

Assessment of Reusable COTS Attributes

Marco Torchiano and Letizia Jaccheri

Department of Computer and Information Science, Norwegian
University of Science and Technology, Trondheim Norway
{marco, letizia}@idi.ntnu.no

Abstract. Among the main activities involved in COTS-based development there are identification, evaluation, and selection of COTS products. Several techniques have been developed for these activities; all of them are based on measurement of attributes. 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. Several studies show that it is possible and convenient to develop a reusable attribute framework. We propose a set of simple and generic criteria can be used to validate the set of attributes and improve them.

1 Introduction

The software development process using COTS (Commercial-Off-The-Shelf) products has been studied in a few works such as [10] and [5]. Even if no general agreement has been reached, a group of key practices that has been identified in these studies are related to the identification, characterization, evaluation, and selection of the products.

The common feature in these approaches is the use of a set of attributes to identify, characterize, evaluate and select COTS products. In some studies the attributes are designed to be reusable. The advantage of reusing attributes is to capitalize the effort spent to measure them by reusing the collected information across projects and organization units. The main critic to the reuse of attributes is that each project has different requirements that demand for completely different attributes and criteria. An intermediate solution is to have a core of reusable attributes, which are intended to be extended by project dependent specific attributes.

We assume that reusing evaluation attributes is possible. We address such research questions as: how can we understand how good are the attributes? What are the common problems? How can they be improved, both in terms of selection of attributes and their measurements?

The goal of this paper is to propose a set of assessment criteria for reusable attributes. The problems discovered will be tracked to their causes, enabling a process of continuous improvement.

This work was conducted in the context of a research and educational project (see [7] and [14] for more information). Here we present a set of classification and characterization attributes together with their measurement scale.

There were 36 students participating the project, they evaluated a total of 36 products; each student evaluated four different products and each product received four evaluations. The attributes measured by the students have been collected in a database, which served as a means to record and share the knowledge about the products.

This paper is organized as follows. Section 2 describes related works and provides the motivations for this paper. Section 3 describes the attributes we used the software evaluation course. Section 4 describes the attribute assessment criteria, the results of their application to our attribute framework, and discusses how they can be generalized. Finally section 5 draws some conclusions and describes future work.

2 Background and Motivation

In the literature there is a wide range of works about COTS attributes in terms of how the attributes should be defined and how they can be reused.

We try to resume the most representative positions within this spectrum.

At one end there are the Iusware [12], RCPEP [9] and OTSO [8] 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 depend on the requirements therefore they vary at each project.

At the other end we find CAP[13], the approach proposed by Boloix et al.[4], and the proposal of the eCOTS[3] and CLARiFi[2] 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.

We have so far considered attributes that are aimed at selection of products. Several recent works characterize COTS products by means of attributes with a different purpose. Carney and Long [6] and Morisio and Torchiano [11] used attributes to identify relationship between characteristics of products and their impact on COTS based development. The use of a fixed set of attributes is a key requirement in this case.

3 The attribute framework

The attributes framework was defined and used in a software evaluation course[1] of the fifth year in the computer science master degree at Norwegian University of Science and Technology (NTNU), Norway.

We wanted the student to reflect on new software technologies in general. In particular we decided to use the term “software item” to be as inclusive as possible. Thus the attributes were used to investigate a field much larger than only COTS products.

Since the student started from a large set of items they were invited to reflect on questions such as “What is the difference between ‘software technology’ and ‘COTS product’?”

We defined an attribute framework that is made up of classification attributes and characterization attributes. The classification attributes can be used to identify products and organize them into groups of homogenous items; the characterization attributes provide a more detailed description of the COTS products and form the basis for their evaluation and comparison.

The attribute framework makes use of both qualitative and quantitative attributes. Several attributes have both kinds of values. The rationale for this duality is that in such an indefinite areas as COTS products, qualitative evaluations bring a useful richness of information that is easily lost when coded into quantitative attributes.

Classification attributes are all quantitative, while characterization attributes can be both. But all the qualitative attributes have been coded into quantitative values.

As shown in **Fig. 1** it is possible to look at a software item both from a qualitative and a quantitative point of view. Estimation provides an imprecise but reach in content qualitative value; it is possible to measure some feature of an item into a quantitative attribute. A qualitative value can be coded into a quantitative value.

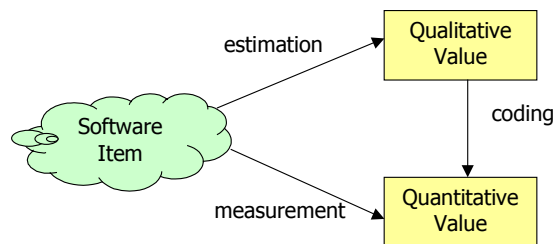


Fig. 1. Quantitative and qualitative attributes.

