A Survey of Software Development with Open Source Components in Chinese Software Industry

Weibing Chen[1], Jingyue Li[2], Jianqiang Ma[1], Reidar Conradi[2], Junzhong Ji[1], and Chunnian Liu[1]

[1] Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology, College of Computer Science and Technology, Beijing University of Technology (BJUT), Beijing 100022, China
{weibingchen, jianqiang.ma}@gmail.com

[2] Department of Computer and Information Science, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway
{jingyue, conradi}@idi.ntnu.no

Abstract. Chinese software companies are increasingly using Open Source Software (OSS) components in system development. Integrating such components into new software systems leads to challenges related to component selection, component integration and testing, licensing compliance, and system maintenance. Although these issues have been investigated industrially in other countries, few state-of-the-practice studies have so far been performed in China and with a representative subset of software companies. It is therefore difficult for Chinese software companies to be aware of special issues, or to plan improvement of OSS-related processes. This paper describes a questionnaire-based survey in Chinese software companies of software development with existing OSS components. Data from 47 finished development projects in 43 companies have been collected. The results show that use of web search engines was the most common method to locate OSS components. Local expertise combined with requirements compliance was the most decisive factors when choosing an identified component. To avoid legal exposure, the common strategy was to use components without licensing constraints. About 84% of the components needed bug fixing or other code changes, rarely relies on support from the OSS community. However, close participation with the OSS community was rare, although most developers meant that this was important.

1 Introduction

Building new software systems by pre-fabricated components is an attractive way to achieve lower cost, shorter time-to-market, higher quality, adherence to industrial standards etc. [11]. It has recently become more and more popular to reuse Open Source Software (OSS) components in system development [2, 5, 16, 18]. Such components offer many advantages, such as free and changeable code. Indeed, many OSS components are recognized for their high reliability, performance, and

robustness [17]. On the other hand, reusing OSS component (and "external" component in general) raises challenges in selecting the right component and to successfully integrate and test the selected component [12]. In addition, it is important to select and integrate OSS component with proper license, if the developed system is going to be distributed or sold to the general market [2, 17].

Many previous studies of OSS-based development are based on theoretical proposals (especially around component selection) [2, 6] and industrial case studies [5, 14, 16]. One major survey has been performed to investigate the state-of-the-practice of OSS-based development in three European countries [11]. Although China has become a major actor to employ OSS software in industry, especially regarding software platforms like Linux, little research has been performed on the challenges of efficient reuse of OSS components in Chinese software industry.

Our questionnaire-based survey focuses on three main issues in reusing OSS components for software development in Chinese software industry, namely component selection, licensing terms, and system maintenance. We have used membership lists from a national Chinese software organization (CSO for short)[1] to achieve an almost representative subset of software companies. We have gathered information from 47 finished projects in 43 companies. The results show that use of web search engines was the most common method to locate OSS components. Local expertise combined with requirements compliance was the most decisive factors in deciding upon an identified component. To avoid legal exposure, the common strategy was to use components without licensing constraints or to package proprietary code separately. About 84% of the components needed bug fixing or other code changes, rarely relies on support from the OSS community. In addition, close participation with the OSS community in so-called OSS projects was rare on most issues, although most developers meant that this was important.

The rest of this paper is organized as follows: Section 2 describes the background. Section 3 discusses the research approach. Section 4 presents results and discussion of research questions, Section 5 contains a general discussion, and a conclusion and ideas for future work are presented in section 6.

2 Background

2.1 Concepts used in this study

In this study, we define a **software component** as in [10]:"Software components are executable units of independent production, acquisition, and deployment that can be composed into a functioning system." An **OSS component** is defined as a software component that:

- Is provided by the OSS community
- Is subject to licensing constraints

[1] The name of this organization was omitted for confidential reasons.

- Is not a platform software (e.g., OS like Linux, DBMS, or similar software).

2.2 State-of-the-art

There have been two main kinds of empirical studies of OSS:

- *Cultural-oriented studies* concentrate on how to make new OSS software's and components, the OSS project itself and its organization as an OSS community, the participators' motivation, and the evolution of the OSS project [22, 24].
- *Technical-oriented studies* like this one, concentrates on process issues in reusing existing OSS components to develop new software [13, 17].

