Empirical Study on COTS Components Classification

Jingyue Li, Finn Olav Bjørnson, Reidar Conradi
Department of Computer and Information Science,
Norwegian University of Science and Technology, Trondheim, No-7491, Norway
{jingyue, bjornson, conradi}@idi.ntnu.no

Abstract

COTS-based development is gaining more and more attention. Effective COTS component classification will help integrators to successfully control the development process, such as selection and integration. In this paper, we present an empirical study to investigate the characterized classification proposed by previous research. From the result of this study, we conclude that some attributes are not good because they are either not measurable or unnecessary. We also propose one other attribute that will affect the development process dramatically. Our future study will focus on investigating these attributes in a larger sample.

1. Introduction

Commercial-off-the-shelf (COTS) components are now widely used in software development. Unfortunately, the lack of a standardized and well-defined COTS classification scheme, results in many organizations making large investments in COTS components selection and evaluation. Furthermore, unsuccessful selection leads to many difficulties and risks in integration and future system maintenance and evolution.

Previous studies have proposed several attributes to classify COTS products. The values of these attributes are supposed to impact on development process of the delivered system [1][3].

To investigate how the development process is affected by COTS classification attributes, we have designed a structured survey to test some of these attributes. The survey is still not completed. So far, we have personally interviewed team members from five different projects in Norway. From the initial survey results, we have some preliminary insight on the validity and necessity of COTS classification attributes.

We found that size and type of functionality are not good classification attributes because they are either hard to measure accurately by the integrator (i.e. COTS user) or they have no actual effect on the development process.

Our result also found that the difference between developers' experience on COTS dramatically affects the future development process, such as selection, integration and maintenance. For other classification attributes, our initial study still cannot give conclusion on them. More data will be gathered to test them in the future.

The rest of this paper is organized as follows: In section 2 we give our definition of COTS components, in section 3, we describe the design and result of this study. Conclusion and future work are discussed in Section 4.

2. COTS components definition

In this study, we define a component as ***"A separable piece of software in the form of program code (source or executable), that will be integrated into a software system (consisting of components plus application), excluding platform software (commodities like OS, DBMS etc.)."***

The COTS component is defined as ***"A component that is not developed inside the same organization, excluding platforms and commodities like OS, DBMS etc. It is either provided by some other organization in the same company, or provided by external companies as a commercial product. It is evolved by the provider, and used by the customers without source code modification."***

3. Research design and result

From a literature review, we extracted several hypotheses related to COTS-based development. The purpose of our study is to test these hypotheses by data from real projects in IT companies. Some of the hypotheses are designed to test the relationship between the COTS classification attributes and the development process. This survey design includes two phases of pre-testing. After these pre-tests, the questionnaire will be distributed to more than 100 companies to obtain statistically significant conclusion on our hypotheses. The purpose of the first phase pre-test is twofold. First, we want to test the reliability and validity of each question. Second, we want to know whether some hypotheses based on the classifications come from literature are valid or not. We did five personal interviews in five projects in the first phase pre-test. Based on the result of this pre-test, some classifications were deleted and some classifications were clarified. The revised questionnaire is on the way of

the second pre-test. In this paper, we will discuss only the result of the first phase pre-test.

3.1. Background

Many researchers have proposed their classification on COTS component. Morisio et al. summarized some of the previous work and gave a proposal on how to classify COTS [1]. These classification attributes are supposed to be linked with software process (e.g. cost models, selection methods, architectures, testing and validation techniques etc.). The classification attributes come from previous literature. There is still no empirical study to evaluate them. Basili et al. proposed to classify the COTS products as hard and easy from their empirical study. Hard COTS are COTS products likely to raise risks. These risks depend on several properties, both of the organization using the COTS and of the COTS product itself [3]. They also proposed that the staffs' experience on the product and on COTS based development is the main factor that lowers the risks. Several hypotheses were proposed based on this theory. Torchiano et al. also proposed some COTS classification attributes [2]. These attributes came from students in one course in a university. There is still no future evaluation in IT companies.

