TÉCNICO LISBOA

Reference Requirements Documents Manager

Denise Sofia Tavares Pedro

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. José Luís Brinquete Borbinha

Examination Committee

Chairperson: Prof. Miguel Nuno Dias Alves Pupo Correia

Supervisor: Prof. José Luís Brinquete Borbinha

Members of the Committee: Prof. Alberto Manuel Rodrigues da Silva

November 2015

Contents

Acknowledgements

To my supervisor, Professor Doutor José Borbinha, for all the patience, the attention and the time spent helping me. The patience to show me what sometimes was obvious but I couldn't see, the attention given even when time was scarce, and all the help to finish this work.

To all the people involved in this work, especially António Higgs, who had a patience of steel with my nerves, worries and fears, and mostly bugs on the code.

To all my colleagues, especially Jorge Miguel Saldanha, who despite being completely out of the scope of my thesis made an effort to help me when I couldn't see any light. To all others, forming a group of persons that have been close to me through these five years of college, and shared the same problems, joys and victories.

To my wonderful family, my lovely grandmother, aunts and cousins that gave me the strength to go further.

To my awesome brother and my little sister, for whom I always had to be an example of good behavior and goals achievement.

To my Father, and my Mother; the two human beings that put me in this world, moved mountains and passed through hell just to give me the opportunity of having a graduation, of being "someone" in this world. Nothing would be possible without your education, discipline, and mostly love. The two persons who always fought to give to their children the life they never had, working day and night for nothing to be missing in this path.

To every each and one of you that believed in me. Thank you.

"Nothing will work, unless you do" – Maya Angelou

Resumo

Em tempos posteriores, quando surgia a necessidade de desenhar e implementar um sistema de informação para resolver um determinado problema, este desenho e implementação eram feitos muito à medida das necessidades de cada organização, mas os detalhes desses sistemas eram em muito semelhantes já que os objectivos gerais eram os mesmos. Entretanto, a estabilização dessas práticas trouxe a necessidade de criar referências com requisitos e boas práticas para particularidades desses sistemas. Um exemplo são os documentos com requisitos de referência fundamentais para o funcionamento de um sistema de gestão de documentos de arquivo que contém políticas e processos que garantem que as organizações gerem os seus documentos de arquivo correctamente.

Em dias correntes, não só se procura seguir as orientações existentes nestes documentos, como se procuram soluções e/ou componentes que já incorporem as mesmas. No entanto estes documentos foram desenvolvidos por especialistas na área da gestão documental e são essencialmente procurados por desenvolvedores de sistemas que na sua maioria não conhecem em detalhe as preocupações da gestão documental. Surge por isso uma necessidade de criar um meio que permita aos desenvolvedores tirar melhor partido da informação destes documentos.

O objectivo deste projecto é portanto o desenho de um sistema que, suportando o armazenamento da informação existente nos documentos, permita uma navegação mais fácil e centrada às necessidades de cada stakeholder, e explorar o uso de técnicas de engenharia de requisitos orientada a objectivos para permitir um melhor entendimento dos requisitos presentes nos documentos pela abstração de objectivos gerais.

Palavras-Chave: Engenharia de Requisitos Orientada a Objectivos, KAOS, Gestão Documental, MoReq2010, Sistemas de Gestão de Documentos de Arquivo.

Abstract

In later days, when the need of designing and implementing an information system to solve a specific problem appeared, these exercises were made much fitted to the needs of each organization; however the details of those systems were very similar since the general goals were basically the same. In the meantime, these practices became stabilized and with that came the need of creating references with requirements and good practices to some particularities of those systems. An example are the reference requirements documents, fundamental to the behavior of a records management system, which contain policies and processes that guarantee a correct management of documents by organizations.

Nowadays, not only one seeks to follow the guidelines presented in these documents, but also there is a search of solutions and/or components that already incorporate them. However these documents were developed by specialists in the field of records management and are essentially seek by system developers that, in most cases, are not familiar with records management concerns. This way emerges a need of creating a means to allow developers to take better advantage of the information present in these documents.

The objective of this Project is to design a system that, supporting the storage of the information available in these documents, allows for an easier navigation and centered to the needs of each stakeholder. Also explore the use of goal-oriented requirements engineering techniques for a better understanding of the requirements present in the documents by abstracting general objectives.

Keywords: Goal-Oriented Requirements Engineering, KAOS, Records Management, MoReq2010, Records Management System

Tables List

Figures List

Acronyms List

RE – Requirements Engineering

GORE – Goal-Oriented Requirements Engineering

KAOS – Keep All Objects Satisfied

RMS – Records Management System

MCRS – MoReq2010 Compliant Record System

MoReq – Modular Requirements for Records Systems

ACL – Access Control List

1 Introduction

This chapter will introduce the problem and the motivation for the engagement in this project, the objectives of this work and a summary of the obtained results, and the expected structure of the present document.

1.1 Problem and Motivation

Organizations have been increasingly dealing with large quantities of information. Time is also more and more important; the challenging and dynamic environment forces quick responses and great adaptability to changes, there is a need to quickly sense and seize opportunities. It is therefore crucial that a management of existing information is efficiently made, with a high efficacy and great performance to produce quick results. Having these observations in account it is expected that any big organization will seek for a good records management system.

Currently, organizations with mandated recordkeeping obligations can find multiple solutions of systems to manage their records in the market. Examples can be found in the market from big companies such as HP (HP Records Manager), Microsoft (Microsoft SharePoint) and IBM. The available solutions aim to provide tools for dematerialization of documentation and associated conduct processes, documents normalization, centralized management of the organization's records, fast search and access of documents, more control and security of information flows, among others, constituting an asset for business growth.

The implementation of this type of systems is relatively new. Before, organizations kept records in whatever form they felt appropriate without the benefit of retention schedules, disposition guidelines, or other formal information life-cycle procedures. With the creation of massive volumes of information came the need to achieve ways to properly control them; what should be retained and what could be disposed. This need led to the creation of systems to solve the problem; each organization would create its own solution, adapted to its needs. Records were kept in whatever form each felt appropriate without the benefit of retention schedules, disposition guidelines, or other formal information life-cycle procedures, leading to a lack of uniformity of procedures.

Later there was a need to standardize the procedures and best practices for records management, therefore the creation of documents with reference requirements for records management, consulted when in need to validate or implement a records management system.

A reader in the act of his duties will analyze each requirement individually to see if that requirement is suitable to his objective, or might be validating if the existing system satisfies a set of requirements. In some cases one requirement in a document might have the same objective as another in a different document but they can be presented in different ways and a reader may not become aware of such resemblance, leading to a possible repetition of requirements.

Also these documents are relatively lengthy and complex, forcing the reader to engage into an intensive, very time consuming and cumbersome task of analyzing the documents and their content. These points altogether

present problems that will inevitably lead to a superior consumption of valuable business time, delaying the phase of specification of the system (or its validation).

The motivation for this work comes after acquiring some knowledge about a European project related to records management and their interest in having a solution to the elicited problem, also enriching its scope with an information system solution – the E-Ark project. It aims to "create and pilot a pan-European methodology for electronic document archiving, synthesizing existing national and international best practices that will keep records and databases authentic and usable over time."[1]

Its objective is to provide simple and efficient methods to acquiring, preserving and enabling re-use of information by public and private, large and small, and able to support complex data-types organizations. The E-Ark project will develop practices to reduce the risk of information loss by consolidating existing heterogeneous approaches to keep and archive records.

1.2 Objectives and Results

Having the problems described above in consideration, the proposed solution makes use of an information system. The objective of this work was to design an information system to support the analysis of a set of requirements documents by a stakeholder. This system should allow for the consolidation of documents of reference requirements of records management, considering they are all about related concerns, which must be supported by services for creation of relationships between them and services to explore such relationships as consolidated views.

The focus would be on the five most important documents of reference requirements of records management: ISO 15489, ISO 16175, ISO 30301, MoReq2010 and DoD 5015.02. However, MoReq2010 alone posed a relatively complete challenge; a success in embedding the document on the system would make the other's assessment a simple and quick task, by replication. MoReq2010 specification has a complex structure and an amount of data enough for an almost complete coverage of possible structure on the documents (by structure, one must think of pages, paragraphs, sections, etc, components that will be better addressed and detailed in the problem analysis chapter) .

Since the referenced documents already contain a set of requirements, our objective will be to deduce higher level goals to be achieved, giving an opportunity to create a better understanding of the existing requirements as a whole. This can be made relying on Requirements Engineering techniques, more precisely in the Goal Oriented Requirements Engineering field and its methods.

The expected result would be a system capable of supporting the storage of a document with reference requirements of records management, an easier navigation through it, and the creation of goals supported by its requirements.

[1] E-Ark project - www.eark-project.com/about About E-Ark

1.3 Document Structure

This document is composed of seven chapters. In Chapter 2 and 3 comprise the Related Work; chapter 2 presents Goal-Oriented Requirements Engineering and some definitions of requirements engineering are given, as well as an introduction of goals in the subject to present GORE. Then there is a presentation and description of the major goal-oriented requirements engineering approaches. Chapter 3 introduces concepts about records management, the main area of interest of this project. After presenting some definitions and benefits of records management, the five documents of reference requirements are presented and described. A separated description of MoReq2010 is made, since this specification had the major focus throughout this project.

In Chapter 4 an analysis of the problem is made, dividing it into a black box analysis and a white box analysis where the information system design is explained carefully as well as the result of creating goal models for MoReq2010 core Services.

Chapter 5 presents screenshots with results of what was possible to implement from the design previously explained.

Chapter 6 presents conclusions and future work. Finally chapter 7 with a list of the consulted references, and chapter 8 with appendices.

2 Goal-Oriented Requirements Engineering

Requirements Engineering is a branch of engineering that concerns with analysis, elicitation, refinement, negotiation, documentation, validation and requirements management of a system to be built.

One can find multiple but similar definitions of RE on the literature. Several authors have created their definition of RE, and this situation made available in the literature multiple definitions for it even though these are similar. According to [1] software systems requirements engineering is the process of discovering the purpose of the software by identifying stakeholders and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation. Pamela Zave created a clear definition of RE: *"Requirements engineering is the branch of software engineering concerned with the real-world goals for functions of and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families"* [1].

If we consider the two definitions presented above and reason about them we can say that RE is **a set of activities concerned with identifying and communicating the purpose of a software system, and the context in which it will run. RE acts as the bridge between the real world needs of users, customers, and other constituencies affected by a software system, and the capabilities and opportunities afforded by software technologies.**

RE should be the way to answer the "**what, why and how**" questions. **Why** is some system needed? What is the purpose of this system and what unleashed is need. **What** features does this system need to have? What features are needed in order to satisfy the said **context** in which this system will run. Finally, **how** this said system is to be constructed.

Goal Oriented Requirements Engineering makes use of *goals* for requirements elicitation, elaboration, structuring, specification, analysis, negotiation, documentation and evolution.

Requirements Elicitation is where alternative models for a target system are analyzed to meet the identified objectives, hence identifying requirements and assumptions on components of such models. The specification is where these requirements are precisely formulated and the analysis searches for problems such as incompleteness, inconsistency, etc. and for feasibility. In the negotiation alternative requirements and assumptions are evaluated as well as a risk analysis is made by stakeholders. The best alternatives are then selected. Documentation, made throughout the whole requirements engineering process is where various decisions are documented together with the underlying rationale and assumptions. Evolution comprises modifications to requirements to accommodate rectifications, environmental changes, or new objectives. [2]

In order to understand the Goal-Oriented RE it is important to define what a *goal* is in this context. Requirements result from a set of desired specifications that a system must have and goals are abstractions of

those specifications. A goal in the context of a software system is an objective that must be achieved by the system and may be formulated at different levels of abstraction. A goal may present some properties:

- Can have different levels of abstraction which can range from high-level, strategic concerns, to low-level, technical concerns.
- Can cover different types of concerns. Functional concerns, which are associated with the services to be provided, and non functional concerns, associated with quality of service.
- Can refer to two different types of systems: the current one or the system-to-be. High level goals often refer to both systems. The system-to-be and its environment are seen as a collection of active components called *agents*. These components are humans playing certain roles, devices, and software.
- A goal may in general require the cooperation of several components. However a goal under the responsibility of a single agent in the software-to-be becomes a *requirement*; a goal under the same conditions on the environment of the system-to-be becomes an *assumption*.

Goals are important in the RE process, and there are several reasons for it, some of them are presented next. First, they can help in achieving requirements completeness and relevance:

- Completeness - The requirements specification is complete with respect to a set of goals if all the goals can be proved to be achieved from the specification and the properties known about the domain considered; requirements are complete if they are sufficient to establish the *goal* they are refining.
- Relevance - A requirement is pertinent with respect to a set of goals in the domain considered if its specification is used in the proof of at least one goal.

Goals avoid irrelevant requirements, providing precise criteria for requirements pertinence. A requirement is considered pertinent with respect to a set of goals if its specification is used in the proof of at least one goal. They also provide a rich structuring mechanism (AND/OR Refinement, Abstraction) that allows a better understanding of the requirements to stakeholders, traceability links from high-level strategic objectives to low-level technical requirements and detection of conflicts among requirements. Goal refinement provides a natural mechanism for structuring complex requirements documents for increased readability.

- AND Refinement – the goal is satisfied if all of the sub-goals are satisfied.
- OR Refinement – the goal is satisfied if one sub-goal is satisfied.

Goals can have dependencies (considering G1 and G2 as two different goals): [4]

- “Requires” dependency if G1 requires G2 satisfaction in order to be satisfied (G1 requires G2);
- “Support” dependency if G2 is satisfied with some contribution of G1 (G1 supports G2);
- “Obstruction” dependency if G1 prevents G2 satisfaction (G1 obstructs G2);
- “Conflict” dependency if G1 satisfaction prevents G2 satisfaction and vice-versa (G1 and G2 are in conflict);
- “Equivalence” dependency if G1 satisfaction means G2 satisfaction and vice-versa (G1 and G2 are equivalent).

Goal models also provide a good way to communicate requirements to the customers; the refinement offers a level of abstraction that allows for an involvement of decision makers for validating choices being made among alternatives and even to suggest new alternatives.

2.1 Goal-Oriented RE Approaches

There have been introduced several different approaches using the concept of goals as part of a Requirements Engineering technique. With GORE approaches it is possible to represent one or more stakeholder needs (goals), assign them to an agent (stakeholder or system), and relate it to other goals, frequently describing how can a goal be achieved [5]. Normally there are present in these approaches activities such as goal elicitation, goal refinement and analysis, and assignment of goals and agents relationships. This section provides a view of four main GORE approaches: NFR Framework, i*/Tropos and GBRAM, and in a separate section the KAOS approach.

2.1.1 The NFR Framework

The NFR Framework is concerned with the modeling and analysis of non-functional requirements. The acronym NFR stands exactly for Non-Functional requirements. Its objective is to put non-functional requirements first in the developer's concerns. The main idea is to systematically model and refine non-functional requirements, exposing positive and negative influences of different alternatives on them. Its main activities are:

- Capturing NFRs in the domain of interest;
- Decomposing NFRs;
- Design possible alternatives for meeting NFRs;
- Dealing with ambiguities, tradeoffs, priorities, and interdependencies between NFRs;
- Selecting operationalizations (alternatives), supporting decisions with design rationale;
- And evaluating the impact of some decisions.

This framework introduces the concept of *softgoal*. A softgoal is a goal that is not satisfied via clear-cut criteria [5]. Softgoals can be refined using the AND and OR refinements and their interdependencies can be captured with positive or negative contributions. The framework supports three types of softgoals:

- NFR softgoals – represent non-functional requirements to be considered;
- Operationalizing softgoals – model lower-level techniques for satisfying NFR softgoals;
- Claim softgoals - allow for the analyst to record design rationale for softgoal refinements, softgoal prioritizations and softgoal contributions.

The framework provides a modeling tool, the *softgoal interdependency graph* (SIG), that can graphically represent softgoals, softgoal refinements (AND/OR), softgoal contributions (positive/negative), softgoal operationalizations and claims. The refinement process will reach a point where softgoals are no longer able to be further refined and here the developer can accept or reject them as part of the system. The developer chooses

alternative combinations of the leaf-level softgoals and uses a provided label propagation algorithm to see if the selected alternative is good enough to satisfy the high-level non-functional requirement for the system. Adopting a bottom-top approach, the algorithm starts from the decisions made by the developer; the labeling procedure flows towards the top of the graph determining the impact of the decision on higher-level goals, taking into account the labels on softgoal refinement links.

This framework also supports cataloguing design knowledge into three main types of catalogues:

- NFR type catalogues, which include concepts about particular types of NFRs.
- Method catalogues, encoding knowledge that helps in softgoal refinement and operationalizations.
- Correlation rule catalogues, that embeds knowledge that helps in the detection of implicit interdependencies among softgoals.

In short, the NFR framework provides a process-oriented approach to deal with non-functional requirements. Here the development process is rationalized in terms of non-functional requirements, instead of evaluating the final product to validate that it meets its non-functional requirements.

2.1.2 i*/Tropos

The i* Framework is an agent-oriented modeling framework [6]. It makes use of notations of the NFR framework, including softgoals, AND/OR decompositions, and contribution links, adding tasks, goals, resources, and dependencies between actors (agents). The framework has two main components:

- **Strategic Dependency Model** - represents a network of dependency relationships among actors, capturing the intention of the processes in the organization and what is important for its participants, abstracting over all other details. SD models are used during the late requirements analysis phase, in order to analyze the changes in the organization due to the introduction of the system-to-be. This model allows for the analysis of dependencies of actors (direct or indirect dependencies) and the exploration of opportunities and vulnerabilities of actors.
- **Strategic Rationale Model** - This model is used to explore the rationale behind the processes in systems and organizations. The rationale behind process configurations can be described in SR models in terms of process models (goals, softgoals, tasks, resources). This model represents the intentional aspects of organizations and systems by providing a lower-level abstraction; it provides a capability to analyze internal processes within each actor [2]. This model allows for a better understanding of what each actor's needs are and how are they met.

This framework is centered on the notion of *intentional actor* and *intentional dependency*. The dependency is *intentional* if it appears as a result of agents pursuing their goals. There are four types of dependencies, classified based on the subject of the dependency, which can be a *goal*, a *softgoal*, a *task,* and a *resource*.

Actors are described in their organizational setting and have attributes (goals, abilities, beliefs, and commitments). There is a dependency between actors; they depend on each other to achieve goals, execute

tasks, and supply of resources, giving them the opportunity of achieving more that they would if acting by itself, or not as cheaply, efficiently, etc. However there's the downside of not achieving much if depending on an actor that might not deliver, becoming vulnerable. Actors are strategic; they are concerned with the achievement of their objectives and struggle to balance their opportunities and vulnerabilities. Actors represent stakeholders and agents of the system-to-be. These can be agents, roles and positions:

- Agents are concrete actors, systems or humans, with specific capabilities.
- Roles are seen as abstract actors embodying expectations and responsibilities.
- Positions are a set of socially recognized roles typically played by one agent.

The framework has also the notion of *link*. There are *decomposition* links, and *means-ends* links, which relate the Strategic Rationale process elements, and are used to model AND and OR decompositions of process elements. The means-ends links are mostly used with goals and specify alternative ways to achieve them. The decomposition links connect a goal/task with its components. A softgoal, a goal, or a task can be related to other softgoals with *softgoal contribution links*, similar to the ones in the NFR framework. The links specify two levels of positive ("+" "++") and negative ("–" "--") contributions to the softgoals from a softgoal satisfaction, a goal achievement, a task execution. Here, softgoals are used as criteria of selection to choose an alternative process configuration that best meets the non-functional requirements of the system. The SR model is considered strategic since its elements are included only if considered important enough to affect some goal achievement.

This modeling framework is the basis for a development methodology – Tropos. Tropos is an agent-oriented software development methodology. It has an iterative and incremental development lifecycle and covers the (early) analysis and design & implementation phases. It uses i* for the representation and reasoning about requirements and system configuration choices.

2.1.3 GBRAM

This approach defines a top-down analysis method refining goals and attributing them to agents starting from inputs such as existing diagrams, textual statements (policy and mission), interview transcripts, etc, by assuming that no goals have been documented or elicited from stakeholders [6]. GBRAM involves two main activities:

- Goal Analysis - This activity concerns with the identification of goals by the exploration of various information sources, then the organization and classification of those goals. Therefore the activity is divided in explore activities to explore the available information, identify activities to extract goals and responsible agents from the first activity and organize activities to classify and organize the goals taking into account goal dependency relations.
- Goal Refinement - goals are refined using questions and scenarios. In this activity the evolution of goals is monitored since the moment they are identified until the moment they are translated into operational requirements for the system specification. This activity is also divided into *refine* activities that "clean" the goal set, eliminating "synonymous" goals, *elaborate* activities where an analysis of the goal set takes place and were possible goal obstacles are discovered. There are also constructing *scenarios* activities and

operationalize activities, with the former describing behavioral descriptions of a system and its environment and the latter the translation of goals into operational requirements. GBRAM requires that a identification of goal precedence occurs during this activity. Identification of goal precedence is identifying which goals must be achieved before which other ones..

A *requirement* in GBRAM specifies how a goal should be accomplished by some system. *Constraints* provide additional information of requirements that must be met for the accomplishment of a particular goal. Similarly to the other GORE approaches, in this framework a system and its environment are represented as a collection of *agents*. Agents are defined as entities or processes that thrive to achieve goals within a system based on the assumed responsibility for the goals.

All of the artifacts (goals, agents, stakeholders) are textually specified in *goal schemas*. However GBRAM does not provide a graphical notation to represent goals, goal refinements, agents, etc.

2.2 KAOS

Standing for Knowledge Acquisition in Automated Specification (according to Dardenne and Lamsweerde) [7] or Keep All Objects Satisfied, this framework derives goal refinements, operationalizations, conflict management and risk analysis by reasoning (in a semi-formal or formal way) about behavioral goals; semi-formal when modeling and structuring goals, qualitative to select a goal among a set of alternatives and formal when in need of a more accurate reasoning. KAOS language combines semantic nets for conceptual modeling of goals, assumptions, agents, objects, and operations in the system, and linear-time temporal logic for specification of goals and objects, as well as state-base specifications for operations. A construct in KAOS language has a two-level structure:

- An outer graphical semantic layer - here the concept is declared together with its attributes and relationships to other concepts;
- An inner formal layer – here the concept is formally defined.

Summarizing, the KAOS approach thrives to:

- Focus on Goal elaboration:
  - Define initial set of high level goals and objects they refer to;
  - Define initial set of agents and actions they are capable of.
- Then iteratively:
  - Refine goals using AND/OR decomposition links;
  - Identify obstacles to goals, and goals conflicts;
  - Operationalize goals into constraints (or software requirements) that can be assigned to individual agents;
  - Refine and Formalize definitions of objects and actions;

A KAOS specification is a collection of the following core models:

- Goal model and Responsibility model – representation of goals and assignment to agents;
- Object model – UML model derived from formal specifications of goals;
- Operation model – definition of various services to be provided by software agents.

In the next sections, a detailed view on these models is carried, taking in account the Objectiver methodology [8].

2.2.1 KAOS Goal Model

The KAOS Goal Model is the set of interrelated goal diagrams necessary to tackle a particular problem and is comprised of a number of entities and relationships whose use in its construction enables to represent *how* and *why* a goal is achieved:

- **Goals** are the focus of this model and represent objectives to be met through agent cooperation, prescribing a set of behaviors the system is supposed to reflect.
- **Sub-Goals** are goals that are linked to other goals through means of refinement relationships, contributing to the satisfaction of the goal they refine. Meeting the conditions of all sub-goals should automatically entail the satisfaction of their "parent" goal.
- **Agents** are humans or automated components that are responsible for achieving certain requirements and/or expectations.
- **Requirements** are low-level types of goals whose achievement constitutes a responsibility of a given software agent.
- **Expectations** are goals assigned to agents that interact with the system; they reflect interactions between the system and its environment and their achievement is not a system responsibility.
- **Domain Properties** are assertions about certain objects of the software environment enunciated as domain invariants or hypothesis, i.e. properties that are known to hold in all states of a domain object or properties that are supposed to hold, respectively.
- **Obstacles** are certain conditions that prevent the achievement of system goals. The definition of these undesired behaviors represents a defensive approach to software modeling.
- **Refinement Links** are relationships between a goal and its sub-goals that represent the decomposition of an objective into clearer steps in its achievement.
- **Responsibility Links** represent the connection between a software agent and the requirement whose achievement it is responsible for.
- **Assignment Links** represent the connection between an environment agent (one that interacts with the system) and the expectation whose achievement it is responsible for.
- **Obstruction Links** relate obstacles to goals, representing the impediment an obstacle represents to a goal's satisfaction.
- **Resolution Links** relate goals to obstacles, representing solutions to the presented impediments.

Next figure presents an example of a goal model from the KAOS Tutorial [8], representing the decompositions, interactions and conflicts that may occur when attempting to achieve the goal “Elevator Called”. The blue diamond shape is a Goal; the thick line diamond shape is a requirement. The yellow diamond shape is an expectation. Red links are responsibility links, pink ones are assignment links, yellow is a refinement link and in this case more specifically a AND refinement.

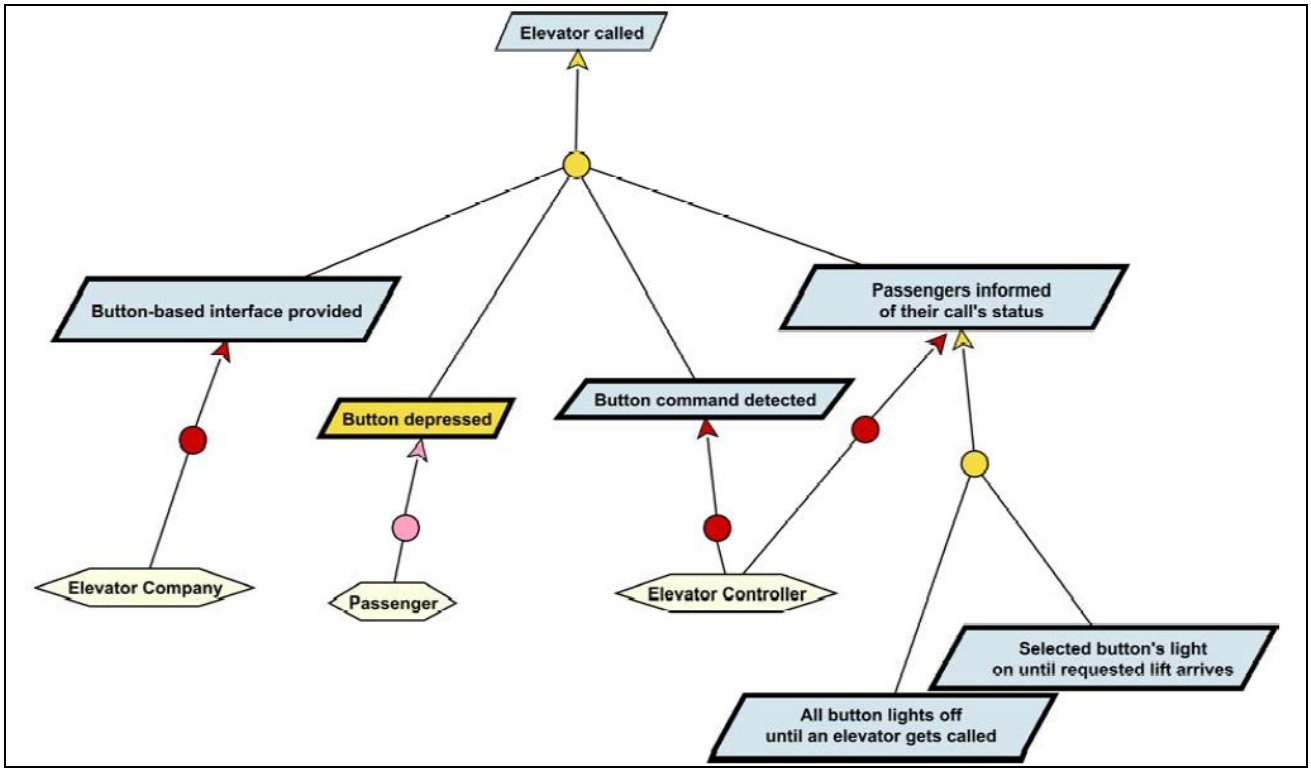


Figure 1 - Example of a Goal Model for a “Elevator Called” Goal

In [8] there are available several completeness criteria for the analysis of the overall system model. The first two are specific to the goal model:

- *“A goal model is said to be complete with respect to the refinement relationship 'if and only if' every leaf goal is either an expectation, a domain property or a requirement."* – **Completeness Criteria 1**.
- *“A goal model is complete with respect to the responsibility relationship 'if and only if' every requirement is placed under the responsibility of one and only agent (either explicitly or implicitly if the requirement refines another on which has been placed under the responsibility of some agent."* – **Completeness Criteria 2**.