The aim of this study is therefore to establish some empirical-based guidelines to make OSS-based development to run more smoothly. Typically, such a development process includes several stages, such as OSS component selection, component integration, and system maintenance.

2.2.1 OSS component selection

Selecting a right component is one key factor for the success of OSS-based development. Typically, the component selection process includes locating candidate components, evaluating components based on pre-defined criteria, and deciding upon components [12, 15]. Most previous studies on component selection focus on selecting COTS (commercial-off-the-shelf) components [1, 15]. Due to the peculiar nature of OSS components, the process and criteria to select OSS components are quite different with those used to select COTS components [6]. The proposed COTS component selection process may not fit OSS selection very well [6].

2.2.2 OSS component integration and OSS licensing issues

After OSS components are selected, the next step is to integrate them into the target system. To ensure the success of integrating the OSS components, the integrators need to consider not only technical issues, such as API and programming language, but also the licensing terms of the selected OSS components. There are more than 50 different OSS licenses [9]. Some licenses have strict constraints on the distribution or resale of the derived system from OSS components. For example, the GPL (GNU Public License)-type licenses do not give the licensee unlimited redistribution rights. The right to redistribute is granted only if the distribution is licensed under the terms of the GPL and includes, or unconditionally offers to include at the moment of distribution, the source code [12, 17].

2.2.3 System maintenance

After OSS components are integrated into a software system, it is important to maintain and update those components properly for a long term use. Most technical supports from OSS communities are in the form of mailing lists and bulletin boards [12]. Since these supports are provided mainly by loosely organized volunteers, it is difficult to control the support quality. To get high quality and long-term support, one proposed strategy is to establish a long-term working relationship with the OSS

community [16]. That is, the OSS component users not only download software from the community, but also upload the modified software to the OSS community [13, 16]. Such a relationship between users and the OSS community is supposed to benefit both practitioners [2].

2.3 State-of-the-practice of OSS-based development in China

China is one of the major countries using OSS in information systems. The Chinese government has played an important role in the process of promoting the Chinese OSS movement. For example, The Japan-China-Korea (JCK) open alliance which announced in November 2003 is an initiative to promote OSS by cooperation [8]. Due to the Chinese government's encouragement on the use of Linux and OSS, more and more Chinese software companies start to use OSS components to develop software. No other country comes even close to the level of advancement that China has achieved in deploying OSS, particularly Linux [8]. The current scale of OSS-based development is large enough to be noticed at the global level. However, there are few empirical studies on OSS-based development in Chinese software industry.

3 Research approach

3.1 Research questions

This study is to investigate the state-of-the-practice of OSS-based development in Chinese software industry. We designed three research questions RQ1 to RQ3 and corresponding sub-questions.

The number of OSS components has increased dramatically these years. More than 137,000 OSS projects have been registered at sourceforge.net. Facing so many OSS components, it is difficult to select the best one to be integrated into a new system. Although researchers have proposed several structured, formal, semi-formal selection processes, and various evaluation criteria, there are few empirical studies have observed the actual selection process used by commercial developers [12]. Thus, our research question **RQ1** and corresponding sub-questions **RQ1.1** and **RQ1.2** are:

RQ1: How the OSS components were selected in practice?

- **RQ1.1**. what methods were used to locate candidate OSS components?
- **RQ1.2**. what evaluation criteria were used to evaluate OSS components?

Many studies claimed that the OSS licensing terms affect the using of OSS components. Although Ruffin [17] discussed major legal aspects of using OSS and related risks mitigation strategy, few studies have illustrated how the licensing issues are managed in practice. So the second research question **RQ2** and corresponding sub-questions **RQ2.1** to **RQ2.4** are:

RQ2: How did OSS license affect the OSS component selection and integration?

- **RQ2.1**. How well did developers understand OSS license?
- **RQ2.2**. Did developers read related OSS licensing terms?
- **RQ2.3**. Did developers encounter OSS license related troubles?
- **RQ2.4**. what strategies were used to avoid the possible OSS licensing troubles?

To get long-term technical support of the integrated OSS components, establishing a long-term relationship by engaging in the related OSS community has been proposed as a solution [7, 16]. However, this proposal lacks support from industry practices. Thus, our research question **RQ3** is:

RQ3: Did the engagement in the OSS community facilitate the maintenance of the integrated OSS components?

