Systematic Construction of Goal-Oriented COTS Taxonomies

THESIS PROJECT

Presented to
The Departament of Llenguatges i Sistemes Informàtics
of the Universitat Politècnica de Catalunya (UPC)
Software Doctoral Program

By: Claudia Patricia Ayala Martínez

Advisors: Pere Botella and Xavier Franch

Claudia P. Ayala Mtz. Pere Botella López Xavier Franch Gutiérrez

Barcelona, June 2005

TABLE OF CONTENTS

1. Introduction

In the last decade, the world of software development has evolved rapidly. In particular an alternative approach to the traditional software development life cycles has been gaining increasingly importance. This alternative consists on building systems by assembling and integrating existent software components in the form of Commercial-Off-The-Shelf (COTS) components -hereafter COTS-, and migrating existing systems towards COTS-Based Systems (CBS)[1] approaches [Meyers-Oberndorf2002], [Carney-Long2000]. The potential benefits of this new technology are mainly its reduced costs and shorter development time [Oberndorf-Brownsword1997]. Thus, as the size and complexity of software systems grow, the interest in CBS increases.

Currently, CBS is highly feasible due to the following:

- The increase in the quality and variety of COTS.
- Economic pressures to reduce system development and maintenance costs.
- The emergence consolidation and growing of component integration technology.
- The increasing amount of existing software in organizations that can be reused in new systems.

In general, COTS are software components that provide a specific functionality and are available in the market to be purchased, interfaced, and integrated into other software systems. More precisely:

> *"A COTS product is a [software] product that is: (1) sold, leased, or licensed to the general public; (2) offered by a vendor trying to profit from it; (3) supported and evolved by the vendor, who retains the intellectual property rights; (4) available in multiple, identical copies; and (5) used without source code modification by a consumer."*

B.C. Meyers, P. Oberndorf. *Managing Software Acquisition*, Addison-Wesley, 2001.

Indeed, COTS can be either software or hardware or a mixture of both. Our work only focus on software COTS, however most of the issues and advices are equally applicable to hardware COTS.

It looks very promising to use COTS in order to improve productivity and quality of software systems development. However, the nature of CBS approaches changes the focus of software engineering from traditional system specification

[1] Some literature uses CBS as Component-Based Systems. In this work, we refer to COTS-Based Systems.

and construction to simultaneous consideration of the system context (system characteristics such as requirements, cost, schedule, operating, and support environments), capabilities of products in the marketplace, and viable architectures and designs. (Fig. 1.1).

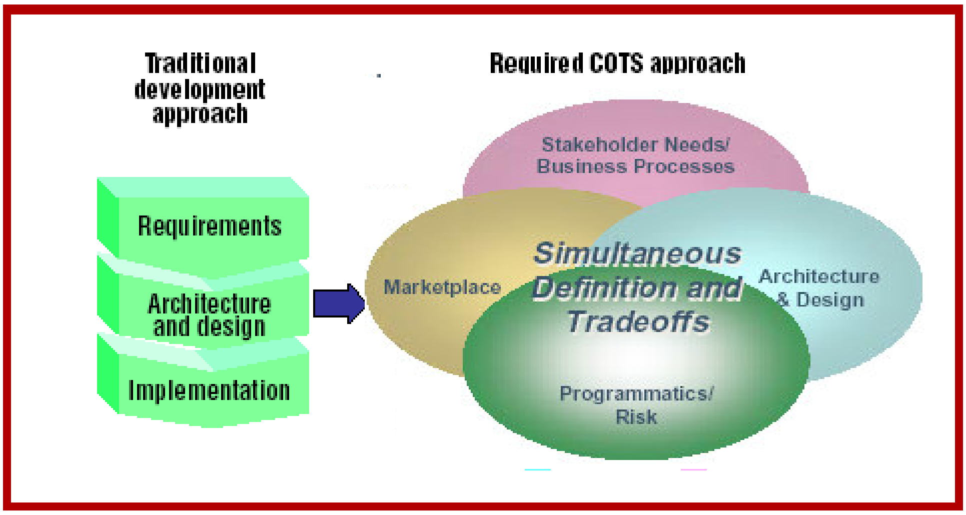


Fig. 1.1 Traditional versus COTS-Based approaches [SEI-a]

This fundamental change of building systems from COTS introduces new problems and risks [Voas1998], [Boehm-Abts1999] that are not generally found in custom development [SEI-b], mainly because:

- New and improved products and technologies are continuously offered by vendors. This leads to a huge COTS marketplace.
- Changes to products are driven by the vendors' market share goals, not by the unique needs of our programs.
- Each product has its own model of how the user will use it, which may be quite different from the end user's actual processes.
- Licensing secures the use of products that we need; data rights and warranties protect us (and the vendors) in the long-term use of those products.
- Vendors release product upgrades (and new products) at times that suit their marketing strategies, not our priorities.
- As consumers, we have little insight into how a COTS has been put together and how it behaves and why.
- Just as each product has its own idea of user processes, each also has its own architectural assumptions and paradigms, making products hard to integrate.
- Some COTS have built-in dependencies on other COTS, which amplify the impacts of product-change events.

Nowadays, the process of developing systems from COTS is an economic and strategic need in a wide variety of different application areas. As a result, a huge amount of COTS have become accessible in the marketplace. This gives raise to a new problem: how to structure the knowledge about these COTS marketplace and in particular how to know which types of packages are available and which are their objectives in such a way that searching the marketplace becomes a feasible task?

> *The marketplace is characterized by a vast array of products and products claims, extreme quality and capability differences between products, and many products incompatibilities, even when they purport to adhere to the same standards.*
>
> Software Engineering Institute, COTS-Based Sytems Overview
> http://www.sei.cmu.edu/cbs/overview2.htm

It is a fact that there are many interrelated technical, legal, management, strategic and business issues involved in effectively using COTS. Therefore, there is an increasing need for arranging the types of COTS available in the marketplace to improve the efficiency and reliability of these involved processes.

This need has been addressed in many different contexts, from IT consultant companies like Gartner [Gartner], professional societies as INCOSE [Incose], commercial web-based companies as ComponentSource [CompSource] to the academic world in which organizations, teams and individuals have presented their own proposals as [Glass-Vessey1995], only for mentioning some of them.

One of the most critical activities in CBS development is the evaluation and selection of appropriate products. The selection of suitable COTS is often a non-trivial task and requires careful and simultaneous consideration of multiple criteria [Ncube-Maiden1999], [Brownsword-et-al.2000].

In general, there are three phases in the COTS selection process: *1) Evaluation criteria definition, 2) Identification of candidate COTS from the marketplace*, and *3) Evaluation of the criteria for these candidates.*

Some methods have been proposed in the last years for dealing with COTS selection. Most of these are more addressed to the first and third phases, and less attention has been paid to the identification of candidate COTS from a huge and changing marketplace. For dealing with this crucial problem, some of these proposals suggest having a repository populated with descriptions of components; however from a practical perspective none of them has studied in deep how to handle this problematic.

In this context, the objective of this work is twofold. On one hand to present the state of the art of the related COTS processes, focusing on one of its main open issues, that is the problematic regarding the lack of structured and widespread descriptions of COTS domains that mainly hampers the accuracy of processes among COTS selection plays a critical role, integration and reuse of knowledge as well as the simultaneous consideration of the marketplace into the nature of CBS. On the other hand, to expose a proposal of solution to this problem as well as its preliminary results.

1.1. The Importance of the Problem

The current lack of structured and widespread descriptions of COTS domains hampers the accuracy of a diversity of processes that range from technical to business, such as COTS selection and integration, communicative skills, contract and market strategies.

As it was mentioned before, COTS selection plays an increasingly crucial role, in particular the process of identification candidate COTS to be evaluated from the marketplace. This is because evaluating and analyzing all the relevant characteristics of candidate COTS takes a great amount of time, typically more than the organization has. Therefore, it is necessary and cost-effective to select the most promising candidates for their evaluation.

A critical consequence of performing this activity poorly is that the whole component selection process is damaged, and confidence on the results of the process diminishes. Nevertheless, there are no studies in deep that help to handle this growing problem.

In this sense, the identification of the most promising candidates COTS for their evaluation is problematic mainly because 3 issues:

- **Growing size of the marketplace**: New and improved products and technologies are continuously offered by vendors. This leads to a huge COTS marketplace.

- **Rapid changes in the marketplace**: Rapid changes in the products marketplace and user needs makes COTS evaluation difficult [Oberndorf-Brownsword1997]. For example, a new release of the product may have a feature that is not available in the release that is currently being evaluated.

- **Diversity of components and interactions among them:** There is a huge amount of COTS accessible in the market. In general they are not developed for working isolated; so many products depend on other products for enabling or complementing their functionality.

Due to the critical nature of the COTS selection process, some methodologies have been proposed in the last years [Kontio1996], [Maiden-Ncube98], [Burgués-et-al.2002], [Briand1998], [Hissam-et-al.2002], [Kunda-Brooks1999], [Morisio-et-al.2000], [Morisio-Tsoukias1997], [Seacord-et-al.2001], [Chung-Kooper2004]. However, they focus their research on the evaluation process itself and do not pay enough attention to how to achieve the identification of candidate components to be evaluated in a huge, non-structured, and highly changing marketplace.

Moreover, as it was stated above, COTS are not designed to operate isolated; instead, they work together and therefore some relationships may be established between them [Franch-Maiden2003]. 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. Specifically, these dependencies help organizations involved in a selection process to find out that some goals that they want to achieve with a COTS will not be satisfied if they do not have or procure other COTS. 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.

Due to this, it is required to think in structuring the COTS marketplace considering above all its changing nature, capabilities of products, the user needs and the dependency relationships among them not only for facing the CBS challenges, but also for the reuse of knowledge that improve the accuracy and efficiency of the involved processes.

1.2. Approximation to the Solution

For tackling these problems we argue the need of structuring the highly dynamic COTS marketplace by means of domain classification able to address a particular activity into the right marketplace segments; this is especially useful for scanning components. The existence of such taxonomy provides a framework for the involved process and structures knowledge on the field. The intermediate nodes of the taxonomy stand for general COTS categories: they are just a classification means, not real COTS domains. Fig. 1.2 presents an excerpt of how this taxonomy may look like. It shows that as many levels as needed may be introduced to catch similarities in the right point.

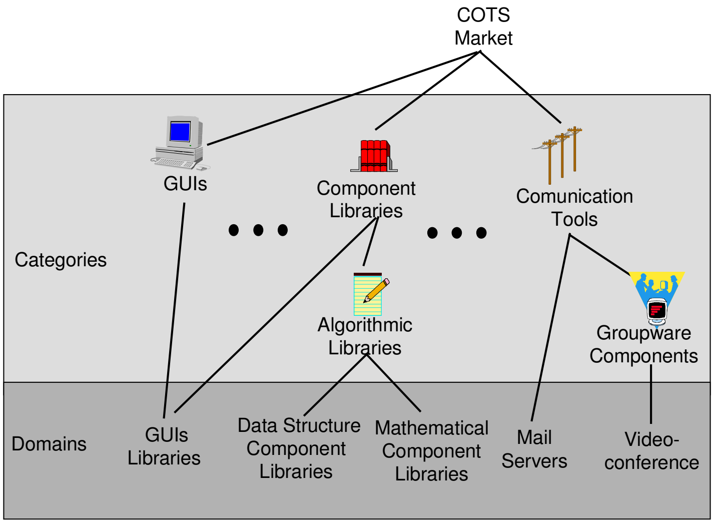


Fig. 1.2 A general taxonomy for the COTS marketplace

It is important to say that this research is addressed to the rationale behind the construction of such classifications. It means that this work is more focused on the ontological description of COTS domains because it supports the reuse of knowledge and changes in the marketplace.

Also, we think that dependencies among COTS domains shall be identified and recorded explicitly for their repeated use during different selection processes. Furthermore, we think that such domain classification provide a great opportunity for including these dependency relationships as an additional element for structuring the COTS marketplace.

We have to say that several works deal with classification of commercial products (as we explain in the state of the art section). In this sense, it is important to stand out the insights for classifying exposed by our research group in [Carvallo-et-al.2004a] from which we initiate this research line. These ideas were oriented to arrange components as a decision tree by means of "characterization attributes" (this concept was taken from [Morisio-Torchiano2002], [Jaccheri-Torchiano2002]) to discriminate among different categories (nodes) and domains (leaves). However, the research performed there was in the context of quality models reuse, and the rationale behind was not rigorous enough.

Therefore, as pointed in [Ayala-Botella-Franch2005a], while those proposals paid attention to the structure of the classification, none has been devoted to the study of how to build this classification taking into account the users requirements, evolution of the domain and trends of the marketplace. Moreover, even less attention has been paid to the methodological aspects required to support its construction.

Thus, this thesis proposal intends to arrange the COTS marketplace and its relationships implied covering the gap among the existent methods for COTS processes and their real application in front of a huge marketplace of products that requires multiple considerations.

In next sections, a detailed explanation is given about the proposal, the state of the art of involved topics, as well as the current state of this research.

1.3. Objectives of the Research

Keeping the previous in mind the general objective of this thesis work is:

> *To provide support for a reliable and comprehensible structuring of the COTS marketplace.*

This research is aimed to advance the state of the art in the study of efficient construction methodologies for organizing and structuring the evolutionary COTS marketplace, understanding and rigorously evaluating the set of information needed in this complex area, applying techniques of classification, requirements engineering and knowledge acquisition that are being used in other areas. It is important to remark that this last point is not referring to invent any new technique, but to offer guidance for the use of the currently existing ones in different situations in the COTS context.

From this perspective, special attention is given to goal-oriented approaches because the research performed so far, points them as a good candidate for extracting, refining, and handling these issues.

As a result, it is expected to answer some practical questions that arise often during CBS development:

- *Which are the market segments of interest of a particular activity?*
- *Which are the relationships among those COTS markets segments and which are their implied needs?*
- *Which lessons learned could be helpful to this particular activity?*

These practical questions can be translated into the next research questions:

- *How can relevant information from the marketplace and related considerations in a CBS be synthesized?*
- *How can this information be used to build goal-oriented taxonomies?*
- *How can this information be maintained for its reuse in different processes?*

Thus, besides the general objective presented above, some specific objectives exist for answering these research questions:

1. To explore the applicability of several classification techniques for structuring the COTS marketplace.

2. To explore the applicability of requirements engineering techniques for the COTS knowledge acquisition under which the classification is constructed.

3. Based on the previous objectives, to propose a methodology to organize the COTS knowledge marketplace in any area, which permits to any organization guiding its activities taking into account its necessities.

4. To explore the applicability of formal modeling notations to support the construction process and knowledge management.

It is recognized that the only valid test of a practical method is in its use on real projects, so from the experimental framework one pursued objective is modeling real cases for validating and enhancing some of the properties observed by this research. Two interesting real scenarios to model are Software Development Applications and Business Applications.

1.4. Structure of this Document

- Section 2 summarizes our proposal solution based on the objectives and the state of the art, stated in Section 1 and Section 3 respectively.

- Section 3 presents the state of the art related with the problem we want to solve, standing out the main open issues we want to tackle as well as our expected contribution. We also relate some topics that contribute to our research.

- Section 4 shows the research planning to fulfill the goals of this thesis project. It states the activities into a three-iteration process preceded by an initial phase 0, as well as the detailed results expected for each iteration.

- Section 5 summarizes our research performed up to date and the expected activities for getting our research objectives. The research performed is further explained in our related publications (that can be found them in the Annex Section). We close this section with topics about our next research activities to achieve our objectives.

2. The Proposal

Based on the analysis of the state of the art and the detailed objectives presented in section 3 and section 1 respectively, the proposal is:

A method for structuring the COTS marketplace.

The main objective of this work is to structure the information about the COTS marketplace; therefore, the proposed solution is related to define a precise method for the construction of COTS taxonomies. We have formulated a goal-oriented method called GBTCM (Goal-Based Taxonomy Construction Method) [Ayala-Botella-Franch2005a].

The GBTCM method was inspired in GBRAM. (Goal-Based Requirements Analysis Method) [AntonThesis]. GBRAM is a widely recognized methodology in the context of Software Engineering. It helps to the Requirements Engineering field to discover, identify, and organize goals for their subsequent translation into a Software Requirements Document (in section 3.6.2, the method is presented in detail). GBRAM provides a helpful framework for integrating the goal-oriented mechanisms and techniques we need for formalizing the construction of taxonomies from goals (see section 5.1).

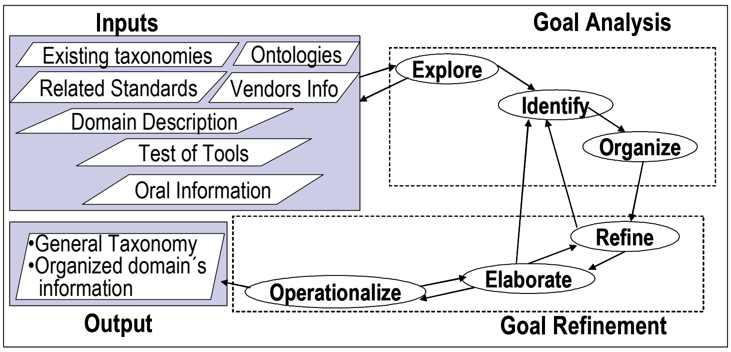


Figure 2.1. Proposed method activities

In Fig. 2.1 we can see the artifacts and activities involved in the proposed method, which are:

- ***Inputs***. Refers to gather information available, considering the process of analysis of information sources that permits to take into account their qualities and factors for classifying them according to their relevance for the simultaneous consideration of issues needed in the COTS context.

- ***Goal Analysis.*** Concerns the exploration of available information sources for goal identification followed by the organization and classification of goals. The goal analysis activities can be summarized as follows:

- ***Explore***. Entails the examination of the *Inputs*.
- ***Identify***. *A*ims at extracting goals applying different techniques.
- ***Organize***. Involves mechanisms for identifying goal dependency relationships and COTS dependency relationships for the classification and organization of goals.

- ***Goal Refinement.*** Concerns the evolution of goals from the moment they are first identified to the moment they are translated into taxonomies. The goal refinement activities are summarized as follows:

  - ***Refine***. *E*ntails the matching of goals from different information sources and the actual pruning of this goal set.
  - ***Elaborate***. *R*efers to the process of more refinement, analyzing the goal set, applying techniques for uncovering hidden goals.
  - ***Operationalize***. Refers to translating the hierarchical set of goals into a customized taxonomy.

- ***Output.*** The final result of GBTCM is not only a goal-oriented taxonomy for a specific field, but also a set of information and knowledge of the domain (repository of the domain) which are the basis for the customized taxonomies construction and maintenance.

Thus, the proposed method intends to guide the gathering and analysis of sources of information, modeling requirements and domains, as well as organizing the knowledge in any segment of the COTS marketplace for their translation to goals and goal-based taxonomies.

The main challenges of the method are related to the COTS problematic context that requires careful consideration, such that this research is addressed to provide the method with guidelines and lineaments for:

- Gathering adequate information taking into account the relevant factors in the COTS framework (for instance, which sources –*Inputs*- should be evaluated, and insights of their recommended use depending on several qualities of the information).

- Performing a goal-based reasoning as a framework for guiding the use of adequate techniques to conduct the stated activities of the method. It means to guide the use of adequate mechanisms and techniques for acquisition, organization, and refine of goals in specific situations, with the primary focus on the construction of goal-oriented taxonomies as decision trees.

