Domain Analysis for Supporting Commercial Off-The-Shelf Components Selection

Claudia Ayala, Xavier Franch

Universitat Politècnica de Catalunya
UPC-Campus Nord (Omega), 08034 Barcelona, Spain
{cayala,franch}@lsi.upc.edu

Abstract. Though new technological trends and paradigms arise for developing complex software systems, systematic reuse continues to be an elusive goal. As a consequence, the need for designing effective strategies for enabling large-scale reuse, whilst overcoming the risks involved in the use of a particular technology, still remains. In this context, the adoption of the Commercial Off-The-Shelf (COTS) technology introduces many challenges that still have not been fully overcome, such as the lack of comprehensive mechanisms to record and manage the required information for supporting COTS components selection. In this paper we present a domain analysis strategy for gathering the information needed to describe COTS market segments in a way that COTS components selection becomes more effective and efficient. Due to the diversity of the information to capture, we propose different dimensions of interest for COTS selection that are covered by different domain models. These models are articulated by means of a single framework based on a widespread software quality standard.

1. Introduction

Systematic reuse is based on the observation that quality and productivity can be significantly increased by shifting the focus of software engineering to a domain-centered view by means of building an infrastructure support. The engineering discipline concerned with building these optimal reusable assets is called *domain engineering* [1]. Domain engineering supports the notion of *domain*, a set of applications that use common concepts for describing requirements, problems, capabilities and solutions.

Particularly, being part of domain engineering, *domain analysis* has been identified as a major factor in the success of software reusability. Domain analysis refers to the process of acquiring and consolidating information about an application domain, so that reusable infrastructure can be designed reliably [2]. Its purpose is to identify the basic elements of the domain, to organize an understanding of the relationships among these elements, and to represent this understanding in a useful way by means of different types of domain models [3]. The different existing views on domain modelling (e.g., [4], [5], [1]) share then the same goal: to facilitate quality software development by reusing the knowledge of the addressed domain.

Reuse is not a context-independent activity. The type of artifact to be reused impacts on the reuse models to be adopted and the reuse processes to be undertaken; therefore, the reuse discipline has to evolve as new paradigms and artifacts emerge. We are interested in one particular case of those software artifacts, namely *Commercial Off-The-Shelf* (*COTS*) *components*. A COTS component is defined as “a software product that is sold, leased or licensed to the general public, offered by a vendor trying to profit from it, supported and evolved by the vendor who retains the intellectual property rights, available in multiple identical copies and used without source code modification by a consumer” [6].

Successful COTS-based systems development requires a unique set of activities to be performed, among which we find the selection of the components themselves, defined as the process of searching candidates and evaluating them with respect to the system requirements. An effective and efficient COTS selection process is essential to deliver full potential to the COTS technology. Several COTS selection methodologies, processes and techniques have been formulated (see [7] for a recent survey). However, though these methods have achieved significant results, they are mainly oriented to individual selection processes. Even in the cases in which a reuse infrastructure is suggested (e.g. OTSO, CARE, PECA), no real support or precise guidelines are offered. Therefore, we may conclude that current COTS selection methods do not provide adequate support to an organization that needs to carry out continuously COTS selection processes (e.g., a consultant company, a third-party software provider, an IT department of a big corporation, etc.)

To solve this problem, it seems feasible to use domain analysis for recording and structuring information about COTS components. However, as far as we know, COTS technology issues have not been explicitly addressed in the domain analysis discipline (although of course many concepts of domain analysis apply to this particular case).

The goal of this report is to present a particular strategy of domain analysis for supporting COTS components selection. In this strategy we produce several domain models covering different dimensions that capture and represent the most important aspects of a particular COTS segment in the COTS marketplace. All the models are synchronized using a unifying framework. We use widespread notations and standards to represent the dimension models. The domain analysis activity is part of our GOThIC (Goal-Oriented Taxonomy and reuse Infrastructure Construction) method, a prescriptive goal-oriented method for building and maintaining a reliable reuse infrastructure in which COTS segments are arranged to form a taxonomy whose nodes are decorated with the domain models built.

2. COTS Technology Characteristics

To tailor conveniently our domain analysis strategy, we need to take into account the most critical factors and characteristics of the COTS marketplace, which are:

- *Growing size and diversity of the COTS marketplace*. New and improved products and technologies are continuously offered. Existent market segments offer more and more products. At the time being, mobile technologies are a good example of both situations. This exponential grow makes it difficult to have a timely description of the COTS marketplace.

- *Marketplace evolvability*. New products, and new versions of existing products, continuously appear not only for improving features of their market segment, but also for offering new services which were previously considered as belonging to other segments (e.g. current mail server systems usually provide instant messaging facilities, even video-conferencing services). This fact points out the need to separate conceptually the COTS components from the services that they cover.
- *Implicit relationships among components*. COTS components are not designed to work isolated, but in collaboration with others, which results in many dependencies among them [9]. For instance, document management systems need document imaging tools for scanning and storing paper documents. Finding this information in the marketplace is not easy.
- *Lack of standards for COTS descriptions*. Component suppliers and brokers do not have a standard for describing components resulting in a variety of documentation styles difficult to compare. A study conducted in [10] evidenced that the required COTS information is highly widespread and unstructured, becoming very difficult to obtain.
- *Lack of reliability information of the vendor supplier*. Supplier information of course tends to highlight strengths and hide weaknesses of the licensed components and services offered.

To sum up, as other recent studies have concluded [11], we may say that the current gap among the COTS marketplace and the informational needs of COTS selection methods is too wide and therefore methods to bridge this gap are required.