3.2 Research design

To answer the research questions, we have used a survey to collect data. First, a preliminary questionnaire with both open-ended and close-ended questions was designed by reading literature. Second, a pre-study was performed to validate the quality of questions in the preliminary questionnaire and to get answers of the open-ended questions. Based on the results of the pre-study, all open-ended questions in the preliminary questionnaires were redesigned into close-ended questions. In addition, the problematic questions in the preliminary questionnaire were revised. Then, the revised questionnaire was used to collect data in a main study.

3.2.1 The preliminary questionnaire

The preliminary questionnaire has 5 sections. Sections 1 and 5 contain questions to collect background information of projects and the respondents. Sections 2, 3, and 4 include questions to investigate our research questions.

3.2.2 The pre-study to verify and refine the preliminary questionnaire

The pre-study included two steps, i.e., individual interviews followed by a group discussion.

Step 1 – Individual interviews. We have interviewed 5 project managers from 5 different companies. All interviewees have solid experience with OSS-based development. Each interview was conducted by two authors of this paper. One was responsible for conducting the interview, and the other recorded answers and asked for clarification if needed. The interviews were also taped for later verification.

Step 2 – A group discussion. After the individual interviews, we revised the open-ended questions in the preliminary questionnaire to close-ended questions and made a second version of the preliminary questionnaire. We then organized a workshop with more than 30 industrial experts to verify and comment on the second version of the questionnaire. Based on comments from the workshop, we revised the questionnaire into a final version. The final questionnaire includes about 35 questions and takes about half one hour to be filled out.

3.2.3 The main study to collect data

In the main study, the data was collected by the cooperating with the CSO. In total, we got 47 questionnaires from 43 companies (4 companies filled in 2 questionnaires each). The sample selection and data collection process are as follows:

1. **Assemble the target population**. We randomly selected 2,000 companies from a database of CSO, which included about 6,000 companies.
2. **Send invitation letters by email to obtain possible participants.** We sent invitation letters by email to the 2,000 selected companies. The invitation letter introduces the survey. We specified that survey participants will be rewarded with either the final report of the survey or an annual membership of the CSO worth of 500 Chinese Yuan. We got about 200 company responses from this step and these companies were used as the original contact list.
3. **Send questionnaires by email to possible participants**. We sent questionnaires (as word files) by email to the 200 companies and asked them to select one completed software development project, which used one or more OSS components, to fill in the questionnaire. Since we cannot get the complete list of relevant projects in such a company, project selection within the company was decided by the respondents themselves. *Therefore the sample selection process was a random selection of companies, followed by a convenience sample of relevant projects within companies.*
4. **Collect filled-in questionnaires with follow up**. From the 200 companies, we first got 40 questionnaires back. To ensure the quality of the data, we excluded 10 questionnaires answered by programmers whose work experiences were less than three years. For the remaining 30 questionnaires, we contacted the respondents again by telephone to clarify possible misunderstanding and to fill in the missing data. At the same time, we contacted the remaining of 160 companies by telephone to persuade them to fill in the questionnaire. By doing this, we got 17 other questionnaires back.

4 Results and discussion of research questions

In this section, we first present background information of the interviewees, participating companies, and projects. We then show the results for each research question followed by detailed discussion.

4.1 Background information

Human respondents. Most respondents have a solid IT background. Five of them are IT managers, 13 are project mangers, 14 are software architects and 7 are senior software developers. Most of them have more than five years of software development and more than two years working experiences with OSS-based development.

Participating companies. According to number of employees, the participating companies include 7 small, 19 medium, 9 large, and 8 super large companies, as

shown in Fig. 1. Comparing with the official number of employees in Chinese software companies [23], as shown in Fig.1, it shows that most of the participating companies are medium and large companies.

Participating projects. Forty-six respondents filled in the actual-used effort of projects. Thirteen out of 46 projects used efforts less than 10 person-months, 18 used efforts between 10 and 100 person-months, and the remaining 15 projects used more than 100 person-months.