- Validating strategies, as *transformation rules* for managing and manipulating goals depending on the users needs for the construction of customized taxonomies (it means to *operationalize* goals to taxonomies). Given correctness and completeness notions, *transformation rules* are defined for arranging goal-oriented hierarchies customizing them to the user needs preserving their correctness and completeness properties.

- As a result of the method, a "goal-mining" and knowledge base of the domain are obtained which are the support not only for an easy evolution of taxonomies, but also their suitability and its dynamic use (i.e. a taxonomy should be constructed depending on the users needs)

Fig. 2.2 shows an overview of this process, which is explained in detail in next sections.

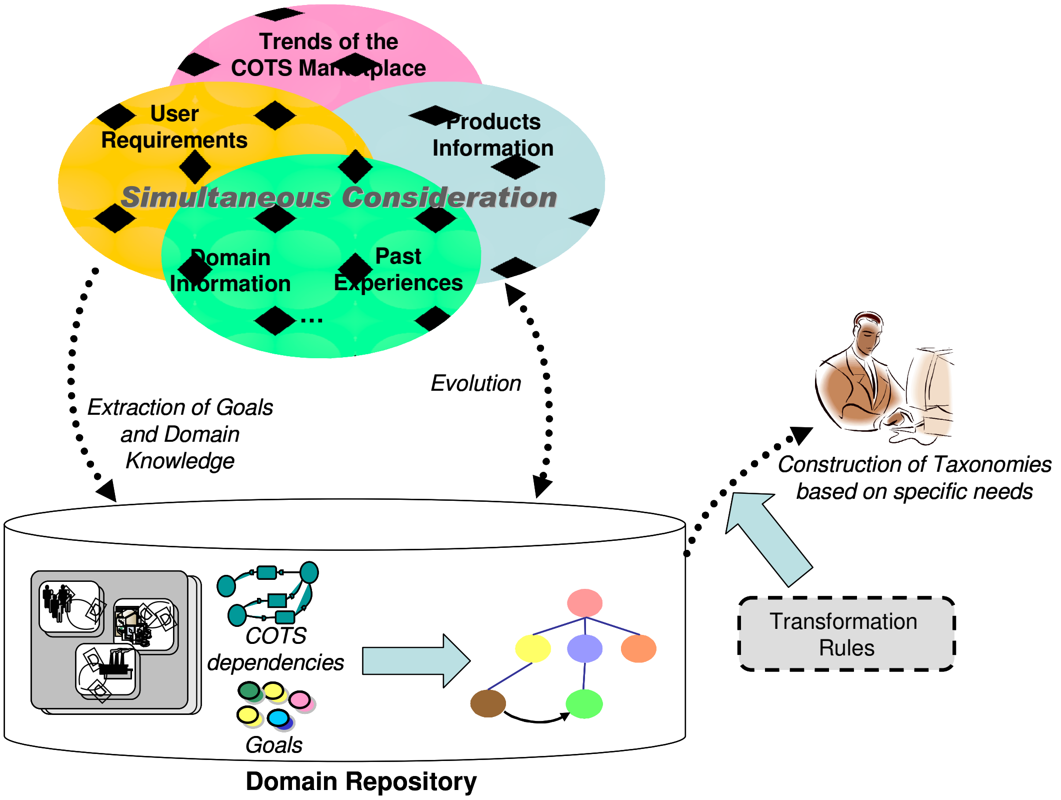


Fig. 2.2 Overview of the proposal

3. State of the Art

3.1. COTS Selection Processes

The use of COTS is perceived as a means to deliver large, complex systems, potentially with significant improvements in quality, development time, and cost. However, as it was mentioned in a previous section, the issues involved with developing CBS present both opportunities and risks.

Thus, effective use of COTS inevitably requires a well-disciplined systematic methodology that would facilitate the exploitation of the benefits of COTS including in a special manner the activities of selection and evaluation.

Nowadays there is a significant body of work that identifies issues or proposes frameworks for pursuing further research for improving new software development processes for CBS.

For instance, in [Brownsword-et-al.2000] the changes required to address the CBS development are identified taking into account real-life lessons and then articulate a framework for organizing the new and changed process elements. In [Morisio-et-al.2000] a report is done about adopted COTS-based processes and proposes a new one.

In a COTS-based development process, early evaluation and selection of candidate COTS components is one of the key aspects of the system development lifecycle.

> *COTS software component selection is a process of determining "fitness for use" of previously-developed components that are being applied in a new system context*
>
> G. Haines, D. Carney, J. Foreman. Software Engineering Institute, 1997.
> http://www.sei.cmu.edu/cbs

The selection of suitable COTS products is often a non-trivial task and requires careful consideration of multiple criteria [Ncube-Maiden1999], above all considering that a marketplace of competing products exists.

3.1.1. Phases of the COTS Selection Process

According to [Kunda-Brooks1999], there are three phases of COTS software selection:

Evaluation criteria definition: The criteria definition process essentially decomposes the requirements for the COTS into a hierarchical criteria set. The criteria include component functionality (what services are provided), other aspects of a component's interface (such as the use of standards) and quality aspects that are more difficult to isolate, such as components reliability, predictability, and usability.

Identification of candidate components: The identification of candidate components, also known as alternatives identification, involves the search and screening for candidate COTS that should be included for assessment in the evaluation phase. This thesis proposal is mainly involved with this issue. In general, evaluating and analyzing all the relevant characteristics of COTS candidates takes a great amount of time, typically more than the organization has. Therefore, it is both necessary and cost-effective to select the most promising candidates for detailed evaluation.

Evaluation of the criteria for these candidates: There are currently three strategies to COTS evaluation: progressive filtering, keystone identification and puzzle assembly. In keystone strategy, products are evaluated against a key characteristic such as a vendor or type of technology. Progressive filtering is a strategy whereby a COTS product is selected from a larger set of potential candidates, in which products that do not satisfy the evaluation criteria are progressively eliminated from the products list. In the puzzle assembly model, a valid COTS solution will require fitting the various components of the system together.

3.1.2. General Considerations in a COTS Selection Process

In [Alves-Castro2001] four main dimensions should be considered in a COTS selection process:

- **Domain Coverage**- The components have to provide all or part of the required capabilities, which are necessary to meet core essential customer's requirements.

- **Time restriction-** Software companies usually operate in a very rigid development schedule, on which their competitiveness depends. Selection is a time consuming activity, where a considerable amount of time is necessary to search and screen all the potential COTS candidates.

- **Cost rating-** The available budget is a very important variable. The expenses when selecting COTS products will be influenced by factors such as: license acquisition, cost of support, adaptation expenses, and maintenance prices. Boehm [Boehm-et-al.1998] provides an economic model for estimating the cost of COTS-based system development.

- **Vendor guaranties**- An important aspect to be considered in the selection activity is to verify the technical support provided by the vendor. Some issues have to be taken into account, for example: vendor reputation and maturity, number and kind of applications that already use the COTS, clauses characteristics of the maintenance licenses.

As work has progressed in the area of COTS research, specific solutions have been proposed to address some of these issues. In this thesis, we consider approaches that offer technical solutions that are applicable to most of the COTS processes, mainly to COTS selection activity.

3.1.3. Current Proposals Involved with COTS

A range of COTS-based development methods has been proposed for dealing with COTS selection [Kontio1996], [Maiden-Ncube98], [Burgués-et-al.2002], [Briand1998], [Hissam-et-al.2002], [Kunda-Brooks1999], [Morisio-et-al.2000], [Morisio-Tsoukias1997], [Seacord-et-al.2001], [Chung-Kooper2004].

For instance, the OTSO (Off-The-Shelf Option) Method [Kontio1996] that provides specific techniques for defining evaluation criteria, comparing the cost and benefits of alternative products, and consolidating the evaluation results for decision-making. The definition of hierarchical evaluation criteria is the core task in this method. It defines four different subprocesses: search criteria, definition of the baseline, detailed evaluation criteria, weighting of criteria. Even though OTSO realizes that the key problem in COTS selection is the lack of attention to requirements and market aspects, the method does not provide or suggest any effective solution. The method assumes the requirements exist; they are used to define the evaluation criteria.

Another work presented in [Kunda-Brooks1999] describes the STACE (Social-Technical Approach to COTS Evaluation) framework, an approach that emphasizes the social and organizational issues to COTS selection process. The main limitation of this approach is the lack of a well-defined process of requirements acquisition/modeling. Moreover, the STACE does not provide a systematic analysis of the evaluated products using a decision-making technique.

The PORE (Procurement-Oriented Requirements Engineering) [Maiden-Ncube1998], [Maiden-Kim-Ncube2002] technique is a template-based approach supporting iterative evaluation and selection of COTS components. The PORE process model identifies four goals: (a) acquiring information from stakeholders, (b) analyzing the information to determine if it is complete and correct, (c) making the decision about product requirement compliance if the acquired information is sufficient, and (d) selecting one or more candidate COTS components. In it, the elicitation of features of existing COTS software and requirements engineering are conducted in parallel, and it is selected the COTS component that almost exactly fits the requirements. The method uses an iterative process of requirements acquisition and product evaluation. Although the PORE method includes some requirements acquisition techniques, it is not clear how requirements are used in the evaluation process and how products are eliminated.

Most of current methods for COTS selection have not adequately treated non-functional requirements. It is worth noting that the role of non-functional requirements becomes more important because COTS components have their functionality already built-in [Carvallo-Franch-Quer2003].

The CARE (COTS-Aware Requirements Engineering) process [Chung-Kooper2004] draws upon the ideas of current methodologies including RUP [RUP] and PORE [Maiden-Kim-Ncube2002], trying to complement them in order to provide a requirements engineering methodology that is agent and goal-oriented that explicitly supports the use of COTS components. For each COTS component, they capture: their goals (soft goals and hard goals) and their detailed specification including: unique identifier, type, name, list of keywords and weights (used for keyword and case based searching); functional overview, domain, vendor, vendor evaluation, version number, operating system requirements, standards compliance, interface, performance, security, availability, reliability, processor requirements, primary memory requirements, disk space requirements, bandwidth requirements, related subcomponents, history, lessons learned, assumptions, and notes. All these descriptions are information like that found on marketing brochures for existing products. These general descriptions are used to determine if the product appears to be potentially useful.

This methodology departs from the description of the COTS components stored and maintained in a knowledge base, or repository. As the goals are defined, analyzed, and negotiated, the RE searches the repository for COTS components that appear to be (possible) matches. As the system develops, goals are defined into requirements and the RE determines if the identified components still seem suitable or if other components seem to be a better fit.

Table 3.1 Summary of methodologies dealing with COTS selection.

Methodologies or Process	Takes into account Requirements and COTS-Marketplace	Reuse of knowledge	Discriminative criteria definition	Mechanism for identification of alternatives	Facility of searching the "set of COTS to be evaluated"
CARE	-	√	√	*	*
OTSO	-	-	√	-	-
PORE	-	-	*	-	-
STACE	√	-	√	-	-

(√) addresses the issue fully (*) deals with the issue but not fully (-) does not deal with the issue

In spite of this methodology pretends to improve reuse by means a repository of COTS descriptions, it present 3 obvious disadvantages: (1) it is not clear how to build this repository; (2) the maintainability of the repository is specially difficult taking into account the rapid evolve of the information of each product and the increasing amount of products in the market and they do not consider this aspect (3) the searching process is not very efficient having to look for components in a widespread range of descriptions.

3.1.4. Remarks

Although the mentioned processes and methodologies deal with the evaluation of COTS components for choosing the most suitable to the requirements, none of them provides adequate mechanisms for helping the identification of candidate components. Nevertheless, many of them explicitly use a "set of components to be evaluated", but they do not specific how to obtain this set of components. It is clearly a gap between the methodologies for selection COTS components and its real application (Table 3.1 shows a summary of the most known methodologies).

Therefore, we strongly believe that it is crucial to develop a mechanism for structuring and reuse the COTS marketplace and knowledge. Their existence would be a basis for the current methodologies for searching components "to be evaluated" into the right segment of COTS marketplace with a high probability of success. It means a significant reduction on time required in the evaluation process and the implicit reuse of the knowledge.

In next section, a survey of the existent proposals for classifying COTS components is presented and we remark why they are weak to solve the explained problematic.

3.2. Arranging COTS Products

As it was mentioned before, among the main activities involved in COTS-based development, identification and selection are of the most critical. One way to facilitate the COTS components selection is the reuse of knowledge about them.

It is well-known that one of the essential problems in reusing software components is locating and retrieving them from a large collection [Prieto-Diaz1987].

Put simply, for reuse to be successful it must be easy to locate COTS with the same or similar functionality. It means classifying them into classes of similar items and to make comparisons in the context of each class. If the collection is organized by attributes that define software requirements, the probability of retrieving nonrelevant components is reduced. In this sense, the process of identify COTS with specific characteristics would be a more feasible task, not only requiring less time but also reducing the amount of COTS that could be evaluated.

The purpose of classifications is to organize the body of knowledge in a field. A well-defined classification structure is essential to the design of an effective retrieval system. However, a classified collection is not useful if it does not provide the search-and-retrieval mechanisms to use it.

Next subsections cite different general ways of organizing information and the current works of classification in the COTS field.

3.2.1. General Ways of Organizing Information

An important part of the knowledge of the people is the categories which they use.

The Kelly approach [Kelly1995] assume that people make sense of the world by categorizing it, and that people can describe their own categorization of the world with reasonable validity and reliability.

There are many ways of organizing information (or things or animals) into categories. In this context, the terms *taxonomy*, *typology*, *ontology*, *directory*, *cataloging*, *categorization* and *classification* are often confused and used interchangeably. In fact, all are ways of organizing information. For clarity, some relevant terminology is defined here [ClassSoc]:

- ***Categorization**:* is the process of associating a document with one or more subject categories. So the entry for a page on cross trainer shoes could go into Running, Manufacturing, and Sports Medicine. All of these are legitimate, depending on the context.

- ***Typology****:* is the study or systematic classification of types that have characteristics or traits in common

- ***Cataloging*** and ***Classification****:* come from libraries, where specialists enter the metadata (such as author, date, title and edition) for a document, apply subject categories to it, and place it into a class (such as a call number) for later retrieval. These tend to be used interchangeably with Categorization.

- ***Directory:*** is an organized set of links, like those on Yahoo or the Open Directory Project, which allows a web site to display the scope and focus of its content. A directory can cover a single host, a large multi-server site, an intranet or the Web. At each level, the category names provide instant context information to users. Rather than a simple list, such as the results of a search, drilling down into the more and more specific categories (for example Shopping>Clothing>Footwear>Athletic) explains how the pages fit into the larger set of information.

- ***Clustering****:* is the process of grouping documents based on similarity of words, or the concepts in the documents as interpreted by an analytical engine. These engines use complex algorithms including Natural Language Processing, Latent Semantic Analysis, Bayesian statistical analysis, and so on.

- ***Thesaurus:*** is a set of related terms describing a set of documents. This is not hierarchical: it describes the standard terms for concepts in a *controlled vocabulary*. Thesauri include synonyms and more complex relationships, such as broader or narrower terms, related terms, and other forms of words.

- ***Taxonomy****:* is the organization of a particular set of information for a particular purpose. It comes from biology, where it's used to define the single location for a species within a complex hierarchic. Biologists have arguments about where various species belong, although DNA analysis can resolve most of the questions. In informational taxonomies, items can fit into several taxonomic categories.

- ***Ontology****:* is the study of the categories of things within a domain. It comes from philosophy and provides a logical framework for academic research on knowledge representation. Work on ontologies involves schema and diagrams for showing relationships in Venn diagrams, trees, lattices, and so on.

On the basis on the differences of classification definitions showed above, in this work, the concepts of taxonomy and ontology are used for categorizing COTS of specific domains.

In fact, formally, an ontology consist of terms, their definitions, and axioms relating them [Gruber1993]; terms are typically organized in a taxonomy. This is where some disagreement among ontology researches arises. Some say that axioms are central to ontology design, and a complete, or high level, taxonomy does not even have to exist (maybe only for visualization) [vanderVet-et-al.1994].Others say that one should first concentrate on defining a taxonomy of fundamental concepts (although they agree there should be axioms or knowledge in some other form associated with the concepts in the taxonomy) [Winston-et-al.1987].

In this proposal, the second approach is used, and we concentrate our research on defining goal-oriented domain taxonomies based on goal similarities. Definitions of concepts of the domain as well as detected relationships among COTS are supporting these taxonomies.

These definitions represent a common vocabulary and conceptual framework to categorize and compare existent COTS in the marketplace, due to this, these taxonomies could be considered an ontological approach to its specific framework of use. Because of that, it was mentioned that this research is addressed to rationale behind the construction of such taxonomies. Such information and taxonomy at their turn could be the basis and the main backbone for manipulating and constructing customized taxonomies for specific purposes; it means they can be “fleshed” with the relevant addition or removing of attributes and other relations among nodes.

From the technical perspective of arranging COTS as taxonomies, as it was proposed, to find the categorizers we need for discriminating elements in the taxonomy is not so easy. The specificity, or granularity, of the classification, and the degree to which the topics are organized can vary. Not all types of categorization are appropriate or necessary for every situation. The type of categorization to choose is dictated by its use, the size of the collection, its depth and breadth, the availability of labor, the rate of change and the number of new items that are being added, as well as the number of users and whether they need immediate access to the latest information.

The process of finding classes for classifying is at least as important as the resulting classification. This topic is related in Section 3.3.

Next, a survey of the currents components classification approaches is presented as well as the analysis of why they do not cover the solution to the addressed problem.

3.2.2. Current Components classification approaches

As it was mentioned before, there is an increasing need for arranging the types of available COTS to improve the efficiency and reliability of related processes with different perspectives and contexts.

From IT consultant companies like Gartner [Gartner], professional societies as INCOSE [Incose], commercial web-based companies as ComponentSource

[CompSource] to the academic world, organizations, teams and individuals have presented their own proposals.

In this last context, many works deal with classification of components proposing different classification mechanisms as [Prieto-Diaz1987] that propose a general faceted schema or [Glass-Vessey1995] that survey taxonomies of application domains, this work emphasizes the confusion present in existing taxonomies between problem-oriented criteria and solution-oriented ones.

It is well-known that the effort devoted to these activities is more valuable if the attributes can be reused. Since the evaluation of COTS is a very project-specific activity, the definition of reusable attributes is difficult.

There is a wide range of works about COTS attributes in terms of how the attributes should be defined. For example: Iusware [Morisio-Tsoukias1997], RCPEP [Lawlis-et-al.2001] and OTSO [Kontio1996] approaches, which advocate redefining the attribute framework every time based on the requirements. Their motivations are:

- Evaluation is different from simple measurement since it is driven by a well-defined goal
- The factors that influence the choice of attributes depends on the requirements therefore they vary at each project.

Other works are more addressed in how the attributes can be reused. For instance: COTS Acquisition Process (CAP) [Ochs-et-al.2001], the approach proposed by Boloix et. al [Boloix-Robillard1995], and the proposal of the eCOTS [eCOTSproject] and CLARiFi [CLARiFiProject] projects. They define a set of generic attributes that can be reused across projects and possibly organizations. Their motivations are:

- COTS evaluation is cost effective only if its outcomes can be reused,
- Using a fixed framework allows evaluating a wider set of products and continuous improvement

Several recent works characterize COTS products by means of attributes with a different purpose that is the use of attributes for identifying relationships between characteristics of products and their impact on CBS.

