Experience Report on the Construction of Quality Models for some Content Management Software Domains

Xavier Franch, Carme Quer, Josep A. Cantón, Roser Salietti
Universitat Politècnica de Catalunya (UPC)
{franch, cquer}@lsi.upc.edu

Abstract

In previous work, we proposed the use of software quality models for driving the formulation of requirements in the context of software package selection. Now, we report two related projects of construction of software quality models in the domains of Document Management, Entreprise Content Management and Web Content Management. These domains may be considered particular cases of a more general category sometimes labeled as Content Management. The goals of these projects are several. First, to assess the scalability of our methods and artifacts. Second, to investigate the degree of reusability when working on domains so closely related. Third, in relation to the previous one, to gain more knowledge of the adequacy and effectiveness of our notion of software domains taxonomy. Fourth, to evaluate the suitability and usability of our DesCOTS system proposed as tool-support for these activities.

1. Introduction

Software package selection [1] is a central activity nowadays in the context of software system development, and more specifically in component based software development [2]. One of the most important undertaken activities is the comparison of user requirements with software package capabilities. In previous work [3, 4, 5] we proposed the use of quality models as an artifact for driving this activity formulating the IQMC (Incremental Quality Model Construction) method. According to [6], a quality model is "the set of characteristics and the relationships between them which provide the basis for specifying quality requirements and evaluating them". Quality models may be used then to express user requirements and software capabilities in a unifying framework, supporting elicitation of requirements, their reuse among different selection projects, negotiation, etc. Fig. 1 shows graphically our view of this process.

In this context, we have undertaken several industrial experiences (see [7] for a summary) as well as academic ones and others that we may consider half-way (as the ones described in this paper, see section 2). During these years, we have gained experience and thus refining our method (see [5] for an abridged summary of lessons learned). We have also developed some tool support, the DesCOTS system [8, 9, 10], to assist the process. We have proposed the notion of taxonomy for organizing software package domains [11]. Currently, our main goal is to consolidate our proposal by means of large-scale experiences, using the results to populate our knowledge bases and discovering the main strengths and limitations of our work.

The goal of this experience paper goes in this direction. We present an experience that we have carried out recently in the form of two related quality model construction projects. We first enumerate the main lines of our research in relation to the experience undertaken (section 2). Then, we provide the contextual details and set the main objectives of this experience (section 3). Next, we illustrate the most relevant points of the development itself (section 4). Last, we highlight some practical advices for future quality model construction processes (section 5) and along with the conclusions we identify some open issues that yield to lines of future research (section 6). Since this paper is a report about experience results on our research, we focus on our own methods, models and techniques instead of others'. We refer to our research publications, especially the latest ones [7, 12, 5], for a thorough analysis of related work and comparison with related approaches.

0-7695-3091-5/08 $25.00 © 2008 IEEE
DOI 10.1109/ICCBSS.2008.32

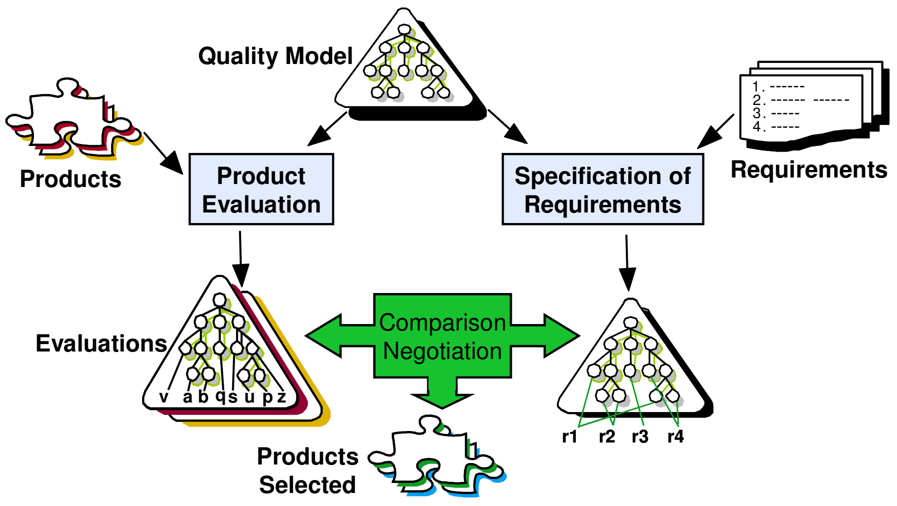


Fig. 1. Use of quality models during software package selection

2. Background

The IQMC method [3] drives the construction of software quality models as a sequence of seven steps. The quality models are built as a refinement of the ISO/IEC 9126-1 quality model [13], which defines three types of quality factors: high-level characteristics, intermediate subcharacteristics and attributes. The first two types are used for classification purposes whilst attributes (that may be basic or derived) are used to measure the quality of software through metrics. The standard proposes 6 high-level characteristics decomposed into 27 second-level subcharacteristics. We have extended the standard along two axes. On the one hand, we have defined an extension called the extended ISO/IEC model that adds 60 new subcharacteristics and attributes that we have found over and over in our software package selection experiences. On the other hand, we have included nontechnical attributes (about cost, supplier, etc.) so that technical and non-technical factors may be treated uniformly. Therefore, we could say that our updated IQMC proposal starts from that model (extended NTISO/IEC model) instead of the original ISO/IEC 9126-1. Fig. 2 shows the result of this approach. Details on the contexts of the extension may be found in [7].