2.2.2 KAOS Agent or Responsibility Model

The KAOS Responsibility Model is the set of responsibility diagrams that can be derived from the goal model and displays the requirements or expectations an agent is responsible for. It therefore comprises three of the

previously described elements: an Agent, its assigned **Expectations** and/or **Requirements**. Figure 2 presents such a model, representing the responsibilities of an Elevator company, following the previous example.

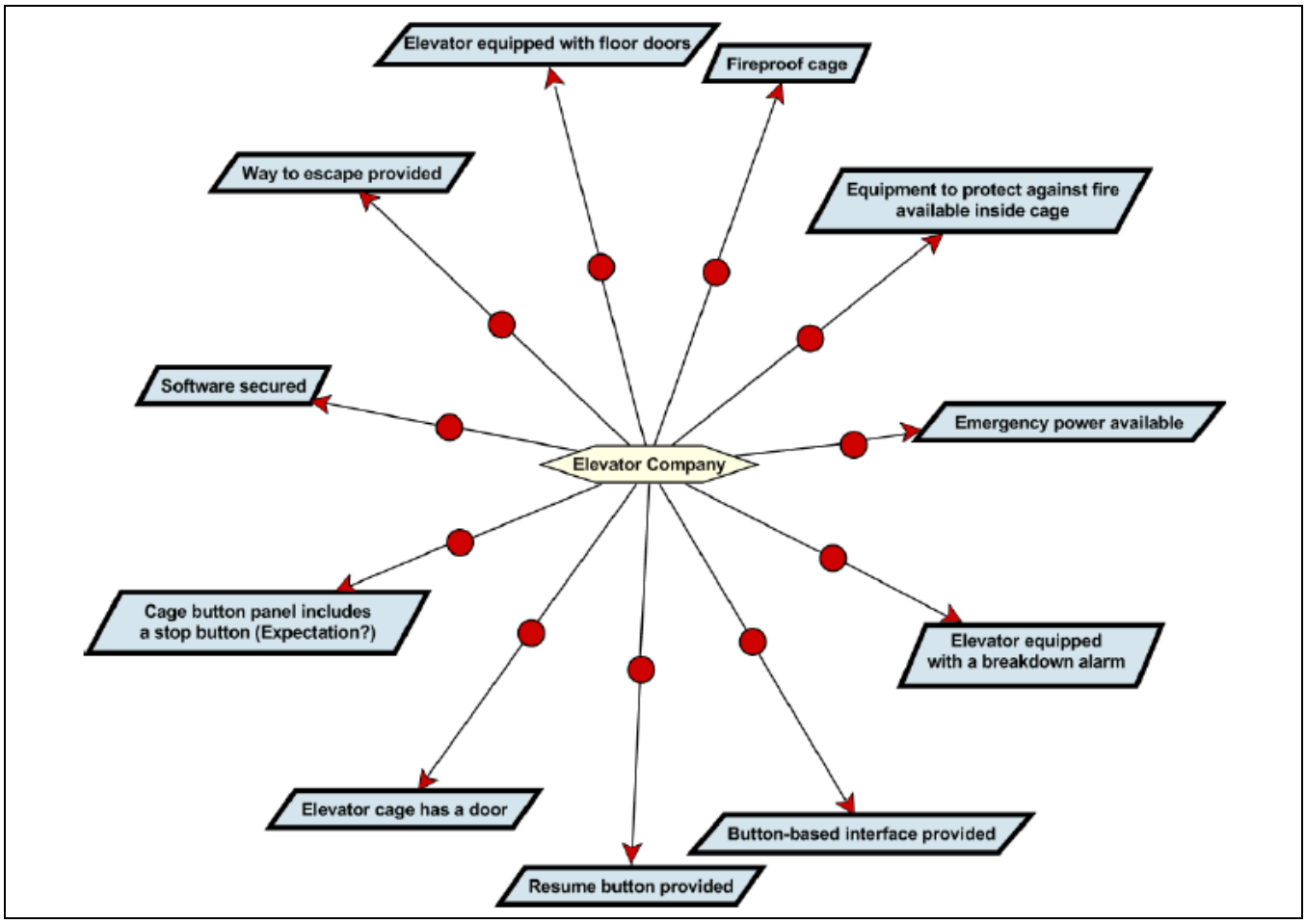


Figure 2 - Example of a Responsibility Model for an Elevator Company Agent

2.2.3 KAOS Object Model

This model defines the several concepts of the domain that may be relevant with respect to the known requirements or that represent constraints on the system itself in order to satisfy requirements. The considered objects may be defined as belonging to three types:

- **Entities** are autonomous, passive objects whose definition is independent from other objects.
- **Agents** are active objects that perform operations in order to achieve several types of goals.
- **Associations** are passive objects whose definition depends on the objects they link.

Their identification is parallel to the process of goal identification and definition, or may result from browsing the goal model or even from discovering system components that are necessary for a requirement's satisfaction. Figure 3 presents an example of such a model, representing the components of an Elevator System.

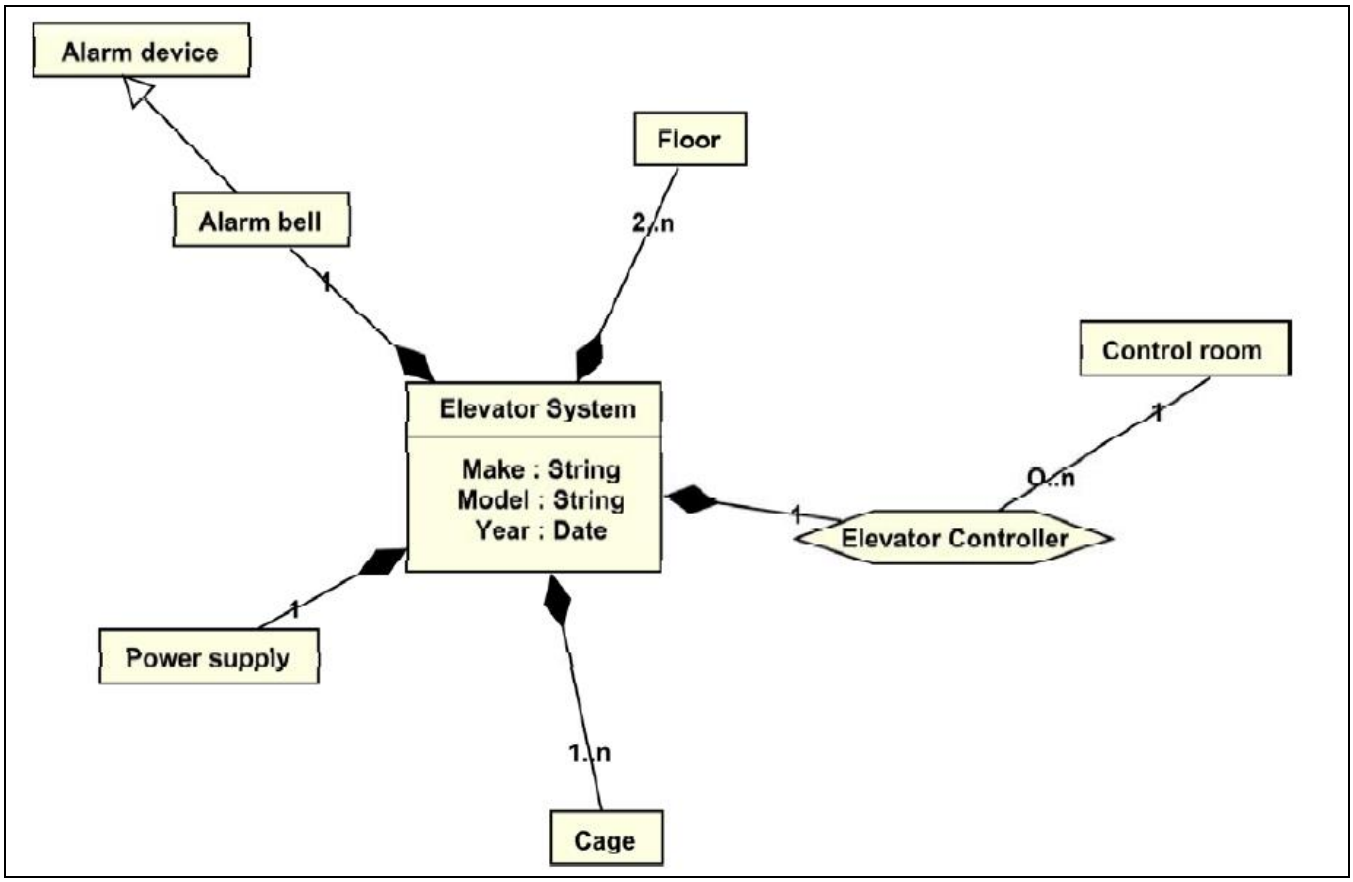


Figure 3 - Example of an Object Model for an Elevator System

The KAOS object model is compliant with UML class diagrams in that KAOS entities correspond to UML classes. KAOS associations correspond to UML binary association links or n-ary association classes.

2.2.4 KAOS Operation Model

This model comprises all the behaviors necessary for an agent to fulfill its requirements. These behaviors are translated into *operations* that work on the previously defined objects, being responsible for their creation, the triggering of object state transitions and the activation of other operations through **events**.

Their elicitation happens during the stakeholder interviews, in case the stakeholders find it necessary to describe certain behaviors in order to define a system goal, or by observing the modeled requirements, representing the operations as how the requirements have to be realized.

The operationalization (fulfillment) of requirements follows a set of heuristics defined as follows:

- ***Static*** requirements are translated into *objects*;
- ***Dynamic*** requirements are operationalized into *operations*;
- Requirements that are both ***static*** and ***dynamic*** are operationalized into interacting *objects* and *operations*.

The operation model therefore requires the definition of the following elements:

- **Operations** are performed by agents and specify objects' state transitions.
- **Events** are ephemeral objects that trigger operations performed by agents and can be external or produced by other operations.
- **Input Links** are established between objects and the operations they serve as input for.
- **Output Links** are established between objects and the operations that produce them as output.
- **Cause Links** relate events to the operations they initiate or terminate.

Figure 4 presents an operation model depicting operations for a button pressed in an elevator system. Operations are represented as ovals. Concerned objects are connected to the operations by means of Input and Output links. Events are represented as those traffic signs that are used to indicate directions.

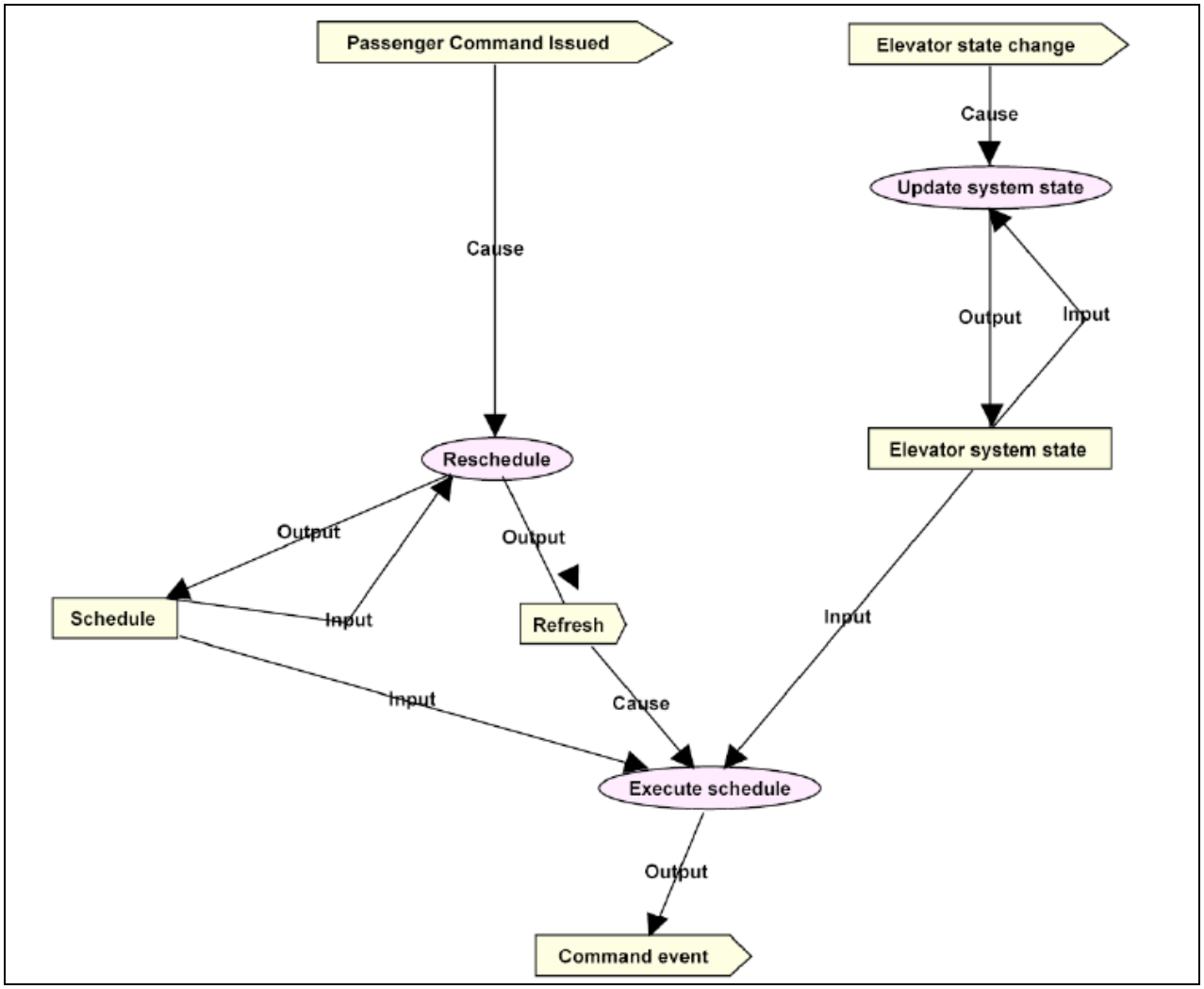


Figure 4 - Example of a Operation Model

This model, just like the goal model, also has completeness criteria that define rules for their evaluation:

- "*To be complete, a process diagram must specify:*
  a. *The agents who perform the operations*
  b. *The input and output data for each operation."*

- *“To be complete, a process diagram must specify when operations are to be executed."*
- *“All operations are to be justified by the existence of some requirements (through the use of operationalization links)."*

2.2.5 KAOS Summary

The KAOS approach, as explained by the Objectiver methodology [8], addresses requirements identification and of the intervening agents by relying on the construction of a requirements model segmented into four types of sub-model: the goal model, the responsibility model, the object model and the operation model. The following figure presents a summary of all models and their relation to each other, as well as all concepts and notations found in the methodology. Appendix IX contains a list explaining each concept.

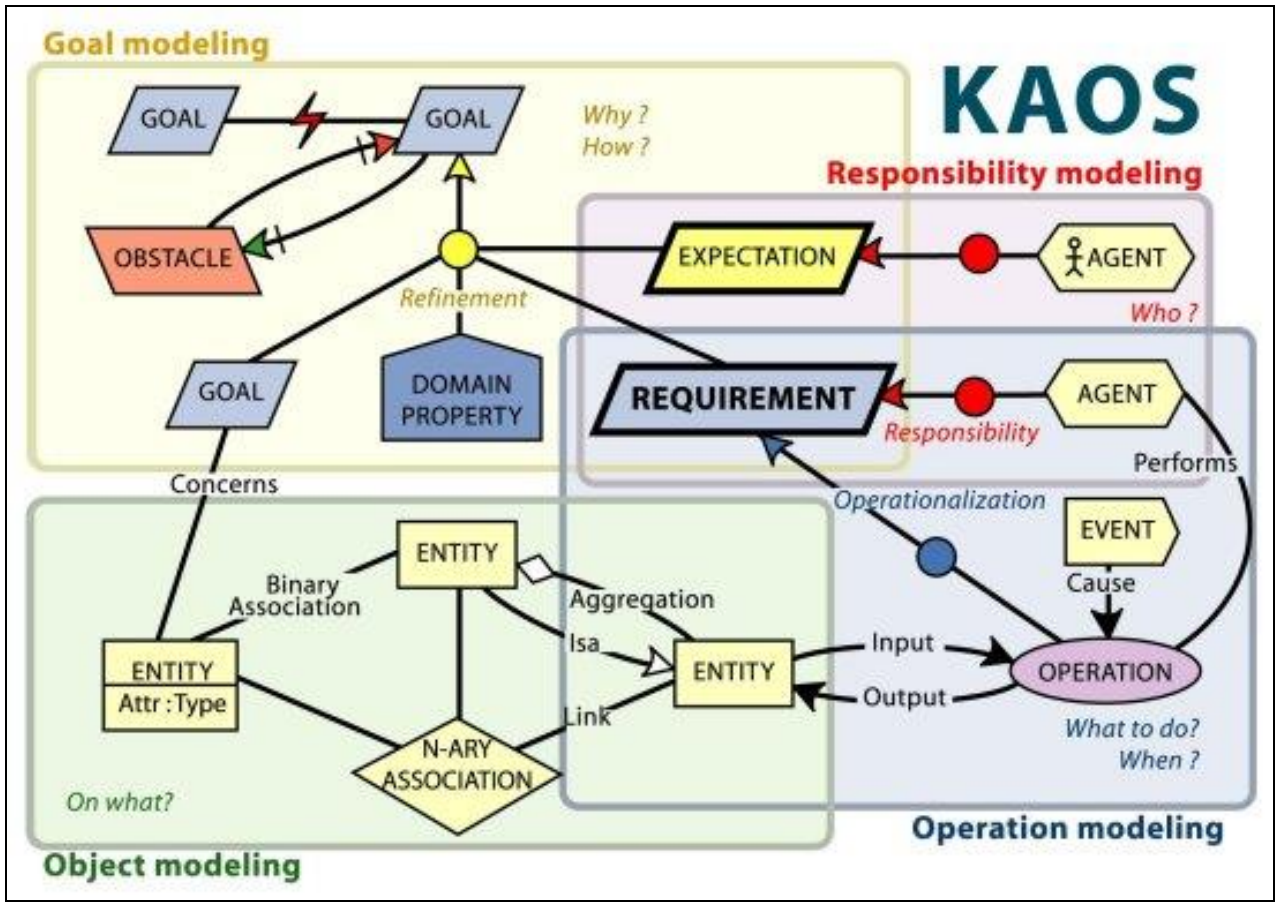


Figure 5 - KAOS Model

KAOS addresses the issue of traceability between the problem description and the solution description stages by operationalizing the several requirements that require completion. This way, analysts have a means to ensure that their intake on the system is translated into the expected solution, and the developers can have some context surrounding the solution they need to develop. By defining completeness criteria, KAOS ensures that the completion of the established goals is clearly defined and that every requirement is attributed to an agent.

Due to these particular capabilities and the clearly defined and unambiguous modeling aspects, the KAOS approach using the Objectiver framework provides an almost "elegant" method for approaching requirements modeling and therefore is the method preferred to be used in the scope of this project.

Unlike other approaches, KAOS does not provide a method to evaluate the impact of design decisions on non-functional requirements. However Lapouchnian [6] states that a variation of the NFR framework and its qualitative analysis approach can be integrated into KAOS with no problems.

3 Records Management

"Field of management responsible for the efficient and systematic control of the creation, receipt, maintenance, use and disposition of records, including processes for capturing and maintaining evidence of and information about activities and transactions in the form of records" [9].

Records Management not only drives manager's practices but also anyone who creates or uses records while conducting their business activities. As defined in [9], records management includes setting policies and standards, assigning responsibilities and authorities, establishing and promulgating procedures and guidelines, providing a range of services relating to the management and use of records, design, implementation and administration of specialized systems for records management, and integrating records management into business systems and processes.

ISO 15489 defines a record as it follows:

"Information created, received, and maintained as evidence and information by an organization or person, in pursuance of legal obligations or in the transaction of business".

A **record** holds valuable information, becoming a valuable resource and a relevant business asset, hence the importance of managing the records and this management being an essential task for organizations to preserve and protect records as evidence of actions.

A **records management system** (RMS) is responsible for the management of records throughout their life-cycle, since its creation, to preservation and/or disposition. The existence of a good records management system can be a source of information about business activities, supporting subsequent activities and business decisions. Also it ensures accountability to present and future stakeholders. Records enable organizations to conduct the business to a path of success by several reasons; for example by delivering services in a consistent and equitable manner, supporting and documenting policy formation and managerial decision making, provide continuity in the event of a disaster, provide evidence of business, personal and cultural activity, among others.

3.1 Main Reference Requirements for Records Management

This section presents an overview of the most relevant documents containing requirements and good practices in the records management field.

3.1.1 ISO Standards

The Technical Committee ISO/TC 46 provided guidelines to ease records and information management; their scope is the standardization of practices relating to libraries, documentation and information centers, archives, records management, museum documentation, indexing and abstracting services, and information science.

According to [10] the standards developed under ISO/TC 46 documentation and information thrive to “facilitate access to knowledge information and to help to develop appropriate automated tools, computer systems and services to obtain the information owned by libraries, archives, museums and enterprises. These standards give rules to identify, index, classify, access, select, exploit, communicate, exchange and preserve, both paper-based and digital information.”

This group prepared several standards to archives and records management that can be divided into “Management Systems Standards (MSS) for Senior Management” and “Standards for Implementers/Practitioners”. In the first group is the series of ISO 30300 and in the second one are:

- ISO 15489 – which mainly concerns are connected to records management;
- ISO 22310 - presenting Guidelines for standards drafters for stating records management requirements in standards;
- ISO 23081 - concerned with records metadata;
- ISO 16175 – presenting principles and functional requirements for records. This standard was prepared by the International Council on Archives (as International Council on Archives and the Australasian Digital Recordkeeping Initiative Principles and Functional Requirements for Records in Electronic Office Environments) and was adopted by Technical Committee ISO/TC 46.
- ISO/TR 26122 – concerned with work process analysis for records;
- Standards related to digitization (ISO/TR 13028 and ISO 13008) and distribution of records (ISO/TR 17068).

ISO 15489

ISO 15489 is a best-practice model to records management, providing detailed specifications for the structure, content, and implementation of a records management system and serves as basis for other standards. [11] This standard is divided in two parts.

In the first part we find the core standard used to establish the fundamental principles and model. The goal is to define the value and key processes of the theme in terms that any concerned reader can understand (management and non-technical personnel). It presents principles to guide the achievement of best practices in records management of originating organizations, either public or private and for internal or external clients, and is intended to be used by individuals concerned with the creation and maintenance of records. It presents the necessary requirements for records management from functional, social and organization perspectives, conception and implementation of a records system (its characteristics, creation and implementation methodology and its deactivation) and the processes and management control of records. It also addresses the importance of knowing the regulatory environment in which the organization is inserted, the identification and definition of policies and responsibilities, and the control, revision and adjustment processes to ensure that the system adapts to organizational changes. The standard presents essential characteristics that a record must have:

- **Authenticity**: An authentic record can be proven to be what it purports to be, created by the person who purports to have created it and at the time purported.

- **Reliability**: A reliable record has accurate information and can be trusted on.
- **Integrity**: A record with this characteristic is complete and unaltered.
- **Usability**: A useable record can be located, retrieved presented and interpreted.

Also it presents Records Systems characteristics:

- **Reliability**: a records management system should be capable of continuous and regular operation in accordance with responsible procedures. The system should also be responsive to environmental changes (changing business needs) without impact on the characteristics of the records it holds.
- **Integrity**: control measures should be available to prevent any action that might compromise the records characteristics.
- **Compliance**: systems should be managed in compliance with all requirements arising from current business, regulatory environment and community expectations existing in the context in which the organization operates.
- **Comprehensiveness**: records systems should manage records resulting from the complete range of business activities for the organization in which they operate.
- **Systematic**: the creation, maintenance and management of records should be made systematically.

The second part is an implementation guide with a methodology to facilitate the implementation of the principles presented in the first part, i.e. eases the comprehension of the standard to organizations wishing to comply with those rules; however not all principles presented in part one are detailed in part two. This part is targeted to records and information management personnel and other technical professionals. ISO 15489 describes in detail the core features of the records management process, presented in the order these would be managed if they were in a physical form [12]:

- **Capture** – Determines basic records information, which records should be kept, for how long and who have access to them. Metadata connected to the documents should be kept in a way that can describe the context, content and structure of the record, and the information about the people involved in the transaction.
- **Registration** – Provides evidence of creation or capturing of a record in an electronic records management system and a way of formalizing the capture of the record into an electronic records management system. Records are registered at several levels of aggregations.
- **Classification** – Process of identifying the category of business activity and to group records within them. This process is done in order to ease the description, control, links and determination of disposal, and access of records.
- **Access and security classification** – Related to classification, this process manages the access restrictions to a determinate record according to its classification; it can have restrictions because of legally classified records to protect personal information, intellectual property rights, and other legal and professional privileges. A record restriction is related to specific requirements by business needs or laws.

- **Identification of disposal status** – Process for identification of the disposal status and retention period of a record. This is done by determining the business activity, the records class and the relevant retention period and also which metadata is to be retained and which is to be destroyed.
- **Storage** – Process to ensure that records are protected, accessible and managed cost-effectively.
- **Use and tracking** – In this process user permissions associated with individuals are identified as well as the access and security status of records for their use. Whenever a record is used this should be registered in its metadata since this can affect its access and disposal status.
- **Implementation of disposal** – It should be possible to retain electronic records and associated metadata removed from the system to ensure that the information is available through the whole retention period. When the retention period is over the records should be destroyed.

The standard specifies that these processes can take place simultaneously or in a different order from the specified one. All of these processes generate detailed descriptive information, called metadata, which is linked to the record. This information depends on the elaboration of the records system, which is determined by the business and accountability requirements of the organization.

