Characterization of a Taxonomy for Business Applications and the Relationships among them[¥]

Juan P. Carvallo[1], Xavier Franch[1], Carme Quer[1], Marco Torchiano[2]

[1] Universitat Politècnica de Catalunya
UPC-Campus Nord (C6)
08034 Barcelona, Catalunya, Spain
{carvallo, franch, cquer}@lsi.upc.es
http://www.lsi.upc.es/~gessi

[2] Politecnico di Torino
C.so Duca degli Abruzzi, 24
10129 Torino, Italy
torchiano@polito.it
http://softeng.polito.it/torchiano

Abstract. In this paper we propose a taxonomy for classifying COTS business applications, i.e. products that are used in the daily functioning of all types of organizations worldwide, such as ERP systems and document management tools. We propose the identification of characterization attributes to arrange the domains which these products belong to, and also we group these domains into categories. We define questions and answers as a means for browsing the taxonomy during COTS selection. We show the need of identifying and recording the relationships among the domains and propose the use of actor-oriented models for expressing these relationships as dependencies. Last, we explore the definition of quality models for the domains, to be used in COTS selection, focusing on their reusability and stepwise definition downwards the hierarchy.

1. Introduction

The amount of Commercial Off-The-Shelf (COTS) products [1, 2] available on the market is growing more and more. This tendency is due both to the increasing adoption of component-based software technologies by the community, and to the continuous creation of new communication and marketing channels that bridge the gap between providers and consumers of those products. Therefore, there is an increasing need for organizing the types of available COTS products to achieve more efficient and reliable selection processes.

Such a need is especially felt in the *business applications* (BA) [3] context. All the aspects of the daily operations of small, medium and large organizations, either private companies or public administrations, heavily depend on the existence of adequate software products to undertake crucial tasks, such as accounting, human resources management, document administration, team work, people communication, business processes monitoring, etc. Therefore, having specific means for discovering which BAs satisfy the needs of an organization is utterly convenient in order to select the most suitable.

[¥] This work is partly supported by CICYT TIC2001-2165 and WISE IST-2000-30028.

Carvallo, J.P. [et al.]. Characterization of a taxonomy for business applications and the relationships among them. A: International Conference on COTS-Based Software Systems. "COTS-Based Software Systems: Third International Conference, ICCBSS 2004, Redondo Beach, CA, USA, February 1-4, 2004: proceedings". Springer, 2004, p. 221-231.
The final authenticated version is available online at https://doi.org/10.1007/978-3-540-24645-9_38

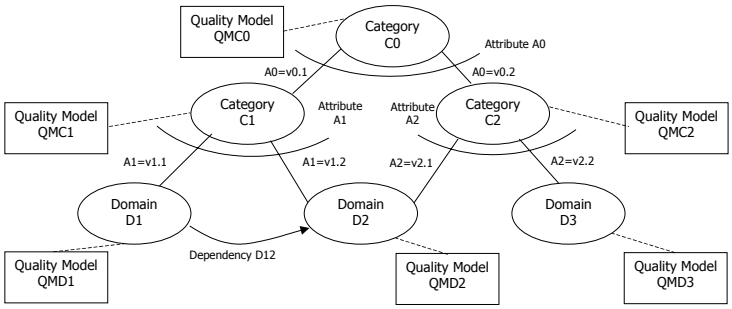

Fig. 1. The fundamental elements of a taxonomy

The purpose of this paper is to provide support for improving the BAs selection processes. Our proposal is built upon a taxonomy of BAs (see fig. 1). We consider that BAs belong to one or more BA domains which appear as leafs in the taxonomy. A domain encloses a significant group of functionality; they are grouped into categories, which in their turn can be grouped to form a multi-level taxonomy. We also represent dependencies among the domains. The taxonomy can be adopted in our COTS selection practices, which use quality models to assess the adequacy of components with respect to requirements [4, 5]: we attach quality models to nodes in the taxonomy, supporting model reuse by inheriting them downwards the hierarchy.

Several taxonomies can be found not only for BAs but for other domains too [6, 7, 8]. But more important than the concrete form that a taxonomy takes, is the rationale behind its construction, i.e. which are properties that may help to organize the BAs and how the taxonomy can be searched. This is especially true when considering not just the construction of the taxonomy, but its evolution. Categories and domains may be arranged with respect to various characterization attributes. We propose to use as rationale the notion of characterization attribute as introduced in [9, 10].

