidelines (4)				
Syntax Guidelines (5)				

Figure 1. The Me4DCAP V0.1 phases

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.

3.3. *The Me4DCAP life-cycle development model*

Throughout a DCAP development, artifacts are being produced to help to reach the aforementioned Component Stages. Figure 2 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 sections 3.6 and 3.8).
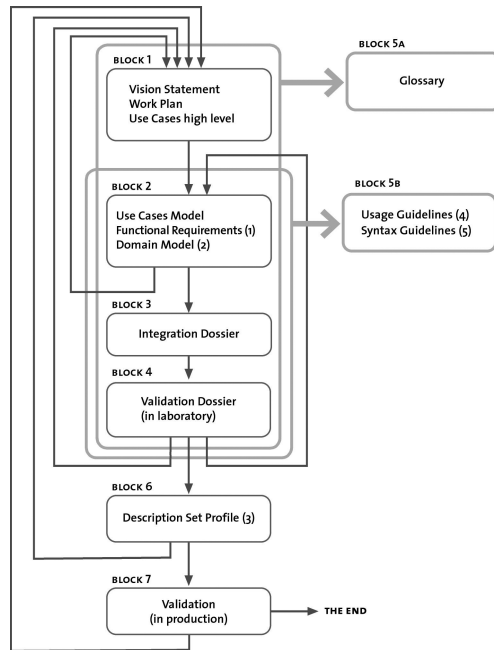
Figure 2. The Me4DCAP V0.1 life-cycle development model and the Me4DCAP artifact sequence
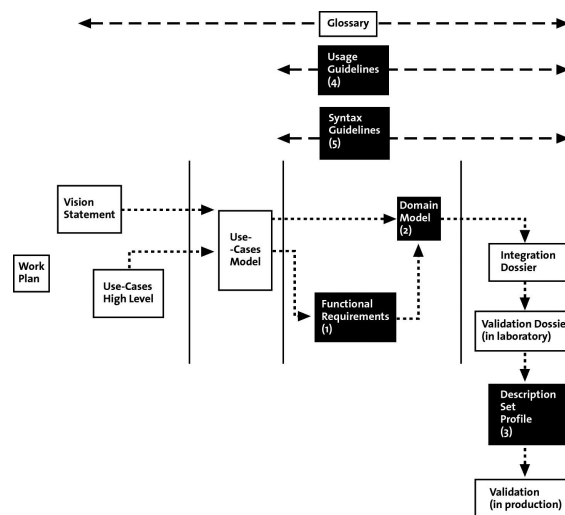
Figure 3. The dependence among the artifacts in Me4DCAP V0.1

In order to develop a DCAP, the DCAP developers will need to follow the life-cycle of the development process, building the artifacts that will be used for the construction

of the Component Stages (see figure 2). Some artifacts 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. Figure 3 shows the dependence among the artifacts.

Next section will describe each Me4DCAP artifact and the techniques that should be used to develop them. This description follows the Singapore Framework Component Stages order; they are the center of all development.

3.4. Functional Requirements (Component Stage 1)

To build the Functional Requirements Component Stage Me4DCAP defines the need to develop a set of 4 mandatory artifacts: the Vision Statement, the Work Plan, the Use Cases High Level, and the Use-Cases Model. The first artifacts to be build are the Vision Statement, the Work Plan and the Use Cases High Level. After that, follows the Use-Case Model. 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. 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 in a board (physical board or web tool), followed by discussion. In the end, the set of ideas chosen should be organized in simple sentences.

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 artifact 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 the members of the work team, and negotiated among them in order to fit all the members of the team time constraints. It is acceptable that the Work Plan has to be modified as the project evolves.

The Use-Cases Model is build after these 3 artifacts previously described are developed. Having Me4DCAP an iterative life-cycle development model, the previous artifacts 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 artifacts to follow the process, being aware that they will be working on those draft versions sometime later in the process.