3. The GOThIC Method

As a response to the need of organizing the knowledge of the COTS marketplace in a structured manner, we have formulated the GOThIC method [8]. The ultimate goal of GOThIC is to guide the construction and maintenance of goal-oriented taxonomies that describe the contents of the COTS marketplace. The method is articulated by means of several activities, such as the exploration of information sources, the identification of goals and their hierarchization. Among these activities, we also find domain analysis of the COTS marketplace segment being addressed by the taxonomy. This activity has the mission of producing a domain model (representation of important aspects of a COTS segment) that serves as the basis to gain knowledge for identifying the correct goals and to build a reuse infrastructure with several kinds of reusable assets of interest for COTS selection processes.

From an operational point of view, the ultimate goal of the GOThIC method is to populate a knowledge base with data according to the UML [12] conceptual model sketched in Fig. 1. At the heart of this model lies the taxonomy composed of two types of nodes, market segments and categories, which are characterized by their goals. Market segments are the leaves of the taxonomy, whilst categories serve to group related market segments and/or subcategories (e.g., the category of communication infrastructure systems or financial packages). From a semantic point of view, market segments stand for the basic types of COTS components available in the marketplace (e.g., the domain of anti-virus tools or spreadsheet applications), i.e. atomic entities covering a significant group of functionality such that their decomposition would yield to too fine-grained domains. As a consequence, COTS components are associated with market segments and not with categories (although an indirect relationship exists, because market segments belong to categories). Components may cover more than one market segment.

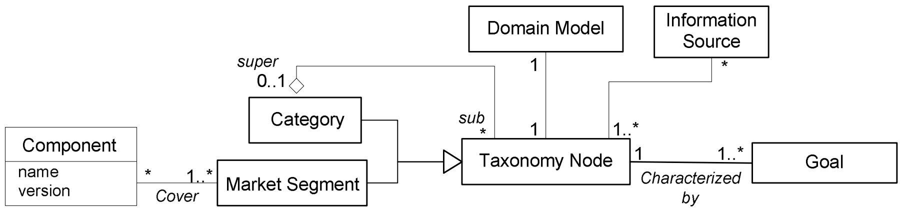


Fig. 1. Conceptual model for goal-oriented COTS taxonomies in the GOThIC method: overview

Taxonomy nodes have a generic domain model bound, which is built during the domain analysis activity. Their construction is a result of the integration of diverse models which are designed from the analysis of some information sources which are gathered, analyzed and prioritized according to several characteristics of the taxonomy construction project. The taxonomy built with GOThIC may then be browsed during COTS selection to locate the market segment (or segments) of interest. Once found, the domain models bound may be used to obtain the appropriate criteria for selecting the most appropriate component. In the rest of the paper we focus on the form that this domain model takes.

4. Domain Analysis for Supporting COTS Selection: Dimensions

In the previous sections we have justified the convenience of having domain models for describing COTS marketplace segments. In this section we identify several dimensions of interest for describing the information of COTS components required during COTS selection processes. Each dimension will be described by a model. To guide the identification of the dimensions, we analyse the informational needs of COTS selection processes that have been reported in the literature, as well as our own experiences in the field (e.g. [13, 14, 15, 16]).

Fundamental concepts.

Every single COTS segment defines lots of concepts that are used over and over. Anti-virus tools have "viruses", e-mail systems have "messages" and "folders", etc. These concepts may be related in many ways, e.g. "messages" are "stored" inside "folders". A poor knowledge of these fundamental concepts may interfere with the efficiency and effectiveness of COTS selection processes, especially taking into account some of the COTS technology characteristics mentioned in section 2 (e.g. growing size and diversity). Currently, it is not usual to find places in the COTS marketplace where fundamental concepts are stated. Most normally, one may find items (products, services, etc., belonging to one or more market segment) whose description uses some terms in a rather obscure way, making those descriptions difficult to use (especially when comparisons among candidates are needed), customize and evolve as the marketplace does [17]. Also the same concept may be denoted by different names in different products or even worse, the same term may denote different products. Therefore a model for representing fundamental concepts is needed. The purpose of this model is to settle the scope of a particular segment, to define its main concepts (both as a vocabulary and as a semantic model) and the relationships that facilitate the understanding of the domain as a whole. To build this model, information sources such as standards and textbooks are useful. We recommend to choose one of these most trustable sources as starting point, then to synthesize the corresponding dimension of the domain model, and last to calibrate this dimension with other informational sources. The resulting model can there be used as a reference framework for the segment.

Functionality.

COTS components have their functionality already built-in. Hence, instead of traditional requirements that specify "must" and "should" needs, requirements for COTS-based systems articulate broad categories of needs and possible trade-offs. Actually, for enabling COTS searching, most of the existent categorization proposals are based on COTS components functionality as characterization attributes for being mapped against the user expected functionality. Thus, COTS functionality is a primary source of information for COTS selection processes. Consequently, a domain model must cover this dimension. But a good balance is needed. On the one hand, the most representative functionalities of a particular segment should be included (e.g., virus repair, automatic resending of messages) and described up to a level of detail that enables efficient survey and evaluation of particular COTS components. On the other hand, if too much detail is given, several mentioned obstacles, remarkably growing size and evolvability of the COTS marketplace, are harder to overcome, since a lot of information would need to be updated continuously. Also, too much detail may commit the description of the functionality to the behaviour of particular components.

Quality of service.

The role of information about quality of service becomes utterly important because COTS components have their functionality already built-in. Therefore, quality factors are likely to break the tie when several COTS candidates provide the required functionality. In particular, quality requirements have been recognized as crucial by the methods and processes proposed so far for driving COTS selection [18]. Thus, efforts are required to obtain reliable and comprehensive descriptions of COTS quality of service in an efficient way. We propose then a dimension for stating quality of service. The resulting model needs to offer a structured description of the COTS segment addressed, organizing the different quality factors hierarchically (e.g., Throughput and Response Time as subfactors of Time Efficiency). The model should also include metrics for the quality factors. The resulting model may serve therefore as a framework in which particular COTS components may be evaluated and compared to user requirements during selection processes.