The rest of the paper is structured as follows. Section 2 introduces a conceptual model for the proposal. Section 3 presents some highlights of the taxonomy for BAs. Section 4 addresses to the relationships among BAs, while section 5 focuses on the use of the taxonomy for the definition of quality models. Finally, section 6 presents the conclusions and some future work.

2. A Conceptual Model for COTS Taxonomies

A formal representation of the concepts that appear in a valid taxonomy can be provided using a conceptual model that uses the UML notation [11] (see fig. 2). A taxonomy is composed of *categories* and *domains,* generalized as *scopes*. Domains are *grouped into* categories and categories on their turn are *grouped into* other categories. A category C shall be partitioned into sub-categories, each corresponding to a *value* of a *characterization attribute* (*attribute*, for short) that *applies to* C. Also *dependencies* of four different types may be declared between domains (see section 4); the ternary association allows more than one dependency among two particular domains.

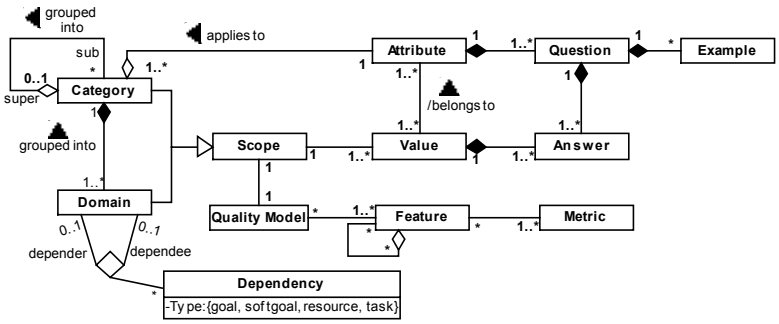

Fig. 2. Conceptual model for COTS taxonomies (integrity constraints are not included).

To simplify the classification of BAs and the identification of domains of interest during selection processes, we associate one or more *questions* to attributes of each category. When two or more questions are associated to an attribute they apply different criteria. People involved in selection processes will answer these questions and, since each *answer* is associated with an attribute value, they will browse the taxonomy at a more abstract level than using directly characterization attributes and their values. *Examples* can be associated to questions to clarify their understanding.

Finally, we define quality models for each scope in the taxonomy. A *quality model* [12, 13] is defined as a multilevel hierarchy of quality *features*. Quality features are measurable and their values are computed using some *metric*. The quality models associated with each sub-category must differ from each other at least by one quality feature. If a partition of a subcategory does not enrich the quality models then it only adds "noise". That is, it makes the taxonomy more complex and adds a question that has no purpose; therefore we apply the Occam's razor ("one should not increase, beyond what is necessary, the number of entities required to explain anything").

Attributes in general are not orthogonal, i.e. an attribute used to split a subcategory depends on the attribute used to split the super-category. In fig. 1, attribute A1 that is used to split category C1, makes sense because attribute A0 has value v0.1 while it could be meaningless if applied to category C2 for which A0 equals v0.2.

3. A Set of Characterization Attributes for Business Applications

Since reuse is the main motivation for this work, we apply it also in our research method. Instead of starting from the scratch, we preferred to take an existing one as starting point and then refactor it to obtain a taxonomy conforming to the criteria defined above. Given the particular type of systems we are addressing, we investigated the way that professional software consultant companies organize the BAs' services that they offer to their customers, and we selected the classification of one of them, which is application-oriented and well-suited for our purposes, compared to others whose classification is based on business areas or that includes not only software but other assets.

The refactoring process we performed was based on the identification of a set of characterization attributes and their associated values, questions and answers. Once the attributes were defined, they were used as the rationale to rearrange the BA categories and domains defined in the original taxonomy, with the goal of obtaining a final taxonomy as close as possible to the original one.

An excerpt of the taxonomy is presented in table 1. As first step during refactoring, we noticed that although the root of the original taxonomy was partitioned into eight sub-categories, we couldn't find an attribute that could be used to support it. Therefore we introduced intermediate categories. At this point an attribute able to discriminate among the original subcategories was needed. We found that a good option is looking at the number of users; we considered therefore two main categories, corresponding to two values of the attribute *number of users*: *single user* and *multi-user*. Single user systems are used typically by one person to work on his/her own data, while multi user systems operate on information shared by several people. We identified two different questions to elucidate the attribute's value:

- *How many users has the system*? Answers *One* or *More than one*.
- *Does the system reconcile the interests of many stakeholders*? Answers *yes* or *no*.

