Observations on Versioning of Off-the-Shelf Components in Industrial Projects (short paper)

Reidar Conradi[1,2] and Jingyue Li[1]

[1]Department of Computer and Information Science,
Norwegian University of Science and Technology (NTNU),
NO-7491 Trondheim, Norway
[2]Simula Research Laboratory, P.O.BOX 134, NO-1325 Lysaker, Norway
{conradi, jingyue}@idi.ntnu.no

Abstract. Using OTS (Off-The-Shelf) components in software projects has become increasing popular in the IT industry. After project managers opt for OTS components, they can decide to use COTS (Commercial-Off-The-Shelf) components or OSS (Open Source Software) components. We have done a series of interviews and surveys to document and understand industrial practice with OTS-based development in Norwegian, German, and Italian IT industry. The perspective is that of a software architect or system integrator, not a developer or maintainer of such components. The study object is a completed development project using one or several OTS components. This paper reports on the versioning aspects of OTS components in such projects. We found that one third of the COTS components actually provided source code, in addition to all OSS components. However, OTS components were seldom modified (i.e. reused "as-is"), even if source code was available. Although backward compatibility of new releases did not cause noticeable problems for a single OTS component, using several different OTS components in a project caused difficulties in maintenance planning of asynchronous releases and system integration of new releases. Several new research questions have been formulated based on the results of this study.

1. Introduction

Software reuse in the form of *component-based software development (CBSD)* has long been proposed as a "silver bullet". It is supposed to offer lower cost, shorter time-to-market, higher quality, and stricter adherence to software standards. Software developers are therefore increasingly using COTS (Commercial-Off-The-Shelf) and OSS (Open Source Software) components in their projects, commonly called OTS (Off-the Shelf) components. COTS components are owned by commercial vendors, and their users normally do not have access to the source code [1]. On the other hand, OSS components are provided by open source communities, with full access to the source code [6].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SCM 2005, September 5-6, 2005, Lisbon, Portugal. Copyright © 2005 ACM

The granularity of an OTS component can be different. Some regard that OTS components could or should include very large software packages such as Microsoft Office. Others limit OTS components to GUI libraries. In this study, we focus on OTS components as *software components*. Such a component is a unit of composition, and must be specified so that it can be composed with other components and integrated into a system (product) in a predictable way [10]. That is, a component is an "*Executable unit of independent production, acquisition, and deployment that can be composed into a functioning system.*" We also 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. That is, platform ("commodity") softwares are not considered, e.g. an OS like Linux, DBMSes, various servers, or similar softwares. This definition implies that we include not only components following COM, CORBA, and EJB standards, but also software libraries like those in C++ or Java. This definition is consistent with the scope used in the component marketplace [9].

To record, understand, and possibly improve industrial practice wrt. OTS-based development, we have carried out several empirical studies of on the usage of COTS and OSS components. This paper will report some results from these studies wrt. versioning of such components. The remainder of this paper is organized as follows: Section two motivates and states the research questions and the research method. Section three describes the results and section four discusses these. Finally, conclusions and future research are presented in section five.

2. Research questions, research method, and data collection

2.1 Motivation and some context

There is a growing literature on OTS-based development, but alas with few representative studies on industrial practice. For instance, Torchiano and Morisio [7] interviewed 7 small IT companies in Norway and Italy on their experience with COTS-based development. Even by this tiny study, they stated six theses that refute many assumptions from the literature. For instance, they claim that OSS and COTS components are used very much in a similar way, e.g. that components are normally not modified even if source code is available.

Based on their study, we first performed a pre-study on COTS components as structured interviews of 16 COTS-based projects in 13 Norwegian IT companies [4]. From the pre-study, we gathered some new insights on COTS-based development and clarified our research questions and hypotheses. The study presented in this paper extended the pre-study in two dimensions. First, it included OSS components because they represent an alternative to COTS components. Second, this study included samples from Norway, Italy and Germany. In addition, the sample was selected randomly instead of on convenience as in the pre-study. The study was performed as a survey

with a web-questionnaire, using a randomized sample of 133 projects from small, medium, and large IT companies [2, 5]. The perspective was largely that of a system integrator.

2.2 Research questions

To investigate the state-of-the-practice of versioning problems in OTS-based development, we first designed research question RQ1 to study separate OTS components. We then designed research question RQ2 to study the whole OTS-based project, which might use several different OTS components.