Carney and Long [Carney-Long2000] propose the classification of COTS products using a bi-dimensional Cartesian space and report some examples that populated this space. The dimensions they define are origin and modifiability.

The origin dimension addresses the way the product is produced and they propose the following possible values: Independent Commercial Item, Special Version of Commercial Item, Component Produced by Contract, Existing Components from External Sources, Components Produced In-house. These values can describe products ranging from completely non off-the-shelf products developed on purpose to commercial components ready to use with a large number of customers.

The modification dimension describes to which extent the product either can or must be modified by the system developer that uses the component. This attribute has five possible values: Extensive Reworking of Code, Internal Code Revision, Necessary Tailoring and Customization, Simple Parameterization, Very Little or no Modification. Two of them assume access to code (extensive reworking, internal code revision), two (necessary tailoring, parameterization) imply some mechanisms built into the COTS to modify its functionality.

Morisio and Torchiano [Morisio-Torchiano2002] extend the work proposed in [Carney-Long2000], proposing a classification framework for COTS products. They depart from the idea that different research works often adopt different implicit definition of COTS, thus making difficult comparing them and evaluating the applicability of proposed approaches. In addition, the term COTS applies to a broad range of products, which exhibit different issues [Carney-Long2000]. The purpose of their framework is twofold: first it is a way to precisely define what is the meaning of COTS, second it represents a way of specifying which sub-classes of products are addressed by a given work. The aim of their work is mainly classification. This proposal is similar to [Jaccheri-Torchiano2002] that emphasizes the assessment of the reuse of attributes.

Hence, the proposals cited before do not resolve completely the problem of classifying COTS products not only for the search-and-retrieval mechanisms from a huge and changing COTS market but also for recording COTS dependency relationships.

It is important to stand out the insights for classifying exposed by our research group in [Carvallo-et-al.2004a] from which we initiate this classification research project. These ideas were oriented to arrange components as a decision tree by means of “characterization attributes” (this concept was taken from [Morisio-Torchiano2002], [Jaccheri-Torchiano2002]) to discriminate among different categories (nodes) and domains (leaves). Also dependencies among domains that belong to the taxonomy were included in the hierarchy. However, the research performed there was in the context of quality models reuse, and the rationale behind was not rigorous enough. Therefore, they have key inconvenients among which we can mention: a) the taxonomy was specific to Business Applications domain, b) it turned out that the way to identify the discriminating characterization attributes (which capture the relevant information for discriminating categories and domains) of the COTS framework was not defined; c) and also, it was not defined how to deal with all information related in a COTS perspective because it departs from an existent taxonomy which was only restructured.

Thus, all of these proposals share a common characteristic that may be considered as a drawback: their categorization relies on experience, knowledge, and observation but they rarely use knowledge engineering and requirements engineering techniques to classify the enclosed items. Sometimes, the meaning of a particular category is not clear without further examining the items, especially if it is absolutely unknown to the user. As a result, the understanding, use, evolution, extension, and customization of the categorization proposal may be difficult.

Therefore, it is a fact that there are not enough studies in depth about the identification of properties that can help to organize these taxonomies. Above all, 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 arrange it and how the taxonomy can be searched (this is especially true when considering not just the construction of the taxonomy, but its evolution).

In addition, it is important to remark that these proposals do not address at the same time and in a suitable manner many aspects that influence the evolution and construction of COTS taxonomies.

3.2.3. Remarks

The proposals cited before do not solve the problem of classifying COTS not only for the search and retrieval mechanism from a huge and changing marketplace but also to improve the reuse of knowledge in a domain that requires multiple considerations.

It is worth to say that our thesis proposal departs from the insights of classification exposed by our research group in [Carvallo-et-al.2004], pretending to fulfill the gap among the practical need of having taxonomies and the rationale behind its construction, by means of a method supported by engineering techniques and not only from the experience.

3.3. Classification Modelling

Data classification is a two-step process [Han-Kamber2001]. In the first step, a model is built describing a predetermined set of data classes or concepts. The model is constructed by analyzing *objects* (or *samples* or *examples*) described by attributes. Each object is assumed to belong to a predefined class, as determined by one of the attributes, called the *class label attribute*. The objects analyzed to build the model collectively form the *training data set*. The individual samples making up the training set are referred to as *training samples* and are randomly selected from the sample population. Since the class label of each training sample is provided, this step is also known as *supervised learning* (i.e. the learning of the model is "supervised" in that it is told to which class each training sample belong). It contrasts with unsupervised learning (or clustering), in which the class label of each training sample is not training, and the number or set of classes to be learned may not be known in advance.

Typically, the learned model is represented in the form of classification rules, decision trees, or mathematical formulae.

In the second step, the model is used for classification. First, the predictive accuracy of the model (or classifier) is estimated using methods. The holdout method is a simple technique that uses a test set of class-labeled samples. These samples are randomly selected and are independent of the training samples. The accuracy of a model on a given test set is the percentage of test set samples that are correctly classified by the model. For each test sample, the known class label is compared with the learned model´s class prediction for that sample.

If the accuracy of the model is considered acceptable, the model can be used to classify future objects for which the class label is not known.

3.3.1. Classification Procedure

1. **Training a Classifier**: The classification algorithm is applied upon the training dataset. It generates a "piece of code", which has learned the properties that characterize the objects belonging to each class, according with the training dataset. This piece of code is the *classifier*.

2. **Tuning the Classifier**: The classification algorithm is applied upon the second dataset, so that the classifier generated in the previous step learns to ignore the properties of the training data set that are peculiar to these objects only.

3. **Evaluating the Classifier**: The classification algorithm is applied upon the test dataset, making class membership information visible to the classifier. The performance of the classifier is measured on how well it can guesses the class membership of the objects in this dataset.

3.3.2. Classification by Decision Trees

A decision tree is a flow-chart-like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and leaf nodes represent classes or class distribution. The top-most node in a tree is the root node.

Their main characteristics are:

- They are classifiers such that the root represents the whole population of the mini-world that is being classified,
- A node represents a subset of the population, which is more homogeneous with respect to its members than the parent node.
- The branches emanating from a node represent a partitioning of the member of that node.
- The contents of a node are described by attributes and value ranges

A typical decision tree is shown in Fig. 3.1. It represents the concept buys_computer indicating whether or not a customer is likely to purchase a computer. Each internal (nonleaf) node represents a test on an attribute. Each leaf node represents a class (either buys_computer=yes or buys_computer=no). That is, it predicts whether or not a customer is likely to purchase a computer. Leaf nodes are denoted by ovals.

In order to classify an unknown sample, the attribute values of the sample are tested against the decision tree. A path is traced from the root to a leaf node that holds the class prediction for that sample. Decision trees can easily be converted to classification rules.

Decision trees have been used in many applications areas ranging from medicine to game theory and business. They are the basis of several commercial rule induction systems.

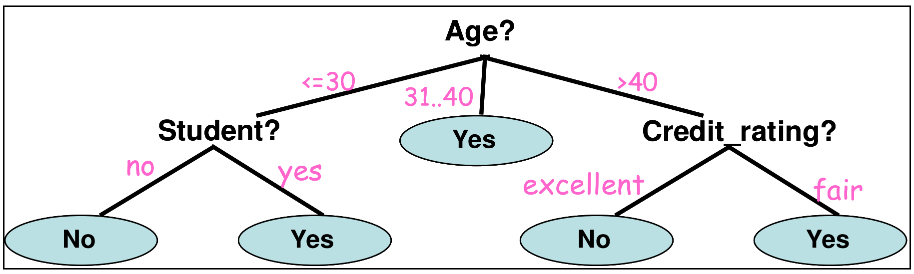


Fig. 3.1. A decision tree for the concept buys_computer

The abstract mechanism for conventional Decision Trees construction is:

- The decision tree is built recursively, starting at the root.

- At each node, we consider each classification variable and look for value(s) that split the node´s population into more homogeneous partitions.

- Among all the candidates, we select the one that makes the “best” partitioning.

- We branch the node creating as many children as the values we consider.

- We invoke the algorithm for each new child, until the population at the node is so homogeneous that it can not be split any further.

There are many algorithms for decision tree induction, among we can mention ID3 [ID3], C4.5 [C4.5] and CART [CART].

3.4. Classification Problem Solving

Since the essential role of problem solving methods is to support reuse, we tackle it in our research for addressing our intended objectives.

Classification problem solving has been thoroughly analysed in the literature. The basic heuristic classification pattern was first identified by [Clancey1985]. [Stefik1995] provides a very good overview of classification problem solving in the context on knowledge systems. In [Wielinga-et-al.1998] their analysis of classification problem solving is carried out in the context of illustrating their competence theory for Knowledge Based Systems development.

The classification problem has also been considered by researches in the case-based reasoning area [Althoff-et-al.1994].

In general, research on problem solving methods [Benjamins-Fensel1998] and ontologies [Gruber1995] aims (among other things) to identify and formalise classes of generic components and reasoning patterns, which can be reused across different domains to support the robust development of knowledge-based systems.

A problem solving method describes a generic reasoning behaviour which can be reused in different applications. For instance, Fig. 3.2 shows the flow of inferences in the *heuristic classification model* defined by [Clancey1985], which can be used to solve classification and diagnostic tasks. According to this model, these tasks can be solved by a sequence of *abstraction*, *matching* and *refinement* operations, by which the available data are abstracted (for instance from a reading of the temperature of a patient we can infer that such temperature is high or low), matched to possible explanations (say, the patient has fever) and these can then be refined (e.g., the final diagnosis could be 'glandular fever').

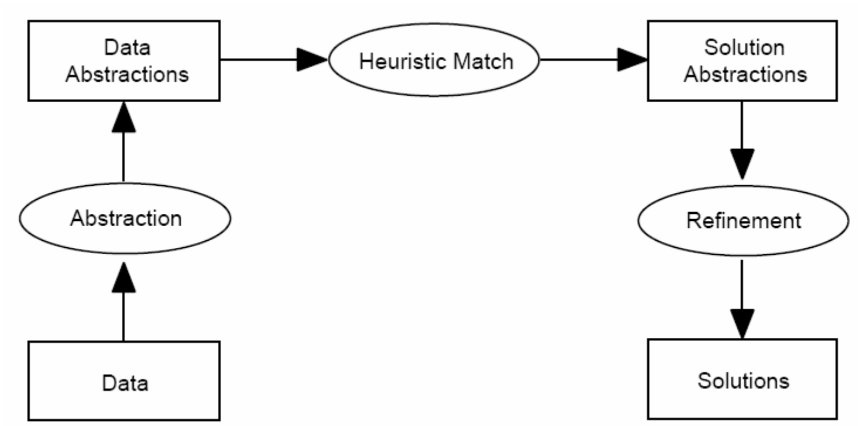


Fig. 3.2. Clancey´s heuristic classification model.

As we mention before, the essential role of ontologies and problem solving methods is to support reuse. Hence, not surprisingly, a number of libraries have been developed over the last years. Some of the libraries focus on problem solving methods, some focus on ontologies; and some include both ontologies and problem solving methods.

The rapid growth in the quantity, quality and size of these repositories attest to the growing importance and maturity of the field.

Ontologies are being developed in a variety of domains to support all sorts of computer-based activities, including intelligent internet searches [Guarino-et-al.1999], knowledge acquisition [Motta-et-al.2000] and system development [vanHeijst1995], [Motta1999]. Several success stories have been reported in the literature. Analogously, success stories have also been reported about the application of problem solving methods to industrial problems.

Equally importantly, advances in knowledge modelling research make it possible to ground knowledge system development on a more solid theoretical and engineering basis, thus aligning knowledge engineering with other engineering disciplines [Schreiber-et-al.2000].

3.4.1. Modelling Classification Problems

As [Stefik1995] explains, "to classify something... is to identify it as a member of a known class". Hence, classification can be seen as the problem of finding the *solution* (class) which best explains a certain set of known facts (*observables*) about an unknown object, according to some *criterion*. In the following we will discuss each of these concepts in turn.

- ***Observables*:** As said above, we use the term 'observables' to refer to the known facts we have about the object (or event, or phenomenon) that we want to classify. Each observable can be characterized as a pair of the form <f, v>, where f is a feature of the unknown object and v is its value. Here, we take a very generic viewpoint on the notion of feature. By feature we mean anything which can be used to characterize an object, such that its value can be directly observed, or derived by inference. We assume that each feature of an observable can only have one value. This assumption is only for convenience and does not restrict the scope of the model.

- ***Solution space*:** The solution space specifies a set of predefined classes (solutions) under which an unknown object may fall. A solution itself can be described as a finite set of *feature specifications*, which is a pair of the form $<f, c>$, where f is a feature and c specifies a condition on the values that the feature can take. Then, we can say that an observable (f, v) matches a feature specification (f, c) if v satisfies the condition c.

- ***Criteria for classification tasks***: generally speaking, classification can be characterized as the problem of explaining observables in terms of pre-defined solutions. To assess the explanation power of a solution with respect to a set of observables we need to *match* the specification of the observables with that of a solution. Given a solution, *sol*: $((f_{sol1}, c_1).....(f_{solm},$

c_m)), and a set of observables, *obs*: ((f_{ob1}, v_1).....(f_{obn}, v_n)), four cases are possible when trying to match them:

- A feature, say f_j, is *inconsistent* if (f_j, v_j) $\in$ *obs*, (f_j, c_j) $\in$ *sol* and v_j does not satisfy c_j;
- A feature, say f_j, is *explained* if (f_j, v_j) $\in$ *obs*, (f_j, c_j) $\in$ *sol* and v_j satisfies c_j;
- A feature, say f_j, is *unexplained* if (f_j, v_j) $\in$ *obs* but f_j is not a feature of *sol*.
- A feature, say f_j, is *missing* if (f_j, c_j) $\in$ *sol* but f_j is not a feature of *obs*.

Given these four cases, it is possible to envisage different solution criteria. For instance, we may accept any solution which explains some data and is not inconsistent with any data. This criterion is called *positive coverage* [Stefik1995]. Alternatively we may require a *complete coverage* - i.e., a solution is acceptable if and only if it explains all data and is not inconsistent with any data. Thus, the specification of a particular classification task needs to include a so*lution (admissibility) criterion*. This in turn relies on a *match criterion*, i.e., a way of measuring the degree of matching between candidate solution and a set of observables.

Classification involves searching the space of possible solutions to find one which explains the given observables in accordance with the given solution criterion. The choice of a method depends on many factors, such as whether the goal is to find one, all or the best solution; whether all observables are known at the beginning or are uncovered opportunistically (maybe at some cost) during the problem solving process; whether or not the solution space is structured according to a *refinement hierarchy*; whether solutions can be composed together, or alternatively, whether each solution presents a different, self-contained alternative.

If we restrict to a scenario in which observables are not acquired incrementally during the classification process, then only a *data-directed* approach can be taken. That is, from the given set of observables we identify the most promising solutions and, in those cases where a refinement hierarchy is provided, we can then search the latter to home in on the most appropriate solutions. A data-directed approach does not however imply that the set of observables remain the same during the problem solving process. In many cases we can have *abstraction mechanisms*, which generate new observables from existing ones, in accordance with Clancey's model shown in Fig. 3.2.

Simpler problem solving methods can simply be defined as 'degenerations' of the heuristic classification model (eg., these may not include abstraction and/or refinement mechanisms). More complex problem solving methods will extend this model with additional inference mechanisms.

3.4.2. Heuristic Classification Model

The heuristic classification model includes two notions:

- *Abstractors*: Data abstraction consists of inferring new observables from existing ones by means of domain knowledge. We model this kind of knowledge by means of abstractors. Formally, an abstractor is a function which takes a set of observables as input and produces an observable as output. Abstractors implicitly introduce a hierarchy structure in the data space: directly observed data are at the lowest level of the hierarchy, higher level data are obtained by applying abstractors to lower level data.

- *Refiners*: Solution refinement consists of taking a solution as input and generating more specialised ones (i.e., solutions which cover a subset of the phenomena explained but the 'parent' solution). In this process, domain knowledge about the solution is used. We model this kind of knowledge by means of *refiners*. Refinement can be seen as the inverse process to abstraction, rather than going from concrete data to abstract ones, here we go from abstract solutions to more concrete ones.

3.5. Knowledge and Requirements Acquisition Techniques

Requirements engineers face a major problem when choosing methods for acquiring requirements of software systems. The problem is not that there is a lack of methods. Rather, no guidance is available to choose methods, to plan a systematic, well grounded acquisition program, or even to sequence methods. Indeed, many requirements engineers are unaware of the methods that are available [Maiden-Rugg1996].

Requirements engineers are familiar with methods such as observation, interviewing and using documentation. However, it is a fact that more than one acquisition method is needed to capture the full range of complex requirements for most complex software-intensive systems.

To complement familiar methods, some researches posit use of ethnographic methods [Sommerville-et-al.1993], [Goguen-Linde1993]. Such methods have received considerable recent interest in requirements engineering. Knowledge engineering has also provided useful methods, such as card sorts and laddering [Rugg-et-al.1992]. Indeed, requirements engineering and knowledge engineering share many concerns. So, Requirements Acquisition (RA) field has paid considerable attention to the Knowledge Acquisition (KA) literature, and has also imported techniques from other disciplines. [Rugg-McGeorge1997].

Although it is clearly essential to choose the correct technique for a task, and to use it correctly, there is surprisingly very little guidance on this in the literature. There is also a scarcity of readily available tutorial papers on the individual techniques; most of the literature on individual techniques assumes previous knowledge of the technique being discussed.

The first published work offering a guidance in selection RA techniques was ACRE (ACquisition of REquirements) by Maiden and Rugg in 1996 [Maiden-Rugg1996], they present an integrated framework for assisting to requirements engineers in choosing methods for RA. The method selection is achieved using questions driven from a set of facets which define the strengths and weakness of each method.

ACRE research has identified six facets which inform method selection:

- Purpose of Requirements: requirements can be acquired for different purposes, such as specification of bespoke systems, selection of components and to provide a legal contract for requirements procurement. Different methods assist each.

- Knowledge types: requirements modeling languages include semantic primitives such as events, states, and agents. Different methods acquire different types of knowledge.

- Internal Filtering of Knowledge: it is often the case that stakeholders are unaware of their own knowledge and its boundaries. Problems can include poor recall and communication incomplete or incorrect knowledge. Methods are offered to overcome these limitations.

- Observable Phenomena: some knowledge can not be communicated by stakeholders but only learned by observing a system and its environment.

- Acquisition Context: method choice also depends on the context of its use. Acquisition does not occur in a vacuum. Complex organizational, political, financial, and temporal pressures influence acquisition.

- Method interdependencies: an acquisition program will include a sequence of methods which influence method choice.

ACRE provides methods for acquiring requirements from stakeholders, rather than for mining requirements out of documents.

Then, in 1997 Rugg and McGeorge [Rugg-McGeorge1997] presented a tutorial paper on the use of sorting techniques applicable to KA and RA.