ISO 30301

This ISO is part of a series of International Standards, the ISO 30300. These series of standards are designed to “assist organizations of all types and sizes, or group of organizations with shared business activities to implement, operate and improve an effective management system for records (MSR)” [14]. The first standard of the series, ISO 30300 is an introductory standard of all series, defining the scope and motivation and the glossary used in the rest of the documents. Published in 2011, these series thrive for establishing the objectives for using a records management system, principles for it, description of a process approach and specification of roles for top management is a concern of these series. [13]

ISO 30301, in particular, presents the specification of the requirements for a MSR. It is directed to organizations that intend to establish, implement, maintain and improve a MSR to support its business, assure conformity with its stated records policies and conformity with the ISO itself. [14] Provides guidelines for the understanding of the organization's context in general (external and internal factors, applicable laws and rules), top management involvement in the implementation of the management system, definition of roles, responsibilities and authorities, planning of achievement of objectives, support to achieve those objectives, operation processes to establishment of the system, performance evaluation methods, and finally improvement processes to correction of undesirable events and continuous upgrading of the system.

ISO 16175

The ISO 16175 [15] is a standard that presents Principles and Functional Requirements for Records in Digital Office Environments, focused primarily in the creation and management of digital records.

Created by the International Council on Archives this standard is a harmonized and generic suite of functional requirements for software products to make and keep records taking in account the existing jurisdiction specific specifications in a manner that is consistent with the ISO 15489 standard. The set of functional requirements presented by the standard is meant to not only inform the development of electronic records management software, but also to help in an eventual incorporation of records functionality into generic business information systems software products. The scope of the standard defines some main objectives of the project, for example an aim to improve the management of records in organizations, supporting their business needs enabling more effectiveness and efficiency of the operations, ensure good governance through good records management, among others.

This suite of principles and requirements will "assist jurisdictions that are developing, or looking to adopt, their own functional specifications, as well as inform the update and revision of previously existing standards"[15] and is divided in three modules:

- First module presents guiding principles that drive the development and implementation of systems to make, keep and use records, divided in records-related principles and systems-related principles. It also presents implementation issues.
- Second module presents the importance of implementing a records management system, an overview of the functional requirements and guidance for implementation of high-level functional requirements.
- The third and last module presents a set of guidelines and functional requirements to organizations that desire to add records management functionalities to existing information systems.

Core features for this standard where identified by simple analysis of the document. The following core features and text citations were possible to understand [11]:

- **Create** – includes:
  - **Capture** – "Digital records management systems uniquely capture, classify and identify records to ensure that their content, structure and context of creation are fixed in time and space".
  - **Supporting import, export and interoperability** – Since records can be transferred to other organizations or systems, the management system must use open formats and industry standards.
  - **Identification** – Every record within the system must have a unique identifier.
  - **Classification** – Similarly to the other standards there must be a classification scheme to aggregate records.
- **Maintain** – includes:
  - **Controls and security** – Access to, or alteration of metadata must be controlled to avoid alteration of records content, structure and context.

- **Hybrid Records** – The system must be able to “ingest and maintain” records management metadata regardless of this record being non-digital or digital.
- **Retention, Migration and Disposal** – The system must be able to retain, migrate and dispose a record taking in account the disposition authorities

- **Disseminate** – includes:
  - **Search, retrieve and render** – “Search tools to locate records by employing a range of searching techniques. Retrieving is the process of preparing the located records for rendering and viewing. “. Render refers to the reproduction of a record in a human-readable representation.
- **Administer** – As the majority of systems there must be administration tools to manage the record management system in terms of user administration, system parameters, backup and restore data, as well as direct alteration or deletion of record content.

3.1.2 DoD 5012.02 - Design Criteria Standard for Electronic Records Management Software Applications

This document is a standard with functional requirements to develop Records Management Application software in the USA Department of Defense Components. Published in 2007 by the American Department of Defense, it is a USA standard, adapted to American laws and regulations, with the main purpose to be used by the American military departments for their records management. However it is used for guidance and certification outside of the USA. The document provides implementing and procedural guidance to manage records.

As present in [16] the standard contains mandatory baseline functional requirements and requirements for classified marking, access control, and other processes and identifies non-mandatory features deemed desirable for records management systems software. In addition it contains a section of the management of classified records, and a chapter of how records should be managed for the privacy act and the freedom of information act. The standard describes the minimum records management requirements that must be met, based on current US National Archives and Records Administration (NARA) regulations. It also contains a testing process in order to enable organizations certifying themselves to the standards.

The DoD 5015.02 core features [12] [16] present a division of general requirements and detailed requirements:

- **General Requirements**:
  - **Managing Recor**ds – a records management system shall manage records in accordance with the standard, regardless of storage media or other characteristics.
  - **Accommodating Dates and Date Logic** – information that contains dates shall be correctly accommodated and processed in current, previous, and future centuries.
  - **Meta-Tagging Organizational Data** – implementation of discovery meta-tagging shall be allowed.
  - **Backward Compatibility** – the capability to access information from their superseded repositories and databases shall be provided.

- **Accessibility** – “The electronic records management system should comply with Americans with Disabilities Act requirements, requirements about text-elements in web-based interfaces and follow standards of electronic and information technology accessibility.”[12]
- **Extensibility** – the capability to provide open standards interfaces in order to integrate into an organization’s information technology enterprise shall be included.
- **Security Compliance** – applicable security standards including Security Technical Implementation Guides shall be supported.

- **Detailed Requirements**:
  - **Implementing File Plans** – mandatory file plan components and mandatory record folder components are specified by the standard.
  - **Scheduling Records** – Phases and actions related to a retention/archiving schedule. [12]
  - **Declaring and Filling Records** – mandatory metadata and authorization requirements are specified by the standard to be able to associate attributes of a record folder with its containing record, as well as links between different records and folders.
  - **Filing Electronic Mail Messages** (E-mail) – the standard specifies that e-mails will be treated the same as any other record.
  - **Filing Records to be Later Transferred or Accessioned to NARA** – additional metadata added to the records to be transferred or accessioned to NARA
  - **Storing Records** – the standard specifies access to repositories requirements and requirements for retrieval of documents from it.
  - **Retention and Vital Records Management** – related to searching and retrieving, destroying, transferring, and retention of records.
  - **Access Controls** – requirements about authorized individuals and user-type roles and responsibilities are specified.
  - **System Audits** – the system shall provide capability to track actions performed in its objects. These track actions include the retrieve, create, delete, search and edit actions in an object.
  - **Product Combinations** – the integration of two or more distinct products (typically one creates records and the other performs the records’ retention schedule tracking) shall meet some requirements.

3.1.3 MoReq2010

MoReq2010, short for Modular Requirements for Records Systems**,** is a model that presents a set of requirements for reference, guidance and normalization. A Records Management System must (totally or partially) comply with this model to manage records in an efficient manner and can then be denominated of MCRS (Moreq2010 Compliant Record System). To get certified according to MoReq2010 a system must fulfill all of the functional requirements specified in the standard.

As mentioned in [17] this specification thrives to avoid a “one size fits all” approach to implement a records management solution. Instead it defines a common set of core services that are shared by many different types of

records systems, but which are also modular and flexible, allowing them to be incorporated into specialized and dedicated applications that might not previously have been acknowledged as records systems.

Created by the DLM-Forum community, its first version was published in 2001, then after reviews and evaluation MoReq2 was published in 2008 and in 2010 the most updated version was published, being this last update (MoReq2010) the document of interest in the scope of this work and the specification chosen to test the proposed solution.

The specification contains base concepts that are necessary to its understanding:

- Entity – An entity belongs to an entity type and normally has associated metadata (system and context), event history, and access control list.
- Metadata – Holds information that describes the entity. It is divided in system metadata (defined by MoReq2010) and context metadata (defined by the supplier and/or the user).
- Event History – It is a set of events associated to the entity, storing information about the different functions performed on the entity.
- Access Control List – It is a list of access control entries that specifies which users and groups can perfom functions on the entity. Specific sets of functionality are collectively defined as roles.

Records are managed in a records system as **entities** (users perform **functions** on entities; records are one of the many other entity types existing in the specification) and the **functional requirements** are bundled into **service definitions**; this service definition is what gives to MoReq2010 its service-based architecture. Entities and services have unique identifiers within an MCRS. An entity has **metadata**, **event history** and **access control list** as information associated with it.

MoReq2010 also has other important concepts that must be taken into account in order to understand its content. Table 1 presents some of those fundamental concepts.

Concept	Description
Destroy	After applying a destroy process to an entity, it becomes a residual entity. Some of its metadata, events, and in records their content, are deleted.
Delete	This process will completely erase an entity from the system.
Open/Close	This process is applied to aggregations. An open aggregation accepts the attachment of entities; a closed one does the inverse.
Inheritance	An entity can adopt characteristics or properties of other entity by heritance.
Disposal Schedule	Schedule that defines the life cycle of a record. Determines how long a record is retained and how is subsequently disposed of at the end of its retention period.
Disposal Hold	Prevents the destruction of a record by imposed legal or administrative orders.

Browse/Search/Discover	Discover entities through exploration of its relation to other entities, and browse across services and their entity types.
Inspect	Examine an entity and its metadata.

Table 1 - MoReq2010 Fundamental Concepts

The specification presents nine core service definitions, and the functional requirements are bundled into their definitions. These services together describe the functionality required by a MCRS according to MoReq2010. The first module (which also describe a service but with different characteristics of the other nine) describes the common functionality required by every MoReq2010 core service. The most important service or heart of the service-based arquitecture of a MCRS is its **Record Service**. Unlike the others, this service cannot be shared with another MCRS, being the only service that distinguishes one MCRS from another. Figure 6 gives an overview of a MCRS seen as a grouping of interrelated services with a service based architecture.

MoReq2010 has two services that are model services - **Model role service** and **Model metadata service**. These services are not required by the specification to be implemented exactly as they are described, despite each one of them presenting a set of functional requirements. However, if the supplier wishes to support advanced modules (such as the import module) the exact description for implementation is advised.

The interface to the MCRS, the **Classification Service** classes' series, and **Record Service** storage feature are areas in which where plug-in functionality is specified. Plug-in modules are sets of functionality that allow for alternatives in achieving the same goal but in different ways, giving the suppliers the opportunity to choose from various approaches.

Next a brief summary of each of the core services is presented, as well as a table with the defined operations in the service.

User and Group Service

Contains features to ensure the ability of users' identification and definition of the groups to which they belong. Data and historical information about users and groups should be available, and every user or group should be represented with entities to represent them. When the user or group is no longer active it should be put in a residual state so that the context of the user or group will still be available. Table 2 summarizes its operations.

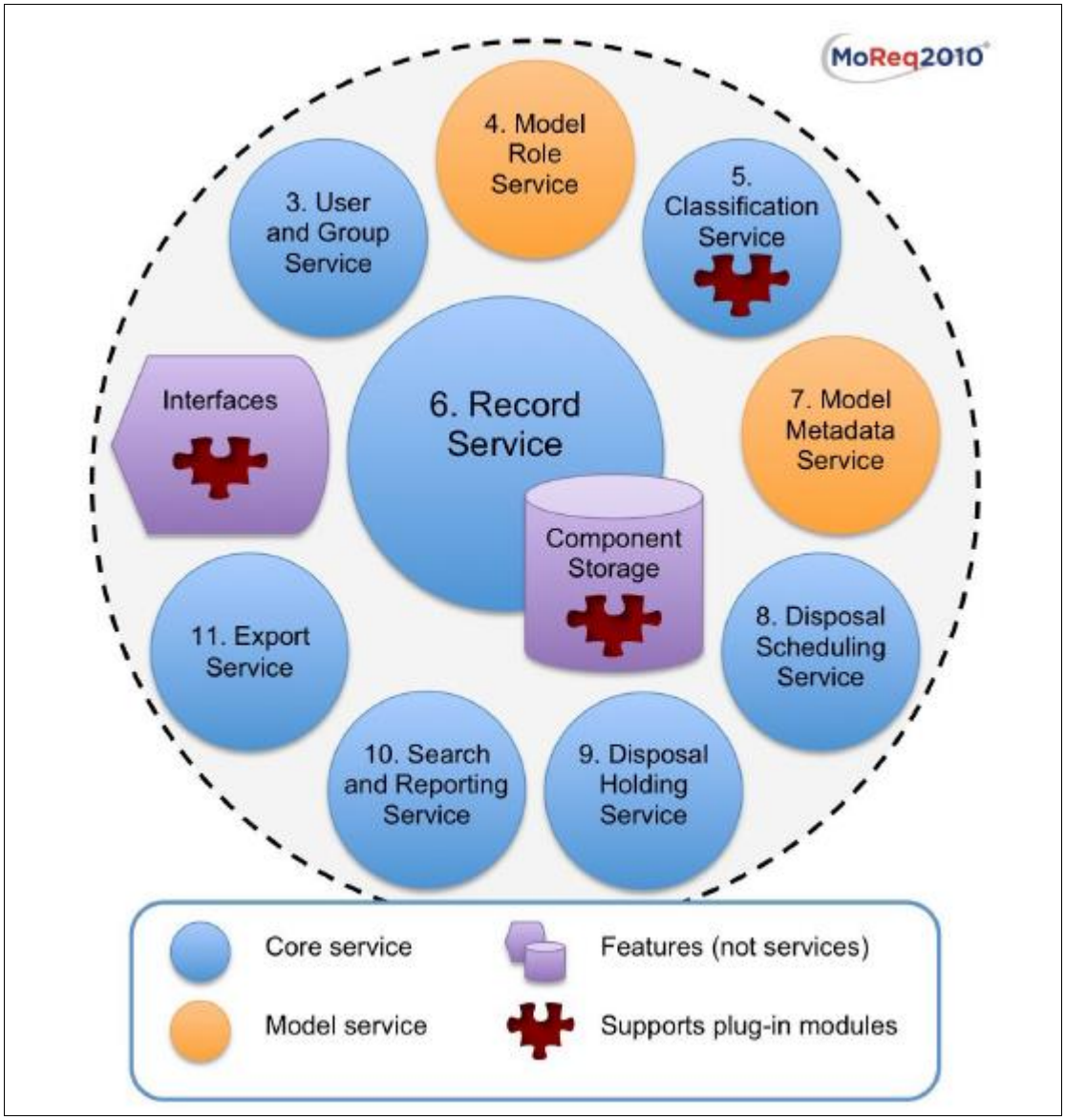


Figure 6 - Service-Based Architecture of a MoReq2010 Compliant Record System

Operations:
Create, delete and destroy Users and Groups, and Modify Metadata details.
Add/Remove Users or Groups
Generate Reports listing active groups that a user belonged to, and listing active users that belonged to a nominated group.
Inspect users and groups of the service.

Table 2 - User and Group Service Main Operations

Model Role Service

Define how the user permissions to perform operations on the system should be managed. A MCRS must always ensure that a user is allowed to perform a certain function by conducting a preliminary check in order to verify that this user has that authority. This authority is given to users by granting them roles.

Operations:
Create, delete and destroy Roles, and Modify Metadata details.
Define a role as administrative.
Modify an Access Control List of an entity.
Associate or Disassociate a function to a role.
Generate Reports listing user permissions, and listing functions associated to a determinate role.
Browse functions through roles; roles through functions; and Access List Controls through entities.
Browse in an Access List Control a role, user, or group
Inspect Roles on the service.

Table 3 - Model Role Service Main Operations

Classification Service

Defines the requirements to the classification plan of the records management system, concerning the inheritance requirements. Every record must be associated with a class entity from its creation to give the record a definitive context and a link to other relevant records; classes represent business functions, activities and transactions and can be defined by the business.

Operations:
Create, delete and destroy Classes, and Modify Metadata details.
Change the disposal schedule of a class for another one.
Replace a nominated class with another class
Browse classes and their associated entities, through disposal schedules and holds.
Inspect classes of the service.

Table 4 - Classification Service Main Operations

Record Service

This service is responsible for the records management on the different aggregation levels. Each aggregation can represent a group of records or other aggregations, and the classifications, access controls and metadata are inherited. There is no possibility to have aggregations and records at the same level, since this is not allowed. Each record is an entity that is made up of its metadata, event history, components and access control list. There can be as many aggregations as needed and it is possible to make duplicates of records to have them in different aggregations for business purposes.

Operations:
Create, delete and destroy Aggregations, and Modify Metadata details.
Replace a nominated aggregation already associated with another aggregation.
Add/Modify/Remove the maximum number of aggregations allowed to each level of the classification plan.
Open/Close Aggregation
Add or Remove an aggregation to another one.
Inspect aggregations of the service.
Browse aggregations through classes, and disposal schedules through aggregations.
Create Records, and Modify Metadata details.
Reclassification of a record.
Add or Remove a record from an aggregation.
Inheritance of the disposal schedule from a class to a record.
Replace a disposal schedule of a record.
Duplicate record, including all its metadata, access control list, event history, components and components content.
Inspect records through its aggregation or class and components, disposal schedule, disposal hold, and components through records.
Browse aggregations and records associated to a class, and records associated to a disposition schedule.
Create components and Modify Metadata details.

Table 5 - Record Service Main Operations

Model Metadata Service

Manages the types of entities and its associated metadata. There are two types of metadata: system metadata and contextual metadata. The first is required by each entity and each service and is specified as part of the functional requirements; this is a set of metadata elements which are required to successfully manage records

within a MCRS and to transfer them to other records systems. The second one is additional metadata specific to a particular implementation; this metadata enriches the historical and operational context of entities.

Operations:
Create, delete and destroy System Metadata (SM) and Contextual Metadata (CM) and Modify Metadata details.
Delete a residual record metadata, registering the event.
Browse classes and their associated entities, through disposal schedules and holds.
Inspect metadata element definitions and templates of the service
Create, delete and destroy Templates and Modify metadata details.
Associate Templates to entities

Table 6 - Model Metadata Service Main Operations

Disposal Scheduling Service

It applies majorly to records and the management of their life cycle. Allows for the management of the selection tables of the system, and corresponding options of end to the document. A record can never be fully deleted as there will always be a remaining residual record to prove its existence. Instead of complete deletion the record will go through a transition from an active entity with complete metadata, history and content to a version with some of its parts erased. The disposal schedule determines when this process will occur. Every record, aggregation and class must have a disposal schedule, having one of the four following outcomes:

- Permanently retain;
- Review at the end of the retention period;
- Transfer at the end of the retention period;
- Destroy at the end of the retention period.

Operations:
Create, delete and destroy disposition schedules and Modify Metadata details.
Replace or Modify an already associated disposition schedule
Alerts to records with retention due date.
Inspect records due to disposal.

Review disposal schedule of a record.
Cancel or confirm a transfer or destruction of a record.
Stop the destruction of a record with an associated disposal hold. Immediate destruction of the record after lifting the disposition hold.
Retention of the disposition schedule of a residual record.

Table 7 - Disposal Scheduling Service Main Operations

Disposal Holding Service

Manages the holding mechanisms for the protection and preservation of documents, according to administrative or statutory requirements. The mechanism of disposal hold prevents the destruction of a record by retaining the disposal schedule in the moment of destruction of the record; the disposal hold only stops this process until it is lifted and then the process of destruction continues normally.

Operations:
Create, delete and destroy Disposal Holds, and Modify Metadata details.
Add/Remove an entity from a disposition hold.
Replace a nominated class with another class
Browse records, aggregations, or classes, through disposal holds.
Inspect disposal holds of the service.

Table 8 - Disposal Hold Service Main Operations

Searching and Reporting Service

Assures that the system possesses several methods to a flexible and effective search for an entity. It must be possible to browse an entity to its related entities, or to search for them based on values in their metadata and for the user to view what is allowed to do with the entity. It should be possible to search in full-text and find all metadata and be able to combine results of searches to perform complex searches. The search results are always expressed as a list of entities, and the appearance of this list should be configurable by the user.

Operations:
Specify Search queries and Criteria

Combine, Chain or Join Results of search queries to answer complex searches
Generate Detailed and Summary reports based on searches
Generate events and include them in the user's event history for both performed searches and reports generation
Save, Modify, Delete and Share report definitions

Table 9 - Searching and Reporting Service Main Operations

Exporting Service

Allows for the complete information description of entities in a sufficient detailed manner, enabling their successful transfer to another records management system; their metadata values, event histories, access controls and contents should be possible to fully transfer to another system. This is done in a common XML data format. Related to export is import where records from another system are imported to the one being used. For every export, an export identifier must be created to make it possible to find the entities exported together. Whenever an export of entities from one system is successfully made to another one (with no loss of business context) interoperability is achieved.

Operations:
Export entities to a XML data file.
Retention of exporting residual entities.
Determine which entities to export in full or as placeholder.
Generate Unique identifier for the export.
Provide a text comment to be included in the export data

Table 10 - Export Service Main Operations

MoReq2010 Non-Functional Requirements

MoReq2010 also has a number of non-functional requirements in separate chapters. These are more difficult to specify universally and to measure and test subjectively, however the specification notes its importance for the use of a records management system. MoReq2010 specifies non-functional requirements for performance, scalability, manageability, portability, security, privacy, usability, accessibility, availability, reliability, recoverability, maintainability, supported, warranted, and compliance.

Performance is related to the responsiveness, efficiency and throughput of the records system under load. Depending on various factors, performance may change according to the amount of use of the system (being accessed at the same time by the whole organization) or the used hardware. **Scalability** is related to the

performance and capacity of the system over time and again under increasing load. **Manageability** takes into account factors as the capacity of the system to make provision for its own management and administration.

Portability is concerned with factors related to the ability of the system to operate successfully in different environments. **Security** refers to the external integrity of the system and its ability to withstand unauthorized access, hacking or tampering, computer viruses, and other forms of accidental or malicious damage; **privacy** enforces security since the system must protect the information it holds. The system may hold sensitive information and is crucial that is secured and private. **Usability** is concerned with the ease of use of the system, especially from the user's perspective as a system to complex or time consuming to use will be difficult to accept. **Accessibility**, which is related to usability, enforces that the system should be easy to use by any type of user and with different capabilities, including users with specific disabilities. **Availability** concerns with which characteristics a system must have to suit a particular business relating to system uptime or down time; some organization's business practices require a system to be available 24/7, others need critical availability at night, etc. **Reliability** concerns with the internal integrity of a system, the precision and accuracy of its software, and its resilience to defects, malfunctions and unexpected operating conditions. **Recoverability** takes into account that even though a system can be reliable there might be particular cases that make the system fail; therefore there is a need to have capacity of recovering organization data largely intact. **Maintainability** is related to the ease of eventual repairs and upgrades of the system. **Supported** and **Warranty** requirements refer to suppliers active support of the systems and warranty to fix any upcoming problems. Finally, **compliance** it's the level which a system complies with industry standards and local regulations.

MoReq2010 Specification Quality

MoReq2010 as a requirements specification has to comply with several quality criteria, not only as a specification as whole but also each individual requirement. There are two important references standards from the Institute of Electrical and Electronics Engineers (IEEE) for documenting requirements; the IEEE 830-1998 - Software Requirements Specifications (SRS) [18] and the IEEE 1233-1998 - System Requirements Specifications (SyRS) [19]. The IEEE standard 830-1998 advocates that a SRS should be divided into three top-level parts as presented in table 11.

According to [20] and [21] MoReq2010 contains information required for the section "Introduction", as present in the first top-level part of the IEEE 830-1998 standard. The information is in chapter 13. Glossary, and also every first section of each chapter. The "Description" is present in chapter 1.Fundamentals and 14. Information Model. The "Specific Requirements" are spread across each chapter in a unique section dedicated to functional requirements.

Apart from the structure, each requirement from the specification has also quality criteria that it must meet. These quality criteria are fundamental to define the whole specification quality, complementing with the specification's structure and as a whole quality. Some of these criteria are:

- **Consistency:** A requirement should not conflict with another one, and its terminology should be consistent with other requirement and glossary terms.
- **Uniqueness:** A requirement should be in one place without missing information.
- **Atomicity:** A requirement should only represent one need. If it's possible to divide an expression of a requirement in two or more expressions this requirement is not atomic.
- **Verifiability:** It must be possible for a requirement to be verified by inspection, demonstration, test, or analysis.
- **Feasibility:** A requirement should allow for an implementation within cost, schedule, and technical limitations.
- **Clearness:** A requirement should be understand as clear as possible, allowing for one and only one interpretation. No ambiguity.
- **Conciseness:** A requirement should be exact and succinct.

Table of Contents		**Description**
Introduction	Purpose	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.
Description	Product Perspective	Describes the dependencies and relations with other products.
	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	Define general constraints that limit the developer's options, such as regulatory policies, hardware limitations, and interfaces to other applications.
	Assumptions and	List of all the factors on which the document content's

	Dependencies	rely, since changes in these factors can have a strong impact on the requirements stated in the SRS.
Specific Requirements		Comprises all the requirements of the document. These can be organized in Functional and Non-Functional. The later can be organized according to the user, relate entities, functionalities, among others.

Table 11 - Summary of IEEE 830-1998 standard on specification structure

As present in [20] and [21] MoReq2010 requirements are no well-formed. A small random sample of requirements was taken and analyzed, resulting in the following table [21]:

Requirement	Expression	Atomic	Unique	Feasible	Clear	Concise	Verifiable
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	Yes	No	No	Yes
R.4.5.08	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	Yes	No	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	Yes	No	No	Yes
R8.4.1.14	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	Yes	No	No	Yes
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	Yes	No

Table 12 - No Well-Formed Requirements from MoReq2010

This table is read as follows:

- R2.4.3 is non-atomic, since it expresses two needs: “browsing services (…)” and “inspect metadata”. Is not clear, as we must avoid referring to any subject without expressing that subject explicitly, since it can cause misunderstandings: “of each” may be about the services, but it can also be about the users, or the system itself.
- R4.5.08 is non-atomic and unclear since it expresses three different needs and is also too long.
- R6.5.20 is non-atomic and unclear; the subject in “its Contextual metadata (…)” is probably the component, but it can also be the user, the MCRS itself or one of the components properties.
- R8.4.14 is non-atomic, as it’s expressing more than one need, unclear, as it offers different alternatives, in a confused way; it’s also ambiguous since does not define clearly when the state update must be done.
- R10.4.16 is unclear, as it makes use of the term “several" in a way that it cannot be verified.

One can conclude that there are no well-formed requirements through the entire document, by extrapolating this sample to the overall document, where the atomicity and clearness are criteria that are frequently violated. However, in [20] it is noted that in comparison to older versions of MoReq specification, MoReq2010 is more normalized than the previous ones, as all the analyzed expressions begin with statements such as “The MCRS must …”, a structural worry not evident in the previous versions, offering a much more clear text, with much more comprehensiveness.

4 Problem Analysis And Design

This chapter presents a description of the problem analysis and design in two different abstractions. First an explanation following a “black box” approach where one is presented with the proposed solution and what should be the system to build. Then a more detailed explanation referred as a “white box” approach, with presentations of internal schemes of the solution.

4.1 Black Box Analysis

The proposed solution can be conceptualized as a system, presented to users as a web application, where the documents in consideration should be available in a form that allows them to perform some pre-defined actions. These actions can be thought as functionalities available in the application interface, allowing a user to see and manipulate data from the reference documents. The most relevant functionalities of the application are presented in the next section with the help of use cases. Still in this analysis, is available an overview on MoReq2010 information as data transformed to be embedded in a system and KAOS contribution in this project.

