

Problem Analysis for Referential Requirements Specifications

João Guilherme de Almeida Henriques Goucha Gaspar

1 Introduction

As the knowledge and practices on specific views of information systems stabilize and it begins to be possible to provide references, there is a natural motivation to a demand and supply of systems or components for these systems that already incorporate those references to the detriment of custom development. The referential specification MoReq2010 is an example of one of those cases, which aims to provide a set of basic requirements that a document management system must meet. However, this specification was developed primarily by specialists in document management, through expedited procedures without clear application of the techniques, methods or fundamental practices of requirements engineering (RE), which is recognized as a high risk for potential inconsistencies, contradictions and gaps. The fundamental concepts of RE imply that a requirements specification should be complete and consistent and their requirements must be well formed, fulfilling a set of quality criteria, including no ambiguity, atomicity, completeness and comprehensibility. In this project the problem handled was the analysis of a referential requirements specification and to determine the degree of alignment with the best practices of RE. With an automatic analysis of the specification MoReq2010 it was possible to analyze some aspects of the quality of its requirements. Thus heuristics were developed according to the RE good practices, aiming to support the analysis of the form of the requirements as well as the dependency relationships between them. This allowed the identification of malformed requirements and inconsistencies and, with the results, propose solutions to improve the specification.

2 Requirements Engineering

The **requirements engineering (RE)** is the field that seeks to define the context and scope of the problem of a project from the systematization of a set of activities such as **elicitation, specification, documentation, analysis, negotiation, validation** and **management of requirements**. All those activities have a purpose in the RE and therefore it is essential that all of them are part of an efficient and appropriate process of requirements engineering. The absence of any of the activities in the process can cause delays, budget overruns, increased maintenance costs, or in the worst case, result in a system that does not support the real needs of users and business.

The success of the activities of RE depends on the identification and proper use of the artifacts recognized in this business area.

The nuclear artifact of this domain is the **requirement**. A requirement can be defined as [1]: 1) a condition or capability needed by a user to solve a problem or achieve an objective, 2) a condition or capability that must be complied with or owned by a system or component to satisfy a contract, standard, specification or other mandatory document and 3) a documented representation of a condition or capability described in 1) and 2).

2.1 Requirements Quality

The **quality** of the requirements is essential to define the quality of a specification (in addition to the quality of the specification structure and of the specification as a whole). The following criteria for evaluating the quality of the requirements are considered:

- **Traceable:** the requirement is traceable if its use, evolution, impact and source are traceable at any moment of the development process;
- **Correct:** the requirement is correct if the stakeholders accept it as such;
- **Consistent:** a requirement is consistent if its expression does not contradict an expression of other requirement;
- **Verifiable:** a requirement is verifiable if a stakeholder can verify that the developed system meets the requirement expressed in the specification;
- **Classified:** a requirement is classified if its relevance and / or stability have been determined and documented;
- **Current:** a requirement must be updated to reflect the current state of the system and its context, such as legal regulations or other requirements inherent to the business;
- **Expressiveness:**
 - **Completeness:** the requirement cannot omit any information that is important for the stakeholders;
 - **Unambiguous:** the requirement can only have one possible understanding. Otherwise it may not be implemented as desired;
 - **Comprehensibility:** the text of the requirement must be simple and easy to understand;
 - **Atomicity:** a requirement should only represent a single need. If it is possible to divide a requirement expression into two or more expressions, the requirement is not atomic.

2.2 Quality of a requirements specification

You can define **quality criteria** for each requirement individually or for the specification as a whole. The quality of a requirements specification can be assessed by its structure and by its content.

Considering the **content**, a requirements specification must be **complete, consistent, legible** and **modifiable** [2]. Considering the **structure**, there are several different recommendations of referential structures. The one proposed by standard IEEE 830-1998 states that a requirements specification must be divided in three different parts. Its description can be seen in Table 1.

Table 1 – Standard IEEE 830-1998

Structure		Description
Part 1: Introduction	Purpose	The motivation for the specification and identification of the target audience.
	Scope	Name of the software product, describing its benefits, objectives, and goals.
	Definitions, acronyms and abbreviations	Terms required to fully understanding the specification.
	References	List of references used in the specification.
	Overview	Overview of the contents and structure of the requirements specification.
Part 2: Description	Product perspective	Describes the dependencies and relations with other products. If the software is part of a larger system, then this section should make explicit the relation to the broader system by identifying interfaces between them.
	Product functions	A summary of the major functionalities that the system will perform.
	User characteristics	Describes the general characteristics of the users of the product.
	Constraints	Defines general constraints that limit the developer's options, such as regulatory policies, hardware limitations, and interfaces to other applications.
	Assumptions and dependencies	List of all the factors on which the document content's rely, since changes in these factors can have a strong impact on the requirements stated in the SRS.
Part 3: Requirements		Agglomerates all the requirements of the document. These are organized into non-functional and functional sections. The latter can be organized depending on the usage mode, the user, the related entities, features, among others.

3 MoReq – Modular Requirements for Records Systems

MoReq stands for "Modular Requirements for Records Systems". The MoReq was developed by the DLM-Forum with the cooperation and support of the European Commission, to be used in public or private sectors that wish to acquire or evaluate a records management system.