Use Cases "offer a systematic and intuitive means of capturing functional requirements" [13, p. 37]. The Use-Cases will be used to develop the Functional Requirements and to understand the objects (and attributes) of the system to be studied. 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 [14];
- the set of all detailed Use-Cases.

For information on how to built an UML Use-Case diagram see [15] or [13].

"Each use-case must include details about what has to be done to achieve its functionality" [16, p.21] 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 [14]. An Use-Case description is a flow of events description, and it should be developed using the template proposed by [16]. 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 jmanagers, giving them the template of the flow-of-events defined by [16] 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 artifacts developed, the Functional Requirements can be built. This Singapore Component Stage 1 is mandatory. 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" [4]. The Functional Requirements Component Stage 1 is a text document, where general goals are mentioned as well as specific tasks [4]. To develop the Functional Requirements the work-team should read, in group, the Use-Cases detailed description 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. Certain ideas speak more to some work-team members than to others so, each functional-requirement- idea should be distributed accordingly, in order to satisfy every member's needs specific requirements. Every member of the work-group should write some sentences describing more deeply the requirement-case that is responsible for. In the end of the process, all the requirements-cases should be put together on the board (physical board or web tool), and the whole group should discuss and review the final result.

3.5. Domain Model (Component Stage 2)

The Domain Model is the mandatory Singapore Component Stage 2. It "captures the most important types of objects in the context of the system." [13, p. 119]. According to [4] "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 build based on the Functional Requirements Component Stage 1 and on the Use-Cases Model artifact described in Section 3.4. The domain model development can also use the help of other techniques depending on the DCAP development context. In situations where access to documentation or the information systems databases is available, it is possible to resort to the Document Analysis technique to define it as well.

Me4DCAP suggests that the Domain Model should be developed using an UML class diagram with details suppressed. The diagram identifies the classes of objects and the relationships among them but the classes' methods and attributes are omitted since the methods' definition is not in the frame of a DCAP development, and the attributes will

be defined in the ORM Diagram data model (see next section). The Entity-Relationship diagram [17] showing the entities and the relationships among them can be an alternative to the UM class diagram technique (attributes should be omitted).

3.6. 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:

- The Integration Dossier;
- The Validation Dossier (in laboratory).

The Integration dossier comprises 3 artifacts: an Object Role Modeling (ORM/NIAM) [18] diagram data model, a State Of The Art and a Document of Integration. All these artifacts are mandatory.

The ORM diagram is a data model with:

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

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 3.7 for details about this artifact). This work should be done by the stakeholders.

After defining the ORM Diagram data model, Me4DCAP defines as next step the application of a metadata scheme property to every attribute of the objects of the Domain 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 attributes with the existing metadata schemes, those attributes should be described with new properties; these new properties have to be created. According to Baker & Coyle (2009) [4], 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;
- To create new properties - in case there are no properties on the metadata schemes of the state of the art to describe some of the identified attributes. 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 will be used to build a Document of Integration. This Document shows, in a matrix, per line, every attribute and its constraints, described by the properties of the metadata schemes and encoding schemes chosen. This work should be done by the metadata programmers of the work-team. An example based on the DCMI Guidelines (c.f. [4]) can be downloaded from the repository of University of Minho accessible through the URL `http://hdl.handle.net/1822/`. 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 artifacts: a Validation Report, a filled-in Matrix and a filled-in Questionnaire.

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

1. 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 3.4). The work-team should make a report (text document) with the conclusions of the meeting and recommendations.

2. 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:

   (a) 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 validation matrix with data referring to each resource. The matrix template should be simple to fill in, where each element of the metadata is populated with the data that corresponds to the resource. This matrix should be accompanied by the 2 Guidelines Component Stage 4 and Component Stage 5. An example of a validation matrix can be downloaded from the repository of University of Minho accessible through the URL http://hdl.handle.net/1822/. The example used is the same as in the Document of integration (see previous point);

   (b) Answer to a set of questions. The final users chosen to do the application of the DCAP to a resource sample (see previous point) 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:

      i. Could you describe all your data with the available elements? If not, please refer the difficulties;
      ii. 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?;
      iii. Did you have any difficulty in particular to describe your data? Were there any ambiguities?;
      iv. 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 2).