Non-technical description.

Despite of the fact that the evaluation of candidate COTS components from a technical point of view (functionality and quality of service) is necessary, experiences in COTS selection show that non-technical information (i.e., information that does not refer directly to the intrinsic quality of software, but to its context, including economic, political and managerial issues) must be taken into account and, in fact sometimes it is even more important than the technical information [19]. As a result, we need to record this information as part of the domain model. This new dimension must distinguish several concepts and focus on the commercial nature of COTS components, stating information about licensing issues, provider reputation, post-sale supporting services, etc. One should be aware that part of the information may be difficult to obtain (e.g., finance information of the provider company) and the corresponding factor may not be included in the model for this reason.

Interoperability.

The analysis of any COTS market segment shows that some relationships among components exist. We have analyzed the types of dependencies that may exist and we have concluded that a COTS component may need another for: *enabling its functionality* (e.g., document management tools need workflow technology to define life cycles); *complementing its functionality with an additional feature*, not originally intended to be part of its suitability (e.g., a web page edition tool can complement a web browser to facilitate web page edition); *enhancing its quality attributes* (e.g., resource utilization can be improved significantly using compression tools). However, in the context of COTS selection, interoperability has been dealt with in a case-by-case basis. Furthermore, some of the COTS selection methods proposed so far just address single component selection, they do not even address the need to select a suite as final solution. Therefore we propose a new dimension to cover this need, otherwise COTS selection becomes not trustable. It is worth remarking that, since we are describing not a particular COTS component but a whole segment, interoperability issues must not be stated in much detail (e.g., data formats, API specificities, etc.); instead the model should include the needs and expectations that one type of component has on others in a very high-level way.

Fig. 2 shows graphically the informational dimensions required for evaluating COTS components.

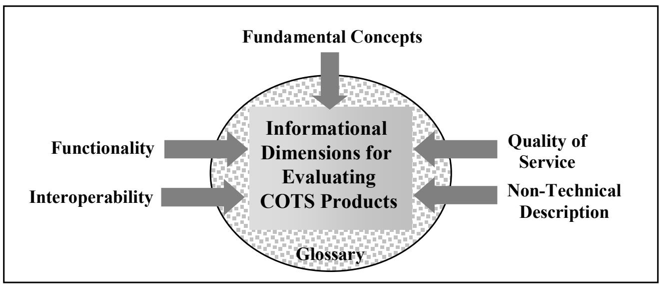


Fig. 2. Informational dimensions for evaluating COTS components

5. Domain Analysis for Supporting COTS Selection: Models

Taking into account the informational dimensions required for the COTS technology, the next step is to decide which are the most appropriate types of models to represent them. A first observation is that, due to the diversity of the different dimensions, various types of domain models will be probably needed, therefore a study of the state of the art in domain analysis is needed.

In the domain analysis field, a variety of methods and techniques have been proposed as: FODA, DARE, ODM, DSSA and PLUS (see [20] and [26] for a survey) which use a diversity of different types of artefacts and mechanisms to record the knowledge that range from the traditional requirements models (namely models of data, behaviour, and function), as Data Flow diagrams [21], Entity-Relationship (ER) models [22], Object Oriented models [23], UML models, Scenarios [24], and Feature models [25], to UML metamodeling techniques [26] as well as more elaborated UML extensions and stereotypes [27], [28] for dealing with domain structural elements, relations and domain variability, this last is commonly represented into variability models [26].

In practice, these proposals vary in their terms, notations, and emphases, but in general they are focused on designing product lines or product families for promoting reusability between applications by means of a planned reuse plan [26]. Hence, they do not address in an optimal way the fundamental informational needs and facts for assessing COTS components in terms of expressiveness and adequateness, structure, and compatibility as required by the COTS technology. Furthermore, as far as we know, none of these approaches has examined in depth the special kind of relationships and information that the COTS technology requires, for instance those relationships that enable interoperability among components and non-technical information.

Therefore, it is a fact that existent domain analysis strategies have to be somehow adapted and complemented to fully deal with the COTS technology characteristics [20], [29].

Fundamental concepts.

Two types of artifacts are adequate for representing fundamental concepts. On the one hand, conceptual data models or feature-oriented models to express the semantic meaning of the terms in the market segment together with their relationships. On the other hand, a glossary to set up a vocabulary of the domain with information about synonymous and other lexical relationships. One could also think of the general concept of ontology [30] for embracing both needs, lexical and semantic information.

We have chosen UML class diagrams [12] for representing the semantic information due to its expressiveness and acceptance in the community. As for the glossary, the Language Extended Lexicon (LEL) [31] approach provides an adequate level of service since it allows to capture the meaning and fundamental relationships of the particular symbols (words or phrases) of the domain. The glossary will include at least the terms that appear in the rest of the models (e.g., the names of classes, attributes and associations of the UML class diagram).

Functionality.

For describing functionality, any approach based on the concept of scenario seems a good option. As commented in section 3, the important point is to use the right level of detail. We propose the use of UML use case diagrams [12] for defining the functionalities of the COTS segment and a brief format of use cases [32] for describing them individually.

Quality of service.

Quality models [13] provide a measurable framework which precisely defines and consolidates the different views of quality (e.g. performance, reliability, integrity, etc) which are required for COTS components evaluation. Among the different existing proposals, we have adopted the ISO/IEC 9126-1 standard [33] for several reasons, remarkably: it provides a two-level departing catalogue but at the same time it is highly customizable to each different COTS segment; there are some metrics already defined for this standard; and it is widespread. In the next section we give more details of this model.

Non-technical description.

Not only in the domain analysis context but in general, it is not usual to find models for representing non-technical information. Usually some categories are identified and for each of them, a list of non-technical factors identified. We have identified 3 high-level factors and 15 second-level subfactors referring to supplier information (e.g., financial information), cost information (e.g., licensing schemes) and other non-technical information about the product (e.g., history of versions). See [18] for more details.

