Validation of New Theses on Off-The-Shelf Component Based Development

Jingyue Li[1], Reidar Conradi[1,2], Odd Petter N. Slyngstad[1], Christian Bunse[3], Umair Khan[3], Marco Torchiano[4] and Maurizio Morisio[4]
[1]*Department of Computer and Information Science, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway*
{jingyue, conradi, oslyngst}@idi.ntnu.no
[2]*Simula Research Laboratory, P.O.BOX 134, NO-1325 Lysaker, Norway*
[3] *Fraunhofer IESE, Sauerwiesen 6, D- 67661 Kaiserslautern, Germany*
{Christian.Bunse, khan}@iese.fraunhofer.de
[4]*Dip. Automatica e Informatica, Politecnico di Torino Corso Duca degli Abruzzi, 24, I-10129 Torino, Italy*
{maurizio.morisio, marco.torchiano}@polito.it

Abstract

Using OTS (Off-The-Shelf) components in software development has become increasingly popular in the IT industry. OTS components can be either COTS (Commercial-Off-The-Shelf), or OSS (Open-Source-Software) components. A recent study with seven structured interviews concluded with six theses, which contradicted widely accepted (or simply undisputed) insight. Since the sample size of that study was very small, it is necessary to investigate these theses in a larger and randomized sample. A state-of-the-practice survey in three countries – Norway, Italy, and Germany – has been performed to validate these new theses. Data from 133 OTS component-based projects has been collected. Results of this survey support four and contradict two of the initial theses. The supported theses are: OSS components were mainly used without modification in practice; custom code mainly provided additional functionality; formal OTS selection processes were seldom used; OTS component users managed to get required changes from vendors. The unsupported theses are: standard mismatches were more frequent than architecture mismatches; OTS components were mainly selected based on architecture compliance instead of function completeness.

1. Introduction

OTS components (Off-The-Shelf) includes COTS (Commercial-Off-The-Shelf) and OSS (Open Source Software) components. Developing with the OTS components is gaining more attention from both the research and industrial community. A newly performed study, in the form of seven structured interviews, proposed six new theses in OTS based development [12]. Since the sample size of that study was small, it is necessary to investigate the six theses in a larger and randomized sample. We therefore designed a state-of-the-practice study on OTS component-based development to validate these theses. The survey was performed in three European countries (Norway, Italy, and Germany). We have gathered answer from 133 projects using either COTS or OSS components from a sample of 1500 companies.

Data of this survey support four of the six new theses and contradict the other two. How to interpret the survey result is also discussed in detail in order to give recommendations and suggestions on OTS based development.

The rest of this paper is organized as follows: Section 2 introduces the six theses we intend to investigate. Section 3 presents the research questions and survey design. Section 4 presents the actual sample. Section 5 lists results of each research question. Detailed discussion on the survey results is given in Section 6. Conclusions and future work are presented in Section 7.

2. Six new theses on OTS-based development

After performing seven structured interviews of small and medium software companies in Norway and Italy, Torchiano and Morisio summarized six new theses which contradicted statements or implicit assumptions from previous literatures [12].

2.1 New Thesis T1: Open source software is often used as closed source.

In previous literatures, OSS components are usually considered to be completely different from closed-source components [6]. OSS products are provided by open source communities with source code available. COTS components are provided by a commercial vendor. The source code of COTS component is not available in most cases. Torchiano and Morisio discovered that developers didn't look and change the source code of OSS components. The assumed explanation is that the users didn't need to see and modify, or they lacked the knowledge, skill, or resources to do so [12].

2.2 New thesis T2: Integration problems result from lack of compliance with standards. Architecture mismatches constitute a secondary issue.

In previous literature, architectural mismatches were cited as one of the primary sources of integration problems [5, 13]. However, Torchiano and Morisio did not find any instances of this. The integration problem in their interviewed was that the selected components did not fully or correctly support the claimed standards [12].

2.3 New thesis T3: Custom code mainly provides additional functionalities.

The previous literature greatly emphasized the integration aspects of custom code (named glueware or gluecode) [3, 9]. However, Torchiano and Morisio concluded that one of the most common remedial solutions among interviewees was to add code (called addware) to provide missing functionalities [12].

2.4 New Thesis T4: Developers seldom use formal selection procedures. Familiarity with either the product or the generic architecture is the leading factor in selection.

Previous studies have proposed several structured, formal, or semi-formal selection procedures [1, 7]. However, no project interviewed by Torchiano and Morisio used any of them.