2.2.1 Research question RQ1: In RQ1, we want to know whether and to what extent OTS components are actually modified locally. The source code is namely available not only for OSS components, but also for many COTS components [4]. But even if the need and opportunity is there, will such changes actually be performed? The first research question is therefore:

RQ1: *To what extent are OTS components actually modified locally?*

2.2.2 Research question RQ2: Some OTS-based projects integrate several OTS components. Updates (releases) to these components may contain unpredictable functionality and come at different intervals − on which a system integrator has little control. Previous studies indicate that the number of different OTS components used in one project has a strong relationship with maintenance effort [1, 3], even postulates that maintenance costs depend on the square of the number of components. The second research question is:

RQ2: *Is system maintenance perceived to carry a risk due to future versioning incompatibilities?*

2.3 Research design

To clarify **RQ1** and **RQ2** we will use data from the mentioned survey [5]. However, the survey questions were not designed to collect comprehensive data on versioning-related issues. We have selected the survey questions given below, with boldfacing as in the original questionnaire.

For **RQ1**, we first asked the respondent to select the most important OTS component in their project, i.e. providing the most functionality for the actual application. This is named **Comp.1** below. Then we asked questions Q5.4, Q5.5, Q5.6 and Q5.10.

- Q5.4: What was the **source code status** of the selected component **Comp.1**? The options are:
  - OSS, i.e. with source code available
  - COTS, but with source code available
  - COTS, without source code
- Q5.5: Have you **read parts of** the source code of OTS-component **Comp.1**? We used a five-point Likert scale (very little, little, some, much, very much -

plus don't know) to measure the answers to this question. The answers were mapped to ordinal values 1 to 5 later.

- Q5.6: Have you **modified parts of** the source code of OTS-component **Comp.1**? We used the same measurement scale as in Q5.5.
- Q5.10 Did you encounter some of the following **aspects (risks)** with the selected OTS component **Comp.1**? The relevant option is:
  - Q5.10.d: The recent OTS component **versions** were **not backward-compatible** with the pervious version.

  We used only yes, no, and don't know to measure the answer to this question.

For **RQ2**, we asked about possible versioning-maintenance problems of the whole project through question Q4.1 and three sub-questions:

- Q4.1 What is your opinion on the following **aspects (risks)** of your OTS-based project? The relevant sub-questions are:
  - Q4.1.k: It was difficult to **plan** system **maintenance**, e.g. because different OTS components had asynchronous release cycles.
  - Q4.1.l: It was difficult to **update the system** with the **last OTS component version**.
  - Q4.1.m: OTS components were not satisfactorily **compatible with the production environment** when the system was deployed.

For each sub-question, we used another five-point Likert scale (don't agree at all, hardly agree, agree somewhat, agree mostly, strongly agree - and don't know) to measure the answer. The answers were mapped to ordinal values 1 to 5 later.

For **RQ2**, we also investigated whether the number of different OTS components used in the project will influence integration effort as measured in Q4.1. We gathered the relevant information through question Q5.1: How many different OTS components have been used in the project?

2.4 Data collection and analysis

The unit of study for **RQ1** and **RQ2** is a completed software development project. Sampling is described elsewhere [2], and data was mostly collected via a web-tool. According to the focus of the different research questions, we used different data analysis methods:

- For **RQ1,** we first clustered OTS components into two categories, with source code and without source code, according to the answers of Q5.4. We then analyzed the distribution of answers to questions Q5.5 and Q5.6 concerning OTS components with source code. After that, we calculate the distribution of answers to question Q5.10.
- For **RQ2**, we first studied the distribution of answers to Q4.1. We then calculate the correlation between the possible risks (Q4.1.k, Q4.1.l and Q4.1.m) with the number of different OTS components used in the project (Q5.1).

3. Research results

We have gathered results from 133 projects (47 from Norway, 48 from Germany, and 38 from Italy). Three companies gave results for more than one project. In these 133 projects, 83 used only COTS components, 44 used only OSS components, and six used both COTS and OSS components. For these six projects, five of them gave detailed information of one COTS component, and one gave information of an OSS component. In total, we gathered detailed information on 88 COTS components and 45 OSS components.