Quality models are bound to software domains (e.g., workflow systems, e-mail systems, requirements management tools, etc.). Similar domains may be grouped into categories. We have observed that these domains are sometimes difficult to distinguish from each other, so we have proposed to use taxonomies for organizing them [11] and therefore quality models are bound to the nodes of this taxonomy. The taxonomy is in fact a decision tree with classifiers that allow arranging the domains and categories. Quality models are inherited downwards this taxonomy promoting thus reuse.

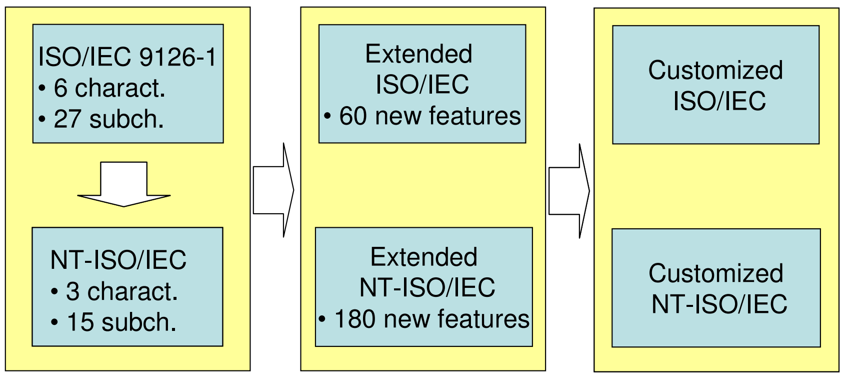


Fig. 2. Extending the ISO/IEC 9126-1 catalogue.

Last, we have proposed the DesCOTS system to support the whole process: definition of quality models and taxonomy management [8], evaluation of products [9] and requirements-driven selection [10]. The tool is currently available at http://www.lsi.upc.es/~gessi/DesCOTS.

3. The Experience

The Software Engineering for Information Systems (GESSI, http://www.lsi.upc.es/~gessi) research Group at the Universitat Politècnica de Catalunya (UPC) offers several assignments as Bachelor's thesis in the Informatics curricula. One of these assignments is a generic project for developing quality models for software package domains and using them for evaluating some products, with the support of our DesCOTS tool. We regularly get some students interested in this project.

In the last semester we had two candidates for carrying out this generic project. Both had a similar profile that we considered appropriate for our purposes: good curricula, high motivation, and they were working in prominent companies where the need of selecting software plays an important role. Furthermore, they were interested in particular in two domains that are highly related: Document Management and Web Content Management. This coincidence both in time and in domain allowed us setting the two projects as one single, two-folded experience.

In addition, the two domains have other interesting characteristics:

- They are really up-to-date, which makes the results of our work potentially interesting for a wider range of practitioners.
- As a consequence, there is currently a great deal of documentation available, which makes the information gathering part of the work challenging.
- The domains themselves are really huge, and it is even difficult to distinguish the barriers among them and other Content Management domains.
- The vocabulary is not uniform and even ambiguous, making then the domain engineering part extremely important.

As a result, we set up the main objectives of this experience as follows:

- To assess the scalability of our methods and artifacts.
- To investigate the degree of reuse when working on domains so closely related.
- To gain more knowledge on the adequacy and effectiveness of software domains taxonomies.
- To evaluate the suitability and usability of our DesCOTS system.

It is worth to remark that for our purposes, we considered convenient to discourage the interaction among the two students. However, they had access to our already existing quality models, both in the form of writing material and the contents of the DesCOTS data bases, since for measuring reuse we consider that knowledge available.

Finally, in the Document Management project, we decided to address also an additional domain, Enterprise Content Management. This possibility arose in the first stage of the project because the student volunteered for that. From our point of view, this made our reuse assessment more powerful, since we became able to assess two kinds of reuse: reuse from one person's work to another (initial goal) but also reuse by the same individual.

The results of the project are compiled in the thesis' dissertations [14, 15] currently available at http://www.lsi.upc.edu/~gessi/DesCOTS/models.html, written in Catalan (although the quality models themselves are in English).

4. Construction of the Quality Models

We present in this section the process of construction of quality models and their use in evaluation of software packages, providing some figures of interest that may help to understand the effort required for such a type of project. Roughly speaking, we distinguish two main parts when building a model, domain analysis and the quality model construction process itself.

4.1. Analysis of the Domain

First of all, the domains of interest were carefully examined and described. The objective of this step is to provide the conceptual basis for the identification of the quality factors to be included in the quality model. Following the techniques proposed in our previous work, the students studied the available information sources, build a glossary, clarified the goals of the domains and the barriers among them, and finally they identified the dependencies among the domains.

Information sources. Up to a total of more than three hundred information sources were used. The types of information sources were: web sites, technical papers, white papers, product information and textbooks. Three particular types that were considered as highly useful were:

- Web sites that provide the comparison of several products (e.g., http://www.cmsmatrix.org/).
- Short reports of important consultant companies (e.g., Gartner's Magic Quadrants).
- FAQ pages of commercial products (available in the vendor's web).