2.5 New thesis T5: Architecture is more important than requirements for product selection.

For COTS component selection, previous literature proposes selection processes mainly based on requirements and their (re)negotiation [2, 7]. However, Torchiano and Morisio discovered that the architecture drove the selection process, and was the fulcrum of all trade-off activities [12].

2.6 New thesis T6: Integrators tend to influence the vendor on product evolution whenever possible.

Literature generally describes the lack of control over the features and evolution of COTS component as an immutable condition that integrators must tolerate [3]. However, Torchiano and Morisio found out that their interviewees were acting to modify this condition [12].

3. Research design

To test the six new theses, we decided to collect more data using a randomized sample. Our initial plan was to transfer the theses directly into hypotheses and make a quantitative survey with standardized questions in a formal questionnaire. We designed six fixed hypotheses according to the new theses and a preliminary questionnaire. The process was as following:

- Jan.-June 2003: Read and discuss about the topic.
- June-Aug. 2003: Make initial research questions and draft questionnaire (in English).
- Late Aug. 2003: first industrial pre-test in five local IT companies and with three internal colleagues.
- Sep.-Oct. 2003: Revision of the draft questionnaire.
- Late Oct. 2003: Second pre-test in five companies.

After these two rounds of pre-testing, we discovered that it was difficult to reliably investigate the six theses without a pre-study to clarify our research questions. For example, we asked respondents whether they had influenced the vendors on product evolution. The question was intent to investigate **T6**. After the pre-tests, we concluded that it was not possible to get reliable results on this question. Some respondents had pushed the vendor to change a COTS component concerning bugs fixing or new functionalities. Although these respondents could tell whether they had got the required changes or not, they were not sure whether they had influenced the evolution of the COTS component.

3.1. A pre-study to clarify research questions

To clarify the research questions, we decided to resort to structured interviews, i.e. a qualitative pre-study, before using a larger and randomized sample. The pre-study was designed to employe more open research questions, not fixed hypotheses, and with a rather detailed interview guide. We used a convenience sample, i.e. a list of large, medium and small IT companies from existing project contacts. We ended up with 16 personal interviews in 13 Norwegian IT companies with an initial

selection of 34 companies. The results of the pre-study were reported in [8].

3.2. Clarified research questions

After the pre-study, we were able to make testable research questions to investigate each new thesis. Although the research questions might not give complete answers to each thesis, they were designed to investigate each thesis in a reliable way.

3.2.1. Research questions for thesis T1. T1 claims that OSS components are being ***used*** as COTS components in practice. From the pre-study, we discovered that we had to interpret the word "used" correctly. The word "used" in [12] means that the source code of OSS component was seldom read and changed. OSS components were therefore "used" as "black-box". As the meaning of "used" in T1 includes two activities, read and change, we designed two research questions T1.RQ1 and T1.RQ2 to study T1.

- **T1.RQ1:** *Had the source code of OSS components been read by integrators during software development?*

- **T1.RQ2***: Had the source code of OSS components been changed by integrators during software development?*

3.2.2. Research questions for thesis T2. T2 indicates that standard mismatches were the most frequent problem in COTS based development, and the architecture mismatches were a secondary issue. From the pre-study, we concluded that it was difficult to conclude that the architecture mismatches constituted a second issue. Although we can answer whether standard mismatches were more frequent than architecture mismatches, some other mismatches might be more frequent than these two. As a result, we limited our research question to compare the frequency of standard mismatches and architecture mismatches, as shown in T2.RQ1.

- **T2.RQ1:** *Were the standard mismatches more frequent than architecture mismatches?*

3.2.3. Research questions for thesis T3. T3 argues that most custom code was used to provide additional functionalities rather than solving mismatches between components, or between components and application. In one word, T3 concludes that more addware (code to add missing functionalities) was built than glueware (code to solve various mismatches) in COTS-based development. In the pre-study, we asked the respondents to give the exact LOC (Line-Of-Code) of glueware and addware they developed in the project. However, most respondents could not give us the exact LOC of glueware/addware. Some respondents claimed that this information was confidential. Others complained that it was time-consuming to calculate this number. To avoid getting wrong or incomplete data, we limited the research question to investigate the occurrences of building glueware and addware instead of comparing the exact LOC. So, the research question T3.RQ1 is:

- **T3.RQ1:** *Was the occurrences of building glueware more frequent than the occurrences of building addware?*

3.2.4. Research question for thesis T4. T4 concludes that formal selection procedures, which generally come from the academia, were seldom used in practice. T4 also concludes that familiarity with either the product or the generic architecture was used as the leading factor in COTS component selection.