3.1 Answers to research question RQ1

For **RQ1**, the answers to Q5.4 show that 29 (or 1/3) of 88 COTS components actually made available the source code to their users, i.e. software integrators.

The general distribution of answers to Q5.5 is shown in Figure 1. It shows that the median value concerning reading of COTS components is 3 (meaning some). The median value of OSS components is the same. This means that 1/3 of the COTS components (having available source code) and all the OSS components are *read* to some degree.

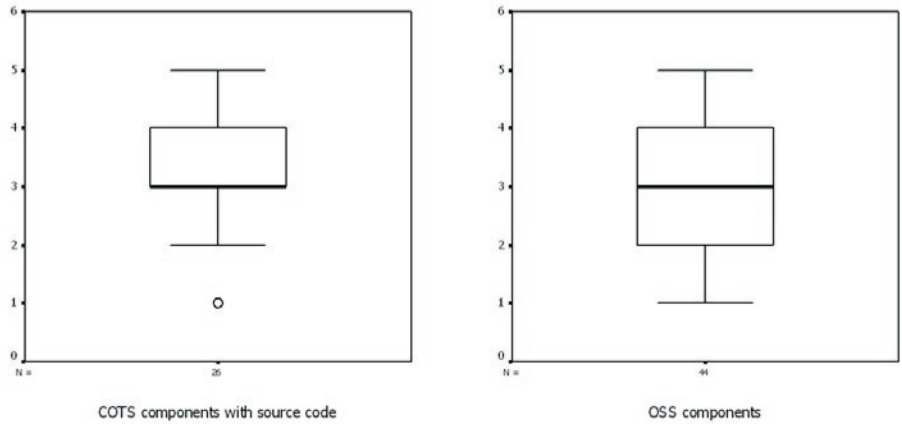

Figure 1. Answers to Q5.5: Has the source code been read?

The detailed answers to question Q5.5 is shown in Table 1. We observe that COTS components with source code were actually read slightly more frequently than their OSS counterparts.

Table 1. Detailed answers of Q5.5 (source code reading)

	Valid answers	**Read somewhat (with value more than 3)**
COTS components with source code	26 out of 29	20 out of 26 (77%)
OSS components	44 out of 45	30 out of 44 (68%)

Figure 2 and Table 2 below shows similarly the answers to Q5.6. Figure 2 shows that the COTS components with source code have been somewhat *modified,* i.e. with a median value of 2 (meaning little). OSS components – all with source code – had also been somewhat modified and with the same median value. The distribution indicates that users *less frequently modify than read* the source code of OTS components, even it such source was available. In Table 1 above, we saw that COTS components with source code were *more* frequently *read* than their OSS counterparts. Table 2 shows that OSS components were *more* frequently *modified* than their COTS counterparts. In the pre-study [4], respondents often expressed that they wanted to perform certain source code modifications of a component, but decided not to perform these for fear of costly maintenance and re-integration with future releases.

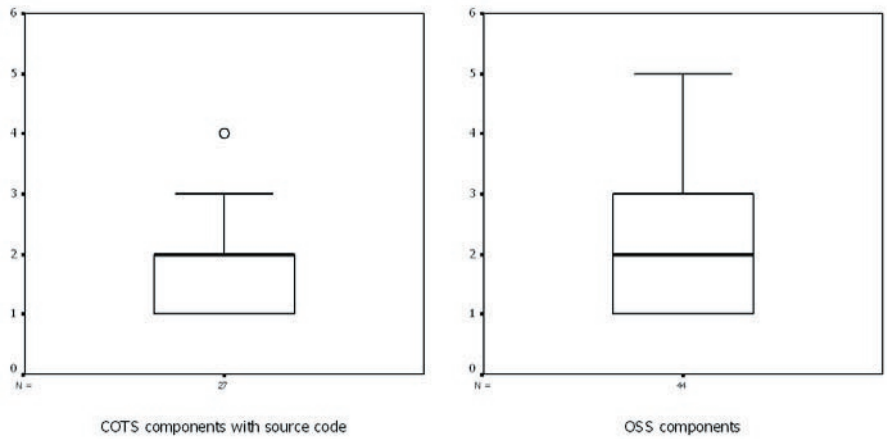

Figure 2. Answers to Q5.6: Has the source code been modified?