However, the use of COTS introduced new problems for requirements engineers, including when to acquire new customer requirements and when to reduce the number of candidate products. In this sense, departing from ACRE, the PORE (Procurement-Oriented Requirements Engineering) method proposed by Maiden, Kim and NCube [Maiden-Kim-Ncube2002], includes some templates for requirements acquisition in response to lessons learned from a COTS selection exercise undertaken by the authors that suggest them the kind of techniques that could be applied. Nevertheless the templates presented in this proposal are not enough for dealing with the specific problematic of requirements and knowledge acquisition in a COTS framework, this is because in one hand they are only suggested as an integral part of the method (which is focused in the whole selection process); on the other hand they do not contemplate the information reuse and others implied considerations of the COTS framework.

3.5.1. Remarks

As a result, taking into account the lack of guidance in applying KA and RA techniques in the COTS field; it is our interest to propose guidance in the applicability of such techniques with the aim not only for acquire requirements and knowledge for building taxonomies but also to reuse them.

3.6. Goal-Oriented Approaches

Requirements Engineering (RE) research has increasingly recognized the leading role played by goals in the RE process [VanLamswerde2001]. Such recognition has led to a whole stream of research on goal modeling, goal specification, and goal-based reasoning for multiple purposes, such as requirements elaboration, verification or conflict management, and under multiple forms, from informal qualitative to formal.

A goal is an objective the system under consideration should achieve. Goal formulation thus refers to intended properties to be ensured.

A critical factor in successful projects is often that developers not only understand what they are developing, but why they are developing a given system, goal driven approaches focus on this issues.

Goals may be formulated at different levels of abstraction, ranging from high-level, strategic concerns to low-level, technical concerns.

There are many reasons why goals are so important in the RE process [VanLamswerde2001]. Their main benefits are concerned with:

- ***Achieving requirements completeness:*** Goals provide a precise criterion for sufficient completeness of a requirements specification; the 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.

- ***Avoiding irrelevant requirements***: Goals provide a precise criterion for requirements pertinence; a requirement is pertinent with respect to a set of goals in the domain considered if its specification is used in the proof of one goal at least.

- ***Explaining requirements to stakeholders:*** Goals provide the rational for requirements. Requirements appear because of some underlying goal which provides a base for it [Yu1997].

- ***Structuring Requirements:*** Goal refinement provides a natural mechanism for structuring not only complex requirements documents for increased readability but also for exploring many alternatives.

- ***Detecting Requirements Conflicts***: Goals have been recognized to provide roots for detecting conflicts among requirements and for resolving them eventually.

- ***Requirements Evolution and Traceability:*** Separating stable from more volatile information is another important concern for managing evolution.

A requirement represents one particular way of achieving some specific goal; the requirement is therefore more likely to evolve, towards another way of achieving that same goal, than the goal itself. So, goals are more stable respect to changes, and the higher level a goal is, the more stable it will be [Anton94].

- ***Identifying Requirements:*** Goals drive the identification of requirements to support them [Dardenne-et-al.1993]

- ***Driving Refinement and Abstraction***: Once a preliminary set of goals and requirements is obtained and validated, many other goals can be identified by refinement and by abstraction, just by asking HOW and WHY questions about the goals/requirements already available.

Goal-based reasoning reviews how goals are used in basic activities such as requirements elicitation, elaboration, verification, validation, explanation, and negotiation; particularly for difficult aspects such as conflict management, requirements deidealization, and alternative selection [VamLamsweerde2001].

It should be noted that the analysis of goals is not unique to software; goals are also addressed in non-computing intensive areas such as goal-based learning and strategic planning.

3.6.1. Why Goal-Based Approaches

The goal-based approaches result appropriate for handling the problem of structuring the COTS marketplace considering its rapid evolution and all the information related in a selection process. This is because: goals are more stable respect to changes in processes, and goal refinement provides a natural mechanism for structuring and exploring many alternatives

Moreover, goals capture, at different levels of abstraction, the various objectives the system under consideration should achieve.

Since a summary and comparison of several of these goal-driven approaches is available in [Green1994] and [Kavakli-Loucopoulos2004], in this section we only discuss two specific approaches for getting and representing goals that we consider helpful to our objectives.

Although goal-based methods stress the need to characterize, categorize, decompose, and structure goals as requirements, they usually fail to offer strategies to identify goals, taking it for granted that the goals have already been documented. We found that the Goal-Based Requirements Analysis Method (GBRAM) try to solve this problem and answer these questions in the Requirements Specification field. Therefore, we explore its applicability to our purposes.

On the other hand we are studying goal-oriented models that help us to represent dependencies among goals, one of the most significant objectives we pursue.

The benefits of goal modeling are to support heuristic, qualitative, or formal reasoning schemes during requirements engineering.

In the last years, the construction of goal-oriented models has become an extended practice in fields such as requirements engineering and organizational process modeling [Yu1997], [Yu-Mylopoulos1994], [VanLamswerde2001].

There exist several proposals of languages for the construction of goal-oriented models. Among them, we are interested in the *i** notation proposed by Eric Yu in the first half of the 90's [Yu-Thesis], [*i**Web] (although other options are valid). This is because *i** allows for the clear and simple statement of actor's goals and dependencies among them, and it offers a natural an easy understanding view of the domain. It also includes a graphical notation which allows for a unified and intuitive vision of the environment being modeled, showing its actors and the dependencies among them. It means that it aligns with our intended objective of represent COTS dependencies. Moreover, *i** is currently one of the most widespread notations used.

Therefore, we are working in the analysis and extension of the *i** SD notation to support the implied needs of the COTS framework. Such work performed in this described next sections.

Next, we explain a briefly overview of these approaches.

3.6.2. Goal-Based Requirements Analysis Method (GBRAM)

With the primary focus on the transformation of enterprise and system goals into requirements, GBRAM was formulated by Annie I. Antón, more specifically to assist analysts in gathering software and enterprise goals from many sources and to support the process of discovering, identifying, classifying, refining, and elaborating goals into operational requirements. We mainly study its suitability to our approach, since it assumes that goals have not been previously documented or explicitly elicited from stakeholders and that the analyst must work from existing information sources, for instance: diagrams, textual statements, policies or transcripts to determine the goals of the desired system. It means it focuses on the initial identification and abstraction of goals from all available sources of information, regardless of the scope of the knowledge base.

The method's chief contribution is the provision of heuristics and procedural guidance for identifying and constructing goals. In other words, it was developed for to identifying, elaborating, refining, and organizing goals for obtaining the software requirements document.

The high level activities of GBRAM briefly explained are:

- Goal Analysis that concerns the exploration of available information sources for goal identification followed by the organization and classification of goals.

- Goal Refinement that concerns the evolution of goals from the moment they are first identified to the moment they are translated into operational requirements for the system specification. It includes activities as refine, elaborate and operationalize of goals.

These high-level activities just described provide an overview of the GBRAM.

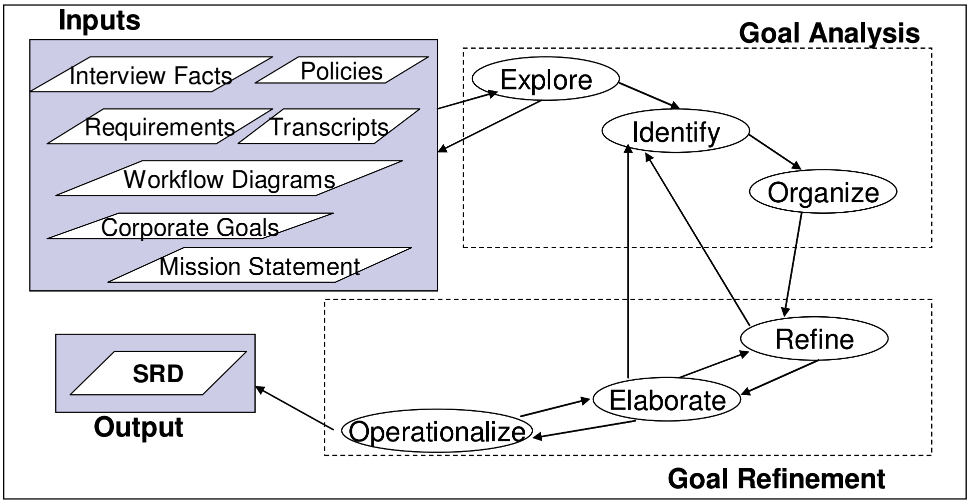


Fig. 3.3. Overview of GBRAM activities.

Fig. 3.3 shows the activities with which an analyst is intimately involved when applying the GBRAM. The ovals located within the dotted box on the upper right corner of the figure denote goal analysis activities.

The ovals within the dotted box on the lower half of the figure denote the activities that take place during goal refinement.

The box in the top left corner contains the possible inputs, which may vary in accordance with the documentation initially available to analysts. The output of GBRAM is always a software requirements document.

The suitability to apply this method to our purpose of constructing taxonomies of COTS products is explained in section 5.

3.6.3. *i** Approach

The *i** approach [Yu1997] was developed for modeling and reasoning about organizational environments and their information systems. The model facilitates the identification and specific representation of what is at stake for whom, and what impacts are likely if a dependency fails.

This approach offers a formal representation of goals and their behaviors with a formal decomposition structure, treating non-functional requirements. It consists of two models components: the Strategic Dependency Model (SD) and the Strategic Rationale Model (SR).

The central concept in *i** is that of the intentional actor. Organizational actors are viewed as having intentional properties such as goals, beliefs, abilities, and commitments. Actors depend on each other for goals to be achieved, tasks to be performed, and resources to be furnished. By depending on others, an actor may be able to achieve goals that are difficult or impossible to achieve on its own. On the other hand, an actor becomes vulnerable if the depended-on actors do not deliver. Actors are strategic in the sense that they are concerned about opportunities and vulnerabilities, and seek rearrangements of their environments that would better serve their interests.

The Strategic Dependency model is used to describe the dependency relationships among various actors in an organizational context. It is a network of dependency relationships among actors. It consists of a set of nodes that represent actors and a set of dependencies that represent the relationships among them, expressing that an actor (depender) depends on some other (dependee) in order to obtain some objective (dependum). The dependum is an intentional element that can be a resource, task, goal or softgoal (see [Yu-Thesis] for a detailed description of their meaning). It is also possible to define the importance (strength) of the dependency for each of the involved actors using three categories: *open*, *committed* and *critical*.

The Strategic Rational model allows visualizing the intentional elements into the boundary of an actor in order to refine the SD model with reasoning capabilities. The dependencies of the SD model are linked to intentional elements inside the actor boundary. The elements inside the SR model are decomposed accordingly to two types of links:

- *Means-end* links establish that one or more intentional elements are the means that contribute to the achievement of an end. The "end" can be a goal, task, resource, or softgoal, whereas the "means" is usually a task. There is a relation *OR* when there are many means, which indicate the different ways to obtain the end. The possible relationships are: *Goal-Task*, *Resource-Task*, *Task-Task*, *Softgoal-Task*, *Softgoal-Softgoal* and *Goal-Goal*. In *Means-end* links with a *softgoal* as end it is possible to specify if the contribution of the means towards the end is negative or positive.

- *Task-decomposition* links state the decomposition of a task into different intentional elements. There is a relation *AND* when a task is decomposed into more than one intentional element. It is also possible to define constraints to refine this relationship. The importance of the intentional element in the accomplishment of the task can also be marked in the same way that in dependencies of a SD model.

The graphical notation is shown in Fig. 3.4 using an example about academic tutoring of students. On the left-hand side, we show the SR model of a tutor and the hierarchical relationships among their internal intentional elements. On the right-hand side, we show the SD dependencies between a student and a tutor.

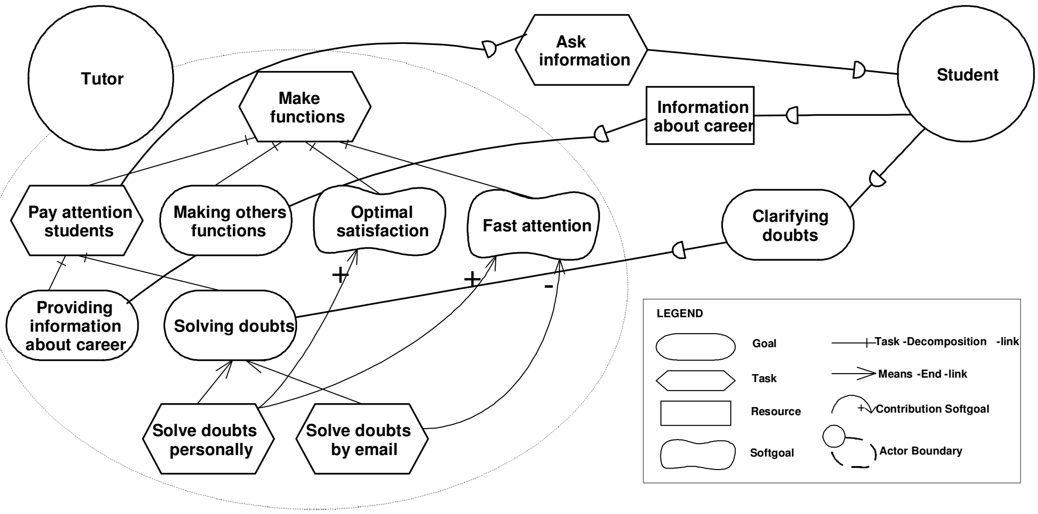


Fig. 3.4. Example of an *i** model for an academic tutoring system.

Actors can be specialized into *agents*, *roles* and *positions*. A position covers roles. The agents represent particular instances of people, machines or software within the organization and they occupy positions (and as a consequence, they play the roles covered by these positions). The actors and their specializations can be decomposed into other actors using the *is-part-of* relationship.

SR models have additional elements of reasoning such as *routines*, *rules* and *beliefs*. A *routine* represents one particular course of action (one alternative) to attain the actor's goal among all alternatives. *Rules* and *beliefs* can be considered as conditions that have to fulfill to apply routines.
The aim of this framework is to provide modeling features, which can lead to semi-automated support facilities to help human users express, manipulate, organize, manage, and draw conclusions from this knowledge.

The SD model and the SR models were built on a knowledge representation approach to information system development.

Specifically the use of the SD models is useful in the framework of this thesis, mainly for 2 issues: in one hand, its use is required for representing common dependencies among actors of the domain, it means as a part of the domain analysis and knowledge representation, previous to the construction of the taxonomy. On the other hand, the most important use proposed is where actors represent COTS and dependencies are established among them; allowing the optimal identification and formal representation of dependency relationships between COTS products from different domains in a taxonomy.

4. Planning

To fulfill the goals of this work, we propose to complete a three-iteration process preceded by and initial phase 0 (Fig. 4.1).

The phase 0 of this research work was focused on the study of the state of the art and evaluated the applicability of suitable methodologies for getting our objectives. At the end of this iteration we chose the goal-based methodologies and in particular GBRAM as the most suitable framework to our target.

- First iteration was used to validate the results of the phase 0 selecting a particular COTS field as a case study. As a result we targeted the obtention of an initial version of our methodology for building COTS taxonomies.

- Second iteration is used for a deeper analysis and a preliminary validation of the results of the previous one. We expect to apply the methodology in additional domains at academic level as well as to evaluate suitable tools for implementing our target. As a result we target the obtention of a refined and enriched version of the methodology.

- In the third and last iteration we expect to consolidate our research work by means of a complete validation of our method, its implementation, as well as the thesis writing and final documentation.

Fig. 4.1(a), shows the Project scheduling and our publications related to each activity. Figure 4.1(b) shows the expected and ongoing activities scheduled by months.

As we can see in such figures; in parallel to the three iterations we are working in the exploration of the application of modeling notation (e.g. *i**, UML) to support COTS dependency relationships and domain modeling; as well as classification techniques.