Interoperability.

Interoperability of COTS components is usually described by means of APIs or data formats. However, as already explained in section 3, we are interested in describing not particular COTS components but the general behavior of all the components belonging to a COTS segment, therefore we need more abstract descriptions. The combination of goal- and agent-oriented models provides a good response to our needs.

Goals allow expressing needs and expectations in a high-level way, whilst agents are an appropriate way to model COTS segments. Then, one COTS segment may state that depends on another to attain a goal. Thus, we have chosen *i** Strategic Dependency (SD) models [34] because they have proven to be useful for representing these dependency relationships. COTS segments may be represented by means of *i** actors; for dependencies, *i** allows stating four different types of relationships: goal dependencies, when an actor depends on another to attain a goal; task dependencies, when an actor requires another to perform an activity in a given way; resource dependencies, when an actor depends on another for the availability of some data; and soft goal dependency, when an actor depends on another to achieve a certain level of quality of service. Actors and dependencies together provide the level of detail that we need in our domain model.

Table 1 summarizes our proposal and makes clear the relationships with other domain analysis approaches.

Table 1. Summary of domain analysis practices for representing COTS dimensions

COTS Dimension	Domain Analysis Practices	Our approach
Fundamental Concepts	ER Models [22], Feature Models[25], UML Diagrams [27,28], etc.	UML Class Diagrams + LEL
Functionality	Data Flow Diagrams [21], Scenarios [24], UML Diagrams [27,28], etc.	UML Use Case Diagrams + brief individual descriptions
Quality of Service	Addressed in Test Cases out of Domain Analysis Stage	ISO/IEC 9126-1
Non-Technical Description	None	Three categories of non-technical factors
Interoperability	Establishment of *"Artifact Dependencies"* considered a special kind of variability, commonly used in Software Product Lines design, represented into variability models [25]. However they not fully deal with particular COTS interoperability relationships	*i** SD Models

6. A Unifying Model for COTS Domain Analysis

The models proposed in section 5 cover the informational dimensions that were identified in section 4. However, it is clear that having these dimensions structured in separate models hampers domain understanding and model management. Therefore, in this section we aim at formulating a strategy to integrate the domain models obtained so far, even considering their different nature, into a single analysis model.

Since the primary goal of COTS segments domain analysis is to characterize COTS components for their evaluation and selection, we need a unifying model which facilitates this goal. From the dimension models given, quality models seem the most appropriate type of artefact. Therefore, if we succeed in putting all the models in an ISO/IEC 9126-1 quality model we will have our goal attained.

6.1 The ISO/IEC 9126 Quality Standard

The ISO/IEC 9126-1 software quality standard is one of the most, if not the most, widespread quality standard available in the software engineering community. It proposes quality models as the artifacts that keep track of the quality factors that are of interest in a particular context.

The ISO/IEC 9126-1 standard (see Fig. 3) fixes 6 top level characteristics: functionality, reliability, usability, efficiency, maintainability and portability. It also fixes their further refinement into 27 subcharacteristics but does not elaborate the quality model below this level, making thus the model flexible. To carry out this refinement, subcharacteristics are in turn decomposed into attributes, which represent the properties that the software products belonging to the domain of interest exhibit. Intermediate hierarchies of subcharacteristics and attributes may appear making thus the model highly structured. Metrics are bound to attributes.

The standard is highly customizable to different purposes and domains; for instance, in our previous work we have created an extension for the particular case of quality of COTS components, in which new subcharacteristics and attributes have been introduced [19].

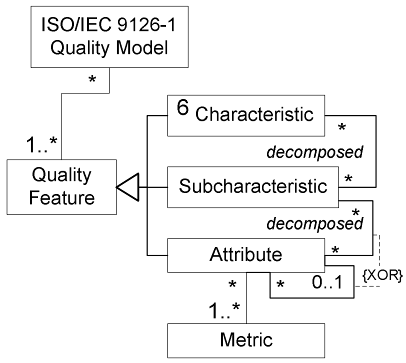


Fig. 3. Conceptual model of the ISO/IEC 9126-1 quality standard

6.2 Integrating all the COTS domain models into the ISO/IEC 9126-1

Functionality.
Regardless of having the same name, the functionality of a COTS segment does not correspond with the ISO/IEC *Functionality* characteristic. Instead, it corresponds to the *Suitability* concept that is a subcharacteristic of *Functionality*. However, since functionality focuses on the services provided but not the data managed, we create a new subcharacteristic *Suitability of Services* that contains the UML Use Case diagram and the individual use case descriptions.

Fundamental concepts.
The UML class diagram is related to two ISO/IEC subcharacteristics. On the one hand, as the case before, *Suitability*, because some of the classes (and their attributes) and relationships are defining part of the suitability of the COTS segment. On the other hand, *Understandability*, which is a subcharacteristic of *Usability*, because having a UML class diagram provides a reference framework that allows testing how much a particular COTS component adheres to it. For the same reason, also the LEL glossary supports *Understandability*. Therefore, we create 3 new subcharacteristics. The first one, *Suitability of Data*, belongs to *Suitability* and contains the UML class diagram. The other two, *Semantic Understandability* and *Lexical Understandability*, belong to *Understandability*. The first one also contains the UML class diagram and the second one the LEL glossary.

Non-technical description.
It is easy to organize non-technical factors in an ISO/IEC-9126-1-form, assuming that the 3 high-level factors are characteristics and the other 15 subcharacteristics.

Interoperability.
Interoperability is also a subcharacteristic of *Suitability* and in this case, we just consider the *i** SD model as the description of *Interoperability*.

Fig. 4 shows an overview of our proposed framework.