From the pre-study, we discovered that the definition of "formal selection procedure" in [12] was too general, because there are several different kinds of formal selection methods. After detailed analysis, we concluded the formal process defined in [12] is a kind of process includes three basic elements: selecting evaluation criteria (factors); collecting and assigning values to these criteria; making decisions using formal decision making methods. In the pre-study, we discovered two other selection processes were in frequent use [8]:

- ***Familiarity-based selection process:*** If the project members had enough previous experience on the candidate COTS components, this experience will be the key factor in the COTS component selection.

- ***Hands-on experimentation based selection process:*** This kind of selection was used when there was no previous experience available. This selection process was more complex than *familiarity-based selection process*, and had three steps:

  Step 1: First, developers used a search engine to browse the Internet, and used keywords to express decided functionality. This usually gave them a handful of candidates.

  Step 2: If there were more than 2 to 3 possible candidates, they selected 2 to 3 of them very quickly based on some key issues, such as license issues, cost, or vendor reputation.

  Step 3: After a small number of candidate COTS components had been decided, they downloaded a demo version of these from the web and tested the COTS components locally.

To test T4, we decided to compare how often these three kinds of selection processes had been used in

practice. So, the corresponding three research questions are:

- **T4.RQ1:** *How often had a formal process been used in practice?*

- **T4.RQ2:** *How often had the familiarity-based selection process been used in practice?*

- **T4.RQ3:** *How often had the hands-on experimentation based selection process been used in practice?*

3.2.5. Research questions for thesis T5. T5 claims that architecture was more important than requirements for product selection. From the pre-study, we discovered that the architecture decision affected most components in the system and there were trade-offs between architecture and COTS component selection. Some COTS components were selected to satisfy the required functionalities. Other COTS components in the same project were selected because they could easily be integrated, although they could not provide all required functionalities. Since different COTS components in the same project might be selected according to different criteria, our research question T5.RQ1 investigate which criteria were more important for the **whole project** rather than just for **a specific COTS component**.

- **T5.RQ1:** *Had architectural compliance been considered more important than functionality completeness in the whole COTS-based project?*

3.2.6. Research questions for thesis T6. T6 argues that COTS users were not completely passive in the COTS market and they managed to influence the vendor on product evolution whenever possible. As mentioned in above, it was difficult to know whether the COTS users actually influenced COTS component evolutions or not, even if it was possible to know whether the required changes (corrections or new functionalities) from the COTS users had been satisfied. To get a reliable conclusion on T6, we limited our research question T6.RQ1 to evaluate how often the required changes had been implemented by the vendors.

- **T6.RQ1:** *How often the required changes (corrections or new functionalities) from integrators had been satisfied by the vendors.*

3.3. Questionnaire design

After we clarified the research questions based on results from the pre-study, we designed a new questionnaire. To validate thesis T1, several questions related to OSS components were also included.

The questions in the questionnaire were organized into three main parts:

1. Questions to collect background information of the company, project, and respondents.
2. Questions to get answers of the research questions mentioned in Section 3.2 (Detailed information of each question is introduced in the Section 5). Some other questions to investigate process improvement and risk management were also included. However, answers to these questions are out of the scope of this paper.
3. Questions to collect information about OTS components actually used in the project. In the questionnaire, we asked the respondents to give brief information (name, COTS/OSS, version) of all OTS used and detailed information of one of the most important COTS/OSS component (the one provides most functionality compared to other OTS components). For the selected most important COTS/OSS component, we gathered detailed background information, such as:
   - Name, version, and functionality
   - How large a percentage of the application functionality in the whole system is provided by the selected OTS component?
   - Whether the source code is available or not?
   - What was the technical base, such as Library, COM; CORBA, EJB etc.

Concepts used in this survey are listed in the first page of the questionnaire. In general, the concepts are consistent with definitions proposed by Torchiano and Morisio [12]. The difference is that we extended the definition to involve OSS component. The key concepts are:

- **Component**: Software components are program units of independent production, acquisition, and deployment and which can be composed into a functioning system. We limit ourselves to components that have been explicitly decided either to be built from scratch or to be acquired externally as an OTS-component. That is, to components that are not shipped with the operating system, not provided by the development environment, and not included in any pre-existing platform.