4.1.1 Proposed Actors and Use Cases

Actors

In order to develop this system there was a need to create scenarios of its manipulation with the use of use cases associated to a set of actors. There are three types of actors which are also the expected users of the system: the **Administrator**, **Expert User** and the **Professional User**. The Administrator is in charge of the uploading of documents to the system, as well as any management action needed. Expert User is an actor capable of performing actions related to the management of associations between documents as well as relationships in a set of requirements. Also this user will be capable of creating possible views of these associations and relationships. This user must have a broad knowledge of records management in order to understand what correlations might exist between requirements; it also must have knowledge of Goal Oriented Requirements Engineering techniques to ensure that the created views comply with the used method. The Professional User, inheriting from the Expert User, will explore, manage, and possibly extract the created views.

Use cases

In the first part of this project one advanced some possible use cases of the system. These use cases were **Upload Document** and **Associate Requirements** for the Expert User:

- Upload Document – the user will upload reference requirements documents in the system for it to create the respective relations with the metamodel.

- Associate Requirements – the user can create associations between requirements of different documents or inside the same document according to his concern. This will be a management of requirements associations' views.

At the time the concept of **metamodel** was a structure that should act as a basis for relations between documents. Throughout the project this metamodel was detailed into the basic structure that the documents should have inside the system, which is later described through this thesis.

It was also created use cases for the Simple User:

- Manage Custom View – the user can manage the view of the requirements according to his concern.
- Explore View – the user can explore a previously created view; scan trough a determined concern across the documents, etc.
- Extract Custom View – the user can extract the view created in a document form.

These simple use cases were object of deeper detailing throughout the project and refinement of what the system should be. The **association of requirements** that was first thought acted as an abstraction for the later introduction of KAOS models to its organization; at the time there was no decision about what GORE method was to be explored. The considered **views** were also an abstraction of what would be a possible way to represent the desired organization of requirements, depending of the decision of GORE technique. The new proposed use cases for the system are represented in the figure 7 diagram, followed by a description of each one.

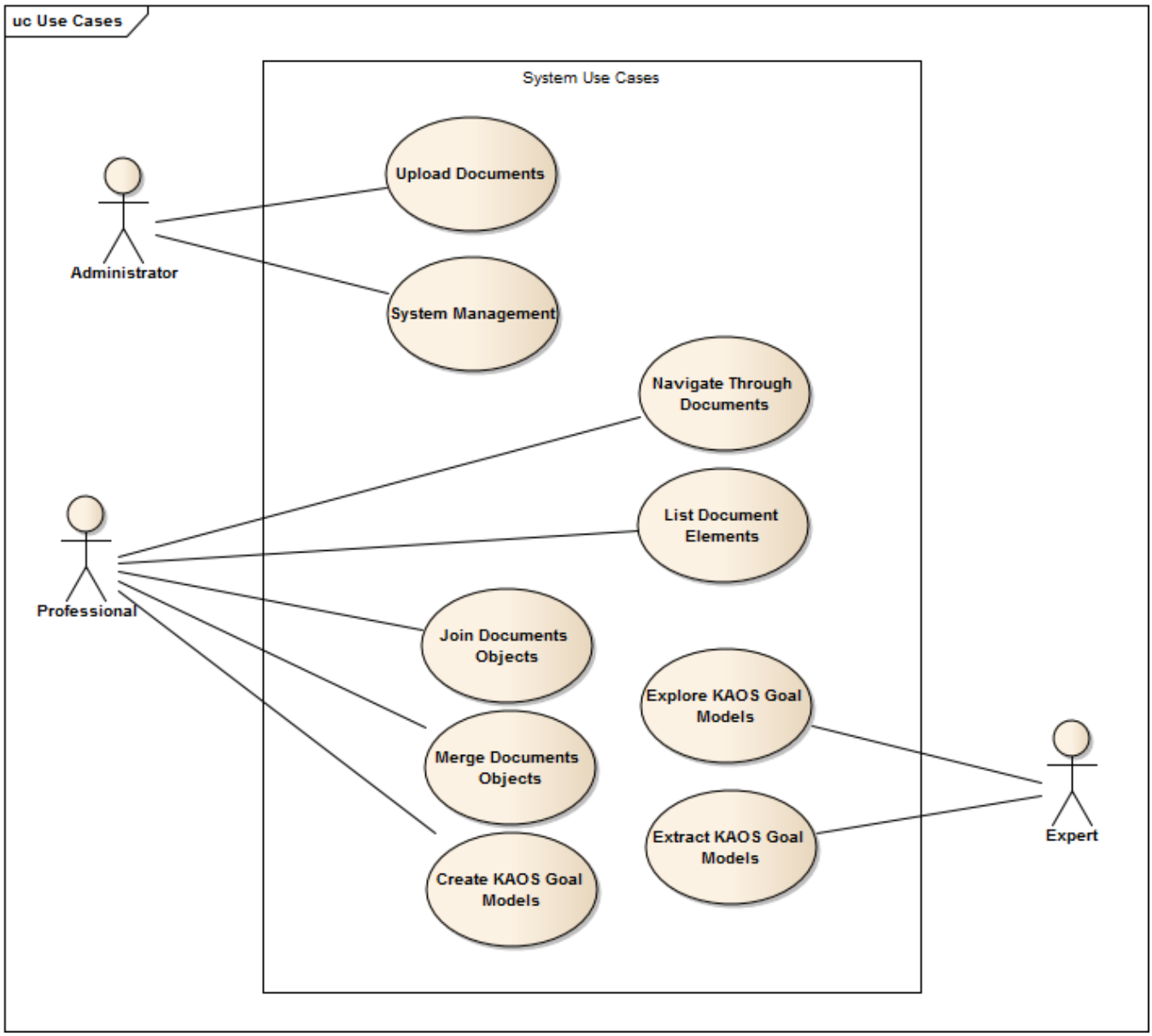


Figure 7 - System Use Cases

Use Case	Upload Document
Actor	Administrator
Goal	Upload a Requirements Reference Document into the system
Pre-Conditions	• System is up and running
Main Scenario	1. The administrator prepares the document to be accepted into the system's basic metamodel. 2. The document is embedded into the system.

Use Case	System Management
Actor	Administrator
Goal	General Management of the system in terms of availability, performance, security, among others.

Use Case	Navigate Through Documents
Actor	Professional, Expert
Goal	Navigate through the documents information
Pre-Conditions	• The document is embedded into the system
Main Scenario	1. The user accesses the application 2. Chooses a document to be visualized 3. The document is presented in a form of visual navigation

Use Case	List Documents Elements
Actor	Professional, Expert
Goal	Present selected information of the document in a listing format
Pre-Conditions	• The document is embedded into the system. • Availability of buttons that list information.
Main Scenario	1. The user accesses the application 2. Chooses a document to navigation 3. Clicks available buttons to present information

Use Case	Merge Document's Objects
Actor	Expert User
Goal	Acknowledge that two objects are equal and fuse them in just one object
Pre-Conditions	• One or more documents are embedded in the system.

	• Two objects of documents express exactly the same thing. • Availability of functionality to make the association.
Main Scenario	1. The user navigates through a document 2. Finds an object that can be associated to another 3. Marks that object and the other one, to be added to an association pool 4. Accesses the association pool and completes the association by linking one to another 5. Choose which object is to be considered from that point on when accessing the information expressed by them.

Use Case	Join Document's Objects
Actor	Expert User
Goal	Acknowledge that two objects are equivalent, creating a new node of aggregation between those objects and enriching the information they hold
Pre-Conditions	• One or more documents are embeded in the system • Two objects express different points of view about the same information • Availability of functionality to make the association.
Main Scenario	1. The user navigates through a document 2. Finds an object that can be associated to another 3. Marks that object and the other one, to be added to an association pool 4. Accesses the association pool and completes the association by linking one to another.

Use Case	Explore KAOS Models
Actor	Simple User
Goal	Explore a previously created diagrams expressing KAOS goal models
Pre-Conditions	• A Goal Model diagram is available in the system

Main Scenario	1. The user accesses a document in the system 2. The user chooses a goal model to explore

Use Case	Extract KAOS Models
Actor	Simple User
Goal	Extract an existing KAOS goal model from the system to a defined format for later use
Pre-Conditions	• A Goal Model diagram is available in the system
Main Scenario	1. The user accesses a document in the system 2. The user chooses a goal model to extract

The use case **Create KAOS Goal Models** is a generalization of 3 use cases, as present in the following figure.

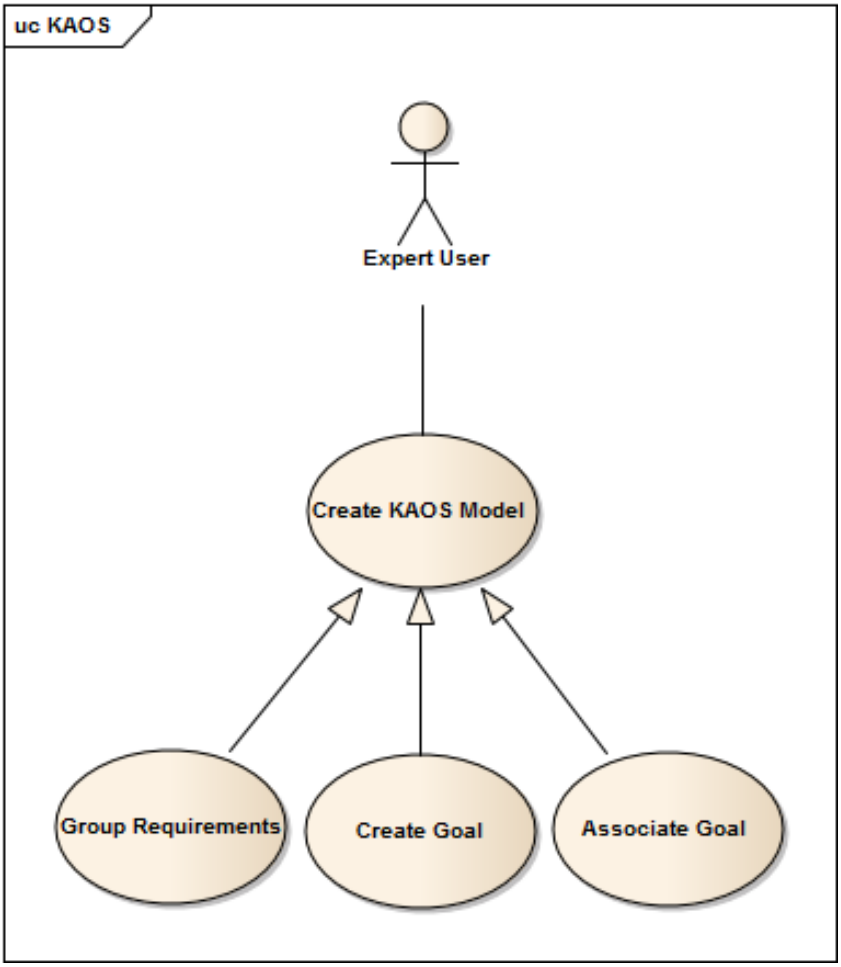


Figure 8 - KAOS Use Cases

Use Case	Create KAOS Goal Models
Actor	Expert User
Goal	Create a KAOS Goal Model using the available requirements. This use case is a generalization of other three.
Pre-Conditions	• A document and its requirements are available in the system. • Availability of Functionality to create a model
Main Scenario	1. The user accesses a document in the system. 2. The user chooses the option of Goal Models Creation.

Use Case	Group Requirements and/or Goals
Actor	Expert User
Goal	Group a set of requirements and/or goals that together fulfill a unique goal.
Pre-Conditions	• A document and its requirements, and/or a set of goals are available in the system.
Main Scenario	1. The user accesses a document in the system 2. Chooses between a set of requirements and groups them

Use Case	Create Goal
Actor	Expert User
Goal	Create a goal for a group of requirements/goals
Pre-Conditions	• A group of requirements and/or goals was made
Main Scenario	1. The user accesses the previously grouped set of requirements/goals 2. Creates a goal for that set

Use Case	Associate Goals
Actor	Expert User
Goal	Associate created goals to a set of requirements/goals
Pre-Conditions	• A group of requirements and/or goals is available • A goal was created
Main Scenario	1. The user accesses the set of requirements/goals previously created 2. Associates the created goal to its set of requirements/goals

4.1.2 MoReq2010 Data

As mentioned in the introduction chapter, the main focus of this project is to embed documents with reference requirements for records management into an information system, in order to perform some operations on its data. To simplify the initial work of preparing a document to embed in the system, in the scope of this project only MoReq2010 was addressed. With more than seven hundred requirements, functional and non-functional, this specification comprises a sufficient amount of material to work with and test the basic functionality of the system.

In a first approach, one must see the document exactly at what it is; a simple “document” or manual, with pages, sections, paragraphs, textual elements as titles and sentences, tables, and images. After analyzing the five documents one can notice that all of them follow this type of document structure. However, despite this similarity, there is an obvious difference of organization and presentation of its contents. For example, Figure 9 and Figure 10 show how ISO 15489 and MoReq2010 present some of their requirements, respectively.

8.3 Designing and implementing records systems

8.3.1 General

A records system should have the functionality that enables it to carry out and to support the records management processes described in clause 9.

Decisions about design and implementation of records systems and the processes they support need to be considered in relation to existing organizational systems.

8.3.2 Documenting records transactions

Records systems should contain complete and accurate representations of all transactions that occur in relation to a particular record. These include the processes associated with individual records. Such details may be documented as part of the metadata embedded in, attached to, or associated with, a specific record. Alternatively, they may be recorded as audit trails which should be kept at least as long as the document to which they relate is retained.

Figure 9 - ISO 15489 requirements presentation

A copy of the MoReq2010 specification was already available in a XML document format from a previous work [21]. This dictated the structure of the data to save on the repository, and also saved and additional work of transforming the specification PDF format or Word format into XML. These files were a set of XML documents containing the whole chapters of the main document, functional requirements, non-functional requirements, entities, functions, metadata, and data structures, all in separate documents and already with the “document” structure.

These documents had XML tags for **categories**, **pages**, **sections**, **images**, **paragraphs**, **texts**, **tables**, **links**, **bolds** and **italics**.

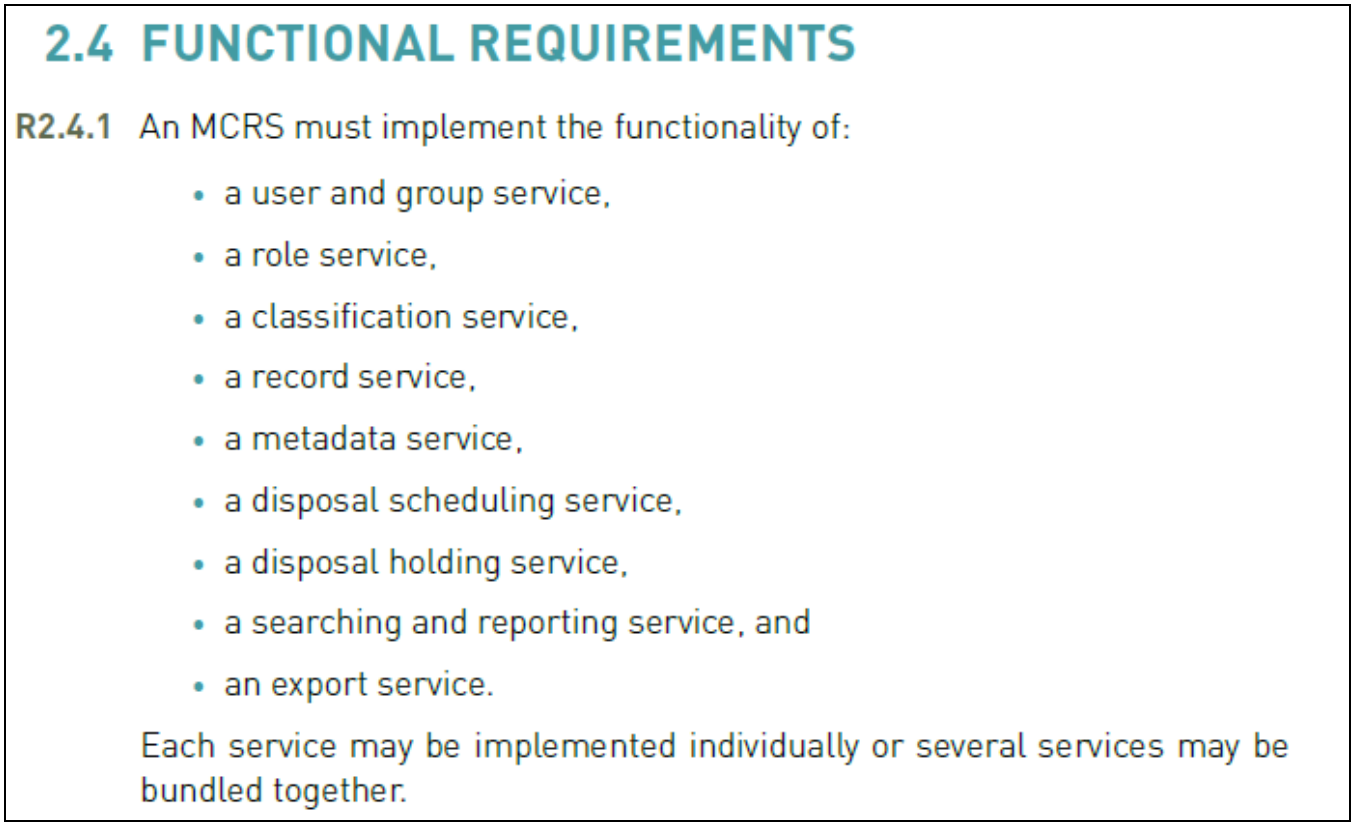


Figure 10 - MoRe2010 requirements presentation

Categories, which are only two in MoReq2010, divided Part One with the Core Services from the Part Two of the document. All of the following elements are children of these nodes.

Pages, despite the definition of what is a page, are not pages of the document; these are the chapters and include the titles. All of the following elements are children of these nodes.

There were two types of **sections**: the section container and the section content. The section container is a parent section for other section contents, typically comprising the titles for subchapters for main chapters. A section content is the last level child of a chapter, acting as a parent for paragraphs, texts, etc. According to figure 11, “1 Fundamentals” is a page and also a section content (purposely repeated since there are chapters where a brief description is included, making it a section with content), “1.1 Important Information” is a section container and “1.1.1 Intellectual property rights” is a section content.

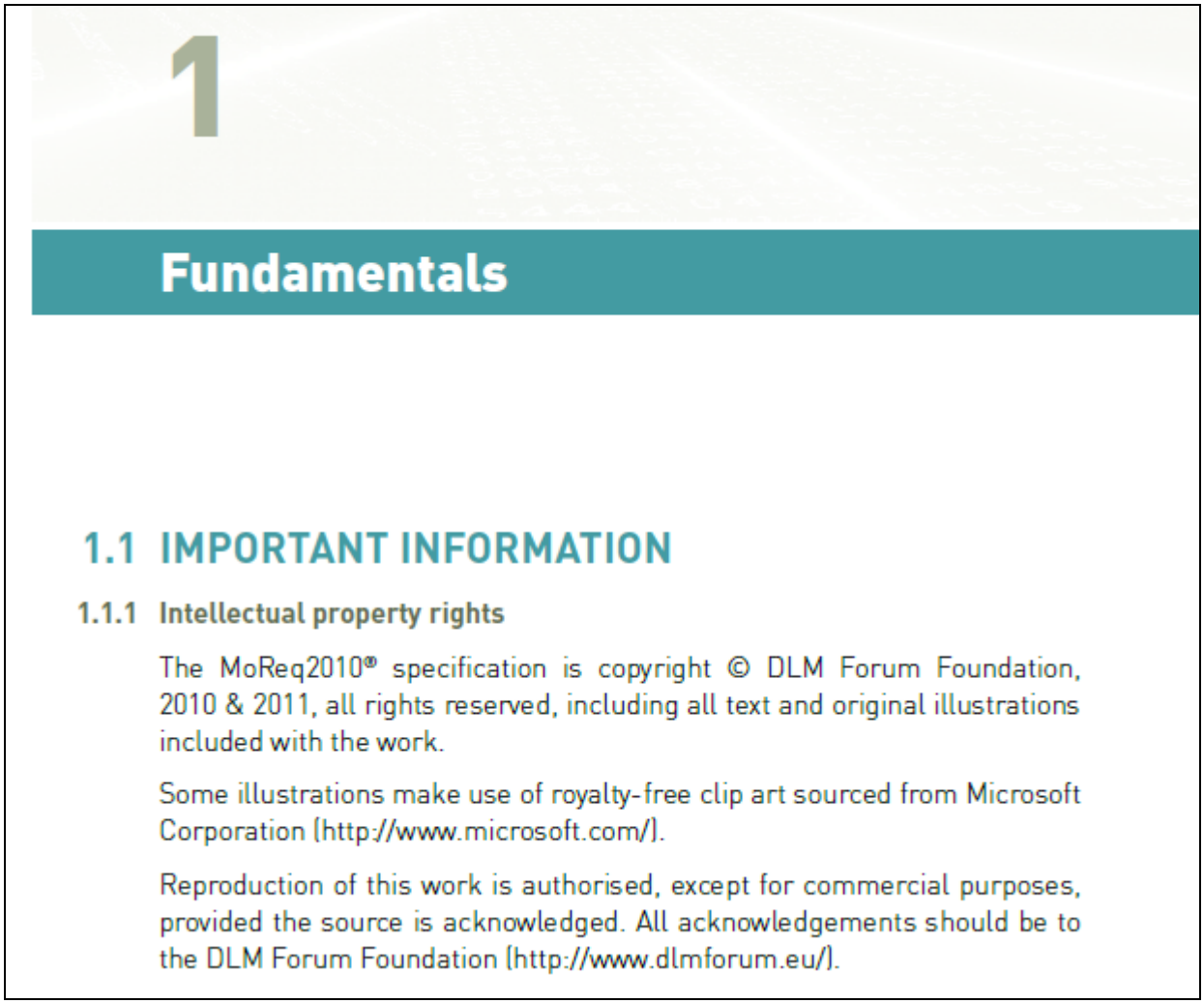
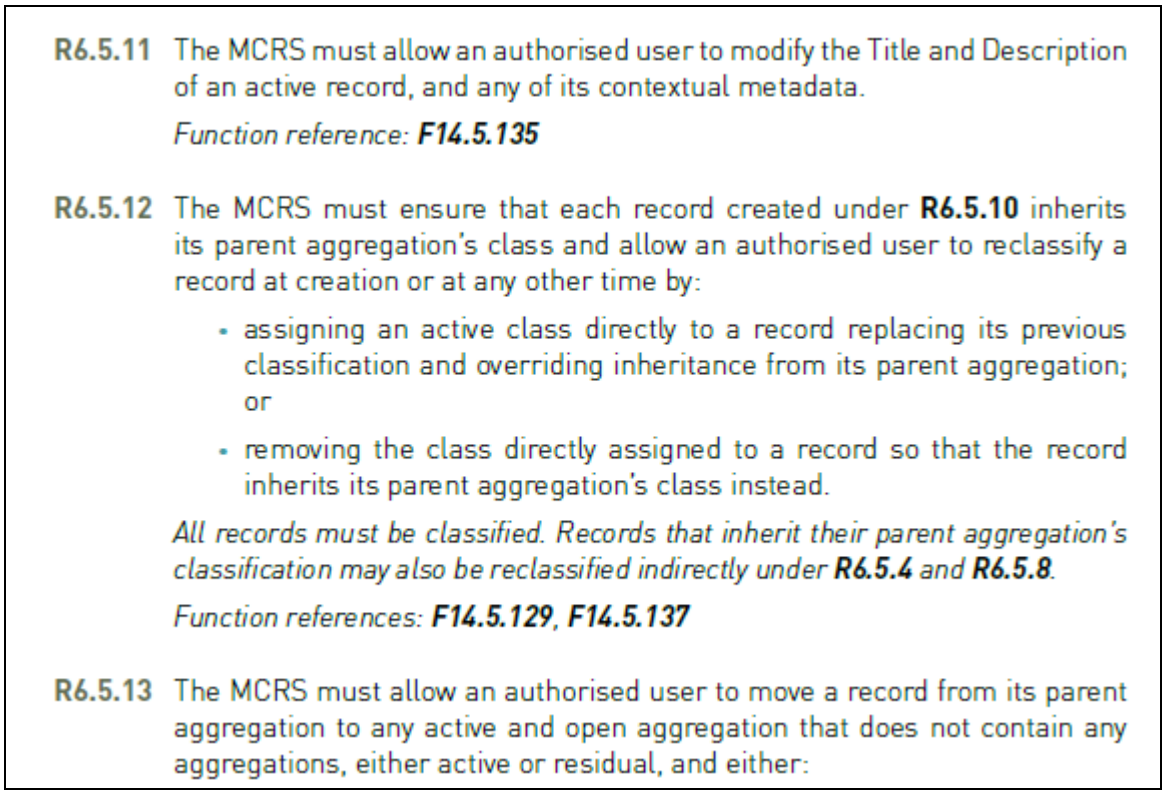


Figure 11 - MoReq2010 sections example

Section contents contain paragraphs, images, tables, links, bolds and italics. Paragraphs contain a text tag with the text existing in each paragraph; each text can have links to other elements inside the document, and can have a bold or italic format.

These elements together facilitated the design of the database model and its organization.

Divided in two major chapters, MoReq2010 specification presents a set of functional requirements and non-functional requirements, presented in a list format. In Part One – Core Services, all functional requirements appear on the last section of each service definition – figure 12; the main non-functional requirements are addressed in a dedicated and separated chapter. Part Two – Plug-in Modules, has sections for functional and non-functional requirements (separated) for each addressed chapter.



R6.5.11 The MCRS must allow an authorised user to modify the Title and Description of an active record, and any of its contextual metadata.

Function reference: ***F14.5.135***

R6.5.12 The MCRS must ensure that each record created under **R6.5.10** inherits its parent aggregation's class and allow an authorised user to reclassify a record at creation or at any other time by:

- assigning an active class directly to a record replacing its previous classification and overriding inheritance from its parent aggregation; or
- removing the class directly assigned to a record so that the record inherits its parent aggregation's class instead.

All records must be classified. Records that inherit their parent aggregation's classification may also be reclassified indirectly under ***R6.5.4*** *and* ***R6.5.8****.*

Function references: ***F14.5.129***, ***F14.5.137***

R6.5.13 The MCRS must allow an authorised user to move a record from its parent aggregation to any active and open aggregation that does not contain any aggregations, either active or residual, and either:

Figure 12 - MoReq2010 Functional Requirements Presentation

4.1.3 KAOS Goal Models

As seen in the previous presented Use Cases, the application should support the creation of KAOS Goal Models. In the related work chapter it is stated that KAOS allows for the creation of four types of models: Goal, Agent or Responsibility, Operation and Object.

In the first part of this work it was made an attempt to compare and align the main processes of a records management system presented by the reference documents, focusing on MoReq2010, ISO 16175 and DoD 5015.02. At the time ISO 15489 was thought to be only a guide from where the others were created, and therefore wasn't considered. After analyzing and reasoning about the exposed requirements in each document one was able to identify similar objectives and grouping them. This exercise is presented in table 13 as a summarized overview of the documents.

At the time one was already able to retrieve what would be the main concerns of these documents regarding the requirements for records management. These three documents all address concerns related to the creation of records, permissions of access to records and their security, classification according to their business context, disposition or records lifecycle management, search of records inside the system, and exporting of records to

other systems. Other concerns are addressed only by some of them, such as additional information for records storage (metadata), audits to the system, security and accessibility, and administration of the systems.