A first version was published in 2001. In 2008 the second version was launched, MoReq2, which differed from the first one fundamentally by assuming a broader context and including test specifications (test framework) and certification (an XML Schema schema)

The MoReq2010, published in June 2010, presents a new structure and content when compared to its predecessors, and now consists of independent modules, each likely to evolve independently of the other [3]. The document is divided into chapters, beginning by presenting "Important Information", the "Context," the "Purpose" and "Key Concepts". After the introductory chapters the functional requirements are then presented (eleven chapters) followed by nonfunctional requirements (fifteen chapters). Finally a number of series of requirements are shown, each with one or more modules. At least one of the modules in each series has to be fulfilled in order to make a system recognized as compatible with the MoReq2010. Table 2 presents the description of each of the MoReq2010's chapters with functional requirements.

Table 2 – Functional Requirements Chapters of MoReq2010

Chapters	Description
2. System Services	Expresses features that are common to all MoReq2010 services; presents a choice of one of the modules from 100 Series - Types of interface;
3. User and Group Service	Expresses the requirements related to users and user groups management;
4. Model Role Service:	Expresses the features related to how users are authorized to execute a function of the MCRS. For example this chapter extends the concept of CRUD, as in document management and clarifies the distinction between destroying and erasing.
5. Classification Service	Aggregates requirements related to the classification scheme, and the classification of records; presents the possibility of choosing one of the modules in the 200 series - Classification Series. It relates to the chapters 6. Record Service and 8. Disposal Scheduling Service;
6. Record Service	Aggregates requirements related to the aggregation of records, including hierarchies of aggregations, and its relations with the processes / business activities and classifications;

7. Model Metadata Service	Describes functionalities that facilitate the interoperability between different systems. The aim is to propose the implementation of a model which can be recognized by any MCRS, in order to export and import documents and their metadata easily;
8. Disposal Scheduling Service	Aggregates requirements that define the functionalities for managing the life cycle of records in a MCRS;
9. Disposal Holding Service	Aggregates requirements that define the features related to disposal hold, from the imposition of the suspension, the impact of the suspension and until its termination;
10. Searching and Reporting Service	Aggregates requirements that define the features related to the searching of objects in a MCRS, expressing different methods, different ways of reporting, as well as features related to the generation and presentation of reports;
11. Exporting Service	Aggregates requirements that define the features related to exporting content, metadata, historical events and access controls. These data must be, migrated, replicated;

A records management system that complies with MoReq2010 specification is called MCRS (acronym for "MoReq2010 Compliant Record System"). For a system to be a MCRS it must meet a number of nuclear requirements and must meet at least one of the modules of each series of requirements. This allows that quite a number of totally different systems can all be MCRS, despite those differences.

4 Problem Analysis

The problem focuses on the need to bring the best practices of requirements engineering to requirements specifications developed by people expert in business but not in requirements engineering. In this context it is necessary to study different heuristics to evaluate the quality of requirements and specifications, and analyze the MoReq2010 in the light of that study.

4.1 Heuristics for Requirements and Requirements Specification Analysis

The Table 3, inspired by [4], expresses a number of quality criteria (for requirements and for specification) and possible heuristics that can be applied to evaluate the criteria in an automatic process mode (this was a preferable condition; however, in some of the criteria, it was not possible to identify heuristics with automatic processing).

Table 3 - Heuristics for Requirements and Requirements Specification Analysis

Requirements Engineer		Heuristics		
Requirements should be	Complete	There are words that need to be combined with others, that define them. For example flow (data, control, ...), access (write, remote, authorized).		
	Traceable	Check links that refer other parts of the specification		
	Unambiguous	Verification of words that can be subjective, as similar, if possible, easy, of course, good bad, etc..		
	Readable	Identification of requirements which are too long, of the excessive use of the words "and" and "or", use of personal or demonstrative pronouns (this, that, he, they) and propositions (up, down, etc.).		
	Consistent	Check the references of the requirements, guarantying that they point to existing sections		
	Atomic	Ensure that the requirements are short		
	Verifiable	See connection of the requirements with objects in the architecture		
	Classified	Enable the use of attributes. Associating attributes of state, for example, may help ensure the desired properties.		
Current				
Requirements specification should be	Complete	Check whether there is the possibility of adding new information, either totally new or additional (not automatic)		
	Consistent	The use of Glossary, the connection between different concepts will help maintain consistency. Check for a glossary and ensure it is used.		
	Readable and editable	The structure of the document must be edited;		
	Unambiguous	lexically		
		syntactically		Generate all possible syntactic trees for each sentence
semantically				
referentially		Find links that may refer to more than one object		

4.2 MoReq2010 Analysis from Requirements Engineering Perspective

In this section it is made an **analysis** of the **structure** of **MoReq2010** and a **verification** of some of its requirements.

From the analysis of Table 1, which describes the structure proposed by the standard IEEE 830-1998 [5], and crossing this information with the description of MoReq2010, we can establish the relationship between them, trying to understand in which MoReq2010's chapter is the information referenced by the mentioned standard. The relation is expressed in Table 4.