Table 2. Detailed answers of Q5.6 (source code modification)

	Valid answers	**Modified somewhat (with value more than 3)**
COTS components with source code	27 out of 29	4 out of 27 (15%)
OSS components	44 out of 45	16 out of 44 (36%)

The result of Q5.10 (backward compatibility) is finally shown in Table 3. The results show that only 17% (10 out of 59) of COTS components and 11% (3 out of 27) of OSS components had back compatibility problems. From this we can conclude that versioning-maintenance problems were not frequent in the selected OTS components. It also shows that there is no significant difference of backward compatibility problems between COTS and OSS components.

Table 3. Result of Q5.10 on backward compatibility problems.

	Yes	No	Don't know	All (N)
COTS component	10	59	19	88
OSS component	3	27	15	45

3.2. Answers to research question RQ2

For **RQ2**, the answers of sub-questions Q4.1.k (asynchronous release cycles), Q4.1.l (last version gives system update problems), and Q4.1.m (last version gives problems with production environment) are shown in Figure 3. We do not tell the differences between projects using COTS and OSS because they are facing the same versioning risk, i.e. that OTS component versioning is out of the OTS component users' control.

Results of Q4.1 show that the median values of Q4.1.k, Q4.1.l, and Q4.1.m are all 2 (meaning hardly agree). It means most OTS component users did not regard the versioning mismatches as a serious maintenance risk in general.

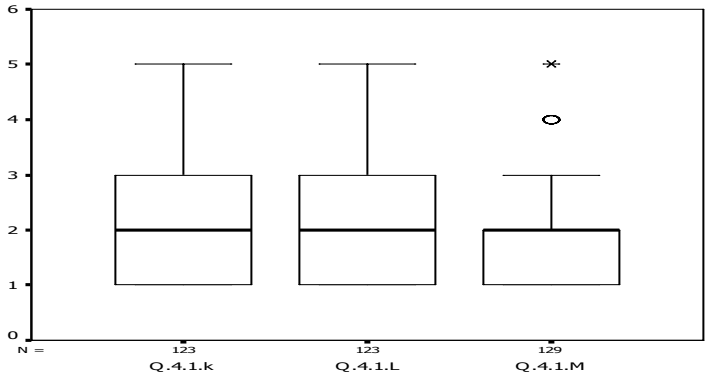

Figure 3. Answers to Q4.1 on some project versioning risks.

To investigate the correlation between the above versioning risks and the number of different OTS components in one project, we used *Spearman* rank correlations in SPSS 11.0. Although the number of different OTS components is an interval (integer) variable, we used it as an ordinal variable. That is, we gave a project using less OTS component a lower rank than a project using more OTS components. The relationship between answers to Q4.1.k, Q4.1.l, and Q4.1.m and the number of different OTS components is shown in Table 4.

Table 4. Correlation between "versioning problems" and number of different OTS components

	Correlation coefficient	Significance (2-tailed)
4.1.k with number	.182	.047*
4.1.l with number	.289	.001*
4.1.m with number	.027	.760

*Correlation is significant at the .05 level (2-tailed).

From Table 4, we can see that the number of different OTS components used in the project have a significant effect on the asynchronous release-cycle problem (Q4.1.k), on the last-version-gives-system-update problem (Q4.1.l).

4. Discussion

This study is basically a state-of-the-practice survey, where we observed some basic trends in OTS-based development in industry. These observations invoked several new research questions that we would like to investigate in the future.

4.1 New Research Question NRQ1: Why OTS source code was seldom modified?

For this study, we discovered that most OTS component users do read the source code when it is available. However, OTS component users did not change the source code very much. Some studies assume that users didn't need to see or modify, or lacked the knowledge, skills or resources to do so [7]. Another possible reason is that the users fear costly future maintenance (cost of reintegration) when a new OTS component version is released [4]. If the reason is the latter one, a new research question **NRQ1** will be: *If it is necessary to locally modify the source code in an OTS component, how to support integration of new OTS component versions (releases) with the local modifications?* An obvious remedy is to apply (semi-)automatic merge tools, often as part of common SCM tools. In the OSS community there is heavy use of OSS's own bug tracking tool, Bugzilla [8] that again uses the open CVS tool for versioning. Furthermore, there is a commitment in the OSS community to *report back* local modifications. Thus the merge/integration job may possibly be delegated to the "owner" of an OSS component. However, we have no specific information in this survey on such integration or any use of SCM tools.