Moreq2010	ISO 16175	DoD 5015.02
User and Group Service Role Service	Controls and Security (Maintain)	Access Controls
Classification Service	Classification (Create)	Implementation of File Plans
Record Service	Identification (Create)	Declaring and Filling Records
Model Metadata Service		Accommodating Dates and Date Logic Meta-Tagging Organizational Data Filing Records to be Later Transferred or Accessioned to NARA
Disposal Scheduling Service Disposal Holding Service	Retention, Migration and Disposal	Scheduling Records Storing Records Retention and Vital Records Management
Search and Report	Search, retrieve and Render (Disseminate)	Storing Records Retention and Vital Records Management
		System Audits
Non-Functional Requirements		Accessibility Security Compliance
	Administer	System Management Requirements
Exporting Service	Supporting import, export and interoperability	Retention and Vital Records Management

Table 13 - Alignment between Reference Documents

However, this type of alignment can only help into having an insight of the addressed concerns in these documents, perceiving what is addressed where and what is missing in some of them, and also in creating a mental organization of the addressed concerns. KAOS models provide a better way to do this exercise, reaching deeper levels of understanding, by addressing requirements and linking them to higher goals.

4.2 White Box Analysis

The conceptualization of the system as a black box offers an abstraction to focus on its main functionality. A white box analysis will provide an insight on how this functionality is achieved. The next sections present the structure of the repository of the system and its different layers and also the exploration of KAOS goal models with MoReq2010 requirements.

4.2.1 Repository Structure

Section 3.1.2 of this chapter presented the expected structure of the documents in general, in terms of pages, sections, paragraphs, among others that should be stored in a repository. The first step was to create a domain model for a database to support the storage of that structure. The Domain Model is presented in figure 13.

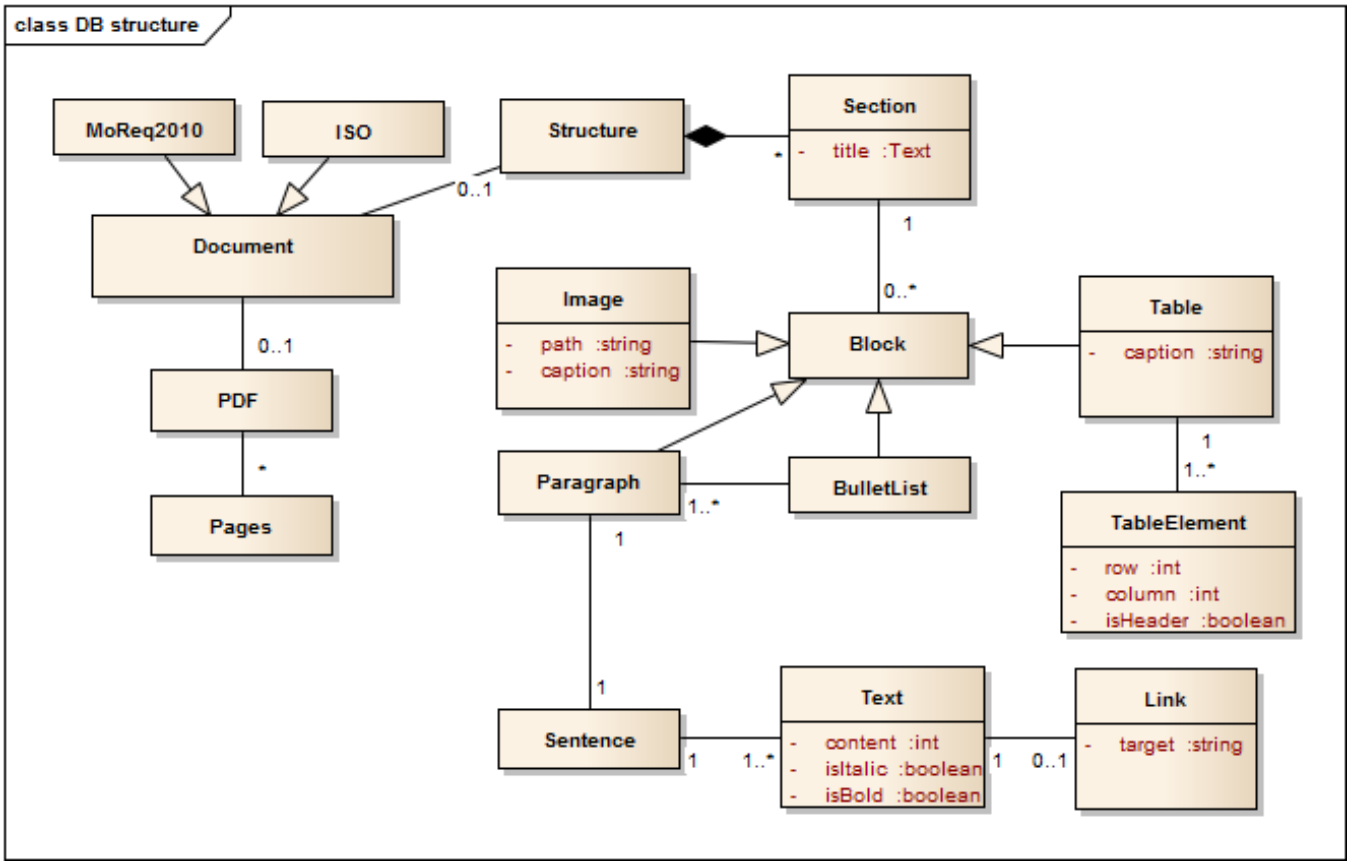


Figure 13 - Documents Structure Model

A **Document** can be any of the five discussed documents in the form of a PDF with several pages. These documents have one structure, which is unique for each one of them, despite the possible similarities in its components. The **Structure** is a set of sections, each of the later having **Blocks**. The blocks contain all of the components that can be found inside any section of the documents.

From this model, one can identify layers for abstraction and better understanding of the proposed system. First, a **Structure Layer** concerning precisely the above presented model of a repository to store the documents.

A user of the system can create certain associations between parts of different documents or inside the same document. The layer where all the services to interact with the documents' contents are made available to a user can be denominated **Business Layer**. The associations can be made combining **components** or **objects**, from the structure layer, where the documents are considered as a whole. The **Objects Layer**, containing the lowest level of granularity, is where single objects are considered. For example:

- Images;
- Texts that can be part of phrases, titles, tables or images subtitles;
- Table elements, etc.

The **Components Layer**, with an upper level of granularity is where compositions of objects from the objects layer are considered. Examples are:

- Sentences;
- Paragraph;
- Image and respective Subtitle, which is a Image + Text composition;
- Sections, etc.

Objects Identification

As mentioned above, one of the possible actions in the system is to merge or divide components or objects from documents. For example one can decide that a requirement in one document is exactly the same in another one, giving the opportunity of keeping only one of them, however a track of this change must be available. In fact, all changes to a determinate document must be tracked down for acknowledgement and validation purposes. In order to achieve this, a URI (Uniform Resource Identifier) system was implemented in order to uniquely identify each object inside the application.

For this feature, one has three important concepts:

- **System URI Path** – Unique identifier of a component inside the system;
- **URI ID** – Unique identifier of a child parting from its parent, and also keeps the order in which this child is positioned in relation to other siblings. If this order changes, so does the URI ID.
- **URI Path** – Unique identifier of a component from a determinate chain of parent components.

Parting from a component, the URI Path of their child (and subsequent children) will be the same; this means that parents keep their children URI ID's. This URI behavior can be better understood by a practical example.

Consider a **section** having three children: a **paragraph**, a **bullet list**, and a **table**. This section "knows" its children URI ID, which is 1, 2 and 3, respectively. This **paragraph** has two **sentences** whose URI ID's are also known by the paragraph. Parting from the initial section, the URI Path for each of the two sentences will always

be the same; if the parent paragraph is the first element of the section then its sentences will have as a URI Path 1.1 and 1.2, parting from the section.

Let's suppose that the considered section is the same in two different documents (with URI ID 1 and 2), and adding a little of complexion to this example, positioned as the first section of document 1 and second subsection of the third section of document 2. For document 1, the System URI Path for the section will be 1.1, and for document 2 will be 2.3.2. Taking again in account the previously considered sentences, in this case their System URI Path would be 1.1.1.1 and 1.1.1.2 for document 1, and 2.3.2.1.1 and 2.3.2.1.2 for document 2, since we only need to add their URI path as a suffix on the section's URI Path.

4.2.2 KAOS Models

The KAOS meta-model contains goals, requirements, expectations on the environment of the system, conflicts between goals, obstacles, entities, agents, etc. These components altogether allow for the creation of diagrams to Goal, Responsibility, Object and Operation modeling. This meta-model presented in section 2.3 KAOS (figure 5) provides the graphic objects to be considered in order to build a diagram. In this work the program StarUML with a Requirements Engineering Toolkit [28] was used in order to create the diagrams.

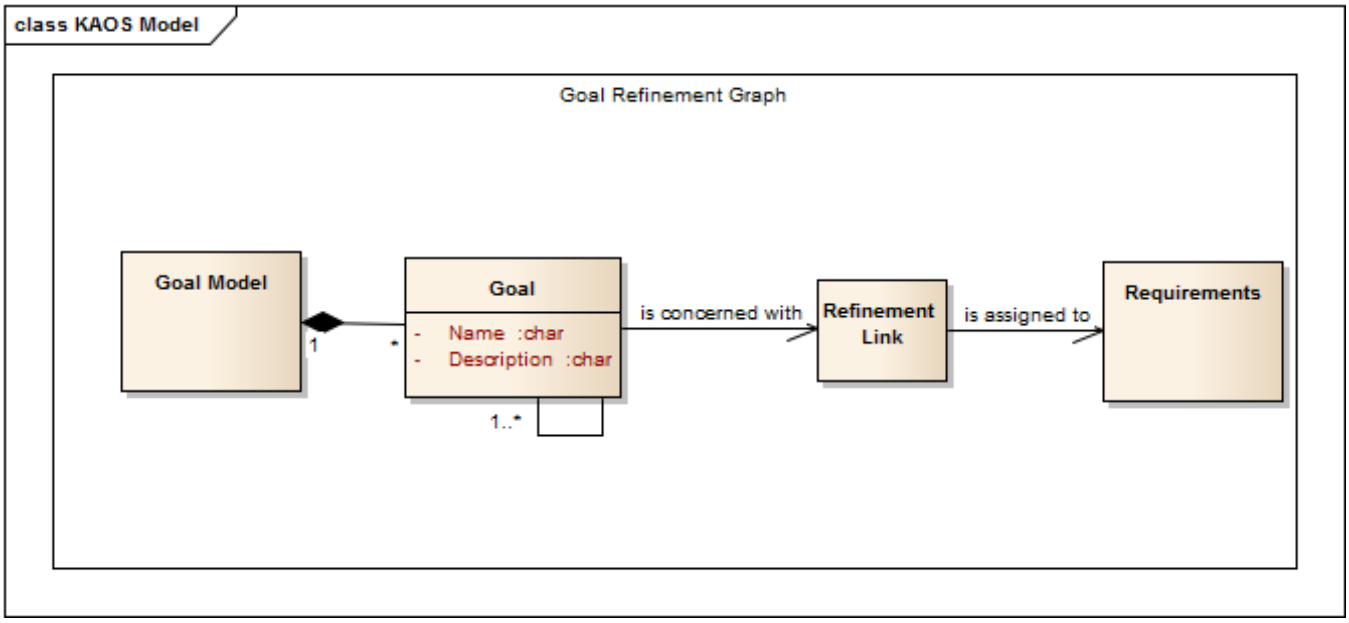


Figure 14 - KAOS Goal Model Support

Figure 14, presents a model to support the creation of Goal Models. A **Goal** is considered the most relevant concept, which can be specialized in **Parent Goals** or **Sub-Goals**. Sub-Goals refine Parent Goals with refinement links - **AND** refinement or **OR** refinement. A Goal can be also refined by a requirement; this refinement is the last step for the Goal Model creation with requirements being the child leaves.

In order to build a KAOS Goal model the next steps are to be followed:

- Identify goals;
- Refine each identified goal until achieving a set of requirements and/or expectations;

However in this project, we already have a set of requirements and the objective would be selecting and grouping these requirements by creating goals which can be achieved with that set. Therefore in order to build the Goal Model an exercise of analysis of a set of requirements must be the first step, then an attempt to group requirements that present a similar function or that can be thought has being part of the same purpose should be done. This refinement should be made in an upward movement until we reach a set of goals belonging to a major objective/function that a records system must have.

In order to test the model of figure 14, a goal model for each core service of the MoReq2010 specification was made. This exercise was carried after a detailed and careful analysis of each requirement of the core services; then a grouping of requirements having a similar objective was made, and the creation of a goal to join the found set. Next figures present the components used in the StarUML [28] program used to create the diagrams:

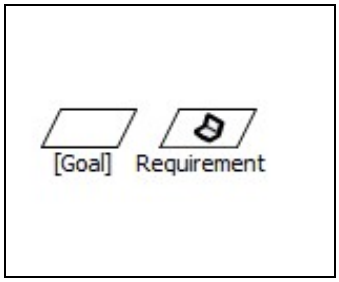


Figure 15 - Goal and Requirement components

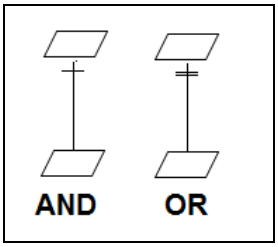


Figure 16 - Refinement Links

Also it was important to find a standard in how one should express the goals in the diagrams, i.e. the naming conventions to models creation. In [7] the words are preceded by a verb in its passive form, for example instead of writing Satisfy Borrower Request it is written as Borrower Request Satisfied, and also [8] uses this convention, stating that the reason to this is to avoid confusion between goals and operations (agent behaviors). In the next exercise this convention was followed being helpful in expressing a text in how to read each diagram.

4.2.3 KAOS Goal Models for MoReq2010 Core Services

User and Group Service

This service concerns with management of system Users and Groups in order to operations perform successfully. Despite not mandating protocols solutions should use for user authentication and user and group management, MoReq2010 presents a set of requirements that act as a wrapper to allow the use of either an external corporate directory service or a custom directory service built into the MCRS; these are simply basic concepts of a user and a group.

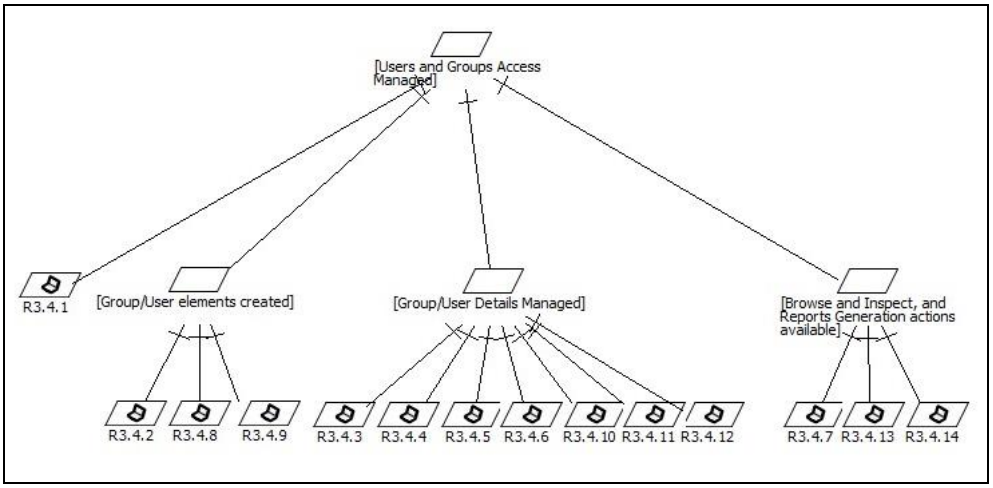


Figure 17 - User and Group Service Goal Model

Requirement R3.4.1 concerns the access to the MCRS, stating that this must be only made by users that are authenticated and for whom there exists an active user entity, describing also the metadata that should be available for that entity. **Group/User elements created** contains requirements concerning with creation of groups and users. **Group/User Details Managed** has requirements updating, deleting, and destroying users or groups, and adding or removing users from groups. **Browse and Inspect/Reports Generation actions available** contains requirements concerned with reports generation for either listing users or active groups, and browse and inspect users and groups. The diagram reads as follows:

In order for the system to manage Users and Groups of the system, only authenticated users must be authorized to access it, elements for Group/User must be created, details from these elements must be managed and actions for Browsing, Inspecting and Generate Reports must be available.

Model Role Service

MoReq2010 authors claimed that at the time that this specification was made, there was no industry standard concerning how users are authorized to perform functions in the MCRS; for that reason a model role service was created in order to cover that matter. A role is granted to a user in order to give it permission to perform specific functions; this user is then defined throughout the specification as an “authorized user”.

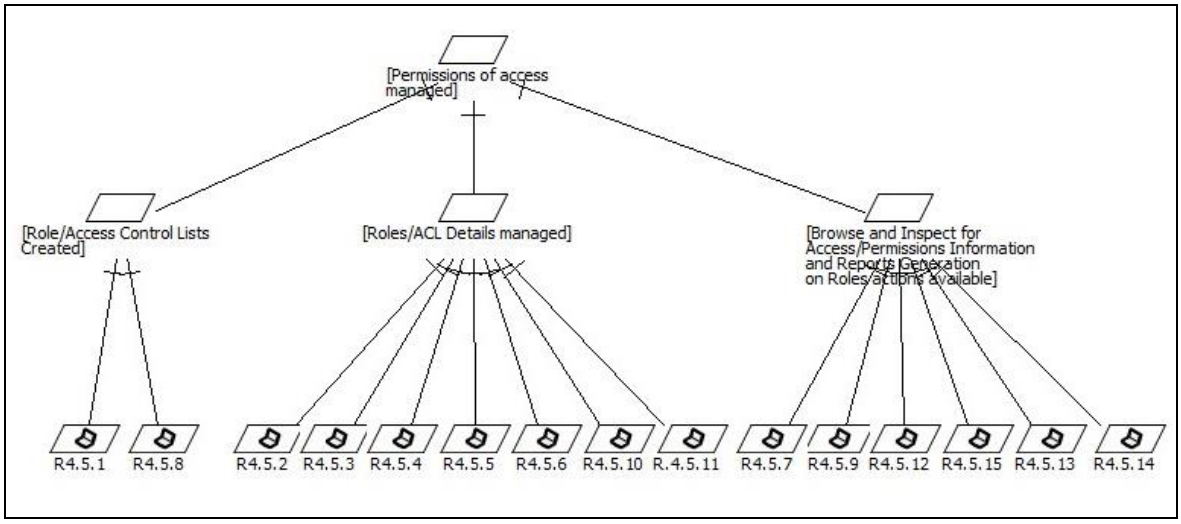


Figure 18 - Model Role Service Goal Model

Role/Access Control Lists Created contains requirements for both creations of a Role and ACL (short for Access Control List). **Roles/ACL Details Managed** has requirements concerning modifications of roles or ACL details, addition and removal of function definitions from active roles, deleting and destroying roles. **Browse and Inspect for Access/Permissions Information and Reports Generation on Roles actions available** contains requirements concerning the browsing and inspection of roles and function definitions and also an entity's ACL, discovery of authorized functions, and reports listing relationships between roles and function definitions. The diagram reads as follows:

In order for permissions of access to be managed, Role/Access Control Lists must be created, Roles/ACL details must be managed and actions to Browse and Inspect Access/Permission Information as well as Reports Generation must be available.

Record Service

Record Service is the service the standard presents to the management of records within the MCRS, which is done with the use of aggregations. An aggregation can either be a grouping of records or a grouping of aggregations, and these groupings are made according to a similarity found in these records characteristics. This service contains 21 functional requirements; Figure 14 presents a diagram with a possible organization of these requirements and their goals.

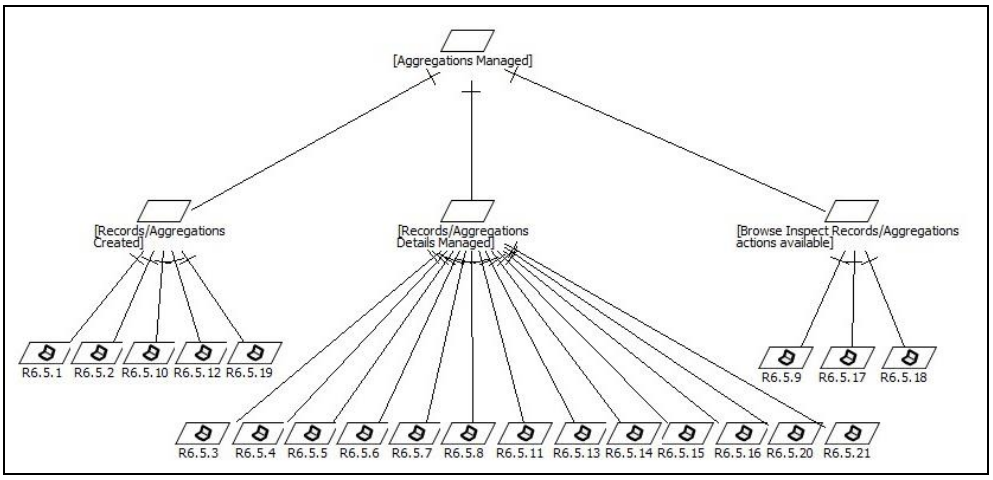


Figure 19 - Records Service Goal Model

The three second-level goals together summarize the functionality that a records service that a MCRS must implement. The goal **Records/Aggregations Created** contains requirements concerning all the aspects of active aggregations or active records creation, their type, and system metadata that must be included. **Records/Aggregations Details Managed** goal contains all requirements concerning opening and closing, moving, deleting, reclassifications, and details modification of both aggregations and records. **Browse and Inspect Records/Aggregations actions available** groups requirements concerned with browsing and inspecting and also searching for records and aggregations. The diagram reads as follows:

In order for aggregations to be managed, Records/Aggregations must be created, their details managed, and Browse and Inspect Records/Aggregations actions must be available.

Classification Service

This service concerns with the need of a record belonging to a class. Classes represent business functions, activities and transactions, and the association of a class with a record provides it with a definitive business context that continues to link the record with the business process that generated it.

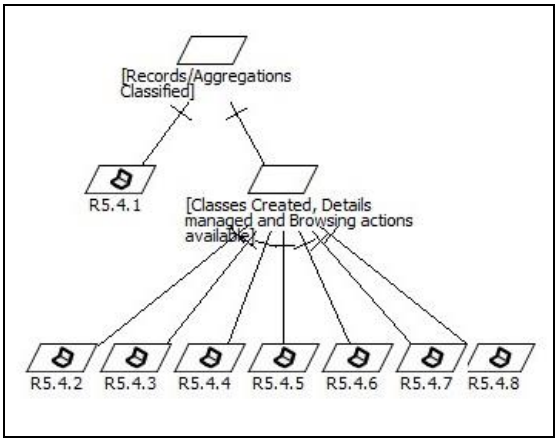


Figure 20 - Classification Service Goal Model

Requirement R5.4.1 states that a MCRS must incorporate a classification service, managing classes within a classification scheme in accordance with one of the modules in the MoReq2010 Classification series. **Classes Created, Details managed and Browsing actions available** goal covers classes creation requirements, details modifications, deleting, replacing, and destroying classes, as well as browsing classes and their associated entities requirements. The diagram is read as follows:

In order to Records/Aggregations to be classified, the system must possess a classification service, classes must be created, its details managed, and Browsing actions must be available.

Model Metadata Service

In order to achieve interoperability in future different MCRS solutions each entity and metadata belonging to it must be recognizable and able to be interpreted universally. This service allows for the specification of a universally understood metadata schema for use by all MCRS solutions, managing entity types and their related metadata element definitions.

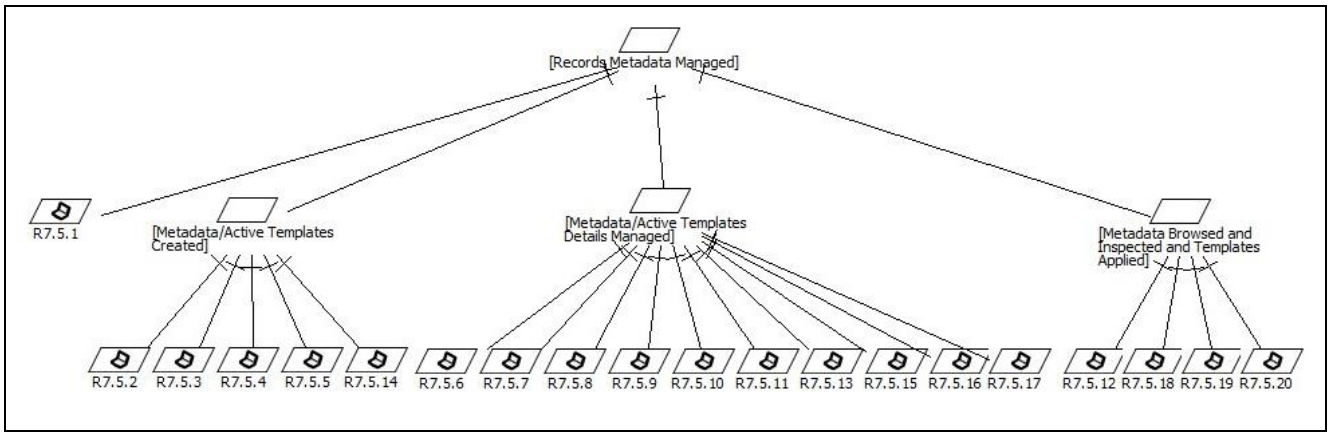


Figure 21 - Model Metadata Service Goal Model

Requirement R7.5.1 states the need of management of metadata elements definition. **Metadata/Active Templates Created** contains requirements concerning the creation of contextual metadata element definitions and active templates. Templates are metadata element definitions created by authorized users to meet local needs. **Metadata/Active Templates Details Managed** concerns with deleting, modifying, and destroying metadata or active templates. **Metadata Browsed and Inspected, and Templates Applied**, having requirements for browse of metadata element definitions and templates in the metadata service and inspection of metadata, and application of active templates to entities. The diagram reads as follows:

In order for Metadata to be managed, Metadata/Active Templates must be created and their details managed, Metadata must be Browsed and Inspected as well as Templates Applied.

Disposal Scheduling Service

A Record has a life cycle and for managing it MoReq2010 presents the Disposal Scheduling Service. Once a record has been created in an MCRS, it can never be deleted in full, as if it had never existed. This concept of accountability is important to good records management: although the complete record and its content no longer exist, there remains a residual record to show that it was once held by the MCRS. A record can be an active entity and transition to residual entity after a destruction event. Destruction is an irreversible process since parts of the entity are erased, preventing a return of this entity to an active state.