3.1 Classification Attributes

We identified three attributes that are useful in order to classify products: architectural level, artifact kind, and life-cycle phase. Each combination of values of these attributes defines a class.

The possible values of the **architectural level** attribute consist of a pair composed of an architectural pattern and a role defined in that pattern. For the course, we choose to adopt 3-tier architectural style. Thus each software item can play the role of either the client, server, or data.

It is important to find out the **kind** of software. We identified three possible kinds of items: executable, standard, service. Executable items include both source and binary pieces of software. A standard is a prescriptive document publicly available

and approved in some forum. A service item consists of a set of functionality provided by a third party, usually through the network; for instance web-based development services (such as project hosting, version control, bug and issue tracking, project management, backups and archives, and communication and collaboration resources).

Another distinction can be found when looking at when, during the life cycle of a system, a given item can be used. We identified two possible values of the **life cycle phase** attribute: development and execution. Development items are tools used to build a software system. Execution items are components used as parts of the system.

3.2 Characterization Attributes

Here we present 16 attributes to characterize COTS products. Only a few attributes have a quantitative scale, most of the attributes were estimated qualitatively, and later on coded into quantitative values. The coding for the qualitative attributes took place when the values were entered into a product database.

Table 1 shows the characterization attributes with their description and how they were originally evaluated. The qualitative column tells whether the attribute has been estimated qualitatively. The quantitative column explains how the quantitative value has been obtained, either by coding the qualitative value or measuring it on the product.

Table 1. Characterization attributes.

Attribute	Description	Qualitative	Quantitative
Product maturity	described by means of a story related to the maturity of the product, e.g. in terms of years on the market and features stability.	yes	coded
Market Share	the ratio/part of the market of similar products that is covered by this COTS product	no	measured (ratio)
Scalability	measured as the number of users it can scale to without sensible decrease in response time	no	measured (ordinal)
Safety/Security	the support offered by the product for developing secure or safety critical systems	yes	coded
Reliability	a story describing how fault-tolerant the product is	yes	coded
Hardware requirements	a list of the hardware characteristics required to use the product	yes	coded
Product support	a summary of the facilities offered to support software development using the product	yes	coded

Documentation	what kind (web, on-line, etc.) of documentation is provided together with the product and what is its size (e.g. num of manual pages)	yes	coded
Usability	the degree of satisfaction on a scale from 1 to 5 and a story describing it in more detail	yes	measured (ordinal)
Modifiability	how easy it is to modify this product	yes	coded
Change frequency	how many releases/versions in the last year	no	measured (ordinal)
License Type	the type of license of the product	no	quantitative (nominal)
Cost	the cost for acquiring the product and deploying it	yes	coded
Software requirements	list of required software to run the component	yes	coded
Standard Conformance	list of standard to which the products conforms	yes	coded

Since the purpose of evaluation attributes is to provide information to select the COTS products and the more trusty the information the better the selection, we collect four independent evaluations for the same product.

4 Assessment

The assessment of the attribute framework aims at finding out whether the attributes are good enough, and how they can be improved for future iterations.

First we describe the assessment criteria and a checklist, then we present the results we obtained on our attribute framework and try to generalize them.

4.1 Criteria

We analyzed the products evaluations gathered using the attribute framework, described in section 3 above, to identify the main problems. We focused on generic issues without taking into consideration the specific semantic of each attribute.

The problems we identified are:

- a characterization attribute could not be evaluated because of lack of information,
- it was not possible to assign a quantitative value to an attribute,
- inconsistent values of an attribute were provided by different people for the same product,
- there are no products in some classes.

Based on these problems we defined a checklist to identify them systematically and possibly track them to their causes. Each item of the list consists of the problem to be revealed and of the method to discover it.

C1: Characterization attribute not evaluated: sometimes it is not possible to find enough information to assign a value to an evaluation attribute for a given product. This problem can be spotted looking at the percentage of not evaluated attribute entries for each product and to the percentage of not evaluated entries for each attribute.

C2: Characterization attribute not quantified: it is possible that even though a qualitative evaluation for an attribute is given, it cannot be coded into a quantitative value. A measure of the gravity of this problem can be found looking, for each attribute, at the percentage of entries that have are not codified.

C3: Classification attribute not quantified: a similar problem can occur when an evaluator is not able to decide which value to assign to a classification attribute. A measure of this problem can be found looking at the percentage of the combinations of classification attributes that include at least one “not classified” value over the whole number of entries.

C4: Inconsistent characterization: since several independent evaluations are provided for the same product it is possible to have different people to provide different values for the same characterization attribute. This effect can be revealed looking at the dispersion of the values in the value range. One of the possibilities is to look at the standard deviation of the evaluations.