3.2. Research methods

In the questionnaire for this survey, we classified COTS components used in projects based on some attributes. We also asked for details of several sub-processes, such as selection, integration, and maintenance for each COTS component. We then investigated whether there were relationship between the classification attributes and those sub-processes.

The attributes proposed by Morisio et al. are grouped into four categories [1]:

- Source: where the produce comes from
- Customization: how much the product can or should be customized
- Bundle: in what form the component is delivered, both to the integrator and to the customer of the system
- Role: what is the intrinsic role the product can assume in the final system

The attributes in these categories are summarized in the following table:

Category	Attributes
Source	Origin
	Cost & property
Customization	Required modification
	Possible modification
	Interface
Bundle	Packaging
	Delivered
	Size
Role	Functionality type
	Architectural level

In our survey, we selected some attributes in the above categories.

- In the source categories, we selected the ***origin*** attribute.
- In the customization categories, we selected the ***possible modification*** attribute.
- In the bundle category, we selected the ***size*** attribute.
- In the role category, we selected the ***functionality type*** attribute and ***architectural level*** attribute.

We listed several alternatives for each selected attribute. The definitions of alternatives are as follows:

Origin:

- By contract: The COTS was acquired by asking a component vendor to create a component for an agreed-upon fee.
- From market: The COTS was bought from the market, where the vendor is selling a product to the world and has no knowledge of the individual purchaser's requirements or constraints.

Possible modification:

- By Plug: Just integrate in the COTS components to assemble an executable system without any extra work.
- By Configuration: Setting or defining shell parameters or macro options available for a COTS component.
- By Glueware: Integrate COTS mostly by using in-house built code around COTS components to solve possible mismatch between components, or between components and application/platform.
- By Addware: Integrate COTS mostly by using in-house built code around COTS components to add the functionalities which are required but not provided by the components.
- By Source Modification: Change the source code of COTS so that it can be integrated.

Functionality type:

- General: The functionality is not specific to a domain, but can be used many different domains e.g. GUI functionality, Web browser functionality, etc.

- Specific: The functionality is specific to a domain and can be reused only in the domain e.g. financial functionality, telecommunication functionality, etc.

Architectural level:

- Core: The COTS components provided the main functionality of the system. The system was built around these COTS components.
- Support: The COTS components provided some functionality to support the central application of the system.

Size:

Morisio et al. proposed a classification scale of COTS size, i.e. small means less than 0.5 MB, medium means 0.5MB to 2MB, large means from 2MB to 20MB, and huge means more than 20MB. Because it is better to get the exact number of the value of interval variable and then define a classification scale afterwards, we used the open answers to let respondents to fill in the exact size of the components as following:

COTS : _____ KB
or ____ (KLOC)

In the questionnaire, we also have questions to ask the respondents' experience on components in order to study whether their previous experience on COTS component have effect on the future development process.

The questionnaire was sent to the respondent one or two days before the personal structural interview. To gather comments on each question, we added three extra evaluation questions behind every question. These three questions are:

1. Is this question easy to read as worded?
2. Do you need classification for this question?
3. If you have comments on this question, please fill in the lines.

In the process of the personal interview, the interviewer went through the questionnaire together with the respondent to gather feedback of questions in case there was something needs to be clarified. The interviewer also gathered answers of questions, which were clear enough.

3.3. Result

The output of the pre-test can be divided into two parts. First, we evaluated the classification attributes proposed to see whether they are measurable and necessary. ***Measurable*** means that the attributes can be quantified and measured. ***Necessary*** means that the value of the COTS components in these attributes will affect future COTS-based development process, such as selection, integration, maintenance, and evolution.

From the result of the pre-test of the questionnaire, we found two attributes that were not good classification attributes.