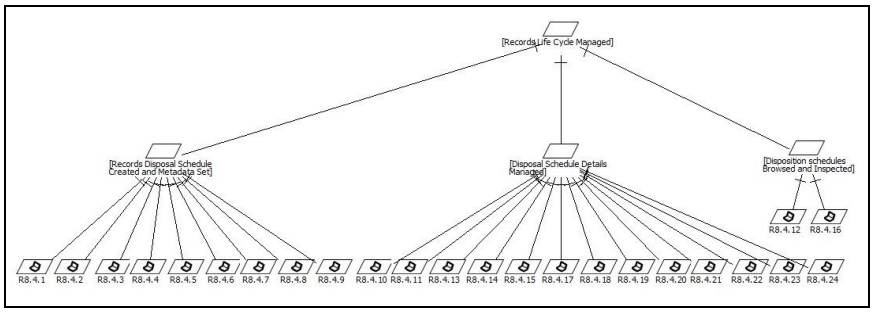


Figure 22 - Disposal Scheduling Goal Model

Records Disposal Schedule Created and Metadata Set contains requirements concerning the creation of new disposal schedules and associated metadata, and setting of values of disposal schedule's metadata elements. **Disposal Schedule Details Managed** has requirements concerning the modification of disposal schedules metadata, deleting, destroying, replacing, and updating disposal schedules. It also contains requirements for actions that must be made according to the disposal schedule metadata values. **Disposition Schedules Browsed and Inspected** has requirements concerning browsing across disposal schedules and active records due for disposal, and inspecting their metadata. The diagram reads as follows:

In order for a Records Life Cycle to be managed, a Disposal Schedule must be created and associated Metadata must be set and the Disposition Schedule details must be managed. Also the Disposition Schedule must be Browsed and Inspected.

Disposal Holding Service

This service is directly related to the Disposal Scheduling service. The later describes the impact of a disposal hold on the disposal scheduling. A disposal hold is a legal or other administrative order that interrupts the normal disposal process and prevents the destruction of some of an organization's records while a disposition hold is in place. It prevents the destruction of a record while the disposal hold is active; once it is destroyed the normal disposal process continues.

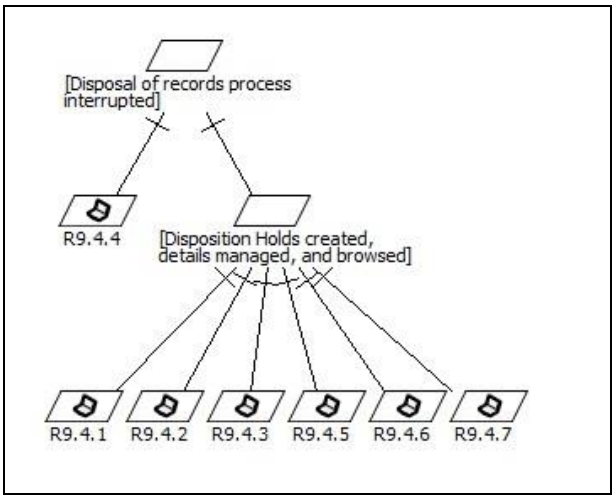


Figure 23 - Disposal Holding Service Goal Model

Requirement R9.4.4 concerns with the major objective of a disposal holding service which is prevent the destruction of records with associated disposition holds. **Disposition Holds created, details managed, and browsed** contains requirements concerned with the creation of disposition holds, association/disassociation of holds with records, deleting, lifting, and browsing a disposal hold and inspect its metadata. The diagram reads as follows:

In order for a Disposal of records process to be interrupted, records with associated disposition holds must not be destroyed, Disposition Holds must be created, its details managed, and must be Browsed.

Searching and Reporting Service

MoReq2010 requires that a record system must have a search engine for finding entities. In a MCRS users can discover entities with two methods: browsing from one entity to its related entities, or searching for entities that match a particular search query.

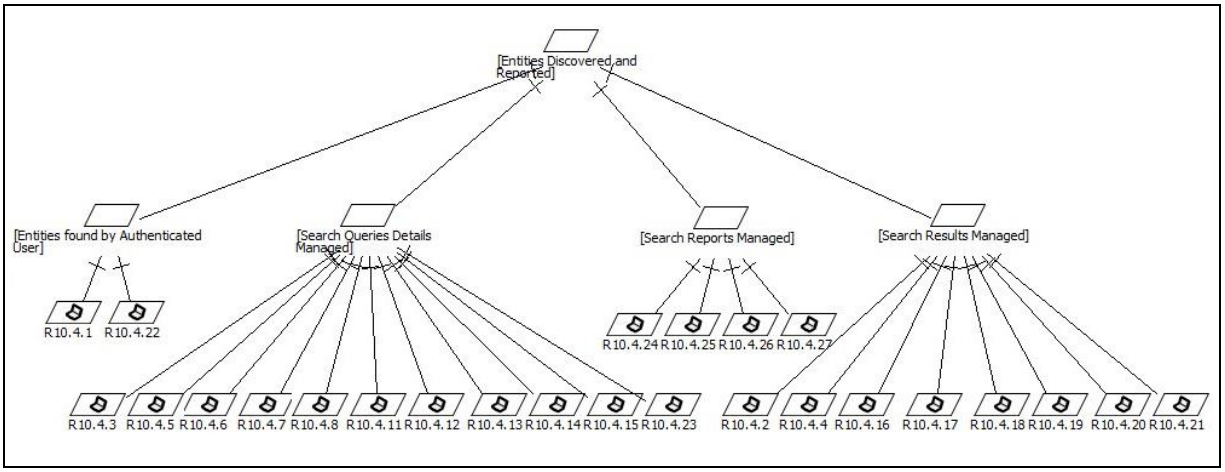


Figure 24- Searching and Reporting Service Goal Model

Entities found by Authenticated Users contains requirements regarding the access of entities in a search query (users must find any entities that they have been authorized to browse or inspect) and modifications on user's event history. **Search Queries Details Managed** has requirements concerning saving, modify, delete and share search queries, as well as search queries specification of criteria. **Search Results Managed** has requirements concerning search results returned, their appearance, entities that should be excluded from the results, among others. **Search Reports Managed** concerns with generation of detailed or summary reports based on search queries, and saving, deleting and sharing report definitions. The diagram reads as follows:

In order for Entities to be discovered and reported, they must be found by authenticated users, also, search queries details, search results and search reports must be managed.

Export Service

This service describes an operation to export MCRS entities. Entities should be described in sufficient detail in a common XML data format, belonging to the specification, so that their metadata values, event histories, access controls and content can be preserved and transferred to another MCRS. Exporting is important for Transferring entities for a different system, organization or archive, Migration where entities are moved from one MCRS to another within an organization, Secondary hosting where entities are regularly copied to one or more secondary read-only systems, and Replication in order to provide a copy for reference or safekeeping of contents of a MCRS.

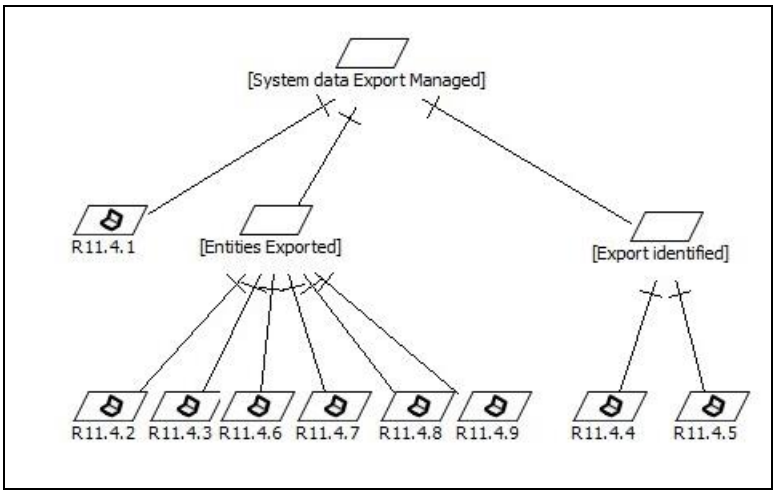


Figure 25 - Export Service Goal Model

Requirement R11.4.1 concerns with the permission of an authorized user to export entities from a MCRS. **Entities Exported** contains requirements concerning with the entities that can or cannot be exported, which entities are to be fully or partly exported, etc. **Export Identified** concerns with actions to identify an export with comments or unique identifiers. The diagram reads as follows:

In order for System data Export to be managed, the system must allow an authorized user to perform the export operation, entities must be exported and the export instance must be identified.

4.2.4 ISO 15489 and MoReq2010 alignment

After completing the exercise of creating goals to group the requirements inside MoReq2010 core services, one can attempt to align these results with ISO 15489 main goals to achieve a good records management system. As mentioned before in chapter 2 Related Work, this standard describes processes of records management in chapter **4.3 Records Management Processes**: Capture, Registration, Classification, Access and security classification, Identification of disposition status, Storage, Use and tracking, and Implementation of disposition. These processes are the core features for records management according to ISO 15489 with that in mind one can infer some relations between them and MoReq2010 core services features. These relations will be made taking the MoReq2010 core services as starting point for simplicity reasons.

User and Group Service

ISO 15489 mentions directly the need for users' management. Access and Security Classification, and Use and Tracking, is where the ISO standard mentions access and security according to groups or users. The groups mentioned in ISO 15489 are business units, while MoReq2010 is more specific and suggests a separated service for the management of users and groups, with specific requirements related to that function. The **access** as in the essence of the word is only described in MoReq2010 in the model role service.

Model Role Service

In this service users access granting, through the attribution of roles, is described directly linking to the core features present in ISO 15489 Access and Security Classification. Despite the access in the ISO being more legally focused, the descriptions are similar with both ISO and MoReq describing that records should only be accessible to users or groups that are granted access to do so.

Classification Service

Both ISO and MoReq put emphasis on this feature. This service answers to Classification and Access and Security Classification in ISO 15489. One can derive two parts of the concept of classification after analysis. There is a general classification where every record must be related to a business activity, either by a classification to each record or through an aggregation system. An aggregation has the purpose of further classifying records and grouping them onto their correct context. ISO 15489 mentions a hierarchical classification representing the business process at three levels, reflecting an analytical process. The first level reflects the business function, the second level is based on the activities constituting the function and the third level is further refinements of the activities or groups of transactions that take place within each activity. MoReq2010 goes further and not only writes about the importance of being able to edit the class of a record, as also recommends a hierarchical classification and puts importance on classifying aggregations as well.

Record Service

In this service, MoReq2010 addresses the use and functions of Aggregations. ISO 15489 does not formulate this matter as a core feature; however it mentions aggregations as well. The difference is that MoReq2010 is clear on the point that it should not be possible to have records at the same level as aggregations - aggregations should have an assigned class. This is not very clear in ISO 15489:2 (2001, p. 16) where it only says that "grouping them [the records], if applicable, into files to facilitate description, control, links and determination of disposition and access status".

Model Metadata Service

ISO 15489 mentions metadata in almost every core feature, making it hard to directly link this service to any of the features. Capturing records demands capturing also metadata associated with them, Registration demands minimum metadata to be present, use of the record may need to be captured to form part of the metadata, etc. In MoReq this is a separate service since the interoperability that is possible with metadata and the importance of having correct metadata.

Disposal Scheduling and Disposal Hold Services

Both services are connected to Identification of Disposition Status, Storage, and Implementation of Disposition of ISO 15849 core features. Both standards mention that it should be possible to retain a record for an unlimited time. The biggest difference is the emphasis that MoReq2010 puts on the fact that metadata about the record should always be saved, even though the actual document is destroyed. This is not mentioned in ISO 15489. To have the disposal process as an automated service is mentioned in both documents.

Searching and Reporting Service

This service answers to a small part of the Storage core feature of ISO 15489. In ISO 15489:2 (2001, p. 18) it says "Electronic records may be stored in a variety of ways that make their retrieval easier or faster". This is the only mention of searching for stored records in the standard. MoReq2010 goes further mentioning that it should be possible to search for records based on related entities, or have a separate search query, and that the search results should be presented in multiple ways chosen by the user, among other statements.

Export Service

ISO 15489 does not directly address any requirements related to the export of data from the system.

Summary

After analysis of the previous comparisons between MoReq2010 and ISO 15489, one can conclude that according to these two standards a records management system should be able to present six basic capabilities:

- Capture of Records;

- Organization of Records;
- Protection/Security of Records;
- Disposal and Retention of disposal of records;
- Access to Records;
- Export/Import of records.

These capabilities and their relation to the MoReq2010 core services can be depicted in a KAOS goal model, as seen in figure 27.

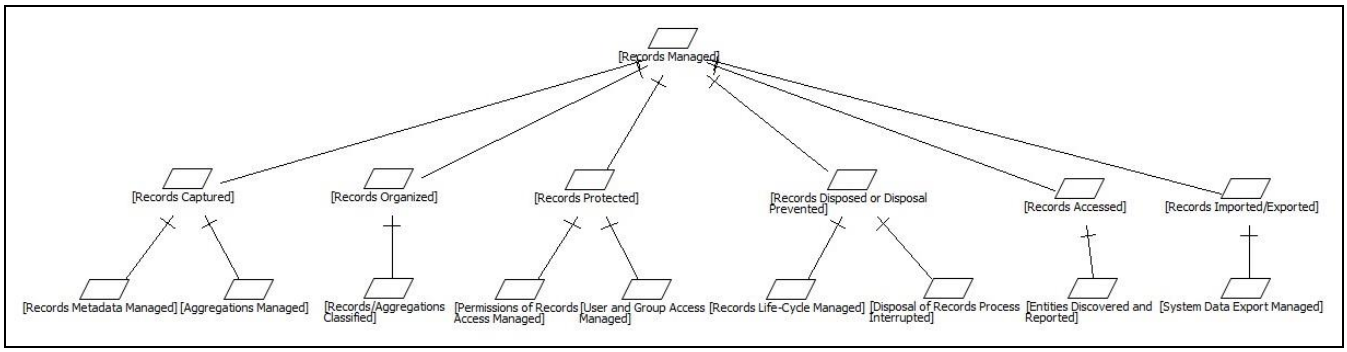


Figure 26 - Goal Model of Records Management functions

This diagram can be read as follows:

In order for Records to be Managed, Records must be Captured, Organized, Protected, Disposed, Accessed and Exported. For Capture, records metadata and records aggregations must be managed. For organization Records and Records Aggregations must be classified. For Protection, permissions of access to records and user and groups of users of the records system must be managed. For Disposal or Disposal Prevention, records life-cycle must be managed and their disposal of records process interrupted. For Access, entities must be discovered and reported. Finally for Exporting, system data export must be managed.

5 Results and Evaluation

This chapter presents results of implementation from parts of the proposed solution. Every detail concerning the construction of the system's structure was made, as well as the integration of MoReq2010 specification into the system. Next sections present the "Reqs" web application with the help of screenshots showing its usage, and also a brief consistency report showing MoReq2010 data saved in the repository.

5.1 "Reqs" Platform

The web application that resulted from the proposed solution is denominated "Reqs". The application comprises functionalities to navigate through the MoReq2010 specification in a different and interesting way. It has three main important pages: **Home**, **Overview** and **MoreqDemo**. The **Home** page is for now a blank page, since the major concern was to first focus on having behavior to visualize MoReq2010 data. Figure 27 shows a screenshot of the **Overview** page.

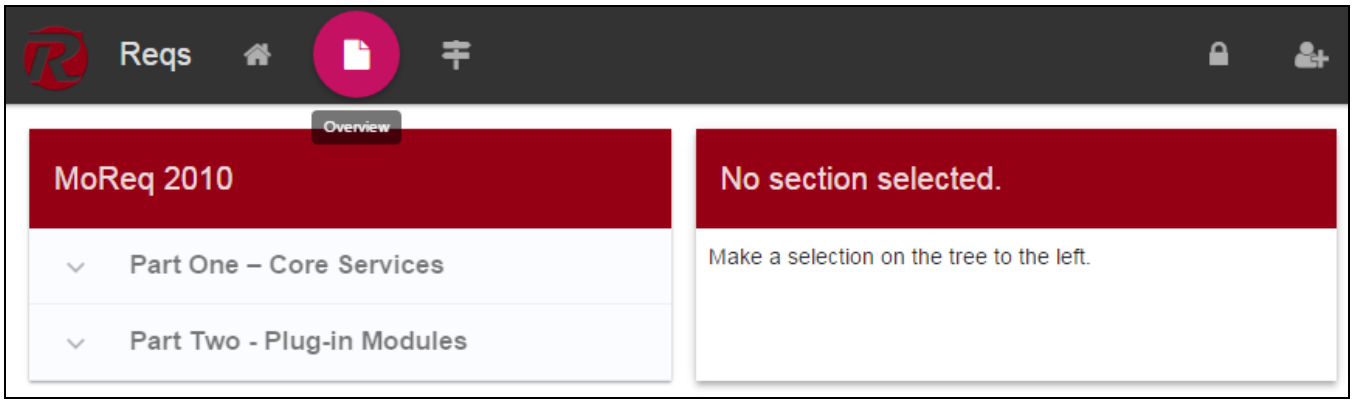


Figure 27 - Overview page for Reqs Application

The **Overview** page allows for a user to navigate through the document with the help of an index. The index appears on the left panel of the screen, listing all the chapters of the document. After choosing an option on the left panel, the right panel shows information of the document regarding that option. Figure 28 depicts an example of choosing an option on the left panel - "1.1.1 Intellectual Property Rights" – and the available information for that option appearing on the right panel.

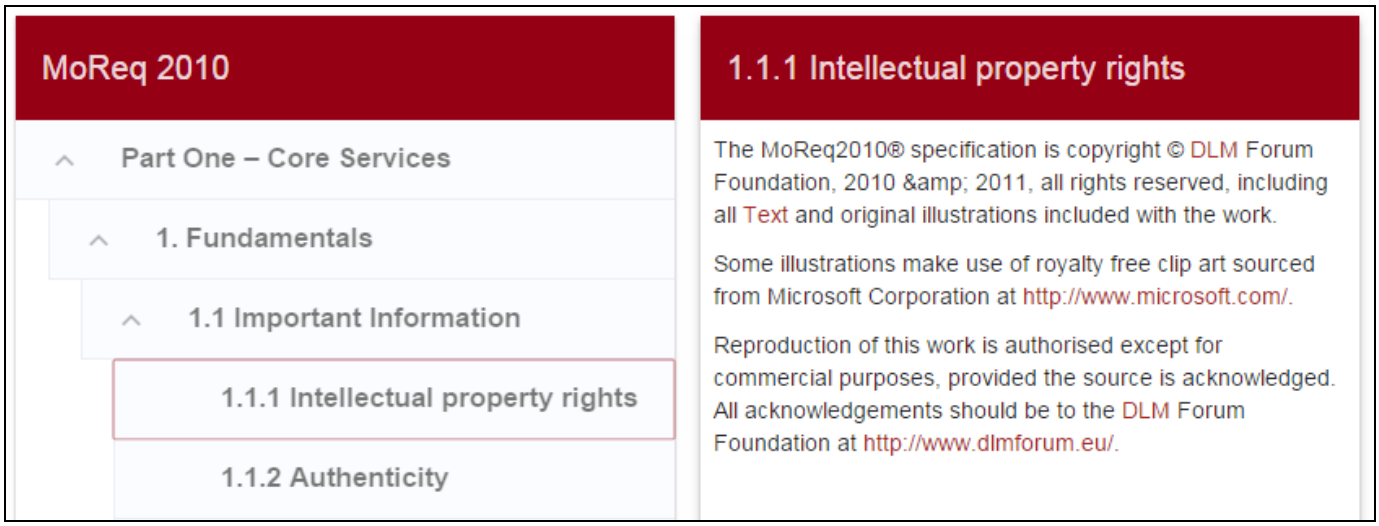


Figure 28 – Example of Overview Page Navigation

The **MoreqDemo** page has options to list some of the documents most important components: functional requirements, non-functional requirements, and information model. The **Intro** tab shows some brief information about how to navigate through **MoreqDemo**.

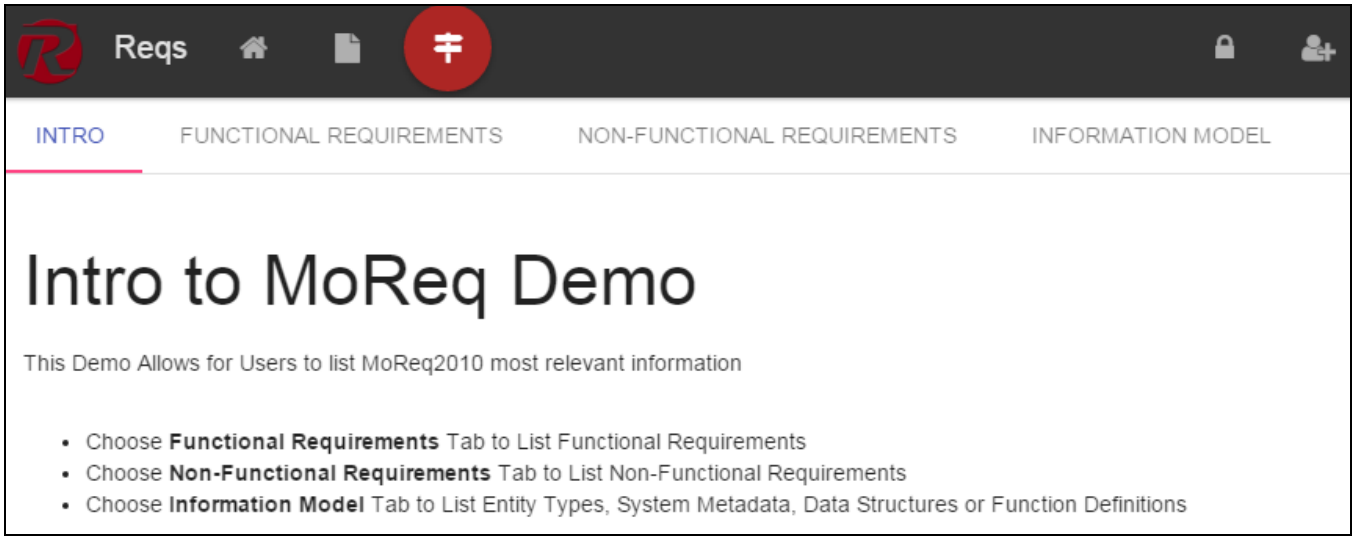


Figure 29 - Intro to MoReqDemo Page

The **Functional Requirements** and the **Non-Functional Requirements** tabs are similar, differing only in the displayed data. Information is presented on these pages in a similar way as the overview page with the use of two panels. The left panel lists all MoReq2010 requirements and by clicking in one of them the right panel shows its description. Figure 30 has an example of choosing the non-functional requirement N12.3.01 for observation.

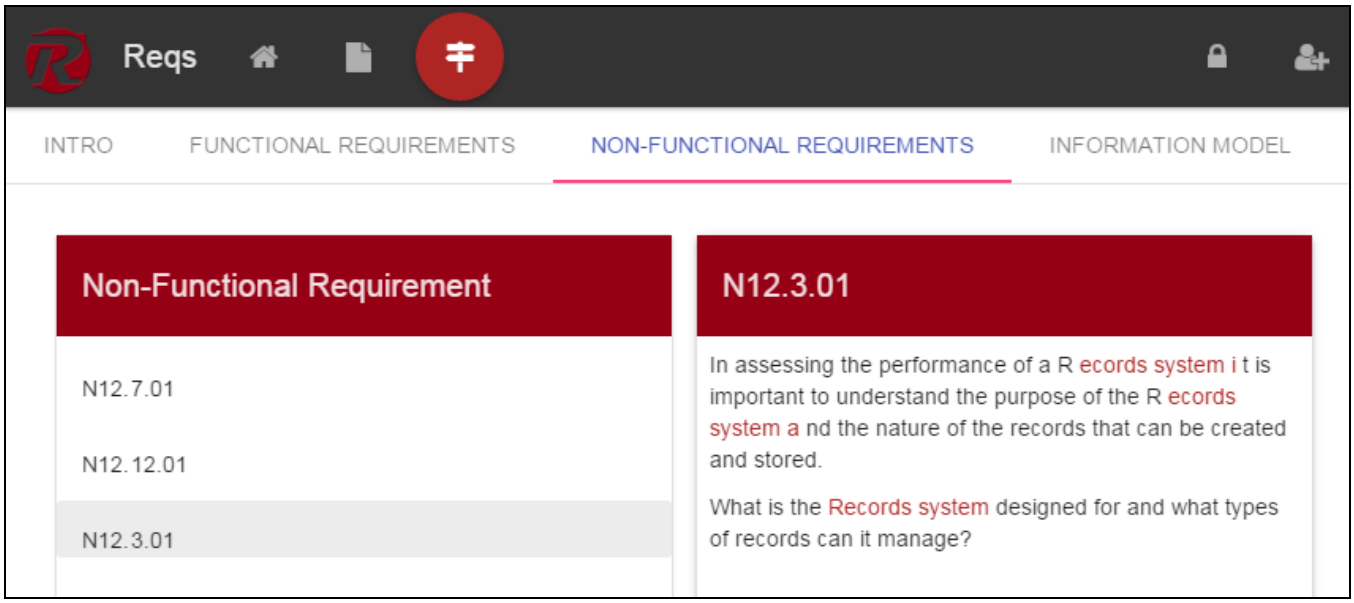


Figure 30 - Example of Non-Functional Requirement of MoReq Demo

The **Information Model** tab also makes use of the two panel visualization. However, instead of having a tab for each component to be listed, the user has buttons available to choose from. There are four buttons as presented in figure 31: **Entity Types, System Metadata Element Definitions, Data Structures,** and **Function Definitions**.

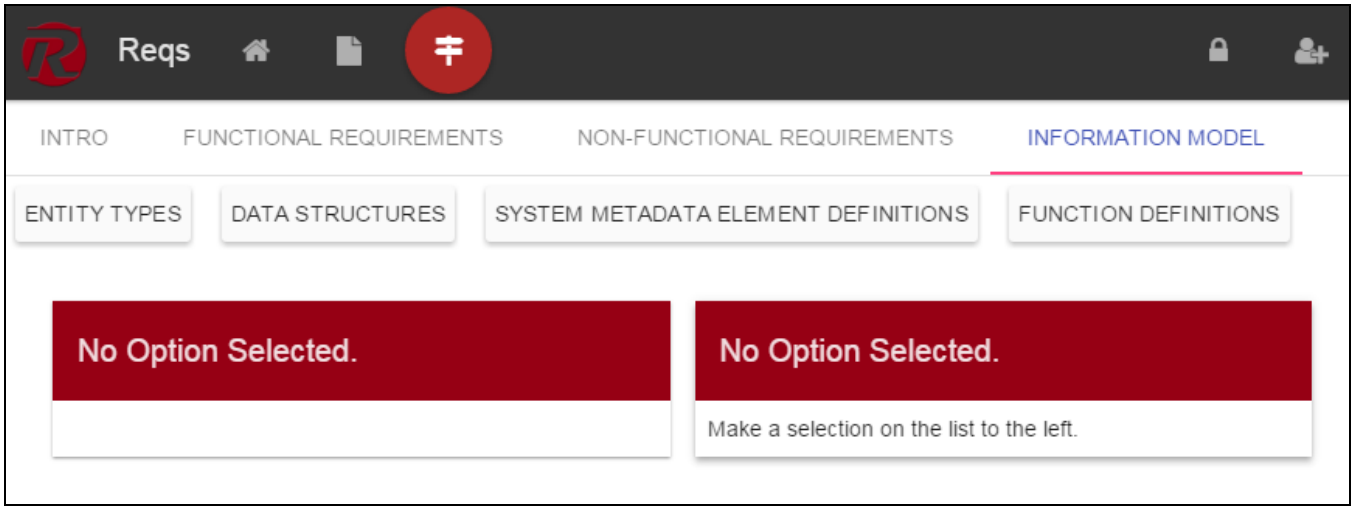


Figure 31 - Information Model tab page

After clicking one of the buttons, left panel lists the elements from that option. Right panel shows tables with information regarding one of the listed elements on the left panel. In figure 32 the button for System Metadata has been clicked, and then one of the metadata definitions “M14.4.103 Template Service Identifier” is chosen for observation.