Starting from this first classification, we identified subsequently other characterization attributes, their corresponding values, questions, and answers (see table 2). The questions are applied at different levels in the taxonomy, and some of them are applied in more than one branch (e.g., see rows 3 and 7 in table 2). Please note the correspondence among the questions' level and the taxonomy structure.

After we finished the process, we identified the differences among the original and the resulting taxonomies. We summarize the results below.

- *Identification of new scopes*. This is the most usual action we have taken, due to the nature of our activity. Many examples exist: categories such as *Single User Systems* and its heirs *Management* and *Operational Tools* or domains as *Workflow*.

Categories					Domains
Level 1	**Level 2**	**Level 3**	**Level 4**	**Level 5**	
a. Single-User Systems	c. Management	…	…	…	…
	d. Operational	…	…	…	…
b. Multi-User Systems	e. Internal Software	f. Information Systems	h. ERP Software		Integrated ERP Software.
					Financial Management Software
			i. Knowledge Management Software	l. Capture and Forms Processing Software	Optical Character Recognition
					Forms Capture Software
				m. Content Management Software	Document Management
					Content Courseware
					Records Management
					Web Content Management.
				n. Knowledge Presentation Software.	Virtual Classrooms
					Information Access
		g. Collaboration Software	j. Team Support Software.		Workflow
					Versioning &Concurrence Control
			k. Workgroup Comm. Suites		Real Time
					Off-Line

Table 1. Partial view of the BA taxonomy.

Level	Cat.	Attribute	Values	Question	Answers
1	Root	Number of users	Single user, Multi-user	How many users has the system?	One, More than one
				Does the system reconcile the interests of many stakeholders?	No, Yes
2	a	Objective	Management, Operational	Is for management or operation?	For Management, For Operation
3	c	Orientation	Data, Process	Is it data or process oriented?	Data-oriented, Process-oriented
4	…	Data processing	Acquisition, Storage, Preparation, Analysis	What type of data processing does it perform?	Acquisition, Storage, Preparation, Analysis
3	d	Utility	Technical, Office	Is it technical or office oriented?	Technical-oriented, Office-oriented
2	b	User's location	Internal, Internal &External	Where are the users located?	Just inside the company, Inside & outside the company
3	e	Orientation	Data, Process	Is it data or process oriented?	Data-oriented, Process-oriented
4	f	Data type	Operation, Support	What type of data does it process?	Operation, Support
4	g	Type of group work	Coordination, Communication	What is the use of the system?	Coordinate teamwork, Communicate people
3	…	User's role	B2B, Customer	Is it for suppliers or customers?	For suppliers, For customers

Table 2. An excerpt of the attributes, questions and answers to browse the taxonomy.

- *Division of existing scopes*. Occasionally, some original scopes were too coarse-grained mixing different concepts belonging to different characterization attributes. This is the case for instance of *Collaboration and Knowledge Management Software* which has been split into *Collaboration Software* and *Knowledge Management Software* (the latter as subcategory of *Information Systems*).
- *Merge of existing scopes.* The other way round, some scopes are so similar that their differentiation does not really makes sense (e.g., the *Business Metrics* and the *Corporate Performance Management Software* domains).
- *Promotion of domains to categories*. Some proposed domains could be actually categories, although we have not found examples of this situation.
- *Degradation of categories to domains*. Complex domains such as *Enterprise Content Management Software* were originally considered as categories although there exist products covering their intended functionalities (which makes them domains).
- *Removal of existing scopes*. Not very often, but some scopes appearing in the original taxonomy have been removed. Three main reasons behind:
  - The scope is not a software domain, e.g. *Voice and Call Processing Equipment*.
  - The scope does not really add value to the taxonomy.
  - The scope does not really belong to the category of BAs. For instance, the *Instant Messaging* or *Audio and Video Conferencing* domains are technical means for some team work domains.

It is important to remark that the taxonomy we obtained is just one of the possible taxonomies for BAs. The goal here is not to find the "best" (if any exists) taxonomy but show how it is possible to build one.

4. Identifying Dependencies among Business Applications

The analysis of any segment of the COTS market shows that COTS products are not designed to operate isolated; instead, they work together and therefore some relationships may be established between them. This is especially true in COTS products of a large granularity, such as the BAs considered in this paper.