Table 4 – Relation between MoReq 2010 and the Standard IEEE 830 - 1998

Standard IEEE 830 - 1998	MoReq201
Introduction	Glossary (chapter 13) and the introduction section in each chapter (2.1, 3.1, 4.1, etc.)
Descriptions	Fundamentals (chapter 1) and Information Model (chapter 14)
Requirements	Requirements Chapters (2.3, 3.3, 4.4, etc.)

Analyzing a small sample of random MoReq2010 requirements and attempting to verify its proper formation, the information partially shown in Table 5 is obtained. There are **malformed requirements**, following almost any criteria. The considered criteria are related to the **expressiveness** of the requirement and other criteria, such as consistency, timeliness and verifiability, are therefore ignored in this analysis.

Table 5 resumes the analysis and validation of a small random sample of no well-formed requirements from MoReq2010:

- R2.4.3 is **considered non atomic** as it expresses two needs: "browsing services (...)" and "inspect metadata". It is also not **clear**, as we must avoid referring to any subject without expressing that subject explicitly: "of each" may be about the services, but can't it be about the users, or the system itself?
- R4.5.08 expresses three different needs, which makes it **non-atomic** and **unclear**. It is hard for a requirement to be clear if it is too long, as this one.
- R6.5.20 is also **non-atomic** and **unclear**: the subject in "its Contextual metadata (...)" is probably the component, but it can also be the user or the system.
- R8.4.14 is **non-atomic**, as it express more than one need, and **unclear**, as it offers different alternatives, in a confused way. How should we understand "immediately"? And if the "update" is required to be done periodically, which period must be considered, besides "daily"?
- R10.4.16 is **ambiguous**, as it makes use of the term "several" in a way that it cannot be verified.

We can extrapolate this analysis to the overall document, where the atomicity and clearness are not frequent, if we consider the rigorous metrics of RE. However, it is important to notice that MoReq2010 is more normalized than the previous ones, as all the analysed expressions begin with statements such as "The MCRS must ...", a structural worry not evident in the previous versions. This improvement offers a much more clear text, with much more comprehensiveness.

Table 5 – Malformed Requirement from MoReq2010

Requirements	Expression	Atomic	Complete	Comprehensive	Non Ambiguous
R2.4.3	The MCRS must allow an Authorised user to Browse across its services, or bundles of services under R2.4.1, and Inspect the Metadata of each as listed under R2.4.2.	No	Yes	No	Yes
R4.5.08	The MCRS must automatically Create an Access control list (D14.3.2) for each Service, or bundle of services under R2.4.1, and for each Entity in the MCRS where so specified, with the following Metadata: Include Inherited Roles Flag (M14.4.43). Each Access control list also has: Access control entries for that Entity.	No	Yes	No	Yes
R6.5.20	The MCRS must allow an Authorised user to Modify the Title and Description of an active Component, and any of its Contextual metadata.	No	Yes	No	Yes
R8.4.14	The MCRS must update the disposal status of any Record when requested by an Authorised user and, either immediately or periodically, and at least daily, the MCRS must update the disposal status of all active records.	No	Yes	No	No
R10.4.16	The MCRS must allow a User to combine, chain, or join, the results of several Search queries so as to answer Complex search enquiries.	Yes	Yes	Yes	No

5 Technological Solution Proposed

To solve the problem presented, the **use cases** shown in figures 1, 2 and 3 and described in Table 6 were considered. These use cases led to the development of the system described in Section 5.1, which shows the **components** and **services** that accomplish these use cases.