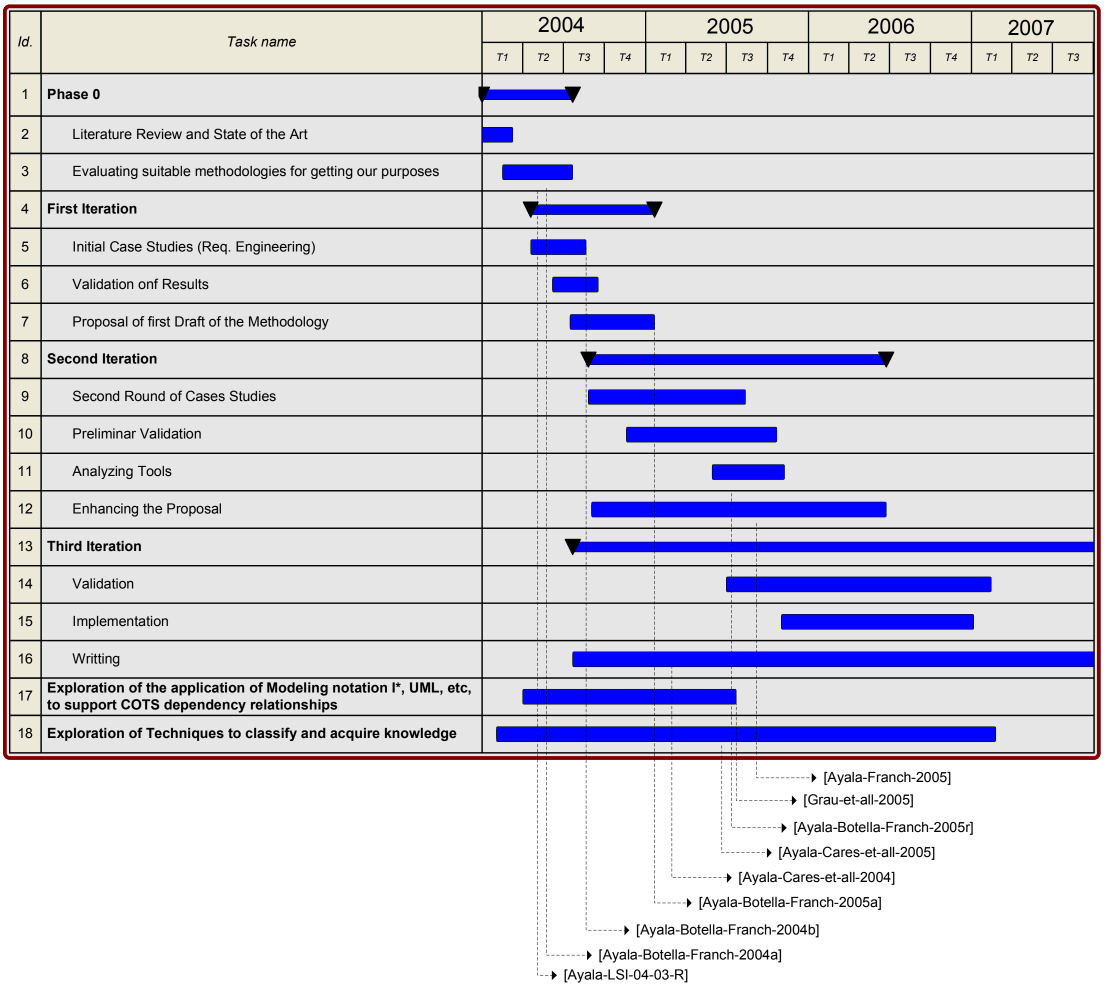


Fig. 4.1(a) Project Scheduling

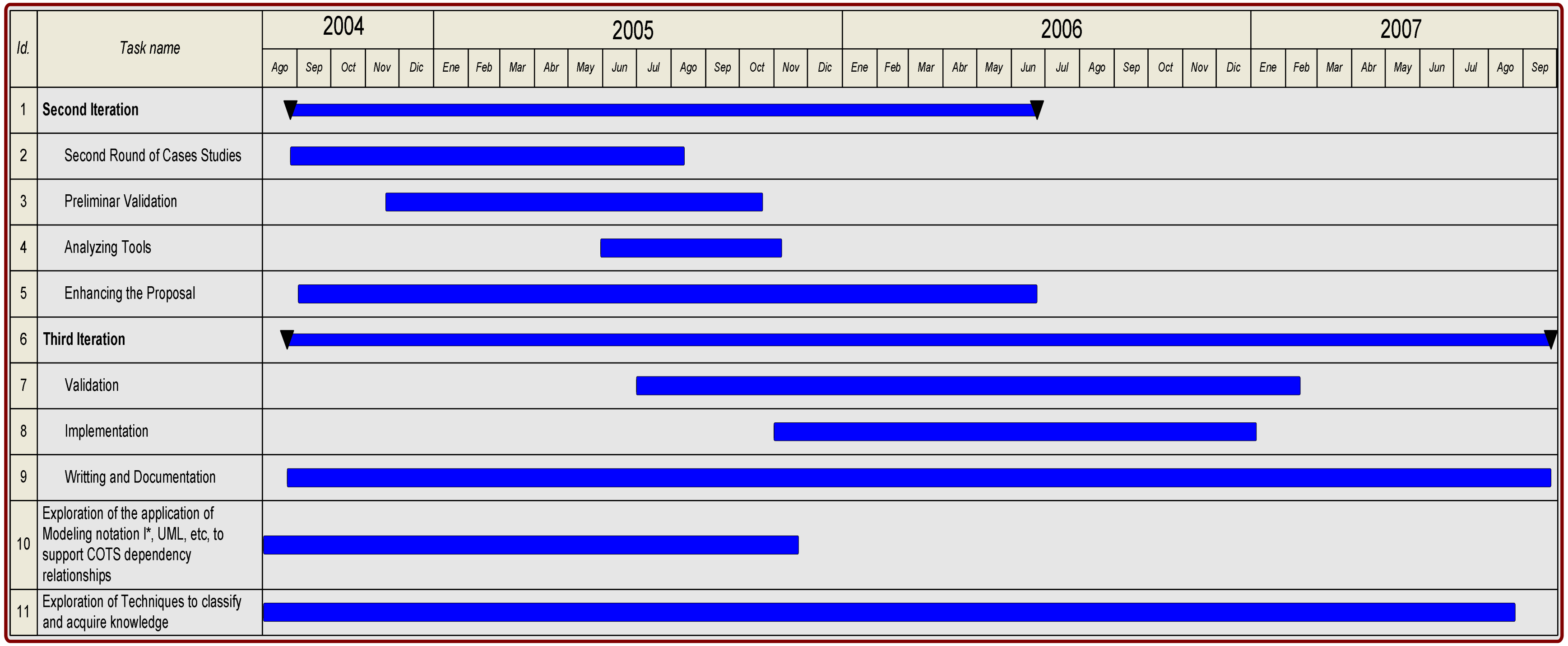


Fig. 4.1(b) Project Scheduling.

5. Research Performed

5.1. Phase 0

As it was stated in section 4, in the phase 0, the applicability of some methodologies for getting our objectives was evaluated.

As it was mentioned before, this research line was departed from the ideas of a previous proposal of our research group [Carvallo-et-al.2004]. The insights of classification exposed there seemed to be satisfactory enough as a basis of our classification studies despite of the inconveniences related in previous sections.

Therefore, we realized a research work to confirm the suitability of these insights. Thus, although some basic techniques for data classification exist, we base this proposal on a COTS classification as decision trees (as pointed in [Carvallo-et-al.2004]) because the study of their properties confirmed that they were the an appropriate way for representing in a natural form a description of the COTS domain marketplace. The main motivation was that they represent rules that can readily be expressed in natural language so that we humans can understand them, by means of these rules we can assign objects to each node.

Besides evaluate the suitability of the COTS taxonomies construction as a decision trees, we studied the applicability of some approaches for getting knowledge and requirements acquisition (from which the taxonomy should be constructed) with the primary focus on generalizing, formalizing, improving and clarifying the process of identify and evaluate mechanisms for constructing the taxonomy as a decision tree in a formal way instead of only the common sense. As a result, the goal-based framework was chosen as the most suitable to the objectives of COTS taxonomies construction and modeling dependencies. This is because goals are more stable respect to changes in processes, and goal refinement provides a natural mechanism for structuring and exploring many alternatives (such reasons are further explained in section 3.6.1).

In this sense, we explored the applicability of goal-based approaches, specifically GBRAM to our purposes. These results were reported and presented in the *"8th World Multi-Conference on Systemics, Cybernetics, and Informatics 2004"* [Ayala-Botella-Franch2004a] as well as in a Technical Report of the LSI department [Ayala-LSI-04-03-R].

Hence, the suitability to apply GBRAM to our purpose of constructing taxonomies of COTS can be summarized as follows:

a) It assumes that goals have not been previously documented or explicitly elicited, so we must work from all available sources of information to determine the goals. This is a very helpful aspect because in the COTS context, we have to simultaneously evaluate much diverse information related with the domain that we want to create the taxonomy.

b) It provides guidelines and heuristics for exploring, identifying, and organizing goals which guide us towards a high probability of success while avoiding wasted efforts.

c) It offers a guide for applying an inquiry-driven approach to goal-based analysis, which can be useful for to enhance the questions-answers mechanism linked to mostly nodes in the taxonomy for helping its browsing [Carvallo-et-al.2004].

Due to this, we consider it suitable for providing the motivation and rationale to justify and identify goals from the available information sources related with COTS approach.

It is a fact that the study and application of many mechanisms and techniques is required for customizing GBRAM to our COTS taxonomies approach (for instance, to guide the applicability of requirements acquisition techniques, a suitable goal-reasoned process for arranging goals as COTS taxonomies and building suitable heuristics that guide the process).

However, the most important is that GBRAM offers a helpful framework for integrating such mechanisms and techniques needed for formalizing the construction of taxonomies from goals; this includes the models and artifacts required for their manipulation and evolution.

5.2. First Iteration

The first iteration was used to validate the results of the first one selecting a particular COTS field as case study. As a result, we obtained an initial version of our methodology for building COTS taxonomies. This version is being enriched as the ongoing work progress.

The first version of the methodology was inspired in GBRAM adapting it to the COTS taxonomies context and using it as a framework for integrating the mechanisms, techniques, and artifacts needed for formalizing the construction of taxonomies from goals as well as their manipulation and evolution.

This work was reported and presented in the "*4th International Conference on COTS-Based Software Systems"* [Ayala-Botella-Franch2005a]. In this paper, we showed how we customize GBRAM. We adjusted the inputs and modified the output. We adapted and pruned some activities in order to obtain the statement goals to be considered in the classification criteria for the construction of COTS taxonomies in any specific area. For clarifying concepts, in this paper we presented some insights of the Requirements Engineering (RE) case study.

Such case study was deeply reported in the "*VII Workshop on Requirements Engineering*" [Ayala-Botella-Franch2004b]. This paper was selected as one of the best papers of the workshop and after a second review process it was included in the "*Journal of Computer Science and Technology. Special Issue on Software Requirements Engineering Vol. 5 - No. 2*" [Ayala-Botella-Franch2005r].

The detailed explanation of customizing GBRAM to the COTS taxonomies context and the RE case study was reported as a technical report of the department [Ayala-LSI-05-25-R].

In this subsection we will present a summary of the method and the case study; however a detailed description of them is founded in the related papers and technical reports.

5.2.1. A Methodology for Building COTS Taxonomies: First Draft

In general, we depart from the general idea of the activities of GBRAM and adequate it to COTS contexts, obtaining the GBTCM method (*Goal-Based Taxonomy Construction Method)* that provides a guide to analyze sources of information and modeling requirements and domains, as well as gathering and organizing the knowledge in any segment of the COTS market. In addition GBTCM contribute and enhance the requirements and knowledge reuse in different process of selection and integration of components.

We use the notion of *goal* as introduced in the context of requirements engineering [VanLamswerde2001]. Goals are the rationale to identify characterization attributes and therefore COTS categories and domains.

Fig. 5.1 shows the activities (ovals) and artifacts (inclined rectangles) involved in GBTCM. The two high level phases are: *Goal Analysis* (concerns the exploration of available information sources for goal identification followed by the organization and classification of goals) and *Goal Refinement* (concerns the evolution of goals from the moment they are first identified to the moment they are translated into requirements).

The general activities are: *Explore* (entails the analysis of available information), *Identify* (aims at extracting goals applying techniques as heuristics), *Organize* (involves the classification and organization of goals according to goal dependency relationships), *Refine* (entails the actual pruning of the goal set), *Elaborate* (refers to the process of more refinement, analyzing the goal set by considering possible obstacles and constructing scenarios to uncover hidden goals and requirements), and *Operationalize* (refers to translating goals into requirements for the final taxonomy).

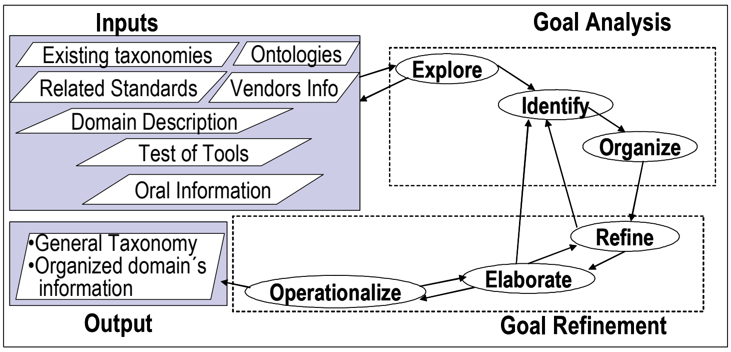


Figure 5.1. GBTCM activities

In table 5.1 we can realize that the output of each activity is the input of the next.

Table 5.1. GBTCM activities and their outputs

Activity	Outputs
Explore	Information sources qualified; Some goals
Identify	Set of goals; Stakeholders and agents; Auxiliary models and artifacts
Organize	Matching of goals from different information sources Dependency relationships among goals Goal hierarchy
Refine	Refined goal set
Elaborate	Scenarios Constraints More Refined Goal Set
Operationalize	Hierarchical structure of Goals Asociated information and models and artifacts Characterization attributes for constructing a general taxonomy of the domain

Certainly, the information sources (inputs) are the base for obtaining goals. The final result of GBTCM is not only a basis for the COTS taxonomy construction in a specific area, but also a set of information and knowledge (repository) that contains: all the activities performed in that area (expressed as goals) and the dependency relationships among COTS domains; in order to concern the impact of changes among domains.

Basically, GBTCM guide to gather, manage, and generalize information related with a domain (e.g. existent taxonomies, standards in the field, and vendors). It also generates artifacts and models (e.g. UML class diagrams, *i** models) that permit to assure consistency and evolution of the repository of knowledge. This repository can be used during different selection process. It is also the source for constructing a general taxonomy and both (the repository and the general taxonomy) could be the basis for organizations to build up the most suitable taxonomy to their processes.

5.2.2. Information Sources in GBTCM

More than anything else we want to emphasize the importance of the information sources (the inputs), since they are the base for obtaining goals.

Table 5.2 summarizes the most important sources of information we have considered in GBTCM. The most information is textual, available in printed form or the web, issued by different organizations or people. However, sources such as domain experts and tools demos still remain.

Table 5.2. Information Sources to consider in our approach.

<table>
<tr><th colspan="3">Information Source</th><th>Information type</th><th>Language</th><th>Examples</th></tr>
<tr><td colspan="3">Existing Taxonomies and Ontologies</td><td>Classifications; Categories; Glossaries</td><td>Natural Language (NL); Tree-like diagrams</td><td>SWEBOK, INCOSE, Gartner, IDC</td></tr>
<tr><td colspan="3">Related Standards</td><td>Descriptions; Glossaries</td><td>NL</td><td>IEEE, EIA, ISO</td></tr>
<tr><td colspan="3">Vendors Information</td><td>Brochures; Evaluation forms; Benchmarks</td><td>NL; Values for attributes</td><td>Rational, Microsoft</td></tr>
<tr><td rowspan="3">Domain Descriptions</td><td>Scientific</td><td>Academic Events, Journals;Textbooks</td><td>Precise and rigorous descriptions</td><td>NL; Models; Formulas; Schemas</td><td>ICCBSS, ICSE, TSE, [MO02]</td></tr>
<tr><td>Divulgation</td><td>Magazines, Forums and Websites</td><td>Descriptions and tips for the general public</td><td>NL; Schemas; Tables</td><td>PCWorld, IEEE Software, COCOTS website, specialized forums</td></tr>
<tr><td>Technical</td><td>White Papers, Surveys and Comparatives</td><td>Papers, Comparative tables</td><td>NL, Tables; Figures</td><td>Gartner, INCOSE</td></tr>
<tr><td rowspan="2">Oral Information</td><td colspan="2">Interviews</td><td rowspan="2">Knowledge; Tips; Practical information</td><td rowspan="2">NL</td><td rowspan="2">ICCBSS panels, SEI courses, Business luncheons</td></tr>
<tr><td colspan="2">Talks, seminars and courses</td></tr>
<tr><td colspan="3">Test of Tools and Systems</td><td>Test results; User's manuals</td><td>Visual data; NL</td><td>Outlook, Rationale Suite</td></tr>
<tr><td colspan="3">Experiences on the field</td><td>Knowledge; Technical reports</td><td>Knowledge; NL</td><td>Past projects made</td></tr>
</table>

The use of one or another information source is determined by several qualities, among which we mention: reliability of the information, availability of the source, acquisition cost, timeliness, scope covered and time needed to process the enclosed information.

These qualities depend on three factors: information source type, organization, or people that create the information, and particular item of information. A complete goal-acquisition program should take these considerations into account.

So we contemplate constructing in next iterations an exhaustive program of analysis of information sources that permits to take into account their qualities and factors for classify them according their relevance as well as to enrich the method with suitable techniques.

5.2.3. Auxiliary Models and Artifacts in GBTCM

GBTCM considers essential the generation of some artifacts and models from the information sources in order to understand, handle, formalize, and remarkably guarantee the integrity and consistency of the information respect to changes and evolution.

Hence, it suggests creating glossaries for homogenizing terms used in diverse information sources and UML class diagrams [UML] for representing a conceptual model of the domain and defining the underlying ontology [Carvallo-Franch-Quer2003]. As a fundamental part of the method, it requires the construction of goal-oriented models, specifically *i** as notation [Yu1997] although other options are valid. These models and artifacts should be refined during all process.

5.2.4. Modeling Dependencies in GBTCM

Modeling dependencies is a crucial issue in our method; *i** aids us in the process of getting goals providing a high level picture of the domain, representing and organizing the knowledge and related activities; on the other hand it is used in order to represent and recorded explicitly the dependencies among domains of COTS for their repeated use during different selection process, moreover the relationships among these domains and the actors that have an interest on them, either as users or as definers.

As it was mentioned before, since *i** is considered essential in our methodology, as a parallel research line we are working in analyzing the goal-oriented modeling framework, specifically in a suitable definition of the *i** methodology, analyzing and evaluating the need of adding it new constructors for its fitness to the nature of the framework we want to model. In this sense, in our research group we written jointly two articles, one of them was presented first in an iberoamerican forum (*"4as. Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería del Conocimiento 2004"*) [Ayala-Cares-et-al.2004] and later in a worldwide forum ("*International Workshop on Agent-Oriented Software Development Methodology 2005")* [Ayala-Cares-et-al.2005].

On the other hand, in our research group we also defined a general methodology for building *i** Strategic Dependency Models, called RiSD for supporting the targets of the different research lines pursued by the group. This work was accepted in the "*17th International Conference on Software Engineering and Knowledge Engineering" (SEKE 2005)* [Grau-et-al.2005].

It is important to say that such work is ongoing in parallel with next iterations.

5.2.5. Construction of a Taxonomy for Requirements Engineering Field

For validating our proposed methodology, we selected an initial COTS category to be used as starting point of this work.

The field selected was the one of Requirements Engineering. This was not just a coincidence; it was selected for three main reasons:

- This is a domain from which we had a basic knowledge to start to work with.

- It is widely acknowledged within the software industry that software projects are critically vulnerable when the activities of software requirements are performed poorly; also reports exist supporting this hypothesis [CHAOS]. Therefore, to improve the efficiency of the activities performed in the area is a crucial need, therefore COTS technology aid RE-related actors to simplify and facilitate their work.

- From the practical point of view, we state as an objective to build a taxonomy for the Software Application Development from which the Requirements Engineering field belongs to.

5.2.5.1 Information Sources Considered

As it can be expected for a topic such this, lots of information sources exist and many of them were gathered.

Table 5.3 summarizes the most important sources of information we consider in this case study, and shows details in a general manner of the kind of information content.

We can note that most information is textual, available in printed form or the web, issued by different organizations or people. Sources such as domain experts and tools demos still remain, but they play a secondary role.

It is important to remark that we pretend to represent information that does not depend on a concrete project or software system, but also to create a general repository with all information as possible related with the RE area, that permits its later applicability and suitability in different organizations.

As it was mentioned above, the use of one or another information source is determined by several qualities.

In general, it is considered as a good practice to fundament goal analysis on the most solid and confident of the sources (mainly solid standards in the field) for extracting the high level goals in order to assure the consistency of the set of goals, and then extracting sub-goals from the remaining sources.

Table 5.3 Main information sources used in the RE case study.

Type of source	Source organization	Information enclosed	Comments
Existing taxonomies	INCOSE	Classification of Software Engineering tools	This section is available free and widely accepted
Related standards	SWEBOK	Main RE areas stakeholder types	Available free, widely accepted
	IEEE std 830-1998	Software activities related with RE	Subscription/payment needed
	IEEE/EIA 12207.1-1997		
	ISO/IEC 12207		
Vendors information	IBM-Rational	Capabilities of products and trends	Exhaustive description of products
	ComponentSource	Capabilities of products and trends focused in platforms	Available free, widely accepted
Tools	RequisitePro	Capabilities of a real RMT	Included in the IBM-Rational Suite
	IRqA		Tool used often in our projects
	EasyWinWin	Capability of a research tool for requirements negotiation	Some tutorials attended and contacts with authors
Academic sites	eCOTS	Trends	Available free, widely accepted
Scientific items	RE-related conferences	Timely state of the art	Subscription/payment needed
	RE&SE textbooks	Areas of RE	
Magazines	Requirements Engineering	Trends and timely state of the art	
WebSites	Volere	RE resources	Available Free
Technical	INCOSE	Trends and concepts in RE	Subscription/payment needed
	Gartner		
Own experiences	Academic records management	Use of RE-oriented tools in a real project	CMM-2 compliant requirements management

Due to the standard nature of SWEBOK in the field, we started with this source for obtaining the high-level goals that guide the whole process (even considering that SWEBOK is not tool-oriented, on the contrary of other sources).

For example, consider the following description in natural language from SWEBOK: “The next topics breakdowns for RE discipline are generally accepted in that they cover areas typically in texts and standards: activities such as Requirements Engineering Process, Requirements Elicitation, and Requirements Analysis, along with Requirements engineering-specific descriptions. Hence, we identify Requirements Validation and Requirements Management as separate topics”. By examining the statement and asking “what goal(s) does this statement/fragment exemplify?” some goals become evident from the description.