- An **OTS component** is a component provided (by a so-called provider) from a commercial vendor or the Open Source community. An OTS component may come with certain obligations, e.g. payment or licensing terms. An OTS component is not controllable, in terms of provided features and their evolution.

The final questionnaire was first designed and pre-tested in English (internal and external reviews). It was then translated into the native languages and published on the SESE web survey tool [11] at Simula Research Lab in OSLO.

3.4. Sample definition and selection

We define the unit of this study as a completed software development project, and its OTS-relevant properties. The projects were selected based on two criteria:

- The project should use *one or more OTS components.*
- The project should be a *finished project*, possibly with maintenance, and possibly with several releases.

Projects were collected randomly from IT companies in Norway, Italy and Germany. Slightly different sampling procedures were used in each country due to limited resources. Detailed discussions on sample selection in this study are reported in [4].

- In Norway, we gathered a company list from the Norwegian Census Bureau (SSB) [10]. We included mainly companies which were registered as IT companies. Based on the number of employees, we selected the 115 largest IT companies (100 IT companies and 15 IT departments in the largest 3 companies in 5 other sectors), 200 medium-sized software companies (20-99 employees), and 100 small companies (5-19 employees) as the original contacting list.
- In Italy, we first got 43580 software companies from the "yellow pages". We then randomly selected companies from them. For these randomly selected companies, we read their web-site to ensure they really are software companies. 196 companies were finally clarified as software companies, and were included in the original contact list.
- In Germany, we composed a company list from a company list from an organization similar to the Norwegian Census Bureau and in a similar manner as for Norway. We then used the existing Fraunhofer IESE customer database to get contact information.

3.5. Data collection procedure.

To avoid bias in the data collection procedure, we made a guideline on how to contact respondents before the survey started. All countries – Norway, Italy, and Germany – followed the process in the guideline.

Possible respondents were contacted first by telephone. If the candidate respondents had suitable OTS-based projects and would like to join our study, a username and password of the survey tool and an electric version of the questionnaire were sent to them. The respondents could use either the SESE web tool or the electronic version to fill in the questionnaire. The respondents who didn't want to answer the questionnaire were also registered. We logged the main reasons of non-response, such as "no software development", "no OTS-based projects", and "busy". Most respondents used the Web tool to fill in the questionnaire. However, phone interview were used in Germany because of the confidential concerns in some companies [4].

3.6. Data analysis method

We used different analysis methods to analyze different research questions.

- Thesis **T1** is relevant to OSS components. To study research questions T1.R1 and T1.R2, we compared the usage of OSS components with COTS components.
- Because **T2**, **T3**, **T4**, and **T6** are not relevant to OSS components, only information from COTS components was analyzed.
- As we mentioned in section 3.2.5, thesis **T5** should be tested using data from the whole project rather than a specific COTS component. The unit to be analyzed for research question T5.RQ1 is therefore the whole project instead of the selected OTS component.

4. Selected sample

4.1. Selected companies

We have gathered results from 133 projects (47 from Norway, 48 from Germany, and 38 from Italy) from 127 companies. In general, we selected one project from each company. However, we selected more than one projects in three Norwegian IT companies because those companies have many OTS-based projects and would like to share more experience to this study.

The responding companies involved small (5-19 employees), medium (20-99 employees) and large (100 or more employees) ones as shown in Figure 1. The main business areas of the sampled companies also cover different types as shown in Figure 2.