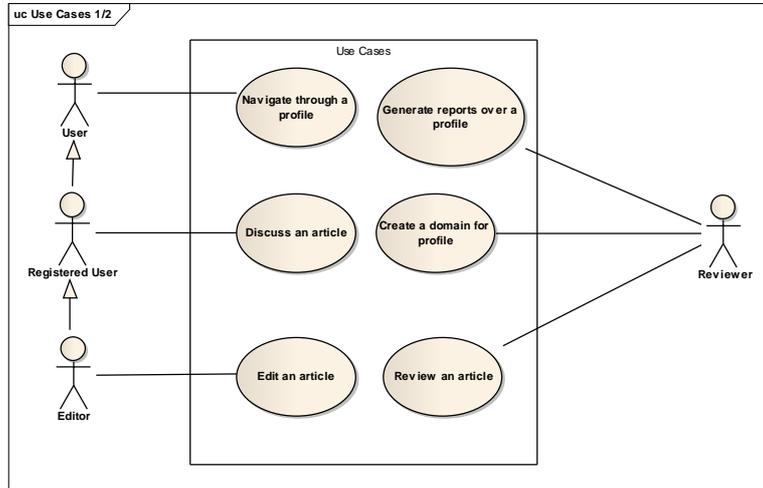


Figure 1 - Uses cases and actors

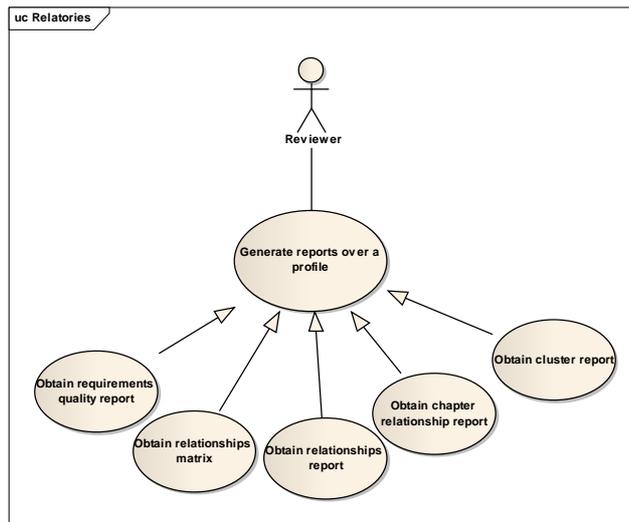


Figure 2 - Use case "Generate reports over a profile"

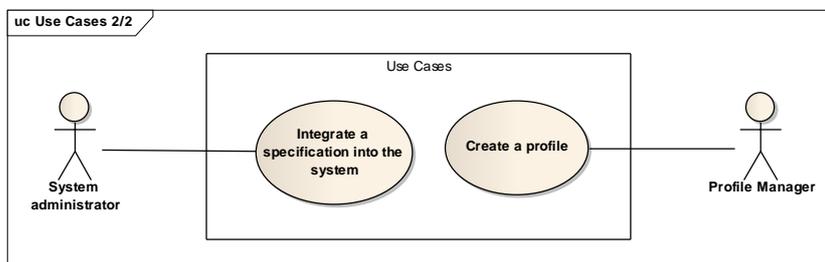


Figure 3 - Use cases for administrative actors

Table 6 – Use cases

Use cases	Goal
Edit an article	Edit an article Requirements Portal
Create a domain for profile	Create a domain for a profile, maintaining the relationship with the domain from which the first profile descends
Review an article	Accept, reject and / or comment on a particular version of an article

Navigate through a profile	Navigate through the articles in a given profile; navigation in the Portal;
Discuss an article	Discuss the choices made to define a requirement or to develop any other article;
Generate reports over a profile	Get the system to generate a report. This use case can be a generalization of five others.
Obtain relationships matrix	Get a matrix that expresses direct relationships between various requirements.
Obtain relationships report	Get a report that expresses the direct and indirect relations that each requirement has with the others
Obtain chapter relationship report	Get a report that expresses direct links between the chapters
Obtain cluster report	Get a report that expresses cluster of requirements
Obtain requirements quality report	Get a report that presents recommendations for improve requirements quality
Integrate a specification into the system	Integrating a document with a specification in the system, automatically. (process whereby MoReq 2010 was integrated into the system);
Create a profile	Create a new profile

5.1 Services and Components

This section presents the services provided by the system, as shown in Figure 4. It is described the system **services** and **components** that support them (Integrator, Portal Requirements and Quality Management System).

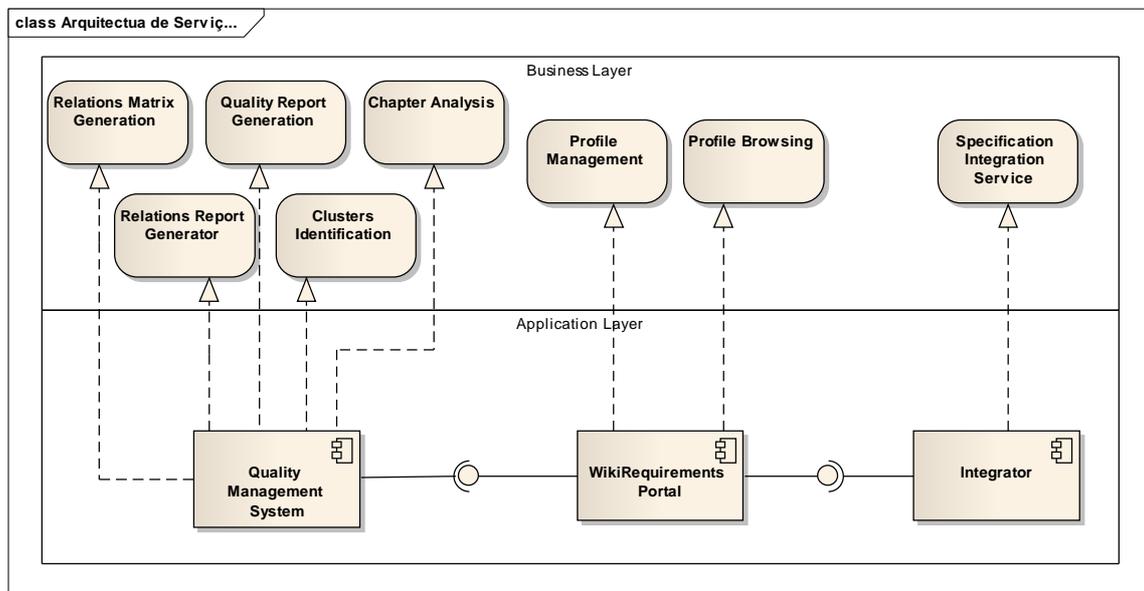


Figure 4 – Services and components (Archimate [6])

The system consists of three components, developed on different technologies and supporting different services:

- **Integrator Component:** This component is designed to integrate any system requirements specification. It has been developed with C#;
- **Requirements Portal Component:** This component was developed with the MediaWiki framework, after considering all the advantages of using a Wiki technology [7]. MoReq2010 document has been adapted to integrate the component;
- **Quality Management System Component:** This component was developed in PHP and interacts with the system Requirements Portal. The component reads data from the Requirements Portal, analyzes them, transforms them and presents the results to the user.