We present some of these goals in the first column of table 5.4. These goals are going to be decomposed in sub-goals by means of the analysis of other information sources and then, applying refinement techniques. We will further describe the refinement process in the next subsections.

Table 5.4. Some goals obtained from SWEBOK

Goals		Agents	Stakeholders
G1:Process of Software Requirements Defined		(RE)	RE, PM,QAM
G2:Requirements Elicitation Performed		RE	RE, Stakeholders
G3:Requirements Analysis Performed		RE	RE, Stakeholders
G4:Requirements Specification Done		RE	RE, users/customer, QAM
G5:Requirements Validation Performed		RE	RE, users/customer, Tester
G6:Requirements Management Done		RE	RE, SCM
	G6.1:Change Management in Requirements Controlled	RE	RE
	G6.2:Requirements Attributes Defined	RE	RE, SCM
	G6.3:Requirements Tracing Controlled	RE	RE, SCM

5.2.5.2 Identifying Stakeholders and Agents

At this stage, we aim at determining who are the stakeholders involved in the achievement of goals. Once the goals and stakeholders are specified, the goals must be assigned to their responsible agent(s).

A stakeholder is any representative affected by the achievement or prevention of a particular goal. Multiple stakeholders may be associated with one goal. Agents are responsible for the completion and/or satisfaction of goals within an organization or system. Identification of stakeholders and agents is crucial to understand the domain at hand and also to identify additional sources of information, e.g. for identifying people to be interviewed.

The stakeholders for each goal are determined by asking "who or what claims a stake in this goal?" and "who or what stands to gain or lose by the completion or prevention of this goal?" For identifying which agents are ultimately responsible for the achievement of each goal, we ask the question "who or what agent [is/should be/could be] responsible for this goal?" In our case, we identified as stakeholders (see table 5.4): Requirements Engineer (RE), Project Manager (PM), Quality Assurance Manager (QAM), Software Configuration Manager (SCM), Testers, Final Users, Customer and Non-Technical Stakeholders (such as regulators, market analyst, system developers; NTS). The only agent is the Requirements Engineer.

Some relationships and dependencies among stakeholders are showed in the *i** SD model in fig 5.2.

5.2.5.3 Auxiliary Models and Artifacts

Since GBTCM considers essential the generation of some artifacts and models from the information sources, in the RE case study, we create: *i** models (we can see an excerpt in fig. 5.2 and fig. 5.3), glossaries, and class diagrams in order to homogenize information from different sources, facilitating the communication allowing confusions. For example, the terms "capturing" and "extracting" that coming from two different sources was unified and defined as "extracting" in our glossary.

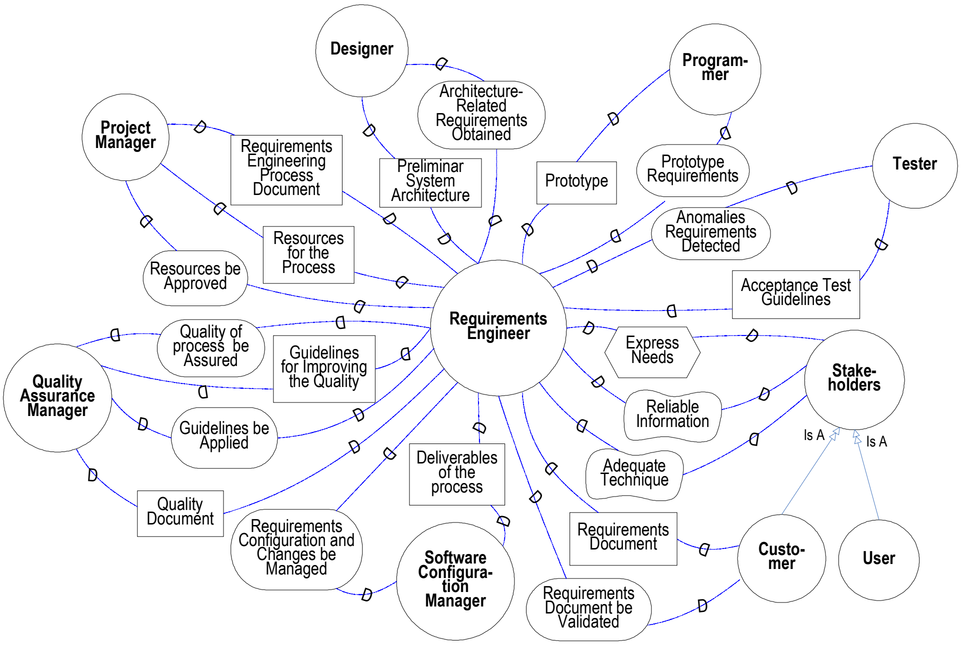


Fig. 5.2. Excerpt of the *i** SD model that shows relationships among system actors

5.2.5.4 Use and Creation of Heuristics

Heuristics aid us by providing prescriptive guidance for managing varying levels of detail in the information available. Previous sections show examples of the application of heuristics that guide the obtention of specific information by means questions. These heuristics are general and do not depend on any particular domain. An important objective of GBTCM is the reuse of knowledge acquired from different case studies. Thus, the creation of new specific heuristics of the domain is a desirable task, in order to their repeated use in other cases, achieving a high probability of success finding goals while avoiding wasted efforts. This section is focus on the general description of the application of GBTCM method to RE field; so we do not present new RE domain heuristics. More than anything else, we are aware that is necessary to perform in deep more practical cases for the foundation of new heuristics. Since is part of our ongoing work

5.2.5.5 Organization and Matching of Goals

Once goals have been analyzed and identified from all information sources, we have to organize the information firstly by means of a matching of goals from all information sources, and subsequently according to precedence relationships.

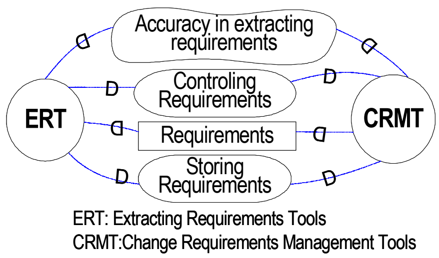
In the matching process we can observe that one goal should be taken into account in the taxonomy construction only if it exist in the market a tool that supports it (although it could be argue that discovering of goals that are not covered by any tool is a significant issue in closing the gap between tools and processes).

We perform the process of organization of goals by means of tables. Table 5.5 is an excerpt of this process. There we can observe the matching among some information sources: identified goals from related standards, scientific items, etc.; vendor's information of tools; and, nodes of INCOSE taxonomy. At the end of the matching process we have a more complete set of goals. This resulting set of goals going to be refined in subsequent steps but firstly dependencies relationships should be specified.

Table 5.5. An excerpt of organization of goals

Goals	Tools	Cathegory of INCOSE Taxonomy
G2:Requirements Elicitation Performed		
G2.1:Requirements Sources Defined and Analizad		
G2.2:Elicitation Techniques Chosen		
G2.2.1:Extracting Requirements	Yes	RequirementsEngineering/Requirements Management/RequirementsCapture&Identification/ ToolsForElicitationOfRequirements
G2.2.1.1:Interviews	Yes	
G2.2.1.2:Scenarios	Yes	Design Domain

Departing from this set of goals, hierarchical dependencies are defined (by means of hierarchical tables, called goal topography). Then, refinement processes are applied concurrently with the identification and specification of dependencies among goals. This specification is done, as we mention before, by means of *i** models; it means the explicit representation of potential dependencies among COTS domains. This mechanism assures the traceability of the impact of changes among domains. As a result, we have a dual representation of goal models, a more technical one (fig 5.3, left) and a more understandable one, easy to index and read (fig 5.3, right).



Goals
M1: Extracting Requirements
M1.1:Accuracy in Extracting Requirements
M1.2:Storing Requirements
M1.2.1: Controling Requirements

Fig. 5.3 An example of a *i** SD model (left) and a hierarchical table (right) involving the RE tools

5.2.5.6 Refinement of the Goal Set

GBTCM is based on identification of goals and organization of knowledge from an iterative refinement process.

Once it was obtained a set of goals from the matching of all information sources, and their hierarchical dependencies were obtained, they are analyzed for refinement.

We used some refinement techniques proposed by GBTCM: the use of scenarios (understood as behavioral descriptions of the system and its environment) that refers to the recreation of the different situations and circumstances in which a goal is executed; the application of the *Inquiry Cycle* [Potts-Takanashi-Anton1994] a formal structure for describing discussions about requirements that consists of a series of questions and answers designed to pinpoint where information needs come from and when; and also the use of glossaries to support the reconciliation of goals and UML class diagrams.

Scenarios were used in the form of "general use-case", and were represented as sequences of actions in natural language prose descriptions. We applied them in order to be more exhaustive and include as much as possible activities performed in the RE phase in the most projects of software development. For instance, by means the construction of scenarios for the goal *Requirements Analysis Performed*, we detected (among other issues) that in many cases the process of analyzing and elaborating requirements demands (in order to be achieved) to identify the subsystems and components that will be responsible for satisfying the requirements, so we had to consider *Architectural Design Done* as one goal in this area. Scenarios were very useful for uncovering and reconciling goals, checking for completeness and conflicts, and communicating with stakeholders.

We have to say that the applicability and precision in the scenarios construction depend on the criticism that the domain requires, it means that a high critical domain (i.e. spatial applications) requires the construction of the most detailed scenarios and special techniques.

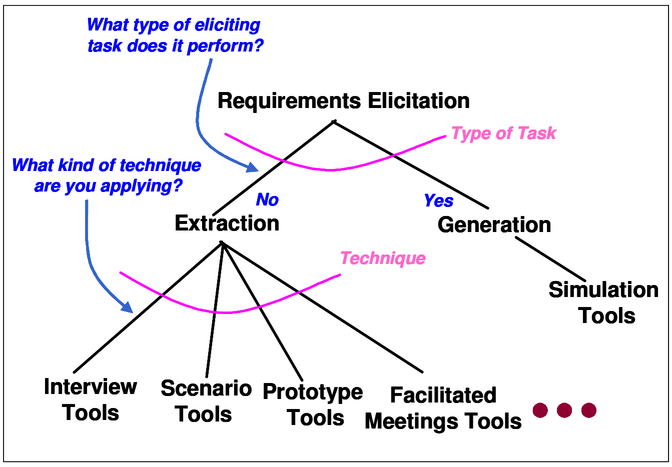


Fig. 5.4 An excerpt of the RE taxonomy obtained

5.2.5.7 Operationalizing Goals

GBTCM is based on identification of goals and organization of knowledge from an iterative refinement process.

Goal information must be ultimately operationalized (related with actions) and the general taxonomy be constructed. This is done by consolidating the goal information, and applying the *Inquiry Cycle*. The *Inquire Cycle* was very helpful for finding the classification criteria in order to organize the resulting information. Fig. 5.4 shows and excerpt of the resulting taxonomy.

5.2.5.8 Remarks

The knowledge (repository) and the taxonomy obtained help to Software Engineers -which usually carry out COTS selection and integration- to structure and reuse better their knowledge for their repeated use during different selection process.

Above all, we claim that the taxonomy and the information and knowledge obtained (repository) could be the basis for organizations to build up the most suitable taxonomy of RE COTS to their processes. It means a better return on investment firstly in time and required effort.

Among some results we contemplate constructing in next iteration an exhaustive program of analysis of information sources that permits to take into account their qualities and factors for classify them according their relevance as well as to enrich the method with suitable techniques.

5.3. Second Iteration: Deeper Analysis (Ongoing Work)

The second iteration is being used to validate the results of the previous one and enrich and improve the stated GBTCM method (Fig. 5.5) with suitable techniques that help to structuring and reuse knowledge of the COTS marketplace with a high probability of success.

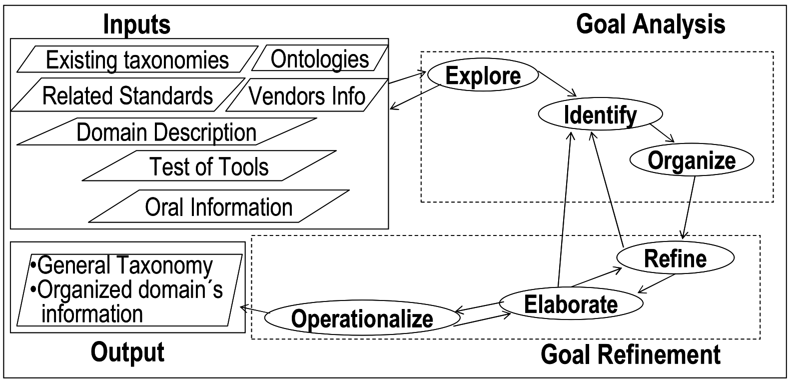


Figure 5.5. Overview of GBTCM

The research lines involved are:

- Preliminary Validation of our Method.
- Construction of a Program of Information Sources Analysis
- Improving Activities of our Proposed Method
- Analysis of Tools for Supporting our Method

5.3.1. Preliminary Validation of our Method

As a part of this iteration, we performed by means of the GBTCM method the case study of Business Applications tools that was before classified by common sense in [Carvallo-et-al.2004] and we compared results. The results were forceful, above all regardless of the new taxonomy comprehension, completeness, and scope.

It originally had 96 nodes, 78 were leafs (i.e., types of software packages) and just 18 intermediate nodes. The longest path from the root to a leaf was of length 6 (just one, and two of length 5), whilst the maximum width was 10 siblings (average width 4,33). Our process resulted in a taxonomy of 188 nodes, 120 of them leafs. The tree changed its form, with a longest path of length 8 (in fact, 16 paths of this length and 21 of length 7) and 7 siblings at most (average width 3,18). As a result, not only we provided rationale for the taxonomy, but also we let it well-suited for our main purpose, to use it for software package selection, with better defined ways to reach a leaf (i.e., a type of software package) from the root having less alternatives to consider at each step.

It was clear that the goals presented a comprehensible rationale behind the taxonomies. The main difference between the empirically obtained taxonomy [Carvallo-et-al.2004] was that the meaning of the particular categories was understandable without further examining the items. Therefore, the use, evolution, extension, and customization of the new categorization proposal are a feasible task.

5.3.2. Construction of a Program of Information Sources Analysis

This activity is being developed in parallel to the case studies; we are analyzing and applying existent requirements engineering techniques to our framework for guiding and improving the analysis of the information sources that should be considered.

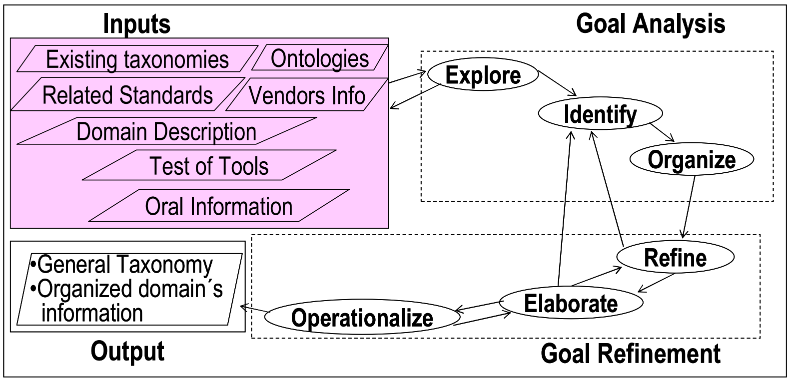


Fig. 5.6 Improving the analysis of Inputs

Above all, we are analyzing characteristics of the information sources that could provide a metric or basis for evaluating their quality and therefore their use in our approach. Among which we can mention: reliability of the information, availability of the source, acquisition cost, timeliness, scope covered and time needed to process the enclosed information.

These qualities depend on three factors: information source type, organization, or people that create the information, and particular item of information.

The goal-acquisition program that we want to achieve is facing these considerations.

Studies have confirmed data quality is a multi-dimensional concept [Huang-et-al.1999], [Ballou-Pazer1985]. We must deal with both the subjective perceptions of the individuals involved with the data, and the objective measurements based on the data set in question.

Subjective data quality assessments reflect the needs and experiences of stakeholders: the collectors, custodians, and customer of data products. If stakeholders assess that the quality of data is poor, their behavior will be influenced by this assessment

Objective assessment can be task-independent or task-dependent. Task-independent metrics reflect states of the data without the contextual knowledge of the application, and can be applied to any data set, regardless of the tasks at hand. Task-dependent metrics, which include the organizations business rules, company and government regulation and constraints provided by the database administrator, are developed in specific applications contexts.

Thus, it is pretended to use a guide for the subjective and objective assessment of data quality that provides insights of the information sources that should be considered as well as their use for the construction of taxonomies in different fields.

5.3.3. Improving Activities of our Proposed Method

5.3.3.1 Improving Goal Analysis, Refine and Elaborate Activities

The activities of goal analysis, refine, and elaborate are deeply interrelated. *Goal analysis* refers to the acquisition and matching of goals from the information sources. The *refine* activity is oriented to prune this set of goals acquired and *elaborate* refers to the process of analyzing these goal set by means of techniques for uncovering hidden goals and requirements.

As a result of performing case studies by means of GBTCM we evaluated and detected the applicability of some techniques for goal acquisition, representation and refinement (some of them were reported in previous iterations, for instance the *i** framework). In this stage we are applying and studying in depth techniques for improving the goal analysis, refinement, and elaborate activities of the method.

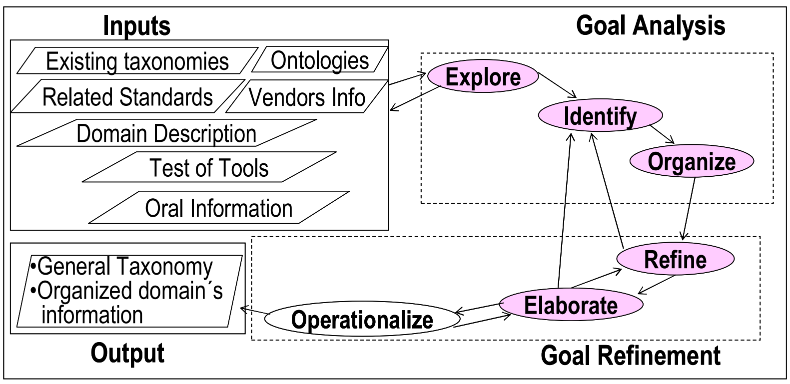


Fig. 5.7. Improving the shadow activities at this stage of the research

Some of these techniques were applied in previous case studies, for instance, the Inquiry Cycle, and Scenario Analysis [Potts-Takanashi-Anton1994]. In this stage we are evaluating how to improve their applicability in our method and we expect to provide heuristics for their application by means of performing other case studies.

On the other hand, we are analyzing the applicability of Viewpoints [Easterbrook1991]. A viewpoint captures a particular responsibility performed by a participant at a particular stage.

We argue that the ability to represent viewpoints and distinguish between them is especially useful for the GBTCM activities of goal analysis and operationalize goals to taxonomies based on the users needs.

5.3.3.2 Improving Elaborate and Operationalize Activities

The operationalize activity refers to translating goals into customized taxonomies to the user requirements.