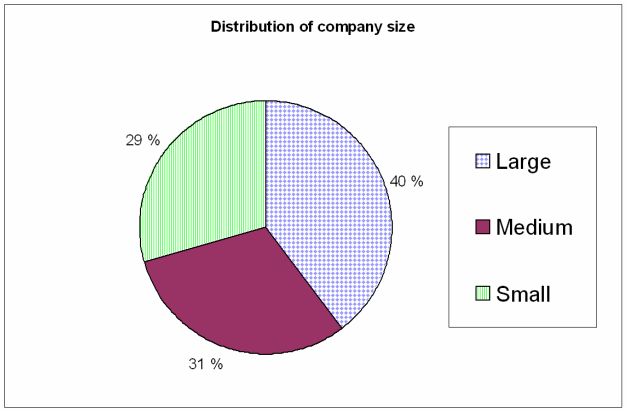

Figure 1. Distribution of the company size

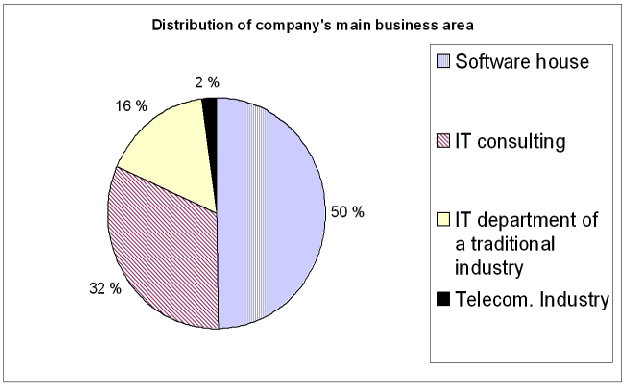

Figure 2. Distribution of the company's main business area

4.2. Selected projects

The final systems produced by the 133 projects cover different application domains as shown in the Figure 3.

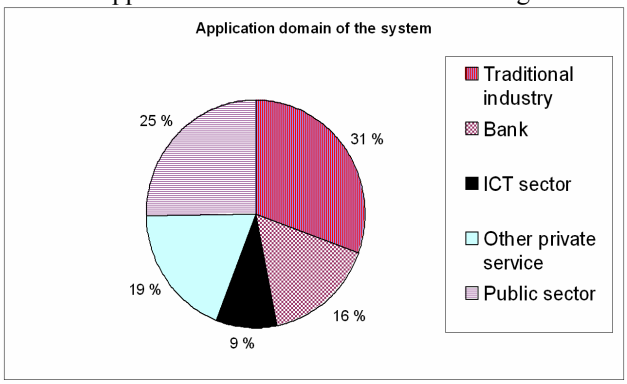

Figure 3. Distribution of application domain of the systems produced by these projects

In the selected 133 projects, 83 used only COTS components, 44 used only OSS components, and six used both COTS and OSS components. For the 44 projects using only OSS components, each of them gave the detailed information of an OSS component. For the 83 projects using only COTS components, 83 different COTS components were selected. For the six projects used both COTS and OSS components, the respondents selected to give detailed information of five COTS components and one OSS component. In total, we got detailed information of 88 COTS components and 45 OSS components.

4.3. Respondents

Most respondents of the 133 projects have a solid IT background. More than 90% of them are IT managers, project managers, or software architects. Most of them have more than 2 year experiences with OTS-based development. All of them have at least a bachelor degree in informatics, computer science, or telecommunication.

5. Answers to research questions

5.1. Answers to T1.RQ1 and T1.RQ2

T1.RQ1 and T1.RQ2 is to test thesis T1, i.e. whether the OSS components had been used in the same way as COTS components. To investigate T1.RQ1, the corresponding question in the questionnaire is:

Have you read parts of the source code of OTS component?

We used a five-point Likert scale to measure the answer. Respondents were asked to answer "very little", "little", "some", "much", "very much", or "don't know".

For question T2.RQ2, we used the same measurement scale as for T1.RQ1, and the corresponding question is:

Have you modified parts of the source code of OTS component?

We assigned an ordinal number from 1 to 5 to the above alternatives (5 meaning "very much"). Results of these two research questions are summarized in Figure 4. The results show that the median value of reading OSS code is 3 (meaning "some"). However, the source code had seldom been changed as the median value of changing the OSS code is 2 (meaning "little"). Our future analysis discovered that 46% of the respondents claimed that they modified "very little" parts of the source code in OSS components.

Another question in the questionnaire asked if the source code of OTS components is available or not. Results show that not all COTS components in this study were "black-box". 29 (out of 88) selected COTS components provided source code to their users. To compare with OSS components, we investigated whether the source code in "white-box" COTS components have been read or modified. Results are shown in Figure 5.