The system provides a set of services, described in Table 7.

Table 7 - Services

Service	Goal	Description of the implementation
Relations Report Generator	Generate relations report, which expresses the relationships between requirements	Construction of a tree where each node is a requirement, and each link is a relation
Clusters Identification	Identify requirement cluster, based on their relationships	Construction of a tree where each node is a requirement, and each link is a relation
Chapters analysis	Identify dependencies between	Identifying direct relationships of requirements, and grouping

	different chapter	those relationships by chapter.
Quality Report Generation	Identify malformed requirements	Application of algorithms base on a set of heuristics.
Relations Matrix Generation	Get a matrix that presents the relations of the requirements selected by the user	Identification of all the requirements referenced by the requirements that the user has chose
Profile Management	Grant the consistency and traceability of the specification or profile	Creation and management of profile, adding new and editing existing articles into Requirements Portal
Profile Browsing	Browse through the articles in a profile	
Specification Integration Service	Integrate any specification with pre-defined XML structure into the system	Processing of XML document, filtering and organizing the necessary information, creation of local objects which will be manipulated; identification of references between them; importation of the objects into the Requirements Portal

6 Results and Validation

This chapter presents the results obtained by the integration of MoReq2010 into an information system, as well as the results of applying automatic analysis tools, developed specifically for this project.

6.1 Consistency and Traceability in MoReq2010

The structured representation of Moreq2010 in the system **improves traceability** compared to the original document, *Word* version. A reference to a certain section of the specification becomes, with the system, a link to that section. The **navigation is facilitated**, the connection between various sections become more obvious and of immediate access.

These links, which replace the internal references in the specification of requirements, also allow to **identify inconsistencies**, as when there are references to the different sections of the specification what is happening, at the application level, is that the system is creating links between pages. If a referenced page does not exist, the profile administrator will be warned and inconsistencies corrected.

Furthermore, **value** was **added** to the original requirements. In the original document, an implicit division of the information stated in the requirements (the expressions itself and a brief explanation or example) already existed. That division was formalized and **new sections** were **added** to assist users in navigation. For example, the requirement R2.4.4, whose original representation can be seen in Figure 5, is represented in the system as shown in Figure 6. In this figure we can observe five different sections. Section 1 is a summary of the requirement, automatically generated, the second section represents the expression of the requirement, section 3 is an explanation or example of application of the requirement, section 4 is an aggregation of references to concepts contained in the glossary of the specification and section 5 is an aggregation of the requirements referenced in the original text plus the its summary.

R2.4.4

The MCRS must allow an authorised user to modify the metadata for each service, or bundle of services under **R2.4.1**, including:

- Title,
- Description,
- Owner Information, and
- Contextual metadata (if any).

The Owner Information gives information about the organisation or organisations using the MCRS, and may include help desk or contact information. The Title and Description should provide the local name of the MCRS and additional descriptive information.