	Characteristic	Subcharacteristics
Quality Of Service	Functionality	Suitability
		Suitability of Services
		Suitability of Data
		Accuracy
		Interoperability
		Security
		F. Compliance
	Reliability	…
	Usability	Understandability
		Semantic Understandability
		Lexical Understandability
	Efficiency	…
	Maintainability	…
	Portability	
	Extended Charact.	
Non-Technical Factors	Supplier	Organizational Structure
		…
	Cost	Licensing Schema
		…
	Product	Stability
		…

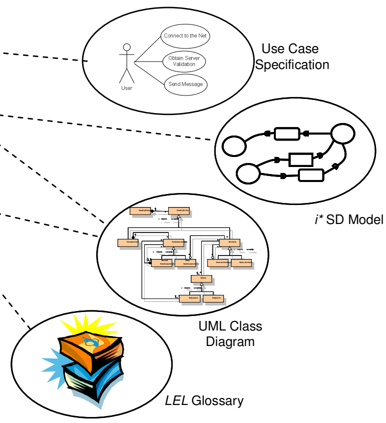


Fig. 4. An overview of the ISO/IEC 9126-1-based analysis model for COTS segments.

6.3 Transforming the models into the ISO/IEC 9126-1 framework

Although we have achieved our primary goal, namely integrating all the dimension models under the same umbrella, there is still a question left that may be considered as a drawback when using the domain model for COTS components evaluation purposes: the fundamental concepts, functionality and interoperability models are expressed with their own formalisms which are not straightforward to evaluate. In this subsection we deal with this problem by providing rules that map the constructs in these models into ISO/IEC 9126-1 quality factors. Furthermore, we state how their metrics are defined. These rules are defined in such a way that they could generate the new, final model automatically from the former models.

Functionality.

For each use case UC appearing in the Use Case diagram, a quality attribute UC belonging to the *Suitability of Services* subcharacteristic is created. The individual use case specifications are part of the description of these quality attributes.

For each obtained quality attribute, an ordinal metric which can take three values, Satisfactory, Acceptable and Poor, is created. These values express how a particular COTS component covers the service represented by the use case.

Fundamental concepts.

For each class or association C appearing in the class diagram that represents a concept provided by the COTS components in the segment, a quality attribute C belonging to the *Suitability of Data* subcharacteristic is created. The elements of the class diagram are part of the description of these quality attributes.

For each obtained quality attribute, an ordinal metric which can take three values, Satisfactory, Acceptable and Poor, is created. These values express how a particular

COTS component provides the data represented by the class or association. These values will be obtained during evaluation by using different criteria (e.g., whether all the attributes are provided, whether the instances are permanent or not, etc.).

Each term of the glossary is included as part of the description of the quality attribute(s) it is related to. The same happens with the elements of the class diagram that were not tackled in the previous step. Last, two numerical metrics are bound to the *Semantic Understandability* and *Lexical Understandability* attributes. The values of these metrics will count the number of semantic and lexical discrepancies of a particular COTS component with respect to the reference models.

Interoperability.

For each agent A appearing the *i** SD model, except the agent S that represents the COTS segment we are modeling, a subcharacteristic A belonging to *Interoperability* is created. For each dependency G among S and A, an attribute G is created.

For each obtained quality attribute, we create an ordinal metric whose values depend on the type of the corresponding dependency: if goal, values are Attained and Not Attained; if resource, Provided and Not Provided; if task, Executed or Failed; if softgoal, Satisfactory, Acceptable and Poor.

Once these rules are applied, evaluation of COTS component may be done in a more uniform and comfortable way. But of course, the original models should be preserved since they are easier to understand and evolve.

7. An Example: the Real-Time Synchronous Communication Domain

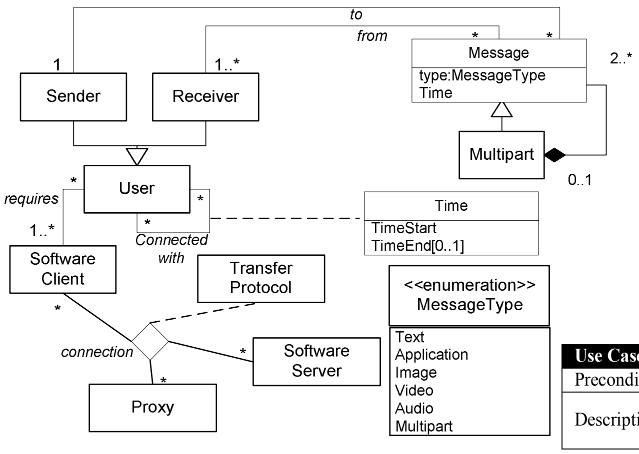
For illustrating our proposal, we present some excerpts of the domain model obtained for the Real-Time Synchronous Communication (RTSC) market segment. This segment embraces the various tools and technologies used to enable communication and collaboration among people in a "same time-different place" mode.

Fundamental concepts.

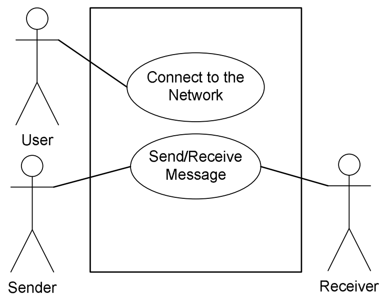
Part of the UML class diagram is presented in Fig. 5a. Several key concepts are stated as classes. These concepts are of different nature, e.g. human roles (e.g., *Server* and *Receiver*), artefacts of any kind (either physical or informational, e.g. *Message*), software and hardware domain-specific components (e.g., *Software Client, Software Server* and *Proxy*), etc. Inside these classes, we identify attributes but just those that play a crucial part in the domain, e.g. *Message* that can be of different types. Domain relationships are also of different kinds. Thus, we can see a high-level relationship among the human roles *Sender* and *Receiver* which are generalized into a *User* class. On the other hand, associations may be of very different nature. For instance, we have permanent or at least very stable relationships (e.g., among *User* and *Software Client*) while others are highly dynamic (real-time connections that are created and destroyed dynamically). OCL restrictions may be used to decorate the model appropriately.