Commonly, COTS products of certain domains depend on others with different aims [14, 15]. Among others we mention:

- *Enabling their functionality*. A product from a domain requires a product from other domain to provide a given functionality. E.g., in order to follow document life-cycles, document management tools need workflow technology to define them.
- *Complementing their functionality*. A product from a domain requires a product from another domain to offer an additional feature, not originally intended to be part of its suitability. For instance, a web page edition tool can complement a web browser to facilitate the edition and modification of web pages.
- *Enhancing their quality attributes*. A product from a domain requires a product from another domain to improve its quality of service. For instance, resource utilization can be improved significantly using compression tools.

As far as we know, in the context of COTS selection, these dependencies among COTS domains have been dealt with in a case-by-case basis, i.e. they have been discovered in each new selection process, with the only exception of product lines architectures. Remarkably, the proposals of domain taxonomies mentioned in the introduction of the paper do not include these relationships.

It is our strong belief that dependencies among domains shall be identified and recorded explicitly for their repeated use during different selection processes. Specifically, these dependencies help organizations involved in a selection process to find out that some goals that they want to achieve with a COTS of a domain will not be satisfied if they do not have or procure COTS from other domains. For instance, an organization selecting a document management tool would discover quickly that they need a document imaging tool for scanning and storing paper documents.

Furthermore, we think that domain taxonomies provide a great opportunity for including this information as an additional element for structuring COTS domains. In fact, we may say that taxonomies are arranged along two different dimensions, one using characterization attributes and the other using dependencies. The selection process can use the first dimension to determine the domain of interest through a repeated question-answer pattern, and then the second dimension to determine the additional products required.

Actor-oriented approaches allow the optimal identification and formal representation of dependency relationships between COTS products from different domains in a taxonomy. Specifically we propose the use of *i** SD models [16], where actors represent COTS products and dependencies are established among them. In a dependency relationship, a *depender* COTS product from a domain depends on *dependee* COTS

products from other domains. The four types of dependencies of i*, namely goals, soft goals, tasks and resources, allow us to represent different relationships between domains:

- *Goals*: The *depender* depends on the *dependee* to achieve a new functionality. For the fact of being a goal dependency, the additional functionality can be provided in different ways. For example, the COTS products in the *Document Management* domain depend on COTS products of the *Web Content Management* domain to offer the functionality of *Visualization of Managed Documents through Web Pages*.
- *Soft goals*: The *depender* depends on the *dependee* to enhance some of its non-functional attributes. For example, the *Efficiency* of COTS products in the *Virtual Classroom* domain depends on others that in fact are not BAs but technologic components (i.e., belonging to a different domain).
- *Resource*: The *depender* depends on the *dependee* for its access to some resource. For example, the *Collaborative Engineering* COTS products need the *Definition of Resources* like users, user types and document types, to define the life cycle and business processes of documents in an organization. This definition will be provided by a *Document Management* COTS product.
- *Tasks*: The *depender* depends on the *dependee* to carry out a task. For example the *Document Management* COTS products depend on COTS products from the *Content Management* domain to *Store Documents in a Certain Format*.

Figure 3 shows an *i** SD diagram that includes a subset of the dependencies identified in the taxonomy of BA during the analysis of the document management domain carried out during a COTS selection process in which we have been involved.

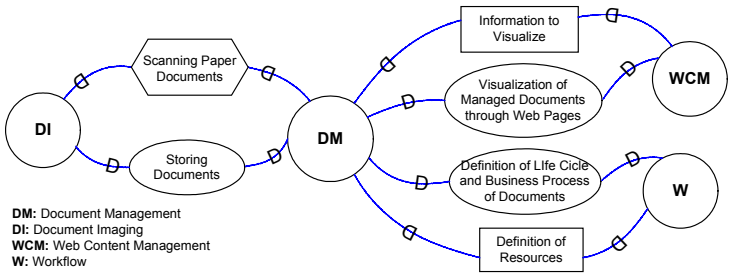

Fig. 3. *i** SD diagram with dependencies involving the *Document Management* domain

5. Organizing Quality Features around the Taxonomy

Quality models provide a general framework to get uniform descriptions of the quality features that apply to a COTS domain. We have used them to support negotiation between requirements and product capabilities as shown in [4]. We have built quality models for some domains both in real-world and academic settings (some of them reported [5, 17, 18]). As a result, we claim that quality models enhance COTS selection reliability but, on the other hand, building quality models is a time-consuming activity. Also, we face the problem that new COTS domains appear continuously and building quality models for them from the scratch is really impractical. For these reasons, we search for methods and techniques that allow quality model reuse.