C5: Inconsistent classification: similarly it is possible that different people classify COTS products in different ways. Since the classification is usually performed by means of nominal attributes, we can measure this issue in a simpler way than the previous one. A simple measure is the number of different classification assigned to a given COTS product.

C6: Empty class: classification of COTS products assigns each product to a specific class. When no product is assigned to a predefined class we have an empty class. Empty classes can be easily found by looking at the number of products assigned to each class.

4.2 Results

We have identified five main possible causes for the problems:

1. the scenario the attributes were designed for does not fit the one that the product is intended for,
2. evaluation or measurement of attributes are affected by the experience or skill of the evaluator,
3. the source of information is scarce,
4. the attribute is not well defined for the following reasons
 - a. the semantic is not well defined
 - b. the coding schema is not well defined
5. the wrong products have been selected.

We applied the checklist defined in the previous subsection and tried to map the problems to previous list of possible causes.

4.2.1 C1: Characterization attribute not evaluated

This problem is mostly related to a specific product. The main cause can be bad or missing documentation or lack of information on the product web site (3). An alternative cause may be an evaluator not skilled enough to find the information (2).

It is also possible that the kind of information required by an attribute is not easily accessible, not only for a specific product but for a full range of products. In this case either the attribute is not well defined (4.a) or the products being evaluated are off the scope of the attribute framework (1). If a single product has a high percentage of attributes not evaluated probably the product should not be evaluated (5).

In some cases the problem can be solved improving the definition of the attribute.

C1 revealed problems both in getting information about specific products and generic problems regarding attributes. For instance in 39% of the cases the market share of the products could not be determined. This can be explained because product web sites rarely report this kind of information, which on the other end can be found from market research firms.

As a general rule, it is important to use several information sources whenever they are available. Otherwise we should be aware that certain attributes cannot be measured and avoid basing our judgment on them.

4.2.2 C2: Characterization attribute not quantified.

This is a major problem can be caused by a misinterpretation of the definition of the attribute, due to either a bad definition of the attributed (4.a) or to an unskilled evaluator (2). It is also possible that the coding scheme for the attribute is not well designed (4.b).

The solution for this problem lies in a better coding schema and a more precise description of the attribute.

C2 revealed that certain attributes could be applied to products of a limited set of classes. For instance 19% of the values of attribute maintainability were not quantified. This can be explained by the dependency that it has on other characteristics of the product, for instance maintainability cannot easily be measured for a product from a third party.

As a general rule it is important to understand and anticipate the dependencies between attributes to avoid this kind of problems.

4.2.3 C3: Classification attribute non quantified

This problem can be due to a limited set of possible values for an attribute (4.b) or an attribute not well defined (4.a). Another possible cause is the presence of a product category that was not considered when defining the attribute (1) or the wrong products are being evaluated (5).

The solution is either to modify the attribute and its coding schema or to filter the set of candidate products.

C3 emphasized the dependency between different classification attributes. For instance 8% of the classifications were Not Classified, Executable, Development, i.e. tools used to develop code for an undetermined architectural level. The explanation is that development tools do not address a specific level, the exception being represented by user interface editors, which are clearly client-level tools. In the case study at first

we focused on web-based three-tier systems and then included a wider range of products.

As a general rule it is important to carefully choose the classification attributes, stating the type of products they can be applied to.

4.2.4 C4: Inconsistent characterization

One possible cause is an imprecise definition of the attribute (4.a) or a bad designed codification schema (4.b) leading to different interpretations.

It is also possible that the information provided by the COTS vendor be scarce or misleading (3).

The solution is to minimize the ambiguity in the definition of the characterization attribute and its coding schema.

C4 revealed several problems with the definition of attributes. On the average each product has two different evaluations per attribute. Several cases of inconsistent evaluations were due to imprecise definitions; for instance attribute reliability, which has 2.8 different evaluations per product suffered of both an imprecise definition and inapplicability to certain categories of products.

As a general rule it is important to provide a precise and operational definition for the attributes.

4.2.5 C5: Inconsistent classification

The most likely cause for this problem is the imprecise definition of the attribute (4.a) or a bad designed codification schema (4.b). It is also possible that the product being evaluated is wrong (5). The scarcity of information (3) can also cause this problem.

The solution is to minimize the ambiguity in the definition of the classification attribute and its coding schema.

C5 discovered a problem with products that also embody a standard. This ambiguity was partly due to the choice of very generic classification attributes and served the purpose of make the student reason about the multiple meanings of products names.

In general it is important to identify precisely the products and avoid buzzwords.

4.2.6 C6: Empty class

A give combination of values of classification attributes may be impossible, for instance because the attributes are dependent on each other due to a bad design (4.a).