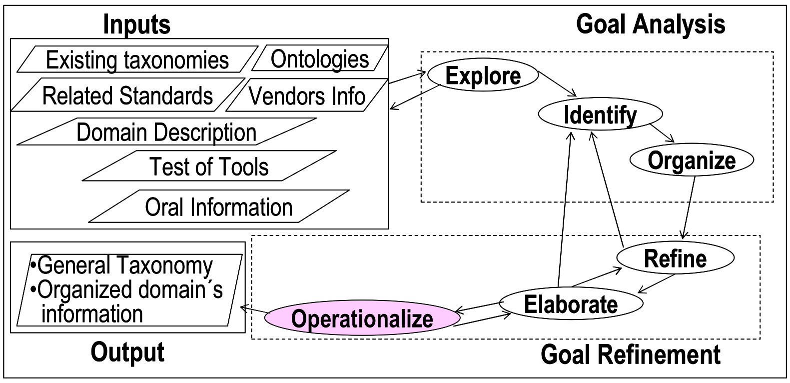


Fig. 5.8 Improving the Operationalize activity at this stage of the research.

Since the result of the elaborate activity is a refined hierarchy of goals, for improving the operationalize activity, our purpose is to provide the GBTCM method of the appropriate transformation rules over hierarchies of goals that allow goal manipulation as required obtaining incrementally a suitable taxonomy to the users needs. As a consequence, we may define the process of taxonomy construction as the repeated application of transformation rules over the nodes of a source hierarchy of goals.

In this sense, in the course of this research, we found that many profit and non-profit organizations have arranged this knowledge by defining categories of services, products, and knowledge, usually structured in a hierarchical form. Some examples of organizations are:

- IT consultant companies such as Gartner [Gartner] or Forrester [Forrester] use these categories to structure their reports and services on IT technology.

- Commercial web-based companies such as ComponentSource [CompSource] and Genium [Genium] group the products commercially available for facilitating searches.

- Professional societies such as INCOSE [Incose] use hierarchies to organize systems engineering knowledge.

- Portals with different registration procedures offer white reports [ITPapers], user's opinions [eCOTS], [IT] or technical products from research projects [CBSE].

- In the academic world, organizations, teams and individuals have presented their own proposals that range from specific of one domain [Arranga2000] to a wide range or even a field [Glass-Vessey1995], being the extreme case proposals as SWEBOK that acts as body of knowledge of a particular discipline [SWEBOK].

Regardless of their comprehension, completeness, and scope, all of these proposals share a common characteristic that may be considered as a drawback: they present a hierarchy of items without a clear rationale behind. The categorization relies on experience, knowledge, and observation but they rarely use knowledge engineering and requirements engineering techniques to classify the enclosed items. Sometimes, the meaning of a particular category is not clear without further examining the items, especially if it is absolutely unknown to the user. As a result, the understanding, use, evolution, extension, and customization of the categorization proposal may be difficult.

However, these arrangements could be profited in our proposed method, being a starting point avoiding structuring the information from the scratch but applying the appropriate transformation rules to this departing classification available. Therefore, we argue that the GBTCM method can handle this drawback, based on the use of goals to provide semantics to the nodes of existing ad-hoc classification hierarchies and then applying these transformation rules.

Transformations rules must satisfy some completeness and correctness conditions with respect to the involved goals. These notions induce naturally a process to apply the transformation rules in a comfortable order that guarantees termination while being expressive enough.

The result of this work as well as its application to the construction of a taxonomy for Business Applications are explained in [Ayala-Franch2005]. In this paper, we define the rules, their syntactic form and also their applicability conditions as properties of the involved goals (please, see this paper in the Annex Section for the detailed explanation of the transformation rules and the case study). Such paper will be presented in the "*16th International Conference and Workshop on Database and Expert Systems Applications*" (DEXA 05) to be held in August 2005 (Copenhagen, Denmark). It is worth to remark that this conference is considered as "Congrès Notable" by the UPC and the acceptance rate of this edition was 23%.

5.3.4. Analysis of Tools for Supporting our Method

Here, we are analyzing the domain and the existent tools characteristics that could be useful for the implementation of our approach; not only for the visual representation of taxonomies, but also supporting the process of creation and evolution.

In this sense we are evaluating tools, performing our developed cases studies by means of the Taxonomy Tool that was developed by our research group for the graphically creation of taxonomies; as well as other tool of general purposes as Protégé. It is important to say that other options will be evaluated too.

This work is ongoing, so in this section we only show a briefly explanation of each one we have analyzed and developed case studies, and a short analysis to their usefulness to our purposes.

The results obtained in this part of the research are so important since they will provide the motivation to implement a new one or adjust some existent tool.

5.3.4.1 Taxonomy Tool

The Taxonomy Tool [Grau-et-al.2004] was created as a module in our research group GESSI for supporting the process of browsing the taxonomies to find out which is the type of COTS component that is needed when the requirements engineer starts to determine the selection requirements. It is integrated to the DesCOTS system (**D**escription, **e**valuation and **s**election of **COTS** components) for supporting the reuse of quality models.

DesCOTS is referred as "system" because, as the COTS selection process is divided into different activities, it was implemented it as a set of tools that interoperate to support the whole process: the *Quality Model Tool* allows to define quality models; the *COTS Evaluation Tool* allows the evaluation of components; the *COTS Selection Tool* allows the definition of requirements that drive the COTS component selection; and the *Taxonomy Tool* allows to organize COTS domains as a taxonomy supporting reuse of quality models.

DesCOTS system was designed with the purpose of being used by different kind of users that will probably not be working in the same location. To allow data interchange between these different users, it was designed following a client/server architectural pattern. Each module (for instance the Taxonomy Tool) has its client program which can be installed independently from the others and just needs to access the server program to get and store the shared data using HTTP/XML, JAVA^{TM} servlets and MySQL.

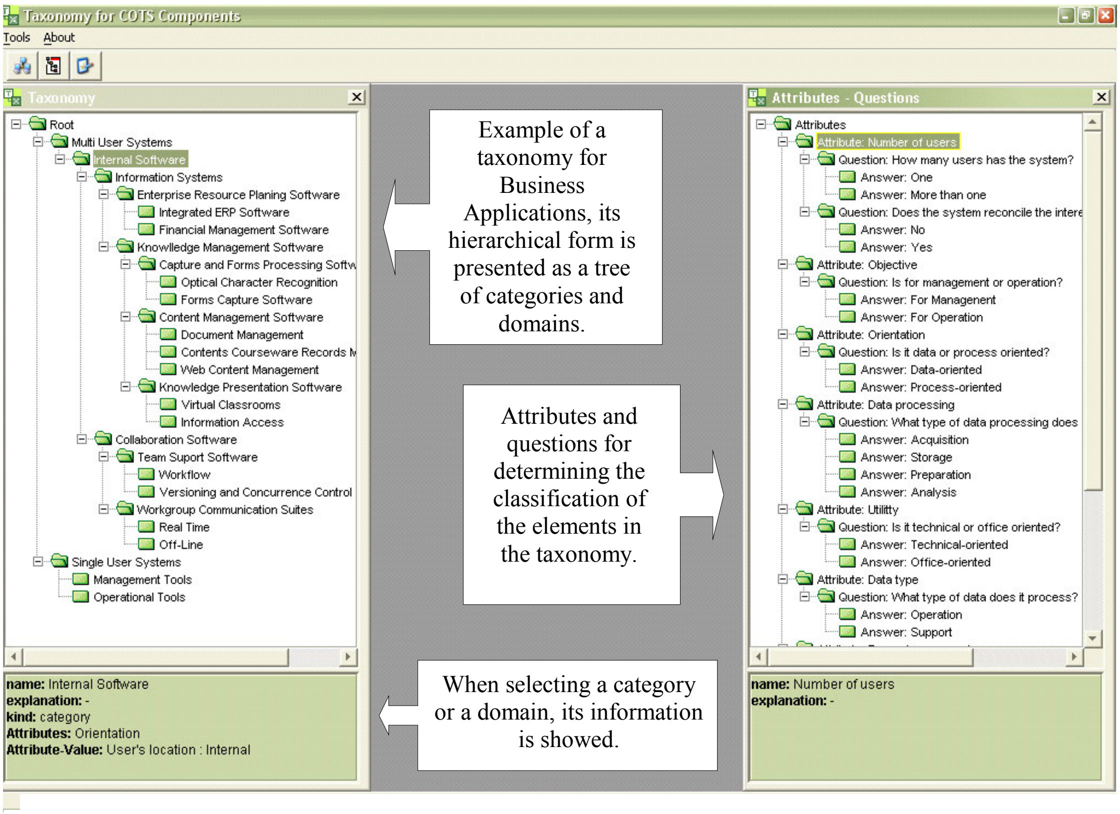


Figure 5.3. Snapshot of the business applications taxonomy with Taxonomy Tool

All the libraries used are open source, following the aim of getting openness and flexibility on the system development and distribution. The client parts of the tools are small applications of 20 Mb on average size, plus 35Mb of the Java Virtual Machine (JVM^{TM}) so it is easy to download and distribute. They run on a low-profile hardware and are easy to install.

Fig. 5.3 shows a snapshot of the Taxonomy Tool for the *business applications* domain.

In the left, the tool shows the taxonomy in its hierarchical form where categories and domains are arranged with respect to various characterization attributes (e.g., number of users, objective, orientation). Overlapping of categories is permitted.

In the right, the tool shows how these characterization attributes can be declared.

Each characterization attribute has its corresponding values, questions and answers in order to make easier the use of the hierarchy.

The questions can be applied at different levels in the taxonomy and some of them are applied in more than one branch.

The Taxonomy Tool offers support for managing and showing graphically taxonomy elements upon the insights of characterization attributes for classifying. However it does not provide support for the procedural taxonomies construction

5.3.4.2 Protégé-2000

Protégé-2000 is a knowledge-based design and knowledge acquisition system developed over more than a decade at Stanford University as a software engineering methodology [Muse-et-al.1995]. It is a free, open source ontology editor and knowledge-base framework. It is based on Java, is extensible, and provides a foundation for customized knowledge-based applications.

It means the tool allows the designer to create custom knowledge-based tools for whatever application is needed.

Protégé assists software developers in creating and maintaining explicit domain models, and in incorporating those models directly into program code. Protégé allows system builders to construct software systems from modular components, including:

- Reusable frameworks for assembling domain models,
- Reusable domain-independent problem-methods that implement procedural strategies for solving tasks [Erikson-et-al.1995]. Protégé allows reuse of frameworks for building domain models through its support for declarative domain ontologies.

The core concept behind the architectural makeup of Protégé-2000 is the design of an ontology or the set of concepts and their relations. This allows for granularity in a domain-specific area, which allows domain experts to use the tool to establish a knowledge base. Using a problem-solving methods specific to that domain, domain experts can then search this knowledge base.

The Protégé-2000 knowledge model has four main concepts that are represented in the software by frames:

- Classes,
- Instances,
- Slots,
- Facets.

The tool uses “classes” and “instances” distinctly and employs a third type of modeling abstraction called “slots”. Classes represent the definitions of concepts, instances represent the specific examples of a concept, slots represent attributes of Esther a class or an instance. Finally there are facets, which are defined as properties of slots, and are constraints on, slot values [Protege].

Fig. 5.4 shows a snapshot of our taxonomy of requirements engineering tool represented by Protégé-2000.

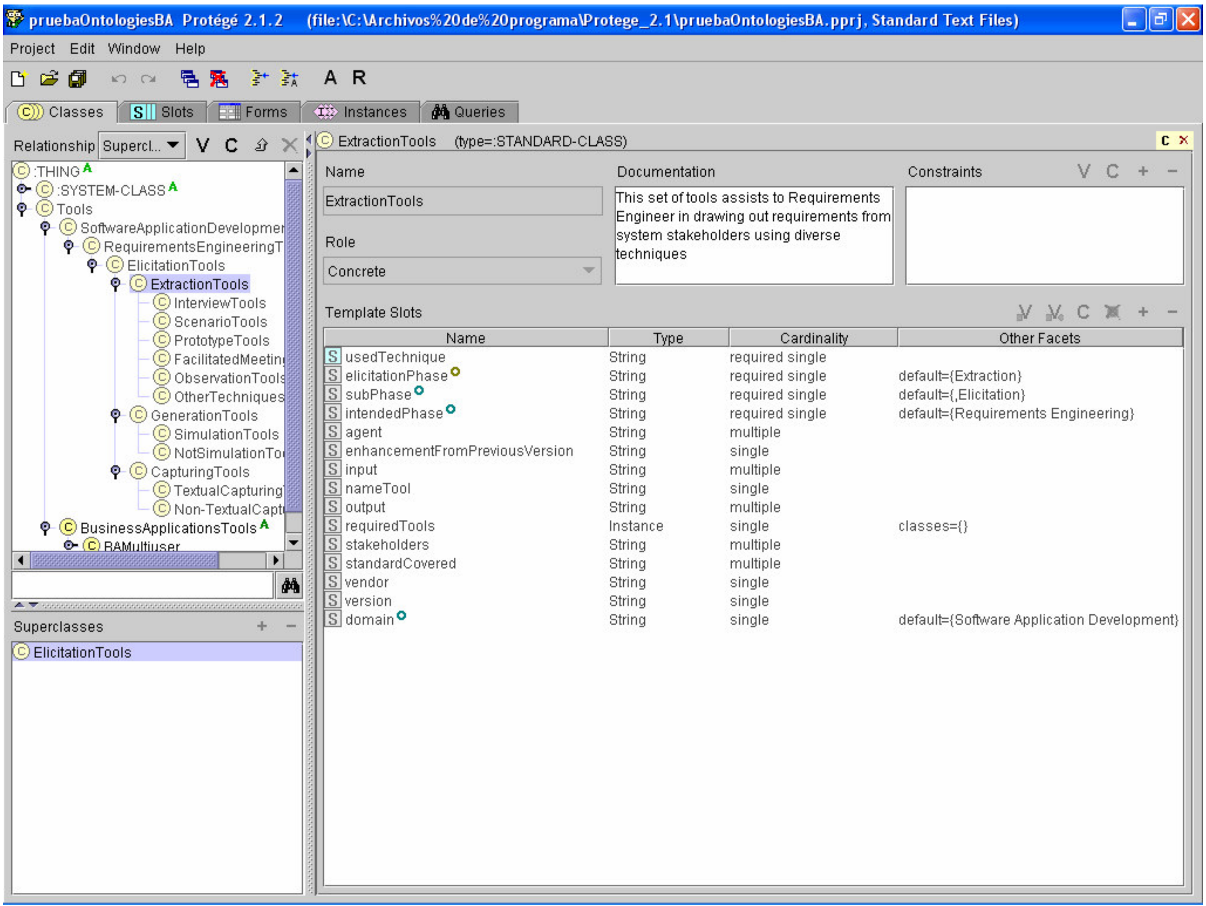


Fig. 5.4 Snapshot of the requirements engineering taxonomy with Prótegé-2000.

In the Protégé approach, the problem-solving methods are first-class entities that have formal parameters that must be mapped to the appropriate referents in the domain knowledge. The separation of problem-solving methods from the domain knowledge on which those methods operate is essential for components reuse.

The language for expressing ontologies in Protégé is a frame-based representation system, in which classes have slots of defined cardinality and data type. Slots may have data that represent instances of other classes in the ontology (e.g., when a class called "prescription" has a slot called "drug-prescribed" that takes on as values instances of another class "drug"). When the data type of a slot is an instance, the ontology-definition language allows the developer to set explicit constraints on the classes whose instances are allowed as values for that slot.

When the data type of a slot is a string, the language allows the user optionally to specify a grammar that restricts the kinds of strings that may be used as values for that slot.

Facets are defined as properties of slots. Multi-inheritance is allowed between classes and every instance of a class is an instance of the superclass of that class.

Classes can also be instances of other classes. The Protégé-2000 environment is divided into *tabs*. Each tab is divided into *panes*. The plug-in architecture of Protégé-2000 makes possible a number of specialized visual tools for entering guideline knowledge. The tool itself is GUI-based so all the design is done using forms and tabs.

The interface is easy-to-use due to the placement of widgets and tabs that give the designer easy access to the tools. The tool also employs a visualization tool that allows the designer to see and editor the ontology structure.

As a conclusion, Protégé-2000 gives the user the ability to construct a domain ontology by using a robust knowledge model. The model uses domain-expert knowledge to design a tool that can be accessed by other applications to tap into its knowledge base.

The case studies we have done, has been represented by means of these explained tools. Therefore, we are evaluating to what extent they can provide support to our purposes and which are not covered as a foundation for the analysis of the implementation of a new tool.

5.4. Third Iteration: Consolidation of the work

In this iteration we expect to achieve:

- Supporting the method with software tool
- The complete validation of our research work
- The final documentation of the PhD. Thesis

5.4.1. Supporting the Method with Software Tool

We expect to begin the implementation or adjustment of some tool for supporting the guided construction of COTS taxonomies based in our method. The results related to the tools analysis that we will obtain from the previous iteration will be a basis for this target, providing the motivation to implement or adjust some existent tool.

5.4.2. Complete Validation of our Research work

While it is easy to propose a method, validation is required before the usefulness of the method may be determined.

Software methods, such as GBTCM that we are proposing, need nearly validation while under development. The initial case studies which seek this validation (that are related in previous section) are best characterized as the formative case studies due to their central role in shaping the method. In contrast, the case studies performed in this phase are best characterized as summative; their primary role will be to validate the method developed during the formative case studies.

Validation of our Goal-Based Taxonomy Construction Method (GBTCM) involves two perspectives of its applications:

- Use GBTCM in the industrial framework, performing real projects of COTS selection, followed by construction of a specific taxonomy based on the results. It allows validating the efficacy of method by means of the usefulness of the resulting product.

- A Case Study involving a large commercial application and multiple stakeholders to evaluate the scalability of the method. Here, we expect to further perform the Software Application Development domain. It allows determining the usefulness and the usability of the method.

- An empirical validation, addressing qualitative research whereby use GBTCM will be compared to the use of other existent alternatives.

While theoretically different in research approach taken, each of these efforts seeks to validate the method.

6. References

[Althoff-et-al.1994] Althoff, K.-D., Wess, S., Bergmann, R., Maurer, F., Manago, M., Auriol, E., Conruyt, N., Traphöner, R., Bräuer, M. and Dittrich, S. ”Induction and Case Based Reasoning for Classification Tasks”. In: H. H. Bock, W. Lenski and M. M. Richter (eds.), Information Systems and Data Analysis, Prospects-Foundations-Applications, Springer Verlag, Berlin-Heidelberg, 1994.

[Alves-Castro2001] Alves, C., Castro, J. “CRE: A Systematic Method for COTS Selection” Proceedings XV Brazilian Simposium on Software Engineering, 2001.

[Anton-Thesis] Antón, A. I.. “Goal Identification and Refinement in the Specification of Software-Based Information Systems”. Ph.D. thesis, Georgia Institute of Technology, June 1997.

[Anton-et-al.1994] Anton, A.I., McCracken, M. W., Potts, C. “Goal Decomposition and Scenario Analysis in Business Process Reengeeniering” Proceedings 6th CAiSE Conference, pp. 94-104, June 1994.

[Arranga2000] Arranga, E. “Cobol Tools: Overview and Taxonomy”. IEEE Software, 17(2): 59-61, 2000.

[Benjamins-Fensel1998] Benjamins, V. R. and Fensel, D., Special Issue on Problem Solving Methods. International Journal of Human Computer Studies, 49(4), 1998.