Functionality.

As stated in section 4, the use case model for functionality focuses on the most characteristic services offered by packages in this domain. Fig. 5b shows some for the RTSC domain, namely *Connect to the Network* and *Send/Receive Message*. Others such as *Send Video Message* or *Connecting Multiuser Session* are not included either because they are not considered general enough but specific of a few COTS components, or because they are considered as secondary. In addition, we can also check that the individual use case specification of *Send/Receive Message* presented in Fig 5c follows the given recommendation of being very abridged.



a. Excerpt of the UML Class Diagram



b. Excerpt of the UML Use Case Diagram

Use Case	Send/Receive Message
Precondition	The Sender and the Receiver are both Connected
Description	The Sender composes a message of any kind and delivers it to the Receiver. The Receiver is notified and then reads the message.

c. Excerpt of Individual Use Case Specification for Send/Receive Message

Fig. 5 Excerpt of some domain models constructed for the RTSC case

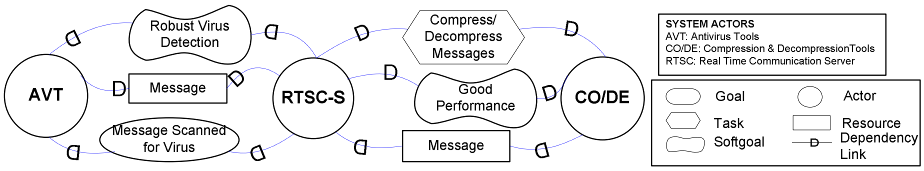


Fig. 6. Some dependencies among RTSC Tools and other types of tools.

Interoperability.

As it is the usual case in COTS segments that offer a lot of functionality, we may identify several relationships with other types of COTS domains. In Fig. 6 we introduce as example two COTS segments related with RTSC, AntiVirus Tools (AVT) and Compression/Decompression Tools (CO/DE), all of them modelled as *i** actors. Among their relationships, we find: a RTSC component relies on a AVT component for detecting viruses (goal dependency, since the AVT decides the best way to do that); a RTSC component depends on a CO/DE one to compress/decompress messages automatically (task dependency, because the RTSC states when and how these automatic activities are done); a RTSC component may improve its performance using a CO/DE component (softgoal dependency, because the concept of "good" performance is matter of negotiation); and both related components need the message to work with from a RTSC component (resource dependency, because it is an informational entity).

Quality of service.

We take into account the particularities of the RTSC segment for defining specific attributes and their metrics. In table 2 we decompose a bit the *Understandability* subcharacteristic with the *Adherence to Best Practices* and *Supported Interface Languages* attributes. We include specific metrics that help to evaluate and compare user requirements. The first metric illustrate the subjective case, whilst the second one illustrates a metric that is both objective and structured (set of values). The description included in the table is part of the glossary but included here for legibility purposes.

Table 2. Excerpt of the quality model for the RTSC case

Quality factor					Metric	Description
3	**Usability**					ISO/IEC 9126-1 Characteristic
	1	Understandability				ISO/IEC 9126-1 Subcharacteristic
		3	Interface Understandability			Effort to recognizing the logical concepts and its applicability by means of interfaces.
			1	Adherence to Best Practices	ADP: 4valueOrder[Ordinal] 4valueOrder = (Optimal, Good, Fair, Poor)	How well events and elements of the interface comply with best practices recognized for user interfaces.
			2	Supported Interface Languages	SIL: Languages = Set(Labels[Nominal]) Labels = (Spanish, Catalan, English, …)	Languages supported by the interface.

Non-technical description.

Table 3 shows an excerpt of the refinement of a non-technical factor of a product, its stability. Note the similarity compared to quality of service description, which facilitates further integration. It should be mentioned that non-technical factors are very similar among different COTS segments.

Table 3. Excerpt of a non-technical factor decomposition for the RTSC case

Non-technical factor				Metric	Description
3	**Product**				Non-technical characteristics of a COTS product that may influence COTS selection
	1	Stability			
		1	Time of Product in the Market	TPM: Time[Ratio] Time = Float	Number of years the product has been in the marketplace
		2	Versions Currently in the Market	VCM: List(Version[Nominal]) Version = String	Versions currently available in the marketplace
		3	In-house made Product	IP: Own[Nominal] Own = (Yes, Not)	Whether the product is in-house or acquired from a third party

Table 4 shows the integration of the presented excerpts in the unifying model using the mapping rules introduced in the section 6.3.

Table 4. The unifying model for the RTSC case (excerpt)