Function reference: F14.5.162

Figure 5 - Requirement R2.4.1 from the original MoReq2010

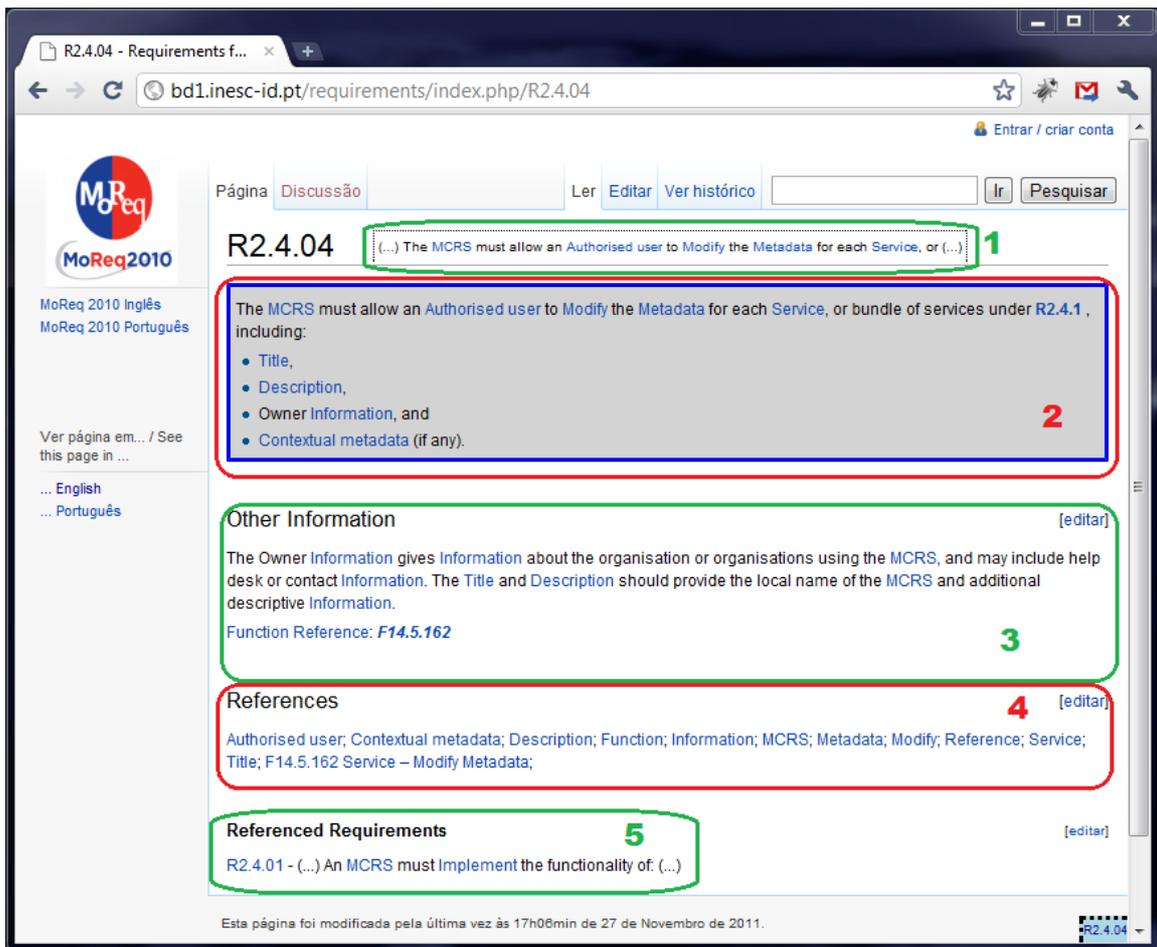


Figure 6 - Requirement R2.4.4 in the Requirements Portal (commented version)

6.2 Requirements Analysis

6.2.1 Results

With the requirements quality report it was possible to obtain the data shown in Table 8. A total of 174 requirements were analyzed (functional requirements of all MoReq2010 Core) and 224 recommendations were obtained. These recommendations were obtained from the application of different algorithms. Associated with each evaluation criterion considered (ambiguity, incompleteness, non-atomicity and incomprehensibility) there is one or two heuristics used to identify potential problems. Each heuristic is responsible for generating the number of recommendations shown in the third column of the table.

Table 8 – Data obtained from requirements analysis

Total of analysed requirements		174
Total of recommendations		224
Evaluation Criteria	Heuristic	# recommendations
Ambiguity	Search of ambiguous expressions	24
incompleteness	Search for expressions which indicates incompleteness	0
Non-atomic	Word count	62
Unreadable	Multiplicity of terms of repetition	102
	Implicit references	14

6.2.2 Validation

The purpose of this section is to validate the results, framing them in the studied problem.

Completeness

The report doesn't present any incomplete requirement according to the used heuristic.

Ambiguity

Twenty four recommendations were identified. Table 9 expresses the frequency of words associated with ambiguity found in the requirements. We also present the false positives identified after analyzing the results and verifying if the requirements were ambiguous or not.

Table 9 – Ambiguous expressions found on the requirements

Expression	# occurrences	% false positives
great	2	100%
several	2	0%
significant	1	100%
at least	16	0%
Total	24	14%

The definition of "Significant" is given in the specification. Although it is usually an ambiguous word, in this case it is defined, with specific limits. There are words that were used, although in a different context from the expected ("great" was a parameter in a desired functionality - search - was thus a keyword, not a word describing the requirement).

Comprehensibility

Table 10 – Results from comprehensibility analysis

Heuristic		Occurrences	% false positives
Multiplicity (more than four words of repetition)	<i>And's</i>	11	0%
	<i>Or's</i>	3	0%
Expressions of implicit reference	That	80	44% ¹
	This	6	50%
	These	3	33%
	Its	12	8%
	they	1	0%
Total		116	35%

In the case of multiple occurrences of the expressions "and" and "or", it appears that the expression of the requirements are usually very long, and non-atomic, and the repetition of this words serves as a connection between what should be more than one requirement. Typically a requirement too long is also a requirement difficult to understand because it is not too sharp and expresses multiple ideas.

In the case of implicit references it appears that there are many false positives, especially in requirements that use the words "that", "this" or "these." These expressions do not exacerbate the understandability of the expression of the requirement if they are used near the subject which they are related to (as in Example 1), or in addition to a verb ("ensure that" as in Example 2).

The MCRS must generate system identifiers for new entities as universally unique identifiers (UUID) and must not allow these identifiers to be modified.

Example 1 - Requirement R2.4.24

The MCRS must allow function definitions to be added to, and removed from, active roles, ensuring that every function definition is at all times associated with at least one active role.