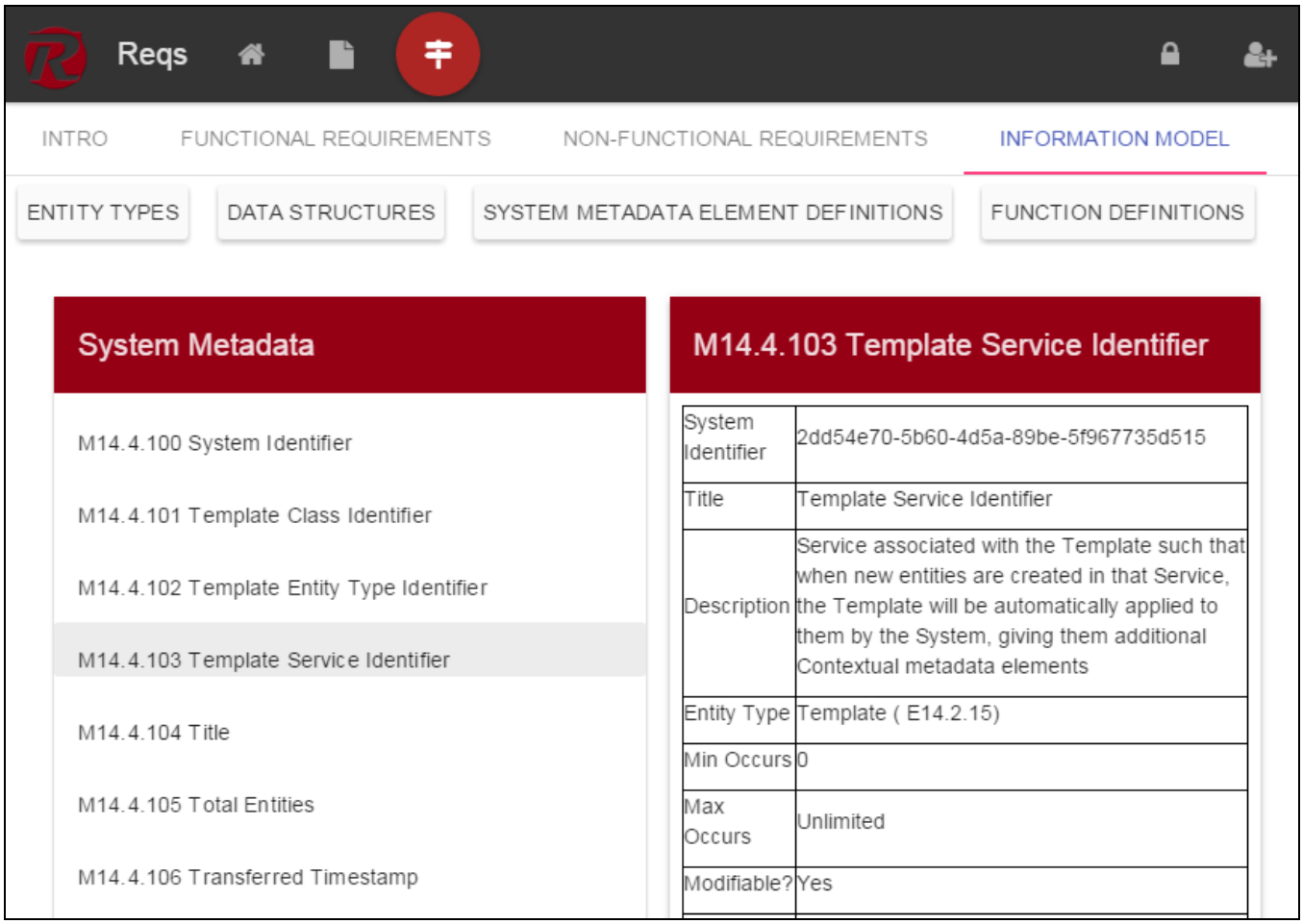


Figure 32 - Example of System Metadata Element Definition observation

Next section will give an insight on the consistency of the MoReq2010 data being presented in these screenshots.

5.2 MoReq2010 in the System

Since one of the outputs of this project is the integration of MoReq2010 on the system, it's important to ensure that not only the data is stored in the database but also that the information is consistent with the original document. One of the chosen ways to prove consistency of a database storing a big size document as the type of MoReq2010 is to count occurrences of some of its components. First, one can count the number of requirements inside each chapter of MoReq2010 and see if the database matches the count. The results are in table 14 for functional requirements and table 15 for non-functional.

Chapter	Original PDF	Database
Part One – Core Services		
2.System Services	28	28
3.User and Group Service	14	14
4.Model Role Service	15	15
5.Classification Service	8	8
6.Record Service	21	21
7.Model Metadata Service	20	20
8.Disposal Scheduling Service	24	24
9.Disposal Holding Service	7	7
10.Searching and Reporting Service	27	27
11.Exporting Service	10	10
Part Two – Plug-in Modules		
101.Graphical User Interface	22	22
102.Application Programming Interface	6	6
201.Hierarchical Classification	16	16
301.Electronical Components	9	9

Table 14 - Total Count of Functional-Requirements

Chapter	Original PDF	Database
Part One – Core Services		
12.3 Non-functional Requirements for Performance	7	7
12.4 Non-functional Requirements for Scalability	7	7
12.5 Non-functional Requirements for Manageability	8	8
12.6 Non-functional Requirements for Portability	5	5
12.7 Non-functional Requirements for Security	6	6
12.8 Non-functional Requirements for Privacy	2	2

12.9 Non-functional Requirements for Usability	2	2
12.10 Non-functional Requirements for Accessibility	1	1
12.11 Non-functional Requirements for Avaliability	6	6
12.12 Non-functional Requirements for Reliability	6	6
12.13 Non-functional Requirements for Recoverability	6	6
12.14 Non-functional Requirements for Maintainability	6	6
12.15 Non-functional Requirements for Support	7	7
12.16 Non-functional Requirements for Warranty	6	6
12.17 Non-functional Requirements for Compliance	5	5
Part Two – Plug-in Modules		
101.Graphical User Interface	11	11
102.Application Programming Interface	10	10
201.Hierarchical Classification	1	1
301.Electronical Components	15	15

Table 15 - Total Count of Non-Functional Requirements

Table 16 shows the total count of elements inside Information Model chapter for each Part of the document. For Part – Two, MoReq2010 original PDF does not present elements from the Information Model in designated chapters; there are three Function Definitions, one System Metadata Element Definition and one Entity Type, in a total of five elements.

Chapter	Original PDF	Database
Part One – Core Services		
14.2 Entity Types	16	16
14.3 Data Structures	3	3
14.4 System Metadata Element Definitions	107	107
14.5 Functions Definitions	196	196
Part Two – Plug-in Modules		
301.7 Information Model	5	5

Table 16 – Total Count of Information Model Elements

Finally, one can also count the number of Images and Tables. In this case the counting was made considering the two main chapters. Table 17 presents the results.

Component	Original PDF	Database
Part One – Core Services		
Images	59	59
Tables	10	10
Part Two – Plug-in Modules		
Images	8	8
Tables	4	4

Table 17 - Total Count of Images and Tables

Another way to show consistency is by comparing some of the information content present in the database and the original content of the document PDF. This will be made by randomly selecting some simple contents, screenshot their appearance on the application and comparing it to screenshots from the original PDF.

First content example shows textual information from a sub-chapter from 1.Fundamentals. Figure 33 is a screenshot from MoReq2010 original PDF and figure 34 is a screenshot from the Reqs application showing the same sub-chapter and its paragraphs.

1.1 IMPORTANT INFORMATION

1.1.1 Intellectual property rights

The MoReq2010® specification is copyright © DLM Forum Foundation, 2010 & 2011, all rights reserved, including all text and original illustrations included with the work.

Some illustrations make use of royalty-free clip art sourced from Microsoft Corporation (http://www.microsoft.com/).

Reproduction of this work is authorised, except for commercial purposes, provided the source is acknowledged. All acknowledgements should be to the DLM Forum Foundation (http://www.dlmforum.eu/).

Figure 33 - "1.1.1 Intellectual Property Rights" Sub-Chapter in original PDF

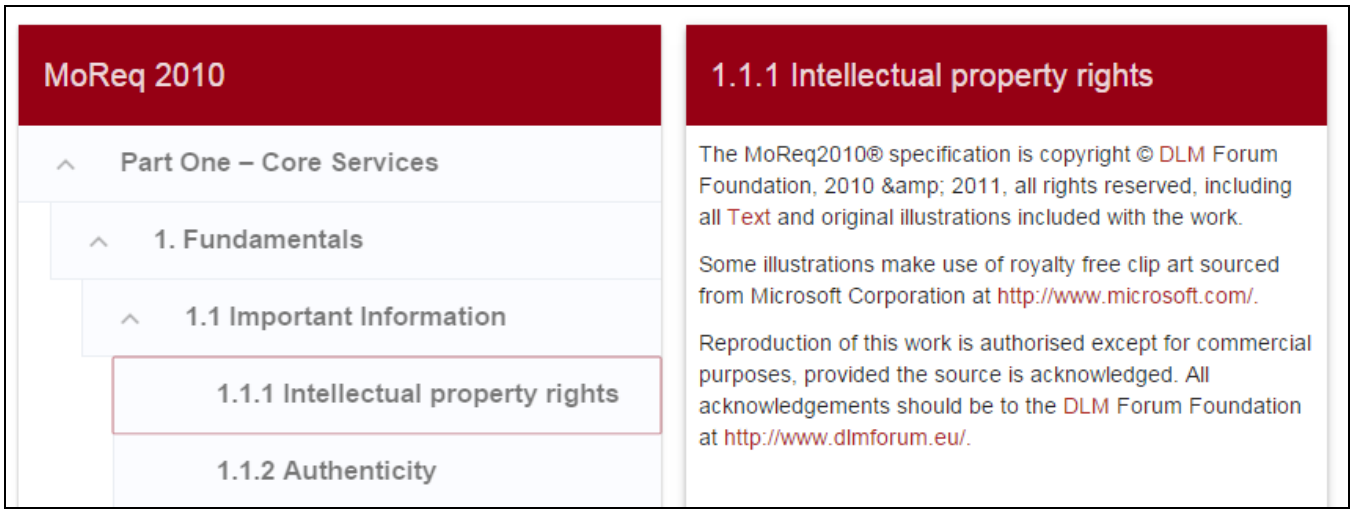


Figure 34 - "1.1.1 Intellectual Property Rights" Sub-Chapter in Reqs application

One can also verify the consistency of the requirements content. Figure 35 and 36 show the information for a requirement from chapter 10.Searching and Reporting Service. Figure 37 and 38 are for a non-functional requirement from the set of requirements for performance in chapter 12.3 Non-Functional Requirements for Performance.

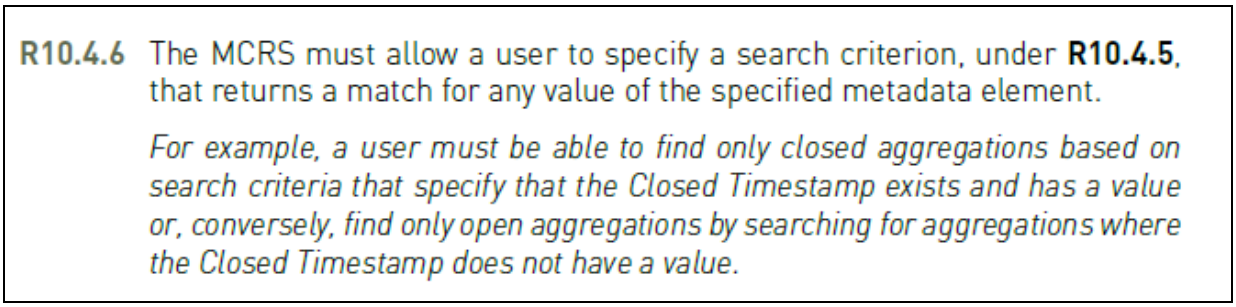


Figure 35 - Functional Requirement in original PDF

The text appearing red in the Reqs Application is for Links to glossary entries, in the case of highlighted names, and to other requirements, function definitions, etc, for others. For example, in figure 38, the highlighted name “Records System” is link for a glossary entry for Records System; the highlighted requirement titles are links to these requirements.

Figure 39 and 40, and last screenshot addresses Part Two – Plug-in Modules, and also tables and their content, by showing the chapter 301.1 Module Information.

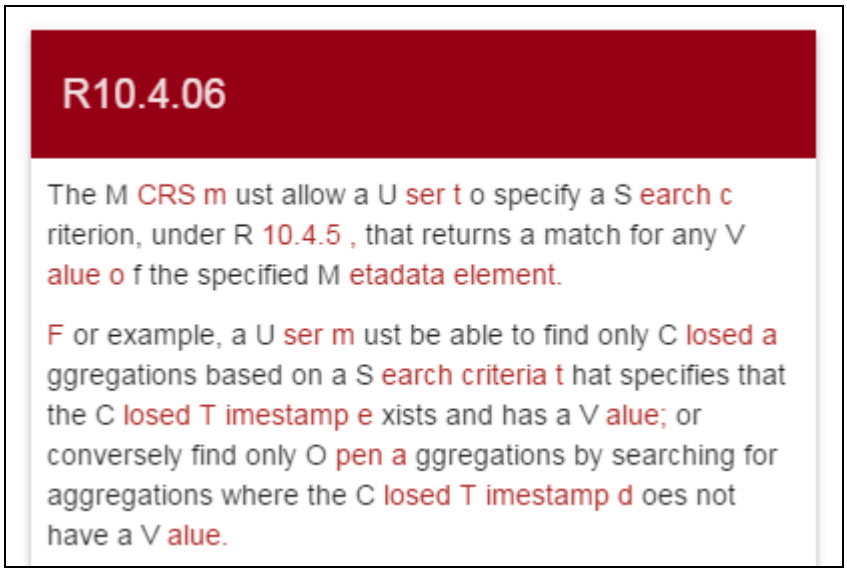


Figure 36 - Functional Requirement in Reqs application

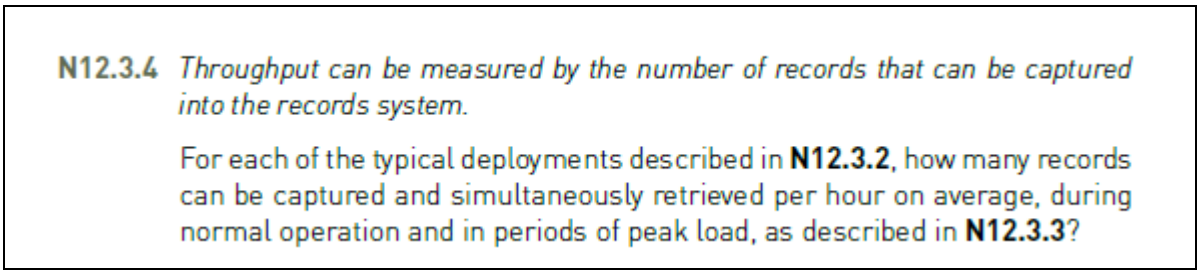


Figure 37 - Non-Functional Requirement in original PDF

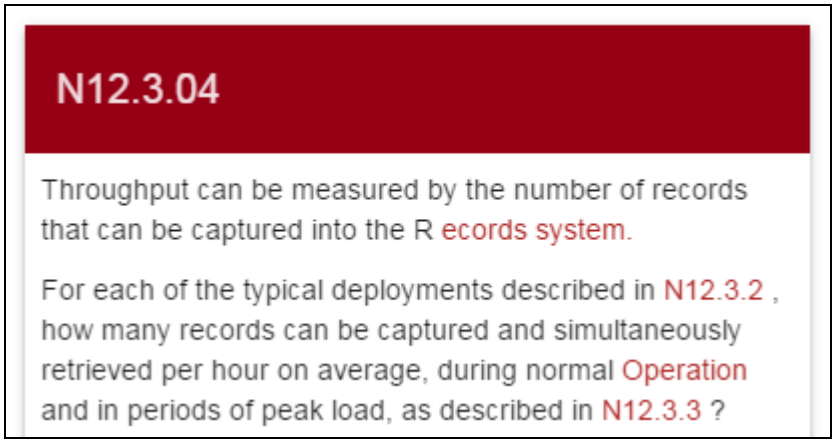
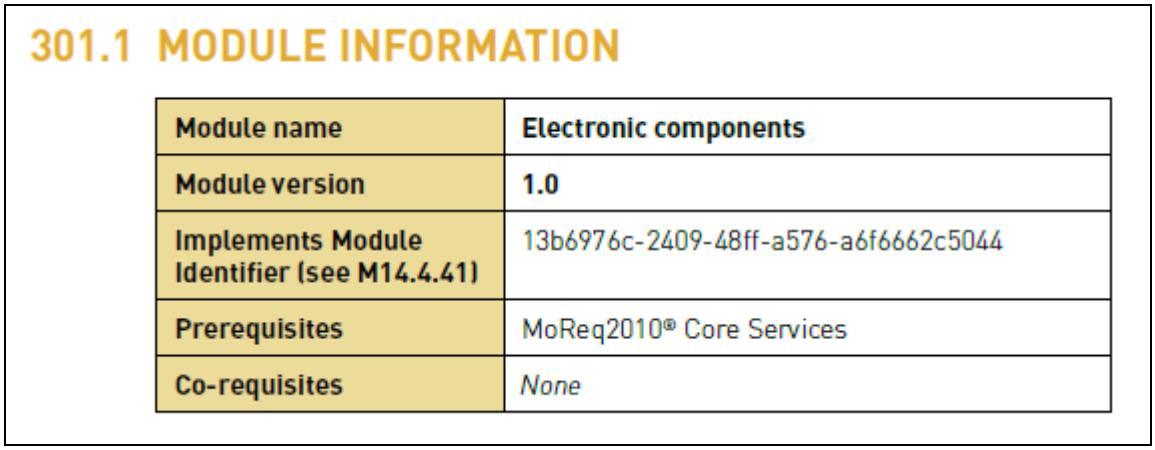


Figure 38 - Non-Functional Requirement in Reqs application



301.1 MODULE INFORMATION

Module name	**Electronic components**
Module version	**1.0**
Implements Module Identifier (see M14.4.41)	13b6976c-2409-48ff-a576-a6f6662c5044
Prerequisites	MoReq2010® Core Services
Co-requisites	*None*

Figure 39 – Part Two "301.1 Module Information" chapter Table Example in original PDF

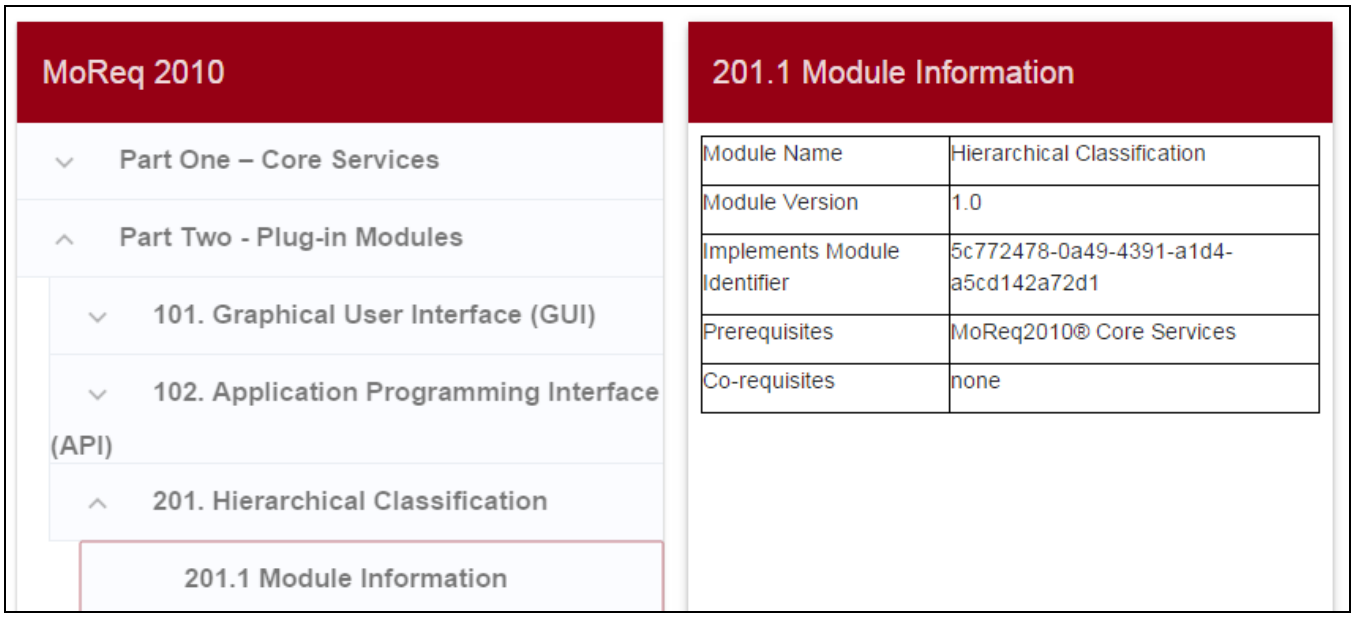


Figure 40 - "301.1 Module Information" chapter Table Example in Reqs application

5.2.1 Data Problems

MoReq2010 specification on its original PDF format already had some problems, more precisely in the chapters' numeration. When treating the information to be saved in the database it was decided to leave these problems as they were, and later add to the application functionality to correct them. This will ease the tracking of eventual changes to the original official document. These problems can be described as follows:

- Chapter 2.System Services has sub-chapters 2.1, 2.2 and 2.4. There is no sub-chapter 2.3. The same happens with every other chapter, except 1.Fundamentals, 14.Information Model, and the chapters mentioned in the next bullet.

- Chapter 6.Record Service has sub-chapters 6.1, 6.2, 6.3 and 6.5. There is no sub-chapter 6.4. Same situation found in chapter 7.Model Metadata Service.
- Chapter 14.Information Model doesn’t present sub-chapter 14.5 Function Definitions in its bookmark index, however the chapter exists and can be viewed by navigating through the pages.
- Chapter 12.3 title has a typo in the word Functional (it appears as Fonctional), in the index, however the title on the document is correct.

6 Conclusion

6.1 Critical Analysis of the Work

The development of this work had several phases. The first phase was related to the definition of the problem and the objective for the whole project. The second and most important phase was the proposal of a system capable of store reference requirements documents for records management, not only structurally but also on its expected behavior. Then came the phase of building the system, starting from the repository were the documents were to be stored; the creation of a parser to read documents in a XML format and store them in the repository and lastly an user interface for data visualization and manipulation. Also, there was a phase were KAOS power was explored, by creating models for MoReq2010 core services to abstract higher goals from the vast set of requirements in the specification.

In this project only KAOS Goal Models where explored, however one could also explore the Object model and the Operation model. MoReq2010 has sufficient detail to objects identification as well as to operations that should be made on these objects. This exploration was not carried away, not only because of time constraints, but also for practical and convenience reasons since it was best to first attempt a simple iteration for testing the concepts before considering anything more complex. When it comes to Responsibility or Agent Model, MoReq2010 very general and generic concepts, which are thought as to fit many different records system as possible, hampers the task of finding specific stakeholders. One can think of a few that could be available in any case: the developer of the system, the user, the buyer, among others.

Several use cases were proposed for the “Reqs” web application (presented in the analysis chapter), however, only those related to the visualization of MoReq2010 information were possible to implement in time to show some results of it, in the form of screenshots of the application, in this thesis. All the functionality regarding the creation of KAOS models and requirements association was not implemented.

6.2 Future Work

The presented system is still in a very initial stage and this work was made to validate that its purpose can be achieved, by exemplifying with MoReq2010 specification. Future steps should be made into adding more reference requirements documents into the system to enrich its repository, as well as the implementation of the rest of the proposed use cases, and possibly test the system functionality with concerned stakeholders.

It would be interesting if the creation of KAOS models could be done in a dynamic way, for example on how it is done in any program built to create diagrams (use case diagrams, UML diagrams, among others). Objects from KAOS models would be available and the user could drag and drop them and create links between them.

One aspect that may also be considered is the extraction of requirements from the system. ReqIF, short of Requirements Interchange Format, is an Organization Management Group standard that allows for the exchanging of requirements data between organizations that do not have a possibility to share the same

repository. The exchanging information is stored in XML documents that comply to the ReqIF format, making them processable by different tools on different organizations, allowing for more interoperability. The requirements extraction should be done in this format to ensure that exchanging requirements from this system between a wide variety of tool implementations is tolerated.

7 References

[1] Zave, P. Classification of research efforts in requirements engineering. ACM Comput. Surv., 1997.

[2] Bashar Nuseibeh and Steve Easterbrook. 2000. Requirements engineering: a roadmap. In Proceedings of the Conference on The Future of Software Engineering (ICSE '00). ACM, New York, NY, USA, 35-46. DOI=10.1145/336512.336523 http://doi.acm.org/10.1145/336512.336523.

[3] Vieira, R., Borbinha, J., MoReq2010 – Uma Apresentação, IST/ INESC-ID.

[4] Vieira, Ricardo. Requirements Engineering Approaches, Research Topics – PhD Program in Information Systems and Computer Engineering, IST - Technical University of Lisbon, Portugal.

[5] Vieira, R., Ferreira, D., Antunes, G., Borbinha, J., Aligning Multiple Requirements Specifications using Goal Models. Information Systems Group, INESC-ID/IST.

[6] Lapouchnian, A. Goal-Oriented Requirements Engineering: An Overview of the Current Research. Department of Computer Science – University of Toronto, 2005.

[7] Dardenne, A., Lamsweerde, A., Fickas, S., Goal-directed requirements acquisitions. Science of Computer Programming 20, (1993) 3-30

[8] Respect-IT, KAOS Tutorial, V1.0, Oct 18, 2007

[9] International Organization for Standardization, ISO 15489: Information and documentation – Records Management – Part 1: General., 2001

[10] International Organization for Standardization, Business Plan 2010 – ISO/TC 46 – Information and Documentation. Vers. N2258, 2010

[11] International Organization for Standardization, ISO 15489: Information and documentation – Principles and functional requirements for records in electronic office environments – Part 2: Guidelines., 2001

[12] Lindqvist, M., Keeping or Discarding Records. A Comparison and a Practical Use of Influential Standards for Electronic Records Management – Bachelor's Thesis. Linkopings University, 2012. ISRN: LIU-IDA/KOGVET-G—12/2009—SE

[13] International Organization for Standardization, ISO 30300: Information and documentation - Management system for records – Fundamentals and vocabulary., 2010

[14] International Organization for Standardization, ISO 30301: Information and documentation - Management system for records - Requirements., 2010

[15] International Organization for Standardization, ISO 16175: Information and documentation – Records Management – Part 1: Overview and statement of principles., 2010

[16] Department of Defense of the United States of America, DoD 5015.02-STD: Electronic Records Management Software Applications Design Criteria Standard., 2007.

[17] DLM Forum Foundation, MoReq2010 - Modular Requirements for Records Systems: Core Services & Plug-in Modules., 2010 & 2011.