More likely the presence of empty, or even scarcely populated class, may indicate a bias in the selection of the initial set of COTS products (5).

The solutions to this problem are a better selection of the candidate products and improved attributes.

C6 revealed that several classes are empty. Even if this may be due to the limited number of products, it can also highlight the scarce number of products belonging to a given class on the marketplace.

The relationships between the problem and the possible causes are summarized in **Table 2**.

Table 2. Problems and causes.

Problem	Attribute	Causes
attribute not evaluated	characterization C1	1) scenario 2) evaluator 3) information 4.a) attribute semantics
attribute not quantified	characterization C2	1) scenario 2) evaluator 4.a) attribute semantics 4.b) attribute coding
	classification C3	1) scenario 4.a) attribute semantics 4.b) attribute coding 5) wrong products
inconsistent values	characterization C4	3) information 4.a) attribute semantics 4.b) attribute coding
	classification C5	2) evaluator 3) information 4.a) attribute semantics 4.b) attribute coding 5) wrong products
empty class	classification C6	4.a) attribute semantics 5) wrong products

5 Conclusions

Our opinion is that in a COTS product evaluation initiative in a large enterprise should be based on reusable attributes in order to be cost effective and gain commitment from all levels in the organization.

If we accept the possibility of reusing COTS-related attributes, it becomes then important to assess the “quality” of the attributes, their measurements criteria, and the collected information. This is the first step in a continuous improvement approach.

Based on our experience we proposed a checklist to perform the assessment of an attribute framework. Each problem can be mapped to a set of possible causes, thus enabling both the evaluation of the quality and the improvement of the attribute framework.

The application of the proposed approach to the case study provided useful insight into the attributes and the collected data.

We found several problems caused by the coding schemas of the quantitative attributes. The main motivation is the educational context in which the attribute framework was defined, one of its main purposes was pedagogical: make the student reflect about COTS products.

In the next version of the attribute framework we emphasize the pedagogical role of the qualitative evaluations and its imprecise feature. Instead we will improve the

quantitative attributes to make them more precise and focused, this will enable collecting more precise data.

As COTS product evaluation becomes a common practice, its assessment gains importance. A lot of work can be done in this area; we foresee two main focus areas:

- development of new and more sophisticated assessment criteria,
- empirical validation of both existing and new criteria.

References

1. SIFT80AT - A course in new software technology. (2001) Available at <http://www.idi.ntnu.no/emner/sif80at/>
2. CLARiFi project home page. (2002) Available at <http://clarifi.eng.it/>
3. eCOTS project. (2002) Available at <http://www.industrie.gouv.fr/rntl/FichesA/E-Cots.html> (in French)
4. Boloix, G. and Robillard, P.: A Software System Evaluation Framework. IEEE Computer 12(8) (1995) 17-26
5. Bronsworrd, L., Oberndorf, T. and Sledge, C. A.: Developing New Processes for COTS-Based Systems. IEEE Software (2000) 48-55
6. Carney, D. and Long, F.: What Do You Mean by COTS? Finally a Useful Answer. IEEE Software 17(2) (2000) 83-86
7. Jaccheri, L. and Torchiano, M.: Classifying COTS products. In European Conference on Software Quality (ECSQ 2002) LNCS 2349 (2002) 246-255
8. Kontio, J.: A Case Study in Applying a Systematic Method for COTS Selection. In IEEE-ACM 18th International Conference on Software Engineering (ICSE) (1996) 201-209
9. Lawlis, P., Mark, K., Thomas, D. and Courtheyn, T.: A Formal Process for Evaluating COTS Software Products. IEEE Computer 34(5) (2001) 58-63
10. Morisio, M., Seaman, C., Parra, A., Basili, V., Condon, S. and Kraft, S.: Investigating and Improving a COTS-Based Software Development Process. In 22nd International Conference on Software Engineering (ICSE 2000) (2000) 32-41
11. Morisio, M. and Torchiano, M.: Definition and classification of COTS: a proposal. In International Conference on COTS Based Software Systems (ICCBSS) LNCS 2255 (2002) 165-175
12. Morisio, M. and Tsoukiàs, A.: IusWare: A methodology for the evaluation and selection of software products. IEE Proceedings-Software 144(3) (1997) 162-174
13. Ochs, M. A., Pfahl, D. and Chrobok-Diening, G.: A Method for Efficient Measurement-based COTS Assessment and Selection - Method Description and Evaluation Results. In IEEE 7th International Software Metrics Symposium (2001) 285-296
14. Torchiano, M., Jaccheri, L., Sørensen, C.-F. and Wang, A. I.: COTS Products Characterization. In Conference on Software Engineering and Knowledge Engineering (SEKE'02) (2002)