Example 2 - Requirement R5.4.04

Atomicity

In Table 11 the results of different runs of the same algorithm are shown. Unaware of the number of words that an atomic requirement in MoReq2010 should have (on average), the "maximum" words number was varied. This value is in the first column of the table. In the second column it is referred the number of recommendations, which are the requirements with more words than the limit expressed in the first column. The third column indicates which is the range within which the conditions are analyzed, and the fourth the number of requirements analyzed. From the analysis of the

¹ From a random sample of 20% of the total (16 analysed requirements)

requirements with more than 30 words the tool returned 112 requirements (a total of 174 requirements). Within these 112 requirements the ones closer to the limit are chosen for a non-automatic analysis. Thirty requirements were analyzed, with 30 to 40 words. After an analysis of the 30 requirements it was concluded that 23 atomic requirements are atomic, although they have been identified by the tool. Therefore for the parameter "words number" of 30 it was obtained 76.7% false positives (from a sample of 30 requirements), which is represented in the last column. The greater the "word count", the lower the percentage of false positives.

Table 101 – Results of atomicity analysis

Word count	# recommendation	Range of words number that define the sample	Sample	% false positives
30	112	30-40	30	76,7%
40	81	41-50	15	40%
50	63	51-75	24	25%
75	37	76-100	15	13,3%
100	21	101-125	7	0%

6.2.3 Critical Analysis

From the analysis of the results presented it is possible to observe that some of the heuristics used are very effective to analyze the quality of requirements, such as counting the number of words in the case of atomicity, or research "or" and", "these "or" its" in the evaluation of understandability, and the expressions "at least" and "several" in the analysis of the ambiguity of the expressions.

However, in some cases there was a high failure rate, after a non-automatic review of the requirements, as in the expressions "that" and "this", in the analysis of comprehensibility, or "great" and "Significant", in the analysis of the ambiguity, where it was found that the expression alone does not decrement the quality of the requirement. It was necessary to read the requirement to really recognize, or not, if the expression affects the quality requirement, and often does not affect.

6.3 Requirements Relations Analysis

6.3.1 Results

It was done an analysis of the relationships between requirements, and the results are shown in Table 12; it was done an analysis of the structure of MoReq2010, by analyzing the relationships between the chapters, the results of which are shown in Table 13, and it was made an analysis of requirements clusters, where it was tried to identify clusters of requirements, but the results were inconclusive.

With the service for retrieving relationships it was obtained some interesting results. Table 12 presents the five strongest requirements for different levels of relationships. The requirements are stronger if they have more relations with other requirements. A level 1 relationship is a direct relationship between two requirements R1 and R2. A Level 2 relationship is when the requirement R3 refers R2 and R2 refers R1. With maximum depth of two levels R1 relates with R2 and R3.

Table 12 – Requirements Relations

Total of requirements		174	
Total of requirements without relationships		67	
Maximum level	Strongest requirements	# relations	Comments
1	R2.4.1	9	And more 10 requirements with five or more relations
	R2.4.13	8	
	R2.4.22	8	
	R10.4.5	8	
	R2.4.15	7	
5	R2.4.1	48	And more 11 requirements with thirty or more relations (five of them also from chapter 7)
	R7.5.1	43	
	R7.5.8	38	
	R7.5.5	38	
	R7.5.2	37	
20	R2.4.1	74	And more 12 requirements with thirty or more relations (six of them also from chapter 7)
	R2.4.2	47	
	R7.5.1	43	
	R7.5.5	38	
	R7.5.8	38	

Table 13 shows the relationships between the chapters of MoReq2010. The relationships were calculated taking into account the relations that each requirement from each chapter has with requirements from other chapters (or with its own

chapter). For example, Chapter 2 (System Services) has 38 internal references (requirements from that chapter referring to other requirements of that chapter) and refers five times the Chapter 7 (Model Metadata Service). The latter refers eight times to Chapter 2.

Table 113 – Relations between MoReq2010 Chapters

Chapter is referenced (chapter number)		Chapter referees (title, chapter number)										% dependency	Total references	Total Requirements
		2	3	4	5	6	7	8	9	10	11			
System Services	2	38	1	0	0	1	5	3	0	1	3	50%	52	28
User and Group Service	3	6	5	1	0	0	0	0	0	0	0	50%	12	14
Model Role Service	4	9	2	4	0	0	0	0	0	0	0	73%	15	15
Classification Service	5	1	0	0	2	1	0	1	0	0	0	34%	5	8
Record Service	6	4	0	0	0	16	0	1	0	0	0	23%	21	21
Model Metadata Service	7	8	0	0	0	0	24	0	0	0	0	40%	32	20
Disposal Scheduling Service	8	1	0	0	4	8	4	45	1	0	0	75%	63	24
Disposal Holding Service	9	2	0	0	0	0	0	1	3	0	0	43%	6	7
Searching and Reporting Service	10	2	2	0	0	0	0	0	0	36	0	15%	40	27
Export Service	11	2	0	0	0	0	0	0	0	0	21	20%	23	10
Total		73	10	5	6	26	33	51	4	37	24			

6.3.2 Validation

It is shown in Table 12 that there are 15 requirements with five or more first level relations, 67 unconnected, and the remaining 92 requirements have one to four relations, from a total of 174 requirements. When searching for higher level (5th or 20th level) relations we observe that a set of requirements of Chapter 7 excel, which can induce that this chapter is very relevant, and many Requirements refer Requirements of that chapter.