One useful information source could have been the one obtained from the installation and use of the products. However, This was not one of the sources since most of the products demo licenses were limited, and the computational resources required for installing them were high.

Glossary. A glossary was built containing up to 85 terms. Of them, 55 were domain independent (highlyreusable) and 30 related to Content Management (thus, usable in every domain belonging to this category).

Clarification of concepts. A fundamental part of the domain analysis was to clearly draw the barriers among the following concepts: Content Management Systems (usually abbreviated as CMS), Web Content Management systems (WCM), Enterprise Content Management systems (ECM) and Document Management Systems (DMS).

In a natural manner, both students felt the need of writing some pages stating very clearly their perception about the meaning of these concepts, which are described in a confusing and sometimes contradictory form in the available documentation.

One of the students built a UML class diagram to make clear the concepts of the domain [15, p. 59]; this model was synchronized with the glossary.

Some abbreviations were discovered to be overloaded, e.g. CMS is used sometimes as "Course Management System" (Moodle is sometimes labelled as being one such CMS).

One of the students included in his documentation Table 1 [14], in which 4 criteria were identified to classify the different systems depending on their values (the second criteria finally did not help classification but it is still useful for clarification purposes). Note how this solution fits with the notion of classifier for taxonomies presented in section 2; we represented this fact in the taxonomy by having a Content Management characteristic with four children, using the first criterion (atomic unit) as main classification attribute.

Table 1. Characterization of Content Management Systems

	WCM	CMS	DMS	ECM
atomic unit	any type of web contents	any type of document contents	any type of document	any type of electronic contents
life-cycle	yes	yes	yes	yes
e-mails	no	no	yes	yes
workflows	yes	no	no	yes

Domain scoping. Once concepts were clear, it became obvious that these domains have some overlapping. For instance, packages available in the market under the form of DMS or ECM systems use to have a graphical interface that may be used directly with a browser and then the final user perceives them as a kind of WCM system. Also, ECM systems incorporate some functionalities that in fact may be considered as belonging to DMS. This is a normal pattern in large packages of any kind, in which related functionalities become merged as new versions are released. It may be said that although the four domains are conceptually independent, software packages available in the market offer functionalities coming from more than one of them, making difficult to classify them as pure solutions.

At this respect, one of the students found useful to use i* SD diagrams [16] (as done in [17]) to represent more clearly the relationships among these domains and also those from Content Management domains to other type of Business Applications such as EDM (Electronical Document Management, for capturing and managing non-electronic documents), Workflow, ERP (Enterprise Resource Planning), etc. Focusing in the DMS part, the SD diagram had 11 actors and 43 dependencies [14, pp. 51-52].

4.2. Quality model construction

Once clarified the domains, the quality models were constructed, and one product was evaluated. Here we describe the aspects that we consider relevant of these activitities.

Functional part. Functionality, represented by the Suitability subcharacteristic in the ISO/IEC 9126-1 standard, took a big part of the whole effort. These types of systems offer a very rich variety of services.
The main issues here are the following:

- Identify these services, accordingly to the characteristics of the domains.
- Discriminate which of these services are really part of the domain and which ones are just marginal or

particular services that should not appear in the (general-purpose) quality model.

- Classify and arrange hierarchically the services improving understandability and later search.
- Decide the most adequate metrics for attributes, which usually just keep track of the different levels or variety of each service.

It is interesting to remark that each student followed a different approach for identifying the main subcharacteristics to classify the quality factors of the functional part. One of the students focused in the main entities of the domains [14]. For instance, in the DMS case, he decomposed the ISO/IEC Suitability subcharacteristic into 10 other subcharacteristics, one for each of the following concepts: Users, Groups, Roles, Documents, Folders, Alias, Query, Life-cycle, Mail, Web Contents.

The other student focused in processes instead [15]. Thus, in the WCM case, the subcharacteristics obtained were: Content Creation, Content Management, Content Delivery, Built-in Applications and User Accounts Management.

Once main subcharacteristics were determined, they were further decomposed into other subcharacteristics and attributes until they obtained several metrics for the different actions. For instance, in the DMS case, Group Suitability was decomposed into one single attribute, Group Management, with 14 basic Boolean attributes, one for each action: Create Group, Add User, etc.