- Size: We do not believe size is a good attribute to classify COTS components because it is not measurable or necessary. First, most COTS components are compiled binary code, the only number that can be calculated is how much space the COTS component takes up in the form of KB on a file. However, compiler differences make it very hard to calculate the lines of source code or use cases from components' physical size in KB. So, the conclusion is that you can get an estimate size but it cannot tell you anything. Therefore, we define it as non-measurable. Second, most respondent do not care about the size of COTS components. When asked this question, almost everyone needs to re-check the system and calculate the size by KB. Morisio et al. supposed the size will affect the learnability [1]. But respondents said they used only part of the functionalities provided by COTS and believed that size could not affects the development process. So, we believe that this attribute is also not necessary.

- Functionality type: We do not believe functionality type is a good attribute to classify COTS components because it is hard to measure by the component user and it will not affect the development process, and is therefore unnecessary. For example, one respondent said they selected a general component from market, but they had to ask the vendor to change the source code to meet the requirement of their specific domain, which changed it into a specific component. So, he didn't know how to define the functionality type from his viewpoint. Morisio et al. supposed the functionality will affect the reusability across projects [1]. The assumption is that specific type components may have better reusability across projects. However, one respondent argued that general type components could also have good reusability across projects if the requirements of both projects are general, for example, PDF file generation. They also said that they searched the components only by the required functionalities not by the required domain. We investigated websites of several main component brokers, such as ComponentSource, Component Vendor Consortium, Xtras.net, VBxtras. We found that none of them classified components based on their usage domain. The possible reason is that

the functionality type of most components in the market is horizontal, i.e. the functionality is not specific to a domain. As a result, we regarded this attribute is also not necessary either.

Another output of our pre-test was the identification of a new classification attribute that affected the component users' decision in the component selection process. The attribute is the component users' or colleagues' experience with the components. From the survey result we found that component users prefer to select components they have experience with, the experience coming either from themselves or from colleagues. If they had the experience, the component selection procedure would become very short, because they would not use time to search and evaluate the component again. In our survey, four respondents had previous experience with the component they used, so they selected the component mainly on their previous experience. If they did not have experience, they would need more time and effort to search and evaluate the components. One respondent of our survey had no experience with the components he used in the new project. Although he used much effort in searching and evaluating the component, he still thought more effort should be used in the component selection phase because he found many problems after the component had been decided. From the interview, we found it is hard to measure the experience of each person. However, some respondents proposed to define the scale of experience on the experience of the whole project. So, our scale for this new attribute is:

- Very much experience: Roughly more than ¾ project members have used this component in previous project
- Much experience: Roughly ½ project members have used this component in previous projects
- Some experience: Roughly ¼ project members have used this component in previous projects
- Little experience: Roughly less than ¼ project members have used this component in previous projects
- No experience: No project members have used the same components before.

4. Conclusion and future work

The contribution of this paper includes two parts. First we did literature study on proposed classification attributes. These attributes are proposed as internal COTS components classification attributes, because they are supposed to impact development process. We evaluated whether the attributes could be quantitatively measured. Then we measured whether the value of the classification attributes could affect the development process. We discovered that attributes like functionality type and size are either not measurable or not necessary. The second contribution is that we proposed developers' experience on COTS components as a new classification attribute that is measurable and necessary.

Based on the above result, the questions about size and functionality type attribute of COTS components were deleted in our revised questionnaire. The question about developers' experience on COTS components was revised and relating scales were added.

We did not find systems using COTS components as the core part of the system or COTS components acquired by contract so far. We could not give any conclusion on the origin and architecture attributes. The small sample could also not support any conclusions on the possible modification attribute. Our future study will investigate these attributes based on a bigger sample.

5. References

[1] Maurizio Morisio, and Marco Torchiano, "Definition and Classification of COTS: a Proposal", in proceedings of ICCBSS 2002, LNCS 2255, pp. 165-175.

[2] Macro Torchiano, and Letizia Jaccheri, "Assessment of Reusable COTS attributes", in proceedings of ICCBSS 2003, LNCS2580, pp. 219-228.

[3] Victor R. Basili, "Evolving Knowledge about COTS-Opening Keynote", ICCBSS 2003, available at http://www.iccbss.org/2003/abstracts.html