[Boehm-Abts1999] Boehm, B., Abts, C. “COTS Integration: Plug and Pray?” IEEE Computer, Vol. 32, No. 1; January 1999 pp. 135-138

[Boehm-et-al.1998] Boehm, B., Abts, C., Bailey, E. “COCOTS Software Integration Cost Model: an Overview” Proceedings California Software Symposium. October 1998.

[Boloix-Robillard1995] Boloix, G., Robillard, P. “A Software System Evaluation Framework” IEEE Computer Vol. 28, No. 12, pp. 17-26, December 1995.

[Briand1998] Briand, L.C. “COTS Evaluation and Selection” Proceedings International Conference on Software Maintenance, 16-19 November 1998, Bethesda, MD, pp. 222-223

[Brownsword-et-al.2000] Brownsword, L., Oberndorf, T., Sledge C.A. “Developing New Processes for COTS-Based Systems” IEEE Software , Vol. 17, No. 4; July-August 2000. pp. 48-55

[Burgués-et-al.2002] Burgués, X., Estay, C., Franch, X., Pastor, J.A., Quer, C. “Combined Selection of COTS Components”. Proceedings 1st International Conference on COTS-Based Software Systems (ICCBSS 2002). Orlando, Florida. LNCS 2255, 2002

[Carney-Long2000] Carney D., Long F.:What Do You Mean by COTS? Finally a Useful Answer. IEEE Software, 17 (2), March/April 2000

[Carvallo-et-al2004a] Carvallo, J.P., Franch, X., Quer, C., Torchiano, M. "Characterization of a Taxonomy for Business Applications and the Relationships Among Them". Proceedings of 3rd International Conference on COTS-Based Software Systems (ICCBSS), LNCS 2959, 2004.

[Carvallo-et-al.2004b] Carvallo, J.P., Franch, X., Grau, G., Quer, C. "QM: A Tool for Building Software Quality Models". In Proceedings of the 12th IEEE Requirements Engineering International Conference RE 2004.Kyoto, Japan. IEEE Computer Society, 2004.

[Carvallo-Franch-Quer2003] Carvallo, J.P., Franch, X., Quer, C. "Defining a Quality Model for Mail Servers" Proceedings of 2nd International Conference on COTS-Based Software Systems (ICCBSS), LNCS, Ottawa, Canada, February 2003.

[CART] Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J., Classification and Regression Trees. Belmont, CA: Wadsworth International Group, 1984.

[CBSE] CBSE web page. www.cbsenet.org/pls/CBSEnet/ecolnet.home, last accessed Feb. 2005.

[CHAOS] Standish Group Report http://www.standishgroup. CHAOS Survey 1994.

[Chung-Cooper2004] Chung, L., Cooper, K. "Defining Goals in a COTS-Aware Requirements Engineering Approach". System Engineering, Vol. 7, No.1, 2004

[Clancey1985] Clancey, W. J. "Heuristic Classification" Artificial Intelligence, 27, pp. 289-350, 1985

[CLARiFiProject] CLARiFi Project home page available at http://clarifi.eng.it/

[ClassSoc] Classification Society of North America http://www.classification-society.org/csna/csna.html]

[CompSource] ComponentSource web page. www.componentsource.com, last accessed February 2005.

[C4.5] Quinlan, J.R. "C4.5: Programs for Machine Learning" Morgan Kauffman, 1993.

[Dardenne-et-al.1993] Dardenne. A, vam Lamswerde, A., Fickas, S. "Goal-Directed Requirements Acquisition" Science of Computer Programming, 20, 1993 pp. 3-50.

[Easterbrook1991] Easterbrook, S.M., "Elicitation of Requirements from Multiple Perspectives" PhD Thesis, University of London, London, UK, 1991.

[eCOTS] eCOTS web page. www.ecots.org, last accessed May 2005.

[eCOTSproject] eCOTS Project. (2002) Available at http://www.industrie.gouv.fr/rntl/FichesA/E-Cots.html (in French).

[Erikson-et-al.1995] Erikson, H., Tu, S., Shahar, Y., Musen, M.A. "Ontology-Based Configuration of Problem-Solving Methods and Generation of Knowledge-Acquisition Tools: Application of PROTÉGÉ-II to Protocol-Based Decision Support", Artificial Intelligence in Medicine, http://www.smi.standford.edu/pubs/SMI_Reports/SMI-94-0520.pdf, 1995.

[Franch-Maiden2003] Franch, X., Maiden, N. "Modelling Component Dependencies to Inform their Selection". Proceedings 2nd International Conference on COTS-Based Software Systems (ICCBSS 03). Ottawa, Canada. February 2003.

[Forrester] Forrester Research Inc. web page. www.forrester.com, last accessed February 2005.

[Gartner] Gartner Inc. web page. www.gartner.com, last accessed May 2005.

[Genium] Genium Software Development http://www.genium.dk/index. xml last accessed February 2005.

[Glass-Vessey1995] Glass, R.L., Vessey, I. "Contemporary Application-Domain Taxonomies" IEEE Software July 1995.

[Goguen-Linde1993] Goguen, J.A., Linde, C. "Techniques for Requirements Elicitation" Proceedings IEEE Symposium on Requirements Engineering, San Diego, California (IEEE Computer Society Press 1993) pp. 152-164.

[Grau-et-al.2004] Grau, G., Carvallo, J.P., Franch, X., Quer, C. "DesCOTS: A Software System for Selecting COTS Components". In Proceedings of the 30th EUROMICRO Conference. Rennes, France. Pages 118-126. IEEE Computer Society. 2004.

[Green1994] Green, S. "Goal-Driven Approaches to Requirements Engineering" Technical Report DoC TR-93-42 1994, Imperial College of Science, Technology and Medicine, Department of Computing Technical Report, London, UK, 1994.

[Gruber1993] Gruber, T.R. "Toward Principles for the Design of Ontologies Used for Knowledge Sharing", KSL-93-04, Knowledge Systems Laboratory, Standford University.

[Gruber1995] Gruber, T. R.,"Toward Principles for the Design of Ontologies Used for Knowledge Sharing", International Journal of Human-Computer Studies 43(5/6), pp. 907-928, 1995.

[Guarino-et-al.1999] Guarino, N., Masolo, C. and Vetere, G.,"Ontoseek: Using Large Linguistic Ontologies for Accessing On-line Yellow Pages and Product Catalogs" IEEE Intelligent Systems, 14(3), pp. 70–80, 1999.

[Han-Kamber2001] Han, J., Kamber, M. "Data Mining: Concepts and Techniques" Academic Press, Morgan Kaufmann Publishers, 2001.

[Hissam-et-al.2002] Hissam, S., Moreno, G., Stafford, J., and Wallnau, K. "Packaging Predictable Assembly" Proceedings IFIP/ACM Working Conference Component Deployment, 21-22 June 2002, Berlin, Germany, LNCS 2370.

[ID3] Quinlan J.R. "Induction of Decision Trees" Kuwler Academic Publishers ISSN 0885-6125 (Paper) 1573-0565, March 1986.

[*i**Web] *i** web page, http://www.cs.toronto.edu/km/istar/, last accessed May 2005.

[Incose] INCOSE web page. www.incose.org, last accessed May 2005.

[IT] IT products guide web page. http://productguide.itmanagersjournal.com, last accessed April 2005.

[ITPapers] ITPapers web page. www.itpapers.com, last accessed May 2005.

[Jaccheri-Torchiano2002] Jaccheri, L., Torchiano, M. "Classifying COTS Products" Proceedings 7th European Conference of Software Quality (ECSQ), Helsinki, Finland, 2002.

[Kavakli-Loucopolos2004] Kavakli, E., Loucopoulos, P. "Goal Modeling in Requirements Engineering: Analysis and Critique of Current Methods"

[Kelly1995] Kelly, G.A. "The Psychology of Personal Constructs" New York, W.W. Norton.

[Kontio1996] Kontio, J. "A Case Study in Applying a Systematic Method for COTS Selection". Proceedings 18th International Conference on Software Engineering, IEEE Computer Society Press.

[Kunda-Brooks1999] Kunda, D., Brooks, L. "Applying Socio-Technical Approach for COTS Selection" Proceedings UK Acad Inform Syst Conference, 7-9 April 1999, University of York, pp. 552-565

[Lawlis-et-all2001] Lawlis, P., Mark, K., Thomas, D. Courtheyn, T. “A Formal Process for Evaluating COTS Software Products”. IEEE Computer, Vol. 34. No. 5, May 2001.

[Maiden-Rugg1996] Maiden, N.A.M, Rugg, G. “ACRE: Selecting Methods for Requirements Acquisition” Software Engineering Journal, pp.183-192, May 1996.

[Maiden-Ncube1998] Maiden, Ncube, C. “Acquiring Requirements for COTS Selection”. IEEE Software Vol. 15, No. 2, March/April 1998.

[Maiden-Kim-Ncube2002] Maiden, N., Kim, H., Ncube, C. “Rethinking process guidance for Software Component Selection”. LNCS 2255, J.C. Dean and A. Gravel (Editors), Springer-Verlang, New York, 2002, pp. 151-164.

[Meyers-Oberndorf2002] Craig Meyers, B., Oberndorf, P. “Managing Software Acquisition”. SEI Series in Software Engineering, 2002

[Morisio-et-al.2000] Morisio, M., Seaman, C.B., Parra, A., Basili, V., Kraft, S., and Condon, S. “Investigating and Improving a COTS-Based Software Development Process”. Proceedings International Conference on Software Engineering, 4-11 June 2000, Limerick, Ireland, pp. 32-41

[Morisio-Tsoukias1997] Morisio, M., and Tsoukias, A. “IusWare: A Methodology for the Evaluation and Selection of Software Products”, IEEE Software Engineering Vol. 144, No. 3, 1997, pp. 162-174

[Morisio-Torchiano2002] Morisio, M., Torchiano, M. “Definition and Classification of COTS: A Proposal”. Proceedings 1st Conference on COTS-Based Software Systems (ICCBSS), Orlando, Florida; 2002.

[Motta-1999] Motta E. ”Reusable Components for Knowledge Models: Principles and Case Studies in Parametric Design”, IOS Press 1999.

[Motta-et-al.2000] Motta E., Buckingham-Shum, S. and Domingue, J. “Ontology-Driven Document Enrichment: Principles, Tools and Applications” International Journal of Human-Computer Studies, 52(6), pp. 1071-1109, 2000.

[Muse-et-al.1995] Musen, M.A., Gennari, J.H., Eriksson, H., Tu, S.W., Puerta, A.R. “PROTÉGÉ-II:Computer Support for Development of Intelligent Systems from Libraries of Components”, Proceeding 8th World Congress on Medical Informatics, pp. 766-770, Vacouver, 1995.

[Ncube-Maiden1999] Ncube, C., Maiden, N. “Guiding Parallel Requirements Acquisition and COTS Software Selection” Proceedings of the IEEE International Symposium on Requirements Engineering 1999, pp. 133-140

[Oberndorf-Brownsword1997] Oberndorf, P., Brownsword, L. “Are You Ready for COTS?” Software Institute Engineering. August 1997.

[Ochs-et-al.2001] Ochs, M.A., Pfahl, D., Chrobok-Diening, G., Nothhelfer-Kolb, B. "A Method for Efficient Measurement-based COTS Assessment and Selection-Method Description and Evaluation Results" Proceedings IEEE 7th International Software Metrics Symposium, London, England, 2001, pp. 285-296.

[Potts-Takanashi-Anton1994] Potts, C., Takanashi, K., Antón, A. "Inquiry-Based Requirements Analysis", IEEE Software, 11 (2), March 1994.

[Prieto-Díaz1987] dPrieto-Díaz, R., Freeman, P. "Classifying Software for Reusability" IEEE Software. January 1987 pp. 6-16

[Protege] Protégé http://protege.standford.edu

[Rugg-et-al.1992] Rugg, G, Corbrigge, C. Major, N.P., Burton, A.M., Shadbolt, N.R. "A Comparison of Sorting Techniques in Knowledge Elicitation" Knowledge Acquisition, Vol. 4, No. 3 pp. 279-291, 1992.

[Rugg-McGeorge1997] Rugg, G, McGeorge, P. "The Sorting Techniques: a Tutorial Paper on Card Sort, Picture Sorts and Item Sorts", Expert Systems, May 1997, Vol 14, No.2.

[RUP] Jacobson, I., Booch, G., Rumbaugh, J. " The Unified Software Development Process". Addison Wesley Longman, Reading, MA, 1999.

[Seacord-et-al.2001] Seacord, R., Mundie, D., and Boonsiri, S. "K-BACRR: Knowledge-Based Automated Component Ensemble Evaluation" Proceedings Workshop on Component-Based Software Engineering, 6-8 September 2001. Warsaw, Poland, pp. 56-63.

[SEI] Software Engineering Institute. COTS-Based System Initiative (CBS) http://www.sei.cmu.edu/cbs/

[SEI-a] Software Engineering Institute. COTS-Based System Initiative (CBS) http://www.sei.cmu.edu/cbs/overview2.html

[SEI-b] Software Engineering Institute. COTS-Based System Initiative (CBS) http://www.sei.cmu.edu/cbs/overview.html

[Schreiber-et-al.2000] Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W. and Wielinga, B. "Knowledge Engineering and Management", MIT Press. Cambridge, MA, 2000.

[Sommerville-et-al.1993] Sommerville, I., Rodden, T., Sawyer, P., Bentley, R., Twidale, M. "Integrating Ethnography Into the Requirements Engineering Process" Proceedings IEEE Symposium on Requirements Engineering, San Diego, California (IEEE Computer Society Press 1993) pp. 165-173.

[Stefik1995] Stefik, M. "Introduction to Knowledge Systems" Morgan Kaufmann, San Francisco, CA 1995.

[SWEBOK] SWEBOK web page, www.swebok.org, last accessed February 2005.

[UML] UML 2.0 Specifications http://www.uml.org/, last accesed February 2005

[vanderVet-et-al.1994] van der Vet, P.E., Speel, P.H., Mars, N.J.I. “The Plinius Ontology of Ceramic Materials” Proceedings 11th European Conference on Artificial Intelligence (ECAI 94), Workshop on Comparison and Implemented Ontologies, Amsterdam, The Netherlands, 1994.

[vanHeijst1995] van Heijst “The Role of Ontologies in Knowledge Engineering” PhD thesis, University of Amsterdam, May 1995.

[VanLamsweerde2001] van Lamsweerde, A. “Goal-Oriented Requirements Engineering: A Guided Tour” Proceedings 5th IEEE International Symposium on Requirements Engineering, Toronto, August 2001, pp 249-263.

[Voas1998] Voas, J. “The Challenges of Using COTS Software in Component-Based Development”. IEEE Computer, Vol. 31 No.6; June 1998. pp. 44-45

[Wiellinga-et-al.1998] Wielinga, B. J., Akkermans, J.K., Schreiber, G. “A Competence Theory Approach to Problem Solving Method Construction”, International Journal of Human-Computer Studies, No.49, pp. 315-338, 1998.

[Winston-et-al.1987] Winston, M.E., Chaffin, R., Hermann, D.J. “A Taxonomy of Part-Whole Relations” Cognitive Science 11:417-444.

[Yu1997] Yu, E., “Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering”, Proceedings 3rd IEEE International Symposium on Requirements Engineering (ISRE), 1997.

[Yu-Mylopoulos1994] Yu, E., Mylopoulos, J. “Using Goals, Rules, and Methods to Support Reasoning in Business Process Reengineering” Proceedings 27th Hawaii International Conference on System Sciences, Vol. IV: Information Systems: Collaboration Technology Organizational Systems and Technology, January 1994.

[Yu-Thesis] Yu, E. “Modelling Strategic Relationships for Process Reengineering” PhD Thesis, University of Toronto, 1995.

6.1. Author´s References

6.1.1. Main Topic (Taxonomies Construction)

[Ayala-Franch2005] Ayala, C., Franch, X., "Transforming Software Package Classification Hierarchies into Goal-Based Taxonomies" To be appear in Proceedings 16th International Conference on Database and Expert Systems Applications (DEXA 05) to be held in August 2005 (Copenhagen, Denmark). Lecture Notes in Computer Science.

[Ayala-Botella-Franch2005r] Ayala, C., Botella, P., Franch, X., "Construction of a Taxonomy for Requirements Engineering Commercial-Off-The-Shelf Components" Journal of Computer Science and Technology. Special Issue on Software Requirements Engineering. Vol. 5 – No. 2 – July 2005. ISSN 1666-6038.

[Ayala-Botella-Franch2005a] Ayala, C.P, Botella, P., Franch, X., "On Goal-Oriented COTS Taxonomies Construction" Proceedings 4th International Conference on COTS-Based Software Systems, Lecture Notes in Computer Science 3412, Bilbao, Spain; February 2005.

[Ayala-Botella-Franch2004b] Ayala, C.P, Botella, P., Franch, X., "Construcción de una Taxonomía de Componentes COTS Orientados a la Gestión de Requisitos", Proceedings VII Workshop on Requirements Engineering (WER 2004), ISBN:950-658-147-9, Tandil, Argentina, December 2004.

[Ayala-LSI-05-25-R] Ayala, Claudia, P. "Goal-Based Reasoning in the Construction of Taxonomies for COTS Componentst" Research Report, Universitat Politècnica de Catalunya, Departamento de Lenguajes y Sistemas Informáticos, LSI-05-25-R.

[Ayala-Botella-Franch2004a] Ayala, C.P, Botella, P., Franch, X., "Goal-Based Reasoned Construction of Taxonomies for the Selection of COTS Products", Proceedings 8th Multi-Conference on Systemics, Cybernetics and Informatics (SCI), ISBN:980-6560-13-2, Orlando, Florida, July 2004.

[Ayala-LSI-04-03-R] Ayala, Claudia, P. "Towards the Definition of a Taxonomy for the COTS Product´s Market" Research Report, Universitat Politècnica de Catalunya, Departamento de Lenguajes y Sistemas Informáticos, LSI-04-03-R.

6.1.2. Goal Modeling Research

[Grau-et-al.2005] Grau, G., Franch, X., Mayol, E., Ayala, C. P., Cares, C., Carvallo, J.P., Haya, M., Navarrete, F., Botella, P., Quer, C. "RiSD: A Methodology for Building *i** Strategic Dependency Models" To be appear in

Proceedings 17th International Conference on Software Engineering and Knowledge Engineering (SEKE 2005). July 2005. Taipei, Taiwán; China.

[Ayala-Cares-et-al.2005] Ayala, C. P., Cares, C., Carvallo, J.P., Grau, G., Haya, M., Salazar, G., Franch, X., Mayol, E., Quer, C. “A Comparative Analysis of *i**-Based Goal-Oriented Modeling Languages”. To be appear in Proceedings International Workshop on Agent-Oriented Software Development Methodology (AOSDM), July 2005. Taipei, Taiwán; China.

[Ayala-Cares-et-al.2004] Ayala, C. P., Cares, C., Carvallo, J.P., Grau, G., Haya, M., Salazar, G., Franch, X., Mayol, E., Quer, C. “Análisis Comparativo de Lenguajes de Modelado Orientados a Objetivos basados en *i**”. 4as. Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería del Conocimiento (JIISIC 2004), ISBN: 84-688-8957-1, November 2004, Madrid, Spain.