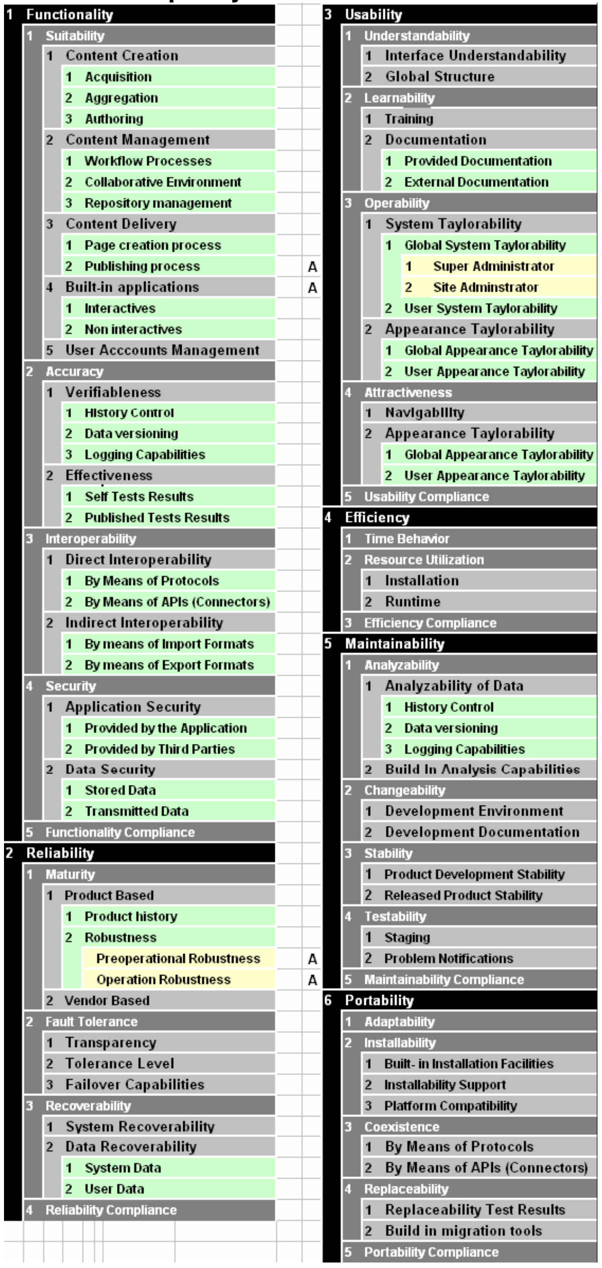
Non-functional part. This part was easier since both students took as starting point our extended ISO/IEC quality model. Basically, they just decomposed the subcharacteristics into attributes and introduced the metrics. The student that developed two quality models reused entirely the whole non-functional part [14].

Table 2 shows a summary of the structure of the quality models. In the "number" columns, "x+y" means "x in the first level and y in the second". As an example, table 3 shows the structure of characteristics and subcharacteristics of the WCM quality model.

Table 2. Form of the quality models

	Functional part (suitability)			
	Subcharacteristics		Attributes	
	levels	number	levels	number
WCM	3	1+5+10 = 16	3	63+84+42 = 189
DMS	2	1+10 = 11	3	19+135+182 = 336
ECM	2	1+3 = 4	3	12+130+72 = 214
	Non-functional part			
	Subcharacteristics		Attributes	
	levels	number	levels	number
WCM	4	26+38+28+4 = 96	3	170+96+6 = 272
DMS	4	26+21+37+50 = 128	3	122+78+19 = 219
ECM	4	26+21+37+50 = 128	3	122+78+19 = 219

Table 3. Characteristics and subcharacteristics of the WCM quality model



1 Functionality					
	1 Suitability				
		1 Content Creation			
			1 Acquisition		
			2 Aggregation		
			3 Authoring		
		2 Content Management			
			1 Workflow Processes		
			2 Collaborative Environment		
			3 Repository management		
		3 Content Delivery			
			1 Page creation process		
			2 Publishing process		A
		4 Built-in applications			A
			1 Interactives		
			2 Non interactives		
		5 User Acccounts Management			
	2 Accuracy				
		1 Verifiableness			
			1 HIstory Control		
			2 Data versioning		
			3 Logging Capabilities		
		2 Effectiveness			
			1 Self Tests Results		
			2 Published Tests Results		
	3 Interoperability				
		1 Direct Interoperability			
			1 By Means of Protocols		
			2 By Means of APIs (Connectors)		
		2 Indirect Interoperability			
			1 By means of Import Formats		
			2 By means of Export Formats		
	4 Security				
		1 Application Security			
			1 Provided by the Application		
			2 Provided by Third Parties		
		2 Data Security			
			1 Stored Data		
			2 Transmitted Data		
	5 Functionality Compliance				
2 Reliability					
	1 Maturity				
		1 Product Based			
			1 Product history		
			2 Robustness		
				Preoperational Robustness	A
				Operation Robustness	A
		2 Vendor Based			
	2 Fault Tolerance				
		1 Transparency			
		2 Tolerance Level			
		3 Failover Capabilities			
	3 Recoverability				
		1 System Recoverability			
		2 Data Recoverability			
			1 System Data		
			2 User Data		
	4 Reliability Compliance				

3 Usability				
	1 Understandability			
		1 Interface Understandability		
		2 Global Structure		
	2 Learnability			
		1 Training		
		2 Documentation		
			1 Provided Documentation	
			2 External Documentation	
	3 Operability			
		1 System Taylorability		
			1 Global System Taylorability	
				1 Super Administrator
				2 Site Adminstrator
			2 User System Taylorability	
		2 Appearance Taylorability		
			1 Global Appearance Taylorability	
			2 User Appearance Taylorability	
	4 Attractiveness			
		1 Navigability		
		2 Appearance Taylorability		
			1 Global Appearance Taylorability	
			2 User Appearance Taylorability	
	5 Usability Compliance			
4 Efficiency				
	1 Time Behavior			
	2 Resource Utilization			
		1 Installation		
		2 Runtime		
	3 Efficiency Compliance			
5 Maintainability				
	1 Analyzability			
		1 Analyzability of Data		
			1 History Control	
			2 Data versioning	
			3 Logging Capabilities	
		2 Build In Analysis Capabilities		
	2 Changeability			
		1 Development Environment		
		2 Development Documentation		
	3 Stability			
		1 Product Development Stability		
		2 Released Product Stability		
	4 Testability			
		1 Staging		
		2 Problem Notifications		
	5 Maintainability Compliance			
6 Portability				
	1 Adaptability			
	2 Installability			
		1 Built- in Installation Facilities		
		2 Installability Support		
		3 Platform Compatibility		
	3 Coexistence			
		1 By Means of Protocols		
		2 By Means of APIs (Connectors)		
	4 Replaceability			
		1 Replaceability Test Results		
		2 Build in migration tools		
	5 Portability Compliance			