				Quality factor	Metric	Description
1				**Functionality**		See ISO/IEC 9126 Description
	1			Suitability		See ISO/IEC 9126 Description
		1		Suitability of Services		See 6.3
			1	Connect to Network	CN: 3ValueOrder[Ordinal] 3ValueOrder = (Satisfactory, Acceptable, Poor)	See fig. 5b
			2	Send/Receive Message	SRMsg: 3ValueOrder[Ordinal]	See fig. 5b
				…		
		2		Suitability of Data		See 6.3
			1	Message	Msg: 3ValueOrder[Ordinal]	See fig. 5a
			2	Connected with	Cw: 3ValueOrder[Ordinal]	See fig. 5a
				…		
	2			Interoperability		See ISO/IEC 9126 Description
		1		Anti-Virus Tools		See fig. 6
			1	Robust Virus Detection	RVD: 3ValueOrder[Ordinal]	See fig. 6
			2	Message Scanned for Virus	MSV: GoalValue[Ordinal] GoalValue = (Attained, Not Attained)	See fig. 6
			3	Message	Msg: ResourceValue[Ordinal] ResourceValue = (Provided, NotProvided)	See fig. 6
		2		CO/DE Tools		See fig. 6
			1	Good Performance	GP: 3ValueOrder[Ordinal]	See fig. 6
			2	Compress/Decompress Messages	CDMsg:TaskValue[Ordinal] TaskValue = (Executed, Failed)	See fig. 6
			3	Message	Msg: ResourceValue[Ordinal]	See fig. 6
	3	…				
2				**Reliability**		See ISO/IEC 9126 Description
3				**Usability**		See ISO/IEC 9126 Description
	1			Understandability		See ISO/IEC 9126 Description
		1		Semantic Understandability	SU: Number[Unit] Number=Integer	See 6.3
		2		Lexical Understandability	LU:Number[Unit]	See 6.3
		3		Interface Understandability		See table 2
			1	Adherence to Best Practices	ADP: 4valueOrder[Ordinal] 4valueOrder = (Optimal, Good, Fair, Poor)	See table 2
			2	Supported Interface Languages	SIL: Languages = Set(Labels[Nominal]) Labels = (Spanish, Catalan, English, …)	See table 2
	2	…				
4				**…other ISO/IEC characteristics**		See ISO/IEC 9126 Description
				Non-technical factor	**Metric**	**Description**
1				**Supplier**		See [19]
2				**Cost**		See [19]
3				**Product**		See table 3
	1			Stability		
		1		Time of Product in Market	TPM: Time[Ratio]; Time = Float	See table 3
		2		Versions Currently in Market	VCM: List(Version[Nominal]); Version = String	See table 3
		3		In-house made Product	IP: Own[Nominal] Own = (Yes, Not)	See table 3
	2	…				

8. Domain Analysis Based COTS Selection

Our domain analysis strategy has been integrated into our GOThIC method, as it was stated before. The way to do that is to consider that the Quality Model class introduced in Fig. 7 is in fact the Domain Model class that appears in Fig. 1.

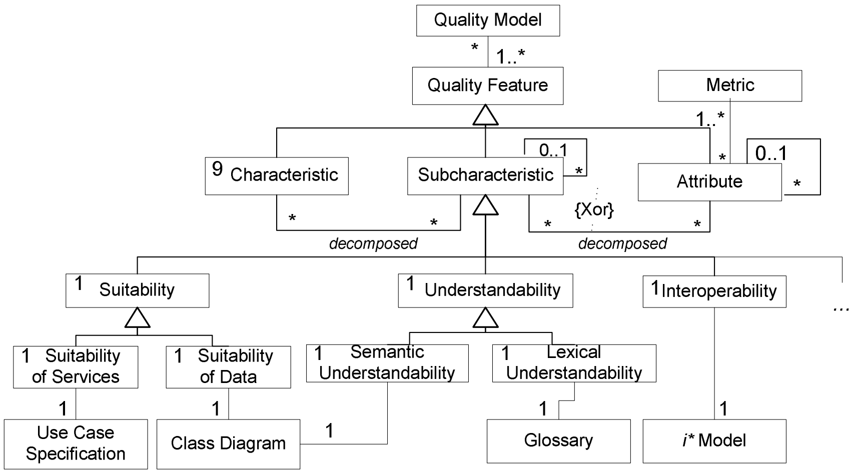


Fig. 7. A conceptual model excerpt of our ISO/IEC 9126-1-based analysis model for COTS segments

As stated in section 3, a GOThIC taxonomy is used to locate the taxonomy node that fulfils the needs of the user in charge of the selection process. Once located, its domain model may be used to guide the rest of the selection process. On the one hand, the factors in the ISO/IEC 9126-1-based model help to elicit and negotiate the requirements, making easier the identification of mismatches among components characteristics and the departing requirements. On the other hand, those factors corresponding to the stated requirements are used to evaluate the capabilities of the candidate components in a uniform way, using the metrics defined in the model. For doing so, we can proceed manually, or use automated support that range from a simple spreadsheet to a more sophisticated tool, e.g. the DesCOTS system [35] that we have developed with this special goal in mind.

9. Conclusions

We have presented an approach for building a reuse infrastructure for supporting COTS selection processes. It is based on the application of domain analysis for recording and representing all the required information.

Our proposal relies on several industrial experiences that have been undertaken under action-research premises, complemented with literature survey and grounded theory. These industrial experiences have been carried out in the field of

Requirements Engineering Tools [15], Workflow Systems [14], Telephony Systems [16] and some sub-categories of Enterprise Applications (with emphasis with those related to Content Management). Industrial experiences have been complemented with academic ones (e.g. Real-Time Synchronous Communication and Message-based Communication Systems) to analyse in more depth some particular aspects of the method.

We believe that our proposal has a positive impact to both the COTS selection context and domain analysis framework. For COTS selection:

- We have put the emphasis on reuse, making a concrete proposal based on the domain analysis technique which allows transferring knowledge from one experience to another.
- Domain analysis not only impacts positively on reuse, but also ameliorates some well-known obstacles for COTS selections success (mentioned in section 2). Remarkably, using domain analysis principles we avoid those semantic and syntactic discrepancies that are common in the COTS marketplace and this helps to overcome the risks in using this technology.
- We have explicitly identified the informational dimensions required for the effective and efficient selection of COTS components.
- We have offered guidance for representing these informational dimensions using appropriate types of domain models.
- Using some mapping rules, we have integrated all these models into a single one, based on a well-known standard, highly oriented to support the evaluation of the candidate components.
- Given this representation, we may use some existing tool-support to conduct the evaluation of candidates in the framework of the ISO/IEC 9126-1 standard.

Concerning domain analysis, existing approaches were not oriented to support reuse in the COTS framework, consequently the need of mechanisms to analyze and create a reuse infrastructure for COTS domains was remaining. In particular, it is required to represent interoperability among COTS components and to analyze non-technical factors that may influence the selection, as well as the need of putting more emphasis to software quality issues.