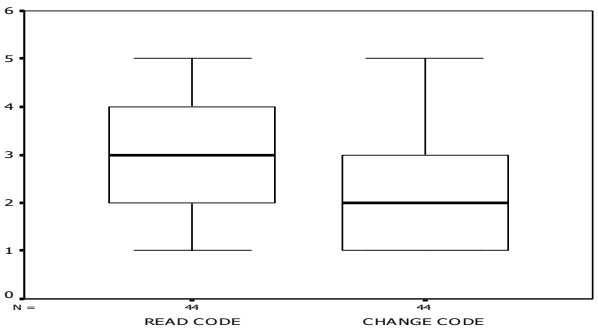

Figure 4. The source code usage of the OSS components

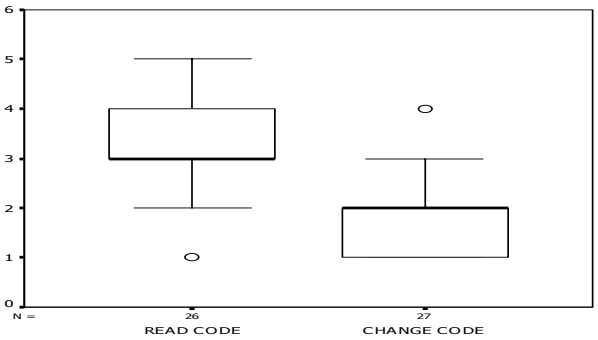

Figure 5. The source code usage of the "white-box" COTS components

From Figure 4 and Figure 5, we can see that the source code of COTS or OSS components had been read to some degree, if it was available. However, the source code in both COTS and OSS components had seldom been changed.

5.2. Answers to T2.RQ1

T2.RQ1 asks whether the architecture mismatches were more frequent than standard mismatches. To investigate T2.RQ1, the corresponding question is:

Did you encounter some of the following aspects (risks) with the selected OTS component?

Standard mismatch and architecture mismatch were listed as option a) and b) of this question:

a) The OTS component did not follow industrial standards (COM, CORBA etc.) as the provider claimed (i.e. glueware was needed)
b) The OTS component had mismatching architectural assumptions towards other parts of the system (i.e. glueware was needed)

Respondents were asked to answer "yes", "no", or "don't know". For the selected 88 COTS components, only 4.5% of the COTS components had standard mismatches, compared with 28.4% COTS components having architecture mismatches.

5.3. Answers to T3.RQ1

T3.RQ1 is to test thesis T3, i.e. to compare glueware vs. addware. To investigate T3.RQ1, we asked respondents the same questions as in T2.RQ1. We listed another option c):

c) The OTS component could not provide all required functionality (i.e. addware needed)

Respondents were also expected to answer "yes", "no", or "don't know".

In data analysis, we added all the "yes" answers for a) and b) options of this question to count the occurrences of building glueware. The rationale is that glueware will be needed in case of either standard, or architecture mismatch. To count the occurrences of building addware, we added all the "yes" answers for option c).

Results show that glueware was needed for 32.9% of the selected COTS components and addware was needed for 65.9% of the selected COTS components.

5.4. Answers to T4.RQ1 to T4.RQ3

To answer T4.RQ1 to T4.RQ3, the corresponding question is:

Have you performed some of the following actions for the selected OTS component?

The listed options are a) to f) as follows:

- *Concerning the identification of OTS components:*
  a) Searched Internet for possible OTS component candidates.
  b) Got recommendation of possible OTS component candidates from the customer
  c) Got recommendation of possible OTS component candidates from a local colleague/OTS-expert
- *Concerning the evaluation of components:*
  d) Used a formal decision-making method to compare possible OTS component candidates, e.g. with weighted evaluation criteria
  e) Limited possible candidates into 1-3 components, by reading literature or other documentation
  f) Did "hands-on" try-out of 1-3 components, e.g. on a down-loaded demo version.

Respondents were asked to answer "yes", "no", or "don't know" for each option. Results of the five options are listed in Table 1.

Data in Table 1 shows that one common criterion for identifying OTS components is familiarity-based. 58% COTS components were selected according to local expert (option c). Formal procedures (option d) were only used in evaluating 19% of the selected COTS components. The most common evaluation method was

hands-on experimentation (option f). It was used to select 65% of the selected COTS components in this study.

Table 1. Results of T4.RQ1 – T4.RQ3

Option	Yes	No	Don't know
a)	59%	36%	5%
b)	24%	73%	3%
c)	**58%**	36%	6%
d)	**19%**	74%	7%
e)	48%	45%	7%
f)	**65%**	27%	8%

5.5. Answers to T5.RQ1

To answer T5.RQ1, the corresponding question is:

What actions were performed during development or maintenance of the project?

We listed several possible actions in the project. The action relevant to T5.RQ1 is: OTS components were selected mainly based on architectural and standards compliance, instead of expected functionality.

Respondents were asked to answer "don't agree at all", "hardly agree", "agree somewhat", "agree mostly", "strongly agree", or "don't know". We assigned an ordinal number from 1 to 5 to the above alternatives (5 meaning strongly agree). Results of the 83 projects using only COTS components are shown in Figure 6.