Updating the taxonomy. Once the model was built, one of the students made the effort to analyse which factors could be bound to quality models in upper levels of the taxonomy of domains [15], see fig. 3. To do so, she started from the taxonomy root, Business Applications, since the upper level the factor is, the more models benefit from its existence. She found that up to 108 factors could be thought of being upgraded to the ancestors of the WCM domain (this means the 23% of the total size of the model), namely Business Applications (46 factors), Multi User Systems (23), Information Systems (14), Knowledge Management Software (21) and Content Management Systems (4). This issue, however, should be further investigated.

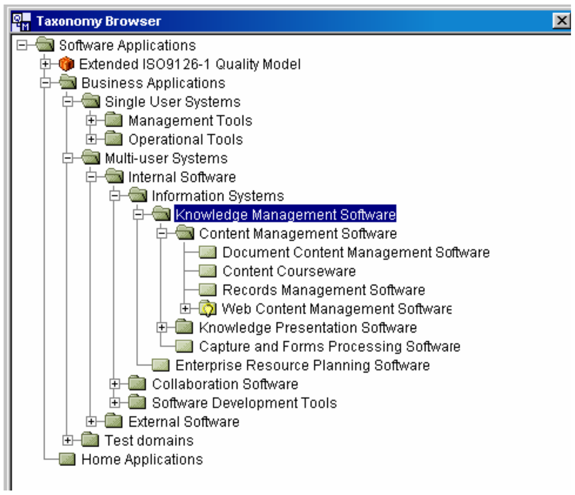

Taxonomy Browser
Software Applications
Extended ISO9126-1 Quality Model
Business Applications
Single User Systems
Management Tools
Operational Tools
Multi-user Systems
Internal Software
Information Systems
Knowledge Management Software
Content Management Software
Document Content Management Software
Content Courseware
Records Management Software
Web Content Management Software
Knowledge Presentation Software
Capture and Forms Processing Software
Enterprise Resource Planning Software
Collaboration Software
Software Development Tools
External Software
Test domains
Home Applications

Fig. 3: Excerpt of the taxonomy for Business Applications

Evaluation of products. Once quality models were available, the students evaluated in detail one product of the domain. One of them was able to use the same product, LiveLink, for both DMS and ECM domains, whilst the other evaluated an open source tool, Plone 2.5.1. In both cases, the quality model was used as a kind of template or even checklist (since a great deal of attributes was Boolean) and provided a prescriptive guidance to evaluation. Evaluation was also a kind of validation of the model, since some flaws were discovered and some information added (in the different artefacts: quality models, glossary, and UML and i* models).

4.3. Distribution of Effort

Table 4 presents the main results concerning effort measured in hours. We could say that a quality model construction project for a large-scale domain such as DMS or WCM takes about 500 hours the first time it is made, whilst the second time for a highly related domain takes much less, about one fifth of the total. We may then consider that 500 hours is a kind of upper bound of this type of project since as we discuss in [5], it will be seldom needed to build a whole model, instead construction will focus in those parts that are more related to the requirements of the selection at hand.

Table 4. Effort invested (in hours)

	WCM	DMS	ECM
Domain analysis	264	200	20
Context analysis	56	100	10
Quality model construction	142	200	30
Tool evaluation	96	100	
TOTAL	558	550	110

5. Lessons Learned

This section includes the lessons learned that the students stated after finishing the construction of the quality models.

Lesson 1. About the skills needed. The skills needed to develop a quality model are radically different of the expertise the students got in our regular University courses. As one of them stated [14], the quality expert needs to be able to locate effectively the adequate sources, to discriminate among relevant and marginal or biased information, to reorganize continuously the collected information as new sources are explored, etc. This kind of skills does not appear as part of the computer curricula, and it is necessary to enforce it as a methodological tool when the students prepare their courses.

Lesson 2. About the usefulness of the approach. Both students agreed that constructing the first quality model implies a big cost on learning how to make and effective domain analysis and in learning the underpinnings of the ISO/IEC standard. For this reason, their opinion is that this construction has mainly sense in the context of a consultant company in which the return on investment is supported by the repeated use of quality models in several projects and the construction of quality models for several domains. The relatively low cost of construction of the second quality model in [14] seems to support this lesson.