In order to make our approach feasible, some premises should follow:

- To be applied to a COTS segment that is of general interest. This means that a great deal of organizations needs to select COTS components from this segment. Some examples are: communication infrastructure, ERP systems, security-related systems, etc. In these contexts, the number of selection processes that take place will be high and then reusability of the models likely to occur.
- The addressed COTS segment offers COTS components of coarse-grained granularity. This makes domain understanding more difficult, time-consuming and cumbersome and therefore domain analysis is helpful. Market segments such as CRM and ECM systems are typical examples, whilst time or currency converters are not.
- An organization is carrying out subsequent COTS component selection processes. This organization will find valuable to have means to transfer knowledge from one experience to another.

Acknowledgements

This work has been partially supported by the CICYT program; project TIN2004-07461-C02-01.

References

1. Prieto-Díaz, R., Arango, G. *Domain Analysis and Software Systems Modelling*. IEEE Computer Society Press, 1991.
2. Frakes, W., Prieto-Díaz, R., Fox, C. "DARE: Domain Analysis and Reuse Environment", Annals of Software Engineering, 5, pp. 125-141, 1998.
3. Software Eng. Institute. http://www.sei.cmu.edu/domain-engineering/, 2002
4. Cornwell, P.C., "HP Domain Analysis: Producing Useful Models for Reusable Software". Hewlett-Packard Journal, August 1996.
5. Neighbors, J. "Software Construction Using Components" PhD. Thesis, University of California, Irvine, 1980.
6. Meyers, C., Oberndorf, P. *Managing Software Acquisition*. Addison-Wesley, 2001.
7. Ruhe, G. "Intelligent Support for Selection of COTS Products". LNCS 2593, 2003.
8. Ayala, C. Franch, X. "A Goal-Oriented Strategy for Supporting Commercial Off-The-Shelf Components Selection". 9th International Conference on Software Reuse (ICSR), 2006.
9. Franch, X., Maiden, N. "Modelling Component Dependencies to Inform their Selection". Procs. 2nd Int. Conference on COTS-Based Software Systems (ICCBSS), LNCS 2580. 2003.
10. Bertoa, M.F., Troya, J.M., Vallecillo, A. "A Survey on the Quality Information Provided by Software Component Vendors". Proceedings QAOOSE Workshop, 2003.
11. Réquilé-Romanczuk, A. *et al*. "Towards a Knowledge-Based Framework for COTS Component Identification". 2nd MPEC Workshop at ICSE'05, ACM 2005.
12. UML Specifications, http://www.uml.org/
13. Franch, X., Carvallo, J.P. "Using Quality Models in Software Package Selection". IEEE Software, 20(1), 2003.
14. Carvallo, J.P., *et al*. "A Framework for Selecting Workflow Tools in the Context of Composite Information Systems". Procs. 15th DEXA Conf., LNCS 3180, 2004.
15. Carvallo, J.P., Franch, X., Quer, C. "A Quality Model for Requirements Management Tools". Book chapter in *Requirements Eng. For Sociotechnical Systems*, Idea Group, 2005.
16. Carvallo, J.P. "Supporting Organizational Induction and Goals Alignment for COTS Components Selection by means of *i**". Procs. 5th ICCBSS, IEEE Comp. Soc., 2006.
17. Ayala, C., Franch, X. "Transforming Software Package Classification Hierarchies into Goal-Based Taxonomies". Procs. 16th DEXA Conf., LNCS 3588, 2005.
18. Maiden, N., Ncube, C. "Acquiring Requirements for COTS Selection". IEEE Software (15)2, 1998.
19. Carvallo, J.P., Franch, X. "Extending the ISO/IEC 9126-1 Quality Model with Non-Technical Factors for COTS Components Selection". 4th WoSQ at ICSE'06, ACM 2006.
20. Ferré, X., Vegas, S. "An Evaluation of Domain Analysis Methods". 4th EMMSAD Workshop at CAiSE, 1999.
21. McMenamin, S.M., Palmer, J.F. *Essential Systems Analysis*. Yourdon Press, 1984.
22. Chen, P. "The Entity-Relationship Model –Towards a Unified View of Data". ACM Transactions on Database Systems, Vol. 1, No. 1, March 1976.
23. Czarnecki, K., Eisenecker, U.W. *Generative Programming: Methods, Tools and Applications*. Addison-Wesley, 2000.

24.Pohl, K., *et al*. "Scenario-Based Change Integration in Product Family Development". ICSE-Workshop on Product Lines, 2001.
25.SEI http://www.sei.cmu.edu/domain-engineering/FODA.html
26.Pohl, K. *et al*. *Software Product Line Engineering*. Springer-Verlag, 2005
27.Clauss, M. "Generic Modelling Using UML Extensions for Variability". OOPSLA, 2001.
28.Schleicher, A. *et al*. "Beyond Stereotyping:Metamodeling Approaches for the UML". 4th Annual Hawaii International Conf, on System Sciences. 2001.
29.Vitharana, P., *et al*. "Design, Retrieval, and Assembly in Component-Based Software Development". Communications of ACM, November 2003/Vol.46, No. 11.
30.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.
31.Leite, J.C.S.P. "Application Languages: A Product of Requirements Analysis". Informatics Department PUC-/RJ (1989).
32.Cockbum, A. *Writing Effective Use Cases*. Addison-Wesley, 2001.
33.ISO/IEC International Standard 9126-1. *Software Engineering-Product Quality-Part 1: Quality Model*. 2001.
34.Yu, E. *Modelling Strategic Relationships for Process Reengineering*. PhD Thesis, University of Toronto, 1995.
35.Grau, G. *et al*. "DesCOTS: A Software System for Selecting COTS Components". 30th EUROMICRO Conference, IEEE Computer Society, 2004.