The Singapore Component Stage 4 DSP is mandatory. The task is to detail the metadata developing their design in the DSP language defined by Nilsson (2008). Further information, including implementation examples, can be found in Baker & Coyle (2009) [4]. This task should be performed by the metadata programmers of the work-team.

3.7. Guidelines (Component Stage 4 and Component Stage 5)

These guidelines are not mandatory in the Singapore Framework. Me4DCAP does not make them mandatory but recommends that they are developed since it helps the final users of the DCAP application to apply correctly the properties and constraints.

Me4DCAP defines that the developing of the Usage Guidelines starts at the same time as the Domain Model. DCMI Guidelines [4] explain: "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" [4] . For detailed information see [4] and [5] .

The development of the Syntax Guidelines needs that the Integration Dossier is developed in a certain stage. This artifact describes "any application profile-specific syntaxes and/or syntax guidelines, if any" [4]. For detailed information see [4] and [5] .

The Usage Guidelines can be developed by both types of members of the work-team, the stakeholders 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.

3.8. Finishing the DCAP development

A validation in production of the DCAP should be performed. 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 2), and every artifact should be checked against this new information.

4. Conclusions and future work

A Dublin Core Application Profile is a very important construct to implement interoperability in a community of practice. Previous studies [7] has shown that there is no method to develop such a construct. 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. Thus, this is a problem that the metadata community has faced; this community needs a common ground of work concerning the development of DCAP.

This paper presents version 0.1 of a method for the development of DCAP (Me4DCAP), based on PhD research in progress project that uses a Design Science Research methodological approach. Me4DCAP establishes the way through the DCAP development: it establishes when activities must take place, how they interconnect, and which artifacts they will bring about; it also suggests which techniques could be used to build these artifacts. Me4DCAP defines a way for the construction of each Singapore Framework [5] component.

As future work we envision the validation process of Me4DCAP under the scope of the DSR approach; this validation is a very important tool to access the adequacy of Me4DCAP to the needs of the metadata community application domain. These validation process will be done using the Focus Group approach and we expect to find projects in developing DCAP using Me4DCAP V0.1 for feedback and fine-tuning A new version of Me4DCAP (V0.2) will be developed, integrating the information reported from the validation process.

Acknowledgements

References

[1] W3C, "Semantic web." `http://www.w3.org/standards/semanticweb/`, 2012. Accessed in 14 Jan 2012. Confirmed 9 Feb 2013.

[2] N. Press, "Understanding metadata," *NISO Press Booklets*, vol. 1, 2004.

[3] A. Powell, M. Nilsson, A. Naeve, T. Baker, and P. Johnston, "Dcmi abstract model." `http://dublincore.org/documents/2007/06/04/abstract-model/`, 06 2007. Accessed in 26.06.2011.

[4] T. Baker and K. Coyle, "Guidelines for dublin core application profiles." http://dublincore.org/documents/profile-guidelines/, 05 2009. Accessed in 26 Jun 2011. Confirmed 1 Feb 2013.

[5] T. Baker, M. Nilsson, and P. Johnston, "The singapore framework for dublin core application profiles." `http://dublincore.org/documents/singapore-framework/`, 01 2008. Accessed in 26 Jun 2011. Confirmed 9 Feb 2013.

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

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

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

[9] A. Hevner, S. March, J. Park, and S. Ram, "Design science in information systems research," *MIS quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

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

[11] A. Hevner, "The three cycle view of design science research," *Scandinavian Journal of Information Systems*, vol. 19, no. 2, p. 87, 2007.

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

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

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

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

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

[17] P. Chen, "The entity-relationship model—toward a unified view of data," *ACM Transactions on Database Systems (TODS)*, vol. 1, no. 1, pp. 9–36, 1976.

[18] T. Halpin, "Business rules and object-role modeling," *Database Programming and Design*, vol. 9, pp. 66–72, 1996.