Lesson 3. About the importance of domain analysis. For an individual who is novel to the domain of interest, the domain analysis phase of the process takes even more time than the construction of the quality model itself (see Table 3). It is important also to say that even gaining experience in the process of dealing with information sources, one cannot realistically expect to reduce drastically this effort, because every large-scale domain requires an important investment on understanding. The only exception has been experienced in [14], in which the second, closelyrelated domain could be analysed much faster.

Lesson 4. About the balanced combination of incremental and sequential development. We have mentioned the inherent incremental nature of the process as new information sources are processed. But on the other hand, it seems that there are three big phases that should not overlap much. On the one hand, the construction of the quality model before a complete understanding of the domain and its context may cause many changes in the structure of the quality model. On the other hand, the definition of metrics before the upper levels of attributes of the quality model are quite stable may provoke the need of redefining these metrics several times.

Lesson 5. About the use of auxiliary artefacts. We have presented several artefacts that the students have used in their work. They considered them highly valuable. The use of glossaries of terms and the taxonomy helped them to undertake the difficult task of determining the limits of the target domain. These difficulties came mainly provoked by the big amount of information sources and the mess of concepts found in them. The i* SD and UML diagrams also contributed to clarify concepts. The i* SD diagrams were considered a big help in establishing the context of the target domain. They were also used for communication means for describing the work to other people.

Lesson 6. About reuse of the functional part. Even in the case of so closely-related domains, reusability of the functional part was very restricted. More than particular subcharacteristics, attributes or metrics, we could talk of reuse of concepts, and some patterns did appear, e.g. in [14] most of the secondlevel subcharacteristics were decomposed into two subcharacteristics for Management and Security. We could eventually expect to define several of these patterns as a way also to improve the effectiveness of quality model construction.

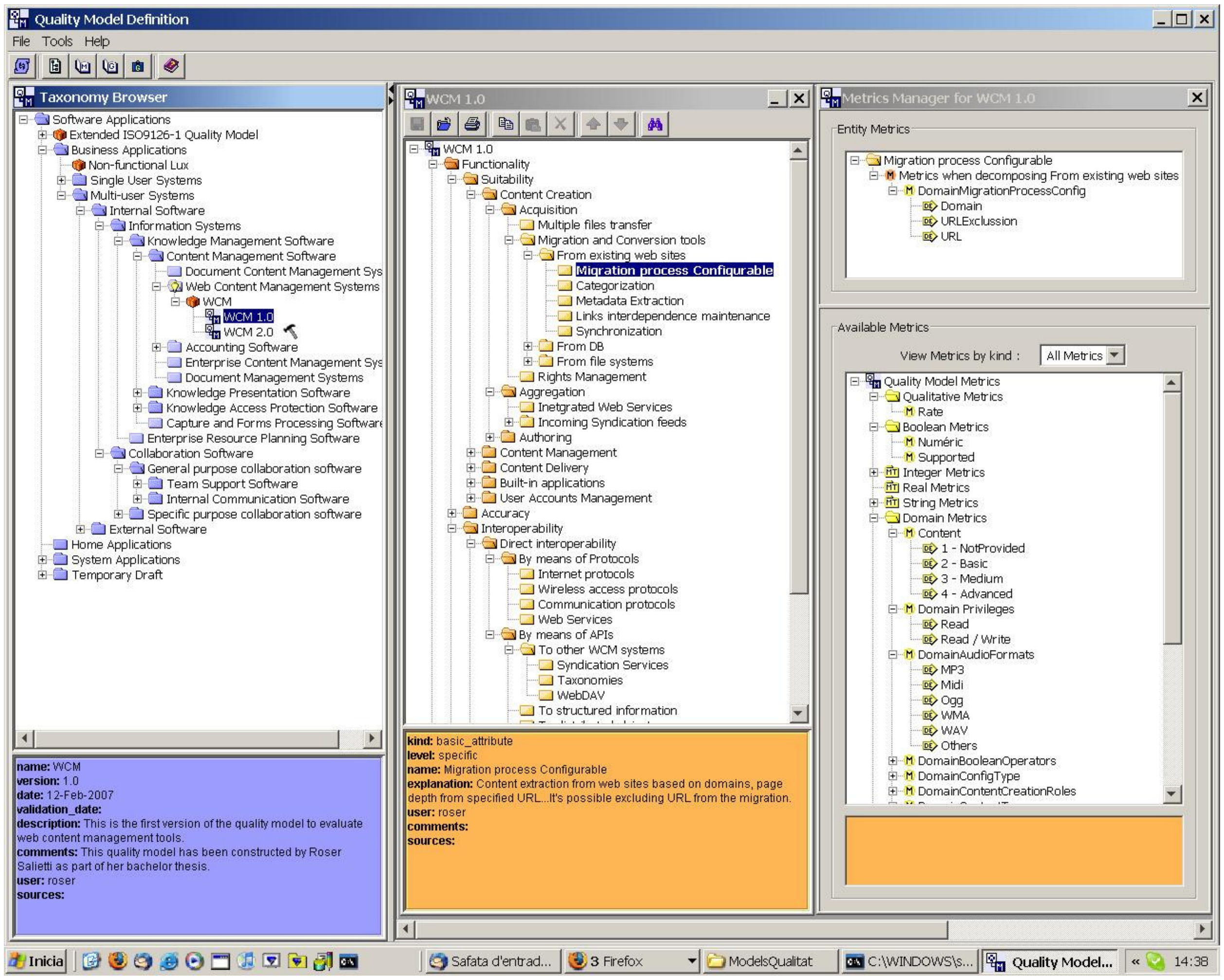
Lesson 7. About reuse of the non-functional part. On the contrary, results on the non-functional part are very good concerning reuse. Both of the projects reused a great deal of quality factors that appeared in previous projects and in particular one of students did reuse entirely the non-functional part in his two models. We are aware that this would not be the case in the case of a real project in which requirements would have forced to pay more attention to some parts of the model and then refine them in more detail. But anyway we believe that the degree of reusability could be still very high.