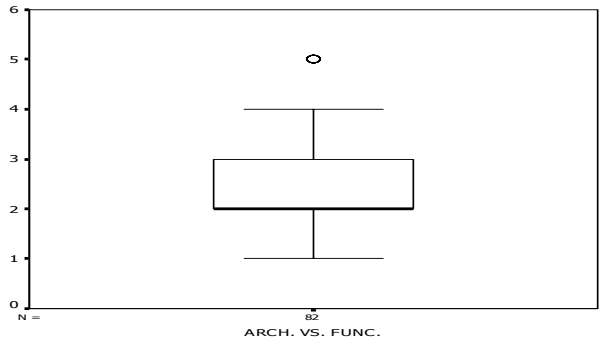

Figure 6. Result of T5.RQ1

From Figure 6, we can see that the median value of this question is 2. It indicates that most respondents **hardly agree** that COTS components were selected mainly based on architectural and standards compliance. So, we concluded that expected functionality still constituted the main concern in COTS components selection.

5.6. Answers to T6.RQ1

To investigate T6.RQ1, we asked the same question as T2.RQ1 with another option e):

e) The provider did not want to or could not change OTS component functionality as requested.

Results show that 60% of the selected COTS components have been changed by vendors according to integrators' requirements.

6. Discussion

6.1. Differences between COTS and OSS components

Our results support thesis **T1**. If "white-box" vs. "black-box" is regarded as the main difference between COTS and OSS components, results of T1.RQ1 and T1.RQ2 show that COTS and OSS component are not as different as people might assume. COTS components were not always "black-box", as 33% of the COTS components selected in this study opened their source code to users.

If the source code is available, both COTS and OSS users tend to read the code. However, both COTS and OSS component users did not change much of the available source code. From this result, we can generally conclude that the OSS component was used the same as COTS component, i.e. integrated without modification.

6.2. COTS component selection and integration

Our results support thesis **T4**. For COTS component selection, results of T4.RQ1 to T4.RQ3 show that developers often choose a sub-optimal product only because they already know it. Formal selection procedures try to find out the most optimal product. However, these procedures might not be cost-effective in some industrial projects. As a result, selection techniques should also include familiarity as a main criterion [12].

Results of T5.RQ1 do not support thesis **T5**. COTS components users believe that functionality completeness is more important than architectural compliance in COTS component selection. However, we need to interpret this result carefully. The result does not claim that architectural and standard compliance are not important in COTS component selection. One lesson learned from our pre-study was that easiness of integration should be seriously considered before you decide to use a COTS component [8]. The possible explanation for result of T5 is that all candidate COTS components support required standards and architectures. So, the only comparable features are their functionalities. Then, COTS components were appeared to be selected based on their functionalities instead of architectural compliance.

For COTS component integration, results of T2.RQ1 contradict thesis **T2,** and show that architecture mismatches were more frequent than standard mismatches. With the maturity of COTS component standard, we believe that less standard mismatches will occur in the future.

Results of T3.RQ1 support thesis **T3.** It means that missing or incomplete features in COTS products raise significant problems. It is therefore important to develop effective techniques that allow seamless extension of existing products [12].

6.3. Vendor relationship in COTS based development

Results of T6.RQ1 support thesis **T6**. It shows that COTS component users actually pushed vendors to change the COTS components either because of bug fixes or new features. However, this result does not imply that most COTS component users actually influenced COTS component evolution. In general, we believe that the COTS component evolution is decided by the vendors' marketing strategy rather than the requirements of a specific user, unless this user is very powerful.

6.4. Possible treats to validity

6.4.1. Construct validity In this study, most variables and alternatives are taken directly, or with little modification, from existing literature. The questionnaire was pre-tested using a paper version by 10 internal experts and 8 industrial respondents before being published on the SESE tool. About 15% questions have been revised based on pre-test results. One possible threat to construct validity is that respondents from different companies may have slight different understanding on the same concept, such as software architecture and architectural mismatches.

6.4.2. Internal validity. We have promised respondents in this study a final report and a seminar to share experience. The respondents were persons who wanted to share their experience and wanted to learn from others. In general, we think that the respondents answered the questionnaire truthfully. However, different persons in the same project might have different opinions on the same project. Asking only one person in each project might not be able to get the whole picture of the project. Due to length limitation of a questionnaire, we asked the respondent to fill in information of only one component in the project. The possible threat is that other OTS components in the same project might give different answers to our questions.