4.2 New Research Question NRQ2: How to manage versioning problems when using several OTS components in the same project?

Results of Q5.10 show that the versioning problems of reusing a single OTS component are very few. However, the versioning risk will increase as the number of different OTS components increases. Our data gives further support the findings in [3], that the most significant factor that influences lifecycle cost of a COTS-based system is the number of COTS packages that must be synchronized within a release.

However, our study shows that using more than one OTS component in a project is sometimes unavoidable. 90 of the 133 projects used more than one OTS component. Therefore, another interesting research question **NRQ2** is: *How to estimate the "optimal" number of OTS components in a project to balance initial development savings with later maintenance costs?* Moreover, some of our investigated projects had very few versioning problems, even if they used more than 10 different OTS compo-

nents in their project. Summarizing their experience by case studies to give guidelines on OTS- based development could be yet another, new research question.

4.3 Possible treats to validity

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 a web tool. About 15% of the questions have been revised based on pre-test results. However, a possible threat to construct validity is that we forgot to give a clear "no" alternative in questions Q5.5 and Q5.6 (not only "very little", "little" etc.).

Internal validity We promised respondents in this study a final report and a seminar to share experience. The respondents were typically persons who wanted to share their experience and wanted to learn from others. We therefore 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 reveal the whole picture of the project. Due to length limitation of a questionnaire, we asked the respondent to fill in information for only one component in the project. The possible threat is that other OTS components in the same project might lead to different answers to our questions.

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. The sample size is generally sufficient for valid statistical conclusions.

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 [2]. 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.

5. Conclusion and future work

This paper has presented results of a state-of-the practice survey on OTS-based development in industrial projects. The results of this study have answered two questions relevant for software configuration management:

- **RQ1**: *To what extent are OTS components actually modified locally?*

Our results show that most OTS component users took advantage of the available source code and read it. However, few of them actually modified it.

- **RQ2**: *Is system maintenance perceived to carry a risk due to future versioning incompatibilities?*

Our results show that versioning problems when using a single OTS component were few. However, the key challenge is to coordinate versioning when several OTS components were used in the project.

Results of this study have shown state-of-the-practice data. By observing the current trend in industry, we discovered several interesting research questions to be studied in the future. The next step is to do a larger qualitative study with personal interviews to further study some of the new research questions.

Acknowledgements

This study was partially funded by the INCO (INcremental COmponent based development, http://www.ifi.uio.no/~isu/INCO) project. We thank the colleagues in this project, and all the participants in the survey. We also thank the local OSS enthusiast Thomas Østerlie for valuable comments.

References

1. Basili, V. R. and Boehm, B.: COTS-Based Systems Top 10 List. IEEE Computer, 34(5):91-93, May/June 2001.
2. 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 (ISESE'05), 17-18 Nov. 2005, Noosa Heads, Australia, 11 pages.
3. Donald, J. R., Basili, V., Boehm, B., and Clark, B.: Eight Lessons Learned during COTS-Based Systems Maintenance. IEEE Software, 20(5):94-96, Sep./Oct. 2003.
4. 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. Submitted to the Journal of Empirical Software Engineering, 29 pages.
5. Li, J., Conradi, R., Slyngstad, O. P. N., Bunse, C., Khan, U., Torchiano, M., and Morisio, M.: An Empirical Study on Off-the-Shelf Component Usage in Industrial Projects. Proc. 6th International Conference on Product Focused Software Process Improvement (PROFES'2005), 13-16 June, 2005, Oulu, Finland, Springer Verlag LNCS Volume 3547, pp. 54 - 68
6. Open Source Initiative (2004): http://www.opensource.org/index.php
7. Torchiano, M. and Morisio, M.: Overlooked Aspects of COTS-based Develop*ment.* IEEE Software, 21(2):88-93, March/April 2004.
8. Bugzilla tool used for OSS (2005): http://www.bugzilla.org/
9. ComponentSource (2004): http://www.componentsource.com/
10. Crnkovic, I., Hnich, B., Jonsson, T., and Kiziltan, Z.: Specification, Implementation, and Deployment of Components. Communication of the ACM, 45(10):35 – 40, October 2002.