[18] Institute of Electrical and Electronics Engineers, 1998. IEEE Recommended Practice for Software Requirements Specifications, New York

[19] Institute of Electrical and Electronics Engineers, 1998. IEEE Guide for Developing System Requirements Specifications, New York

[20] Vieira, R., Ferreira, D., Borbinha, J., Gaspar, G., "A Requirements Engineering Analysis of MoReq," 2012.

[21] Gaspar, J., Análise de Problemas em Especificações de Requisitos de Referência. Instituto Superior Técnico, Maio, 2012

[22] A. van Lamsweerde. Goal-Oriented requirements engineering – a guided tour. In IEEE Computer Society Press, editor, Proceedings of the 5th IEEE International Symposium on Requirements Engineering (RE'01), pages 249-263, Los Alamitos, 2001.

[23] Jennifer Horkoff and Eric Yu. 2011. Analyzing goal models: different approaches and how to choose among them. In Proceedings of the 2011 ACM Symposium on Applied Computing (SAC '11). ACM, New York, NY, USA, 675-682. DOI=10.1145/1982185.1982334 http://doi.acm.org/10.1145/1982185.1982334.

[24] Zave, P & Jackson, M. Four dark corners of requirements engineering. ACM Trans. Softw. Eng. Methodol., 6(1):1–30, 1997.

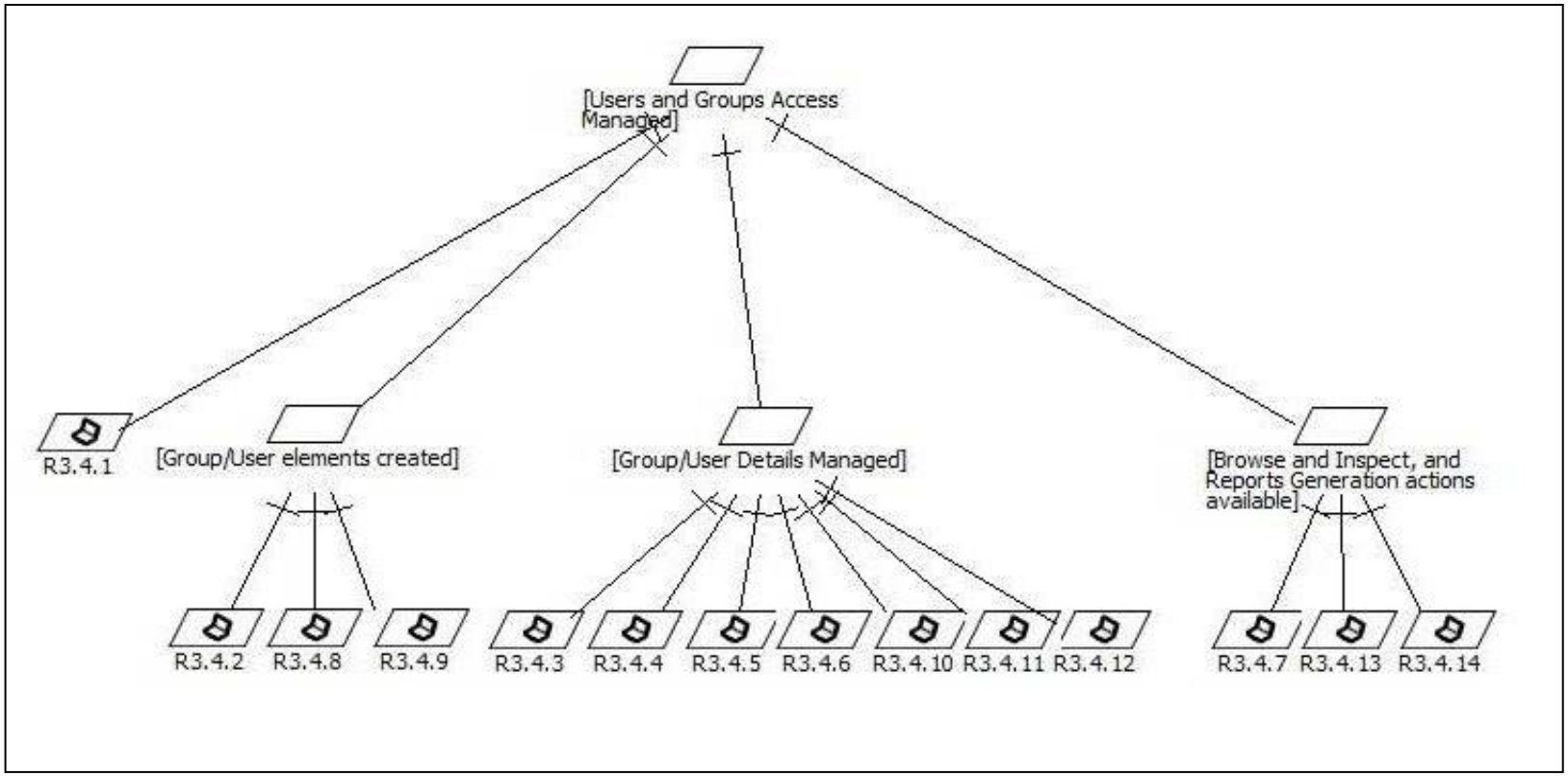
[26] International Organization for Standardization, Business Plan 2010 – ISO/TC 46 – Information and Documentation. Vers. N2258, 2010

[27] J. Mylopoulos, L. Chung, B. Nixon. Representing and Using Non-Functional Requirements: A Process-Oriented Approach. IEEE Transactions on Software Engineering, 18(6), June 1992.

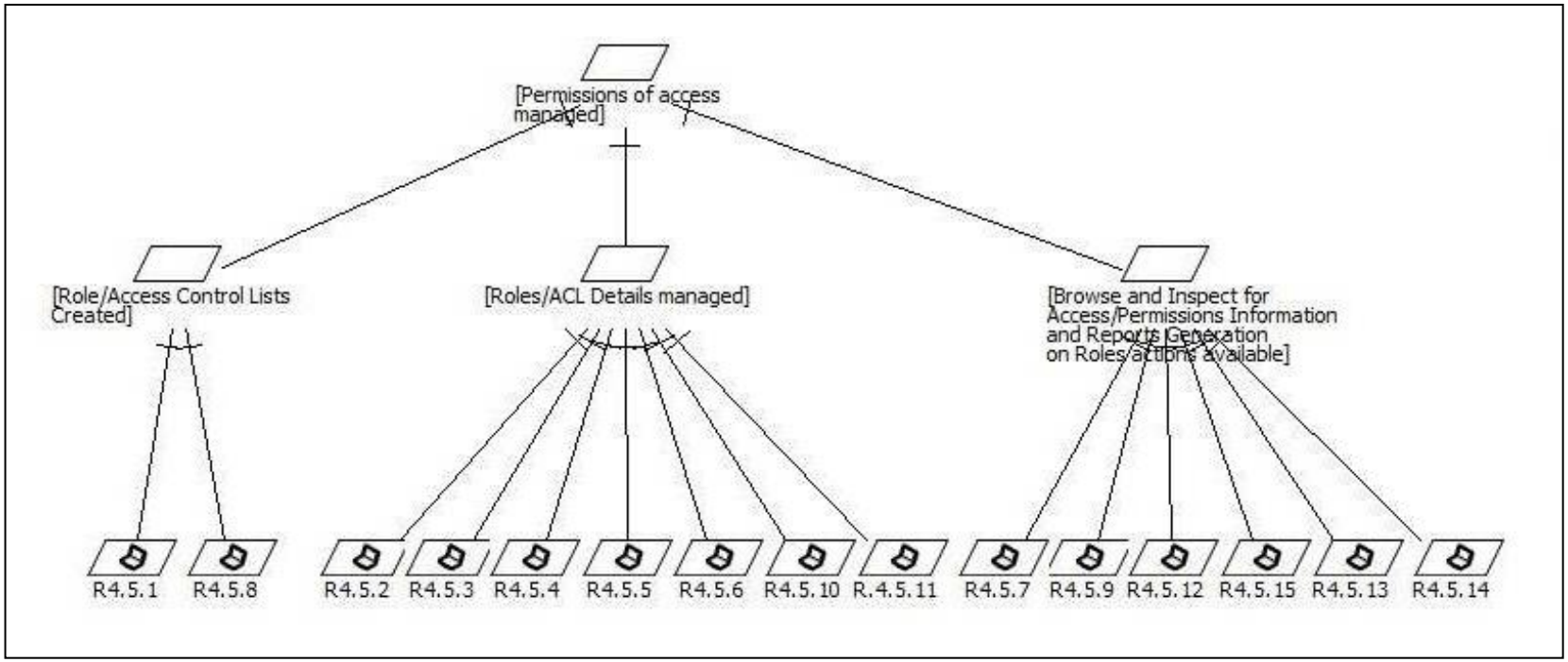
[28] Sam Supakkul. 2008. RE-Tools: A Multi-Notational Requirements Modelling Toolkit. [ONLINE] Available at: http://www.utdallas.edu/~supakkul/tools/RE-Tools/index.html. [Accessed 08 October 15].

8 Appendices

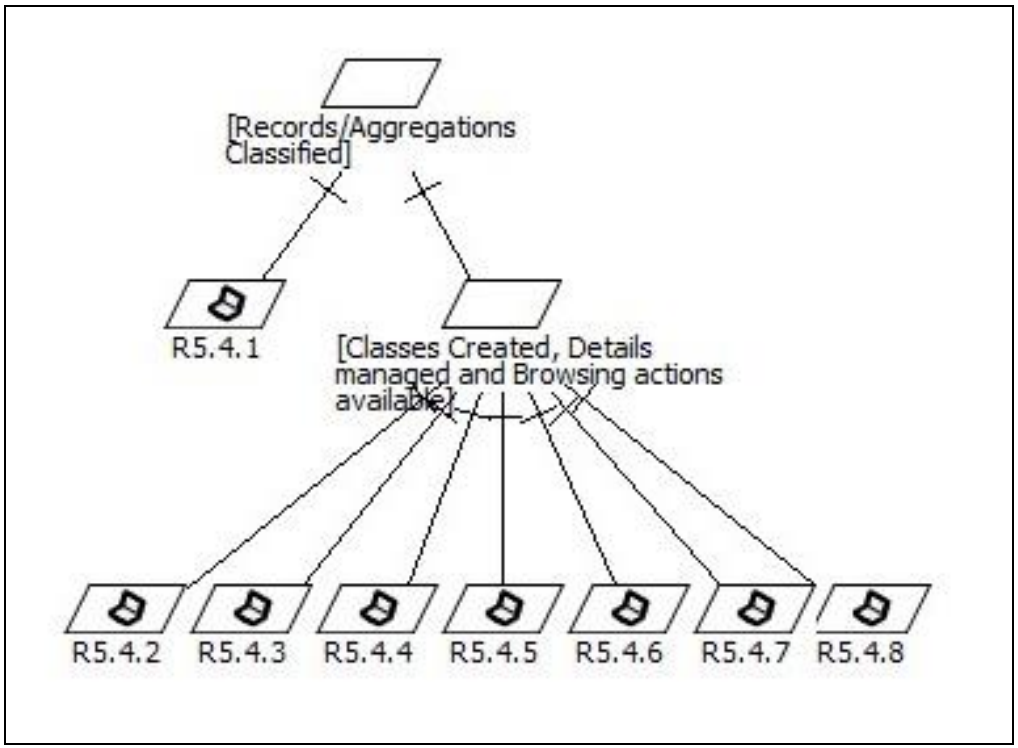
8.1 Appendix I: User and Group Service Goal Model



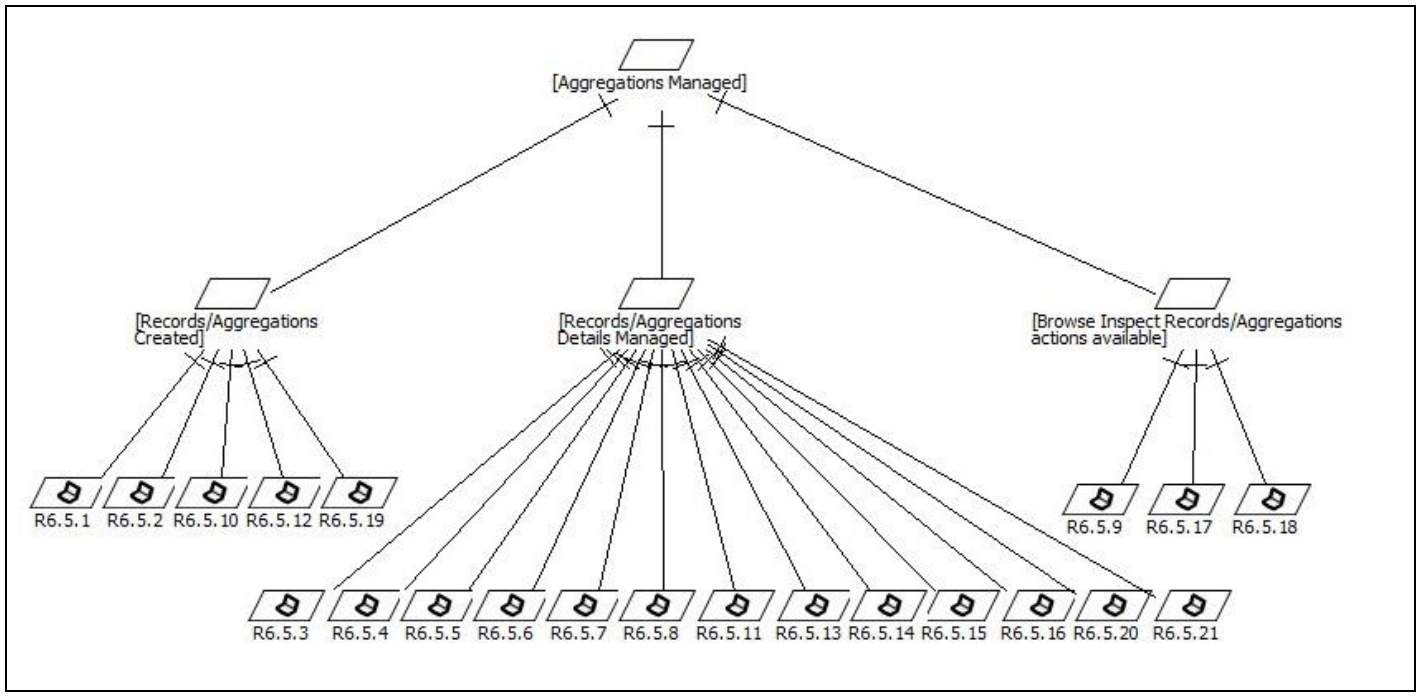
8.2 Appendix II: Model Role Service Goal Model



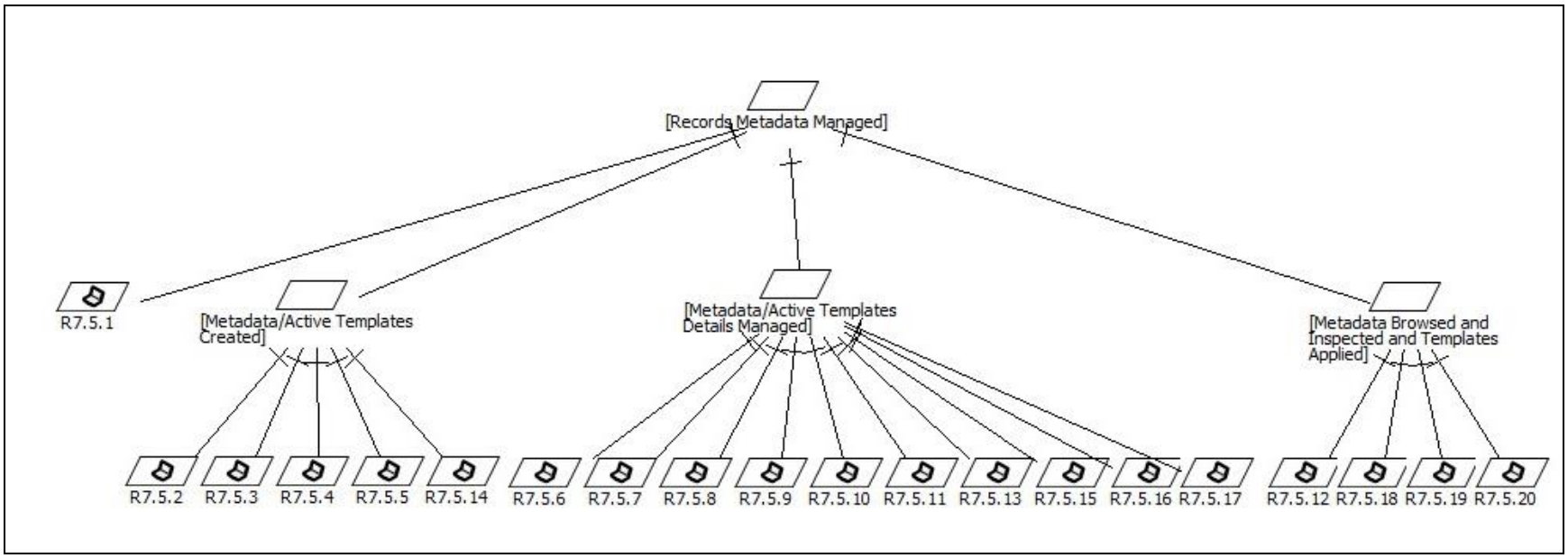
8.3 Appendix III: Classification Service Goal Model



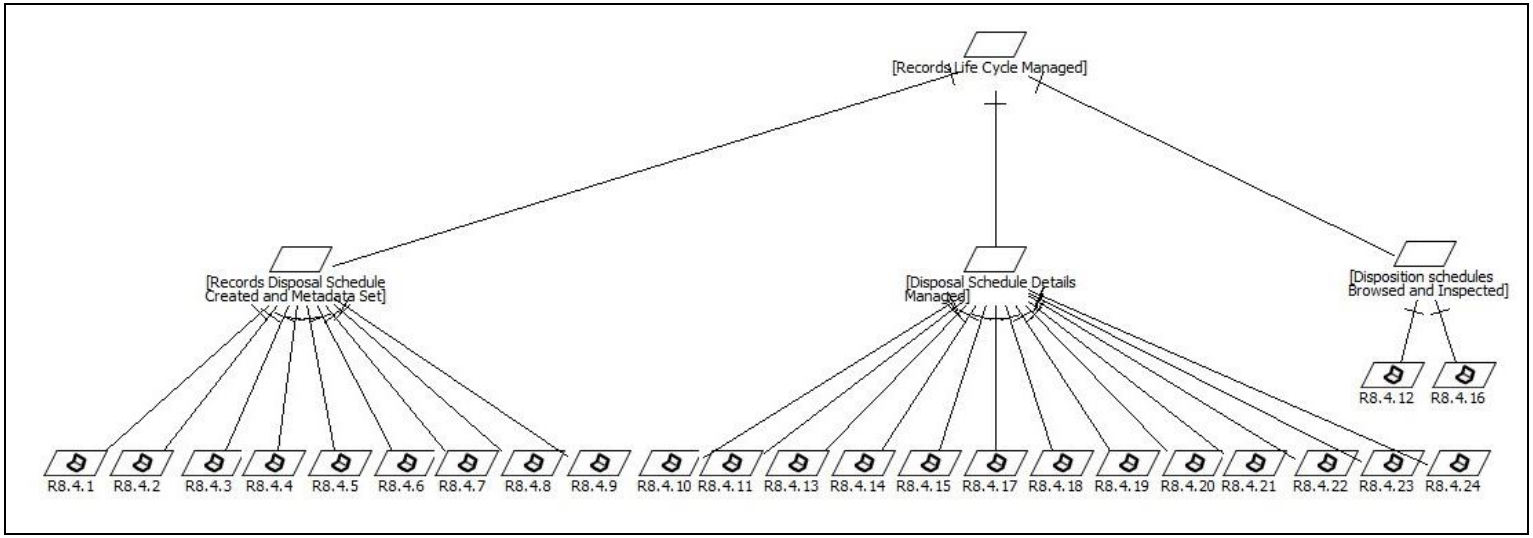
8.4 Appendix IV: Record Service Goal Model



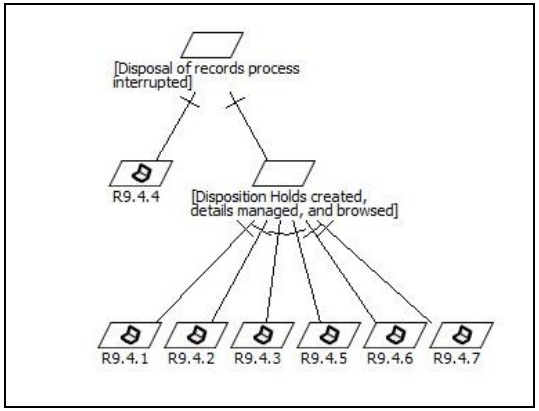
8.5 Appendix V: Model Metadata Service Goal Model



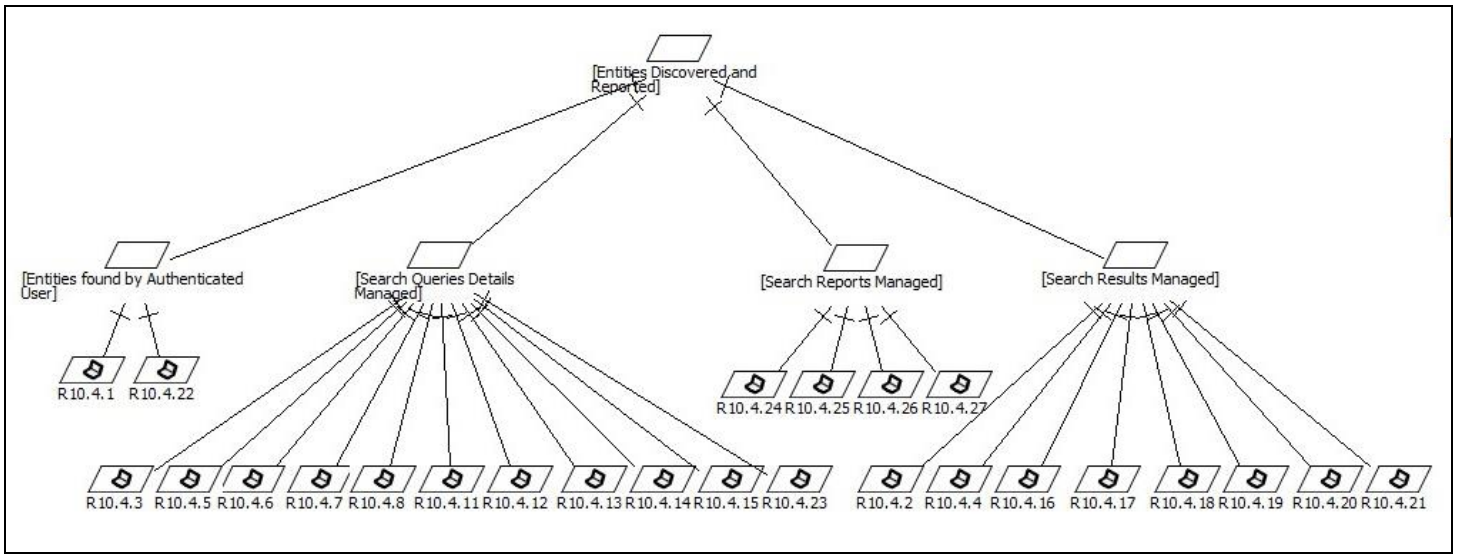
8.6 Appendix VI: Disposal Schedule Service Goal Model



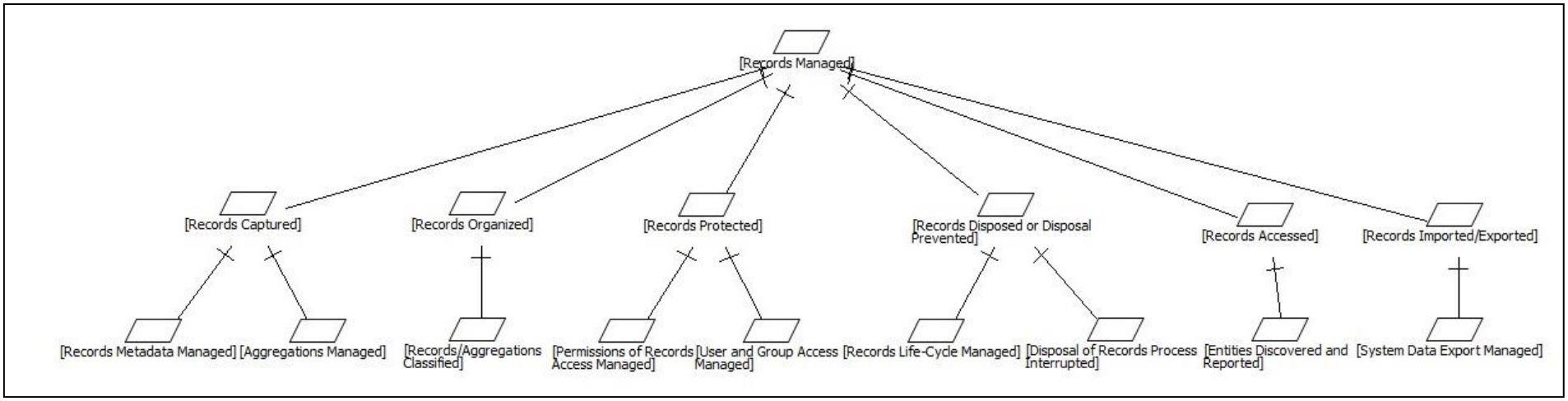
8.7 Appendix VII: Disposal Holding Service Goal Model



8.8 Appendix VIII: Searching and Reporting Goal Model



8.9 Appendix IX: Records Management Goal Model



8.10 Appendix X: KAOS Models Objects List

Agent Active - Object (or processor) performing operations to achieve goals.

Association - Object; the definition of which relies on other objects linked by the association.

Composite system - The software being studied and its environment.

Conflict - Goals are conflicting if under some boundary condition the goals cannot be achieved altogether.

Domain Property - Descriptive assertion about objects in the environment of the software. It may be a domain invariant or a hypothesis.

Entity - Autonomous object, that is, the definition of which does not rely on other objects.

Environment - Part of the universe capable of interaction with the software being studied.

Event - Instantaneous object (that is, an object alive in one state only) which triggers operations performed by agents.

Expectation - Goal assigned to an agent in the environment.

Goal - Prescriptive assertion capturing some objective to be met by cooperation of agents; it prescribes a set of desired behaviors. Requirements and expectations are goals.

Object - Thing of interest in the composite system being modeled whose instances can be distinctly identified and may evolve from state to state. Agents, events, entities and associations are objects.

Obstacle - Condition (other than a goal) whose satisfaction may prevent some goal(s) from being achieved; it defines a set of undesired behaviors.

Operation - Specifies state transitions of objects that are input and/or output of the operation. Operations are performed by agents.

Operationalizations - Relationship linking a requirement to operations. Makes the connection between expected properties (goals) and behaviors (operations).

Refinement - Relationship linking a goal to other goals that are called its subgoals. Each subgoal contributes to the satisfaction of the goal it refines. The conjunction of all the subgoals must be a sufficient condition entailing the goal they refine.

Requirement - Goal assigned to an agent of the software being studied.

Responsibility - Relationship between an agent and a requirement. Holds when an agent is assigned the responsibility of achieving the linked requirement.