Lesson 8. About the taxonomy. The idea of taxonomy is aimed at supporting the identification of domains when making a selection project, which was not really the case in this experience. So the students are able to inform only about the facility to understand the concept and to extend the taxonomy with the new domains. They both agreed on understanding the concept, envisaging the usefulness, and found easy to identify a classifier to extend the taxonomy. In [14] a proposal of intermediate node can be found grouping three of the four Content Management domains to include the concept of Integrated Document Management (IDM) systems that arise often in the related literature.

Lesson 9. About tool-support. Both students agreed in the need of having specialized tool support. They both agreed that spreadsheets are not a suitable support for constructing quality models because it is difficult to represent and maintain the hierarchy of the quality model, and its visualization is hard. Instead, the DesCOTS-QM subsystem helps in the management of this hierarchy. In the tool, the changes in the organization of the quality model are easy to be done and their appearance as a tree, the branches of which may be folded and unfolded, is very visual and compact. Also, some added functionalities such as copy and paste have improved usability. Fig. 4 shows a screenshot of DesCOTS when building the quality model of the WCM domain.

Lesson 10. About evaluation of products. The evaluation of the software packages took more time than foreseen. Installing and executing the products was not an easy task, and it was difficult to obtain values for all the attributes, especially non-functional.

Lesson 11. About compatibility with previous experiences. Since this has not been the first experience, it is good to analyse the results with respect to the previously undertaken ones. Good news is that the lessons above are aligned with these previous experiences. The process behaved similarly, and the quality models were smoothly defined as a refinement of our extended ISO/IEC model (just a few factors that were not of interest were removed), allowing also a significant reuse from existing models in the nonfunctional part. The new targeted domains were accommodated in our taxonomy of domains, and the DesCOTS system supported most of our needs in a good enough way.



```
Software Applications
  Extended ISO9126-1 Quality Model
  Business Applications
    Non-functional Lux
    Single User Systems
    Multi-user Systems
      Internal Software
        Information Systems
          Knowledge Management Software
            Content Management Software
              Document Content Management Sys
              Web Content Management Systems
                WCM
                  WCM 1.0
                  WCM 2.0
            Accounting Software
            Enterprise Content Management Sys
            Document Management Systems
          Knowledge Presentation Software
          Knowledge Access Protection Software
          Capture and Forms Processing Software
          Enterprise Resource Planning Software
        Collaboration Software
          General purpose collaboration software
            Team Support Software
            Internal Communication Software
          Specific purpose collaboration software
      External Software
  Home Applications
  System Applications
  Temporary Draft
```

name: WCM
version: 1.0
date: 12-Feb-2007
validation_date:
description: This is the first version of the quality model to evaluate web content management tools.
comments: This quality model has been constructed by Roser Salietti as part of her bachelor thesis.
user: roser
sources:

```
WCM 1.0
  Functionality
    Suitability
      Content Creation
        Acquisition
          Multiple files transfer
          Migration and Conversion tools
            From existing web sites
              Migration process Configurable
              Categorization
              Metadata Extraction
              Links interdependence maintenance
              Synchronization
            From DB
            From file systems
          Rights Management
        Aggregation
          Inetgrated Web Services
          Incoming Syndication feeds
        Authoring
      Content Management
      Content Delivery
      Built-in applications
      User Accounts Management
    Accuracy
    Interoperability
      Direct interoperability
        By means of Protocols
          Internet protocols
          Wireless access protocols
          Communication protocols
          Web Services
        By means of APIs
          To other WCM systems
            Syndication Services
            Taxonomies
            WebDAV
          To structured information
```

kind: basic_attribute
level: specific
name: Migration process Configurable
explanation: Content extraction from web sites based on domains, page depth from specified URL...It's possible excluding URL from the migration.
user: roser
comments:
sources:

```
Migration process Configurable
  Metrics when decomposing From existing web sites
    DomainMigrationProcessConfig
      Domain
      URLExclussion
      URL
```

```
Quality Model Metrics
  Qualitative Metrics
    Rate
  Boolean Metrics
    Numéric
    Supported
  Integer Metrics
  Real Metrics
  String Metrics
  Domain Metrics
    Content
      1 - NotProvided
      2 - Basic
      3 - Medium
      4 - Advanced
    Domain Privileges
      Read
      Read / Write
    DomainAudioFormats
      MP3
      Midi
      Ogg
      WMA
      WAV
      Others
    DomainBooleanOperators
    DomainConfigType
    DomainContentCreationRoles
```

Fig. 4: Snapshot of DesCOTS for the WCM case

6. Conclusions and Open Issues

The conclusions of this work have to be seen with regard to our main objectives enumerated in section 3:

- *Scalability*. The methods and artefacts subject of study have scaled well with these big domains. The key factors have been: the hierarchical nature of quality models; the incremental nature of our method; the use of auxiliary artefacts.
- *Reuse*. We have confirmed that the nonfunctional part of the quality models is quite similar in different domains, whilst for the functional part we may reuse some patterns of behaviour but not really parts of the model. One consequence is the possibility to enlarge our extended ISO/IEC model to allow more direct quality model construction.
- *Classification*. We have extended the departing taxonomy by adding the targeted domains with little effort, being easy to identify the adequate classifier, and we have identified more than one hundred quality factors that could be upgraded to upper levels. This would be a good improvement of our current proposal.
- *Tool-support*. The suitability, usability and performance of the version 3.2 of the DesCOTS system were considered satisfactory enough for the construction of the quality models, far much better than a single spreadsheet. New functionalities are on the way, remarkably to allow derived metrics whose absence was identified as the main drawback by the students.

Next we present the open issues that we are going to consider in future experiences that are related with three subjects: the definition of the metrics for the functional part of the quality models, the decomposition of that functional part and its influence in the extendibility of quality models.

As it happened in our past experiences of construction of quality models, the metrics assigned to the attributes in the functional part of the quality models were almost all boolean and set-valued metrics. In fact, a basic attribute with a set-valued metrics could be converted in a derived attribute decomposed into several boolean attributes, and the other way round. In one case, the model has more attributes but simpler.

In section 4 we have mentioned that the two students used a different approach for the decomposition of the functional part of the models, concept-oriented vs. process-oriented. Which approach is more convenient? Are there other possible approaches? Which is more convenient?

When students were finishing their work, they observed that new information sources appeared that could influence their work. This is one more example that domains change and also quality models have to evolve to reflect these changes. It is ongoing work to analyse in which way the structure and the decomposition of characteristics of quality models influence on their evolution.

7. References

[1] A. Finkelstein, G. Spanoudakis, M. Ryan, “Software Package Requirements & Procurement”. In *Proc. Int’l Workshop Software Spec. and Design* (IWSSD), 1996.

[2] C. Szyperski. Component Software - Beyond Object-Oriented Programming. *Addison-Wesley*, 2002.

[3] X. Franch, J.P. Carvallo. “Using quality models in software package selection”. *IEEE Software*, 20(1), Jan/Feb 2003.

[4] J.P. Carvallo, X. Franch, C. Quer. “Defining a Quality Model for Mail Servers”. In *Proceedings 2nd International Conference on COTS-Based Software Systems* (ICCBSS’03), LNCS 2580, 2003.

[5] J.P. Carvallo, X. Franch, C. Quer. “Determining Criteria for Selecting Software Components: Lessons Learned”. *IEEE Software*, 24(3), June/July 2007.

[6] International Standards Organization. “Information Technology – Software. Product Evaluation. Part 1: General Overview”. 1999.

[7] J.P. Carvallo, X. Franch, C. Quer. “Managing Non-Technical Requirements in COTS Components Selection”. In *IEEE Proceedings 14th International Conference on Requirements Engineering* (RE), 2006.

[8] J.P. Carvallo, X. Franch, G. Grau, C. Quer. “QM: A Tool for Building Software Quality Models”. In *IEEE Proceedings 12th International Conference on Requirements Engineering* (RE), 2004.

[9] C. Quer, X. Franch, X. López. “DesCOTS-EV: A Tool for the Evaluation of COTS Components”. In *IEEE Proceedings 13th International Conference on Requirements Engineering* (RE), 2005.

[10] C. Quer, X. Franch, X. López. “DesCOTS-SL: A Tool for the Selection of COTS Components”. In *IEEE Proceedings 14th International Conference on Requirements Engineering* (RE), 2006.

[11] J.P. Carvallo, X. Franch, C. Quer, M. Torchiano. “Characterization of a Taxonomy for Business Applications and the Relationships between them”. In *Procs. 3rd Int’l Conference on COTS-Based Software Systems* (ICCBSS), 2004.

[12] C. Ayala, X. Franch. “A Goal-Oriented Strategy for Supporting Commercial Off-the-Shelf Components Selection”. In *Proceedings 9th International Conference on Software Reuse* (ICSR’06), LNCS 4039, 2006.

[13] ISO Standard 9126: Software Engineering – Product Quality, part 1. International Organization for Standarization, 2001.

[14] J. Cantón. “Construction of a quality model for DMS/ECM tools”. Bachelor’s dissertation, available at http://www.lsi.upc.edu/~gessi/DesCOTS/ models.html, 2006.

[15] R. Salietti. “Definition of a Quality Model for WCM Tools”. Bachelor’s dissertation, available at http://www.lsi.upc.edu/~gessi/DesCOTS/models.html, 2007.

[16]. E. Yu. *Modelling Strategic Relationships for Process Reengineering*. PhD Thesis, University of Toronto, 1995.

[17] V. Sai, X. Franch, N. Maiden. "Driving Component Selection through Actor-Oriented Models and Use Cases". In *Proceedings of the 3rd International Conference on COTS-Based Software Systems* (ICCBSS’04), LNCS 2959.