Reusability of quality models downwards categories and domains of the hierarchies is a way to support this objective. We have observed throughout our experiences that some quality features appear over and over, and this repetition is directly connected to the characteristics embedded in the characterization attributes. The recognition of COTS domains and categories improves reusability: once a new COTS domain has been identified, its quality model can be constructed by inheriting the features of the quality models for those COTS categories in the hierarchy which it belongs to. During the process, new categories may be identified, abstracting commonalities of this new domain with others. As a result, a quality model bound to a category of the taxonomy collects all the quality features common to all its subcategories and domains. Since then, any quality model for a particular selection process may reuse the quality model of the corresponding COTS domain.

Fig. 4 shows the stepwise refinement of quality models downwards the hierarchy (see [15] for details). The quality model bound to the root is proposed to be a refinement of the ISO/IEC 9126-1 quality model [13]. For instance, we split the *Suitability* subcharacteristic of this standard into two, *Basic Suitability* and *Added Suitability*, keeping apart those functionalities that are inherent of a COTS domain from those that have been added as the products in the domain are enriched in new versions. Then, each category and domain adds or redefines some quality features which are related to the corresponding characterization attribute's value. For example, the quality model for the *Multi-User Systems* category includes security features such as the support for *Login and Passwords*. This feature has been reused in both the *Internal Software* and the *External Software* category quality models and some additional features such as *Security Communication Protocols* has been added to the quality models of the latest one.

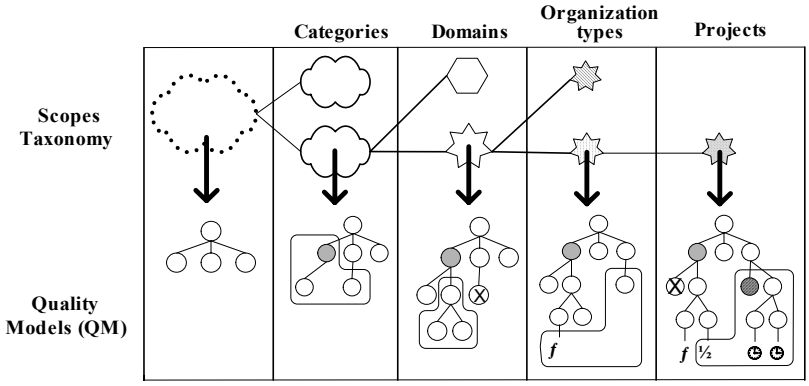

Fig. 4. Refinement of quality models downwards the taxonomy.

In our BA context, we also have observed that even quality models bound to domains such as *Virtual Classrooms* or *Integrated Enterprise Resource Planning Software* can be specialized to particular types of organizations. Consider two furniture manufacturing companies. One of them produces all the parts for its products from the basic materials while the other builds its products by assembling parts that have been previously ordered to different suppliers. As a result, the quality model for first company should be tailored to be more concerned on the control and reduction of manufacturing costs (e.g. *Definition of production budgets and schedules*, *Cost*

ing costs (e.g. *Definition of production budgets and schedules*, *Cost components tracking* and *Quality control*), while the other may be more concerned on the control of the logistics related to the process of buying the parts and their timely reception (e.g. *Purchase orders emission*, *Shipping scheduling and rescheduling* and the *Coordination of shipments from multiple providers*). We reflect this fact by introducing organization-type quality models.

Last, concrete COTS selection processes may take place with even a further refinement of a quality model, specifically tailored to the requirements of this process. For instance, the quality attributes for human-computer interface may be refined, or their metrics defined, to adapt to the particular interface requirements of the context of use of the system-to-be.

6. Conclusions

We proposed characterization attributes as the rationale for building taxonomies in any category of COTS components; questions, answers and examples simplify their use. Although introduced in the particular context of business applications, our approach can be applied to any other broad category of COTS products, such as scientific packages or life-cycle support tools. As far as we know, this rationale distinguishes our approach of other taxonomy proposals as [6, 7, 8].

We have demonstrated the feasibility of our approach in one of the most populated and critical fields of the COTS market. In particular, we have solved some problems observed in the departing hierarchy: categories were not defined precisely (as a consequence, their granularity was not always adequate); categories often overlapped; the criteria for decomposing categories was never declared and often was not evident, making hard the use of the hierarchy.