But to really understand the organization of the chapters it is also necessary to interpret the relationships between them. Table 13 shows the relationship between different chapters. The findings may lead to the conclusion that the requirements specification is poorly organized.

Note that the second chapter is referenced by all the other chapters, which makes sense, given its content, described in Table 2, where the chapter is presented as an aggregator of requirements common to all services. However, it appears that chapters 3 and 4 are very dependent of the second chapter, even more than from themselves, as they relate with requirements of Chapter 2 more times than with the Requirements of their chapters.

Note also the dependence of chapter 8 from the chapters 5, 6 and 7. If the relationship between chapters 5 and 8 would be expected, since it is stated in the introduction of these chapters, the other two are more striking.

It is also interesting to note the percentage of dependency of the chapters. This value was calculated taking into account the amount of existing requirements in a chapter divided by the number of requirements that this chapter relates. Note the large dependence of Chapters 2, 3, 4 and 8, and non-dependence of Chapters 6, 10 and 11.

6.3.3 Critical Analysis

Considering these results it's possible to take different conclusions. Some requirements stand out, which are related to many other (directly or indirectly). This data allows us to state that there could be a formal hierarchy of Requirements in MoReq2010 (since it already exists one informally, as these results show). The requirements could have a relation of specialization, aggregation or composition explicit.

The document structure can also be rethought. There are strong dependencies between some chapters, as opposed to others which are almost completely self-contained. However, the answer to this question would require a more careful semantic analysis.

7 Conclusions and Future Work

This thesis aimed for a qualitative analysis of a generic requirements specification (the MoReq2010), applying techniques from the domain of requirements engineering. Satisfactory results were obtained (from the perspective that points to be improved were found in the aforementioned specification).

It should also be noted that the results of this work will be presented at the meeting of the members of the DLM-Forum, to be held between May 31 and June 1, 2012, in Copenhagen, Denmark. Additionally the results were submitted to an article entitled "Requirements Engineering analysis of MoReq", [8], for publication in the Record Management Journal lying now in review.

The student of Mestrado de Engenharia de Informática e Computadores of IST, Ricardo Sousa is currently developing its master thesis in order to extend and apply the techniques developed in this project for the other referential specification, the "ISO 16175: ICA Principles and Functional Requirements for Records in Electronic Office Environments".

Below are described some opportunities for future work, to continue the developed project

Requirements Specification

With the developed system new functionalities can be developed to support a flexible specifications profile system that maintains the consistency between the different profiles. Currently you can create profiles, but there are not many services to support this feature. Tools can be developed to manage ontologies that help manage the different domains; services to create and manage profiles can also be improved.

Requirements Quality

The heuristics applied to the qualitative analysis of the requirements have been partially effective, but can clearly be improved. There are many possible improvements:

- Parameters can be updated (dictionaries, the number of words, etc.).
- Semantic trees can be generated, and then analysed to identify syntactic ambiguity;
- A more precise analysis of the text can be made, using techniques from natural language;

It can also be considered the possibility of developing an intelligent service, adjusting the parameters in order to provide more accurate results, based on the rates of acceptance or rejection of the users' recommendations. The results of the execution of the service would not only help to improve the quality of the specification, but they could also provide benchmarks for application in other specifications.

It may be interesting in future to do a semantic analysis of the document, starting from the results presented here and propose new expressions for the requirements.

Requirements Relations

Relationships between requirements were identified. If on the one hand the results were interesting, they were somewhat short on the other. The attempt to find requirements clusters was not successful. Probably by applying more complex and appropriate search techniques it might be possible to obtain very interesting results.

It may be interesting in future to do a semantic analysis of the document, starting from the results presented here and to propose a restructuring of the specification.

8 Bibliography

- [1] "IEEE 610.12 Standard Dictionary of Software Engineering Terminology," IEEE Computer Society, Nova Iorque, 1990.
- [2] Klaus Pohl, *Requirements Engineering: Fundamentals, Principles and Techniques.*: Springer, 2010.
- [3] DLM-Forum, "MoReq2010 - a new world standard for compliance & the best practice information management for all sectors," vol. 157, Setembro 2010.
- [4] F. Fabbrini, M. Fusani, S. Gnesi, and G. Lami, "The Linguistic Approach to the Natural Language Requirements Quality: Benefit of the use of an Automatic Tool," in *IEEE Computer Society*, Washington, 2001, pp. 97-.
- [5] "IEEE 830-1998 - IEEE Recommended Practice for Software Requirements Specifications," IEEE Computer Society, 1998.
- [6] Mark Lankhorst, *Enterprise Architecture at Work.*: Springer, 2005.
- [7] David Ferreira and Alberto Rodrigues Silva, "Wiki Supported Collaborative Requirements Engineering," in *Wikis4SE'08 Workshop*, Porto, 2008.
- [8] Ricardo Vieira, David Ferreira, José Borbinha, and Guilherme Gaspar, "A Requirements Engineering Analysis of MoReq," 2012.