6.4.3. Conclusion validity. This study is a state-of-the-practice study. We studied what had happened in industrial projects. However, we did not investigate the cause-effect relation of the phenomena discovered in this study.

6.4.4. External validity. We used different randomization to select samples in different countries. However, the sample selection processes were not exactly the same due to resource limitations [4]. Another possible threat to external validity is that our study focused on fine-grained OTS components. Conclusions may be different in projects using complex and large OTS packages, such as ERP, content management systems, and web services in general.

7. Conclusion and future work

This paper has presented results of a state-of-the practice survey on OTS-based development in industrial projects. The study has investigated six new theses in OTS component based development. Results of this study gave support on four (T1, T3, T4, and T6) new theses and concluded that:

- T1: Source code of OSS components was read frequently with very few modifications. OSS components were de facto used as COTS components, even if the source code was available.
- T3: In the COTS components integration process, more addware had been built than glueware.
- T4: Formal selection procedures were seldom used to select COTS components. Familiarity-based and hands-on experiment based selection process were frequently used in practice.
- T6: In most cases, COTS users managed to get their required changes in the COTS components from COTS vendors.

Results of this study contradict the two (T2 and T5) other theses and concluded that:

- T2: There were more architecture mismatches than standard mismatches in practice.
- T5: Functionality completeness was regarded more important than architectural compliance in COTS component selection.

As this study is a state-of-the-practice study, we could not give cause-effect conclusions based on results of this survey. The next step is to do a further qualitative study with structured interviews to find out the possible explanations for the conclusions of this study.

8. Acknowledgements

This study was partially funded by the INCO (INcremental COmponent based development) project [14]. We thank the colleagues in these projects, and all the participants in the survey

9. References

[1] Albert, C. and Brownsword, L. "Evolutionary Process for Integrating COTS-Based System (EPIC): An Overview," SEI, Pittsburgh, 2002. Available at:

http://www.sei.cmu.edu/publications/documents/02.reports/02tr009.html.

[2] Boehm, B., "Requirements That Handle IKIWISI, COTS, and Rapid Change," *IEEE Computer*, Vol. 33, No. 7, 2000, pp. 99–102.

[3] Boehm, B. and Abts, C., "COTS Integration: Plug and Pray?" *IEEE Computer*, Vol. 32, No. 1, 1999, pp. 135–138.

[4] Conradi, R., Li, J., Slyngstad, O. P. N., Bunse, C., Kampenes, V.B., Torchiano, M., and Morisio, M. "Reflections on conducting an international CBSE survey in ICT industry," submitted to 4th International Symposium on Empirical Software Engineering, Nov. 2005, Noosa Heads, Australia, 11 pages.

[5] Egyed, A., Medvidovic, N., and Gacek, C., "Component-Based Perspective on Software Mismatch Detection," *IEE Proc.-Software*, Vol. 147, No. 6, 2000, pp. 225–236.

[6] Lawton, G., "Open Source Security: Opportunity or Oxymoron?" *IEEE Computer*, Vol. 35, No. 3, 2002, pp. 18–21.

[7] Lawlis, P., Mark, K. E., Thomas, D.A., Courtheyn, T. "A Formal Process for Evaluating COTS Software Products," *IEEE Computer*, Vol. 34, No. 5, 2001, pp. 58–63.

[8] Li, J., Bjørnson, F. O., Conradi, R., and Kampenes, V. B., "An Empirical Study of Variations in COTS-based Software Development Processes in Norwegian IT Industry," *Proc. of the 10th IEEE International Metrics Symposium*, Chicago, USA, September, 2004, IEEE CS Press (2004), pp. 72-83.

[9] Morisio, M. et al., "Investigating and Improving a COTS-Based Software Development Process," *Proc. of 22nd International Conference on Software Engineering,* Limerick, Ireland, June, 2000, ACM Press, pp. 32–41.

[10] Norwegian Census Bureau (2004): Available at: http://www.ssb.no

[11] SESE tool (2004), Available at: http://sese.simula.no

[12] Torchiano, M. and Morisio, M., "Overlooked Aspects on COTS-based Development," *IEEE Software*, Vol. 21, No. 2, 2004, pp. 88-93.

[13] Yakimovich, D., Bieman, J., and Basili, V., "Software Architecture Classification for Estimating the Cost of COTS Integration," *Proc. 21st International Conference on Software Engineering (ICSE 99)*, ACM Press, 1999, pp. 296–302.

[14] INCO Project (2000): INCO project description, Available at: http://www.ifi.uio.no/~isu/INCO.