We represented explicitly the relationships between business application domains using the concept of dependencies present in actor-based models. As a result, the implications of the use of a particular business application become clearer. As far as we know, this information is not included in other taxonomy proposals.

Finally we showed how to define quality models for business application domains reusing part of them along the taxonomy; a category's quality model is inherited from the sub-categories.

Different actors may benefit from our approach:

- Software consultant companies offering assessment for business automation may structure their services better.
- Medium- and large-size companies with their own IT department may be more confident on their own selection.
- Software engineers which usually carry out COTS selection may structure better their knowledge and may aim at a better return of investment.

As future work, we mention:

- Application of the approach to other domains such as application development, software infrastructure and wireless service engineering.
- Instead of having a static taxonomy, the browsing process of the hierarchy could be built upon a dynamic taxonomy. At any moment, any sound attribute (i.e., set of

answers) could be applied to discard some categories and select others. Selection of a category after giving a value to an attribute restricts the set of sound attributes during the browsing process.

- Definition of a COTS selection process around the taxonomy. The selection process will be based on three main activities, in addition to other usual ones:
  - Browsing the taxonomy for locating the COTS domain of interest.
  - Analyzing the dependencies for understanding the implications of the selection.
  - Refining the quality model to be used in the evaluation of components.

References

1. D. Carney, F. Long. "What Do You Mean by COTS? Finally a Useful Answer". *IEEE Software*, 17 (2): 83-86, March/April 2000.
2. B. Craig Meyers, P. Oberndorf. *Managing Software Acquisition*. SEI Series in Software Engineering, 2002.
3. IT Papers, "Business Applications". http://www.itpapers.com/category/busapp.html
4. X. Franch, J.P. Carvallo. "Using Quality Models in Software Package Selection". IEEE Software, 20(1), January/February 2003, pp. 34-41.
5. J.P. Carvallo, X. Franch, C. Quer, "Defining a Quality Model for Mail Servers". 2nd International Conference on COTS-Based Software Systems (ICCBSS), LNCS 2580, Ottawa (Canada), 2003.
6. R. Glass and I. Vessey, "Contemporary Application-Domain Taxonomies" IEEE Software, 12 (4): 63-76, July 1995.
7. INCOSE, "Software Engineering Tools Taxonomy". http://www.incose.org/tools/tooltax/se_tools_taxonomy.html
8. CBSE Net, "Application Domain Taxonomy". Available on-line (previous registration) at: http://www.cbsenet.org/pls/CBSEnet/eco_ricerca_documenti.concept_search_frame
9. M. Morisio, M. Torchiano. "Definition and classification of COTS: a proposal". 1st International Conference on COTS-Based Software Systems (ICCBSS), LNCS 2255, Orlando (USA), 2002.
10. L. Jaccheri, M. Torchiano, "Classifying COTS products". 7th European Conference on Software Quality (ECSQ), Helsinki (Finland), 2002.
11. Unified Modelling Language (UML) 1.4 Specification. OMG document formal/(formal/2001-09-67). September 2001.
12. ISO Standard 8402: *Quality management and quality assurance-Vocabulary*, 1986.
13. ISO/IEC Standard 9126-1 Software Eng. – Product Quality – 1: Quality Model, June 2001.
14. X. Franch, N. Maiden. "Modeling Component Dependencies to Inform their Selection". 2nd Intl. Conference on COTS-Based Software Systems (ICCBSS), LNCS 2580, Ottawa (Canada), 2003.
15. P. Botella, X. Burgués, J.P. Carvallo, X. Franch, C. Quer. "Using Quality Models for Assessing COTS Selection". 5th Workshop on Requirements Engineering (WER), Valencia (Spain), 2003.
16. E. Yu. "Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering". 3rd IEEE International Symposium on Requirements Engineering (ISRE), 1997.
17. P. Botella, X. Burgués, J.P. Carvallo, X. Franch, J.A. Pastor, C. Quer. "Towards a Quality Model for ERP System Selection". Chapter in *Component-Based Software Quality: Methods and Techniques*, A. Cechich, M. Piattini, A. Vallecillo (eds.), LNCS 2693, 2003.
18. X. Franch, J. Marco. "A Quality Model for the Ada Standard Container Library". In *Proceedings of the 8th ICRST-Ada Europe'03, LNCS 2655*. Toulouse (France). June 2003.