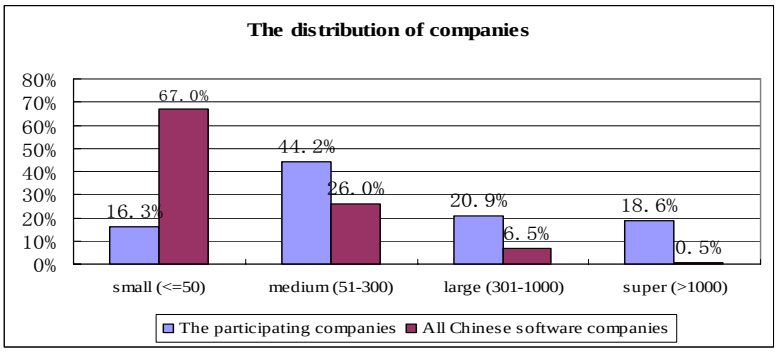

Fig. 1. The distribution of participating companies

4.2 Investigating RQ1: How OSS components were selected

Results of RQ1.1. To answer RQ1.1, we listed possible activities of locating OSS components from our pre-study and literature [15] as following:

- a) Have used it (them) before
- b) From colleagues of same company
- c) From friends of other companies
- d) Through reading related magazines (e.g., Programmer magazine)
- e) Through visiting trade shows and exhibitions
- f) Using search engines (e.g., Google, Yahoo)
- g) Visiting OSS project portals (e.g., sourceforge.net, freshmeat.com)

The respondents were asked to answer whether they have performed such activities to locate OSS components or not. The results are shown in Fig. 2 and reveal that locating OSS components was mostly based either on **search engines** (e.g., Google or the search feature in Sourceforge) or **internal experience** (e.g., having used the components before, reading magazines, getting advice from internal colleagues). **External information channel**, such as getting advices from friends in other companies, was rarely used.

Discussion of RQ1.1. Previous studies have discussed the practices of selecting OSS components. In [12], the authors concluded that most companies use a manual (brute force) method, e.g., searching with Google or Sourceforge. Our data support that conclusion. However, our results show that developers used Google more frequent than Sourceforge. The authors of [12] also proposed that familiarity was the main

attribute to be considered when selecting OSS components. Our results support this. As indicated in [13], companies were willing to listen to experience from other companies and were also willing to share their own experience with others. However, our results show that experience sharing between people in different organizations was not popular. The possible reason is that there is a lack of channels to share experience of using OSS components between different organizations.

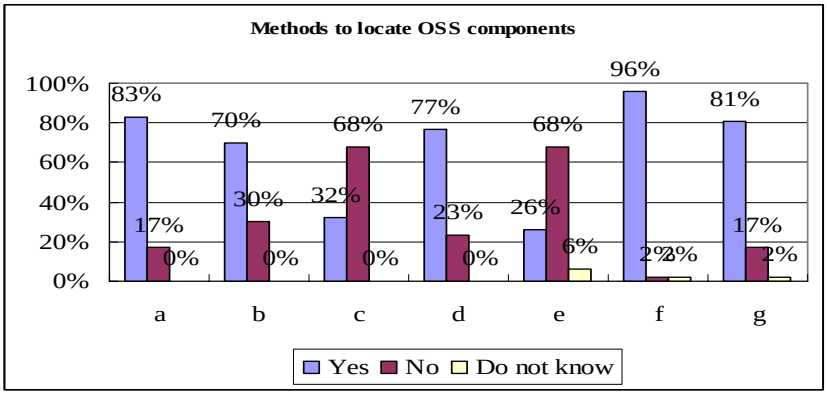

Fig. 2. Distribution of methods to locate OSS components

Results of RQ1.2. To answer this question, we formulated possible criteria to be considered when evaluating OSS components from [3, 12] as following:

- Requirements compliance
- Architecture compliance
- Quality of components (security, reliability, usability etc.)
- Functionality
- OSS licensing term
- Price
- Reputation of components or supplier
- Quality of documentation
- Expected support from the OSS community (updating, bug fixing etc.)
- Environment to be used in (platform, hardware etc.)

Respondents were asked to answer “don’t agree at all”, “very low”, “low”, “medium”, “high”, and “very high”, or “don’t know”. We assigned an ordinal number from 1 to 5 to the above alternatives (5 meaning very high). The results are shown in Fig. 3 and illustrate that **requirements compliance** (i.e., with median value 4) is the most important criteria to be considered. On the other hand, **price and support** are the least important criteria to be considered (i.e., with median value 3). The importance of other criteria, such as component quality and reputation, architecture compliance, OSS licensing terms are between.

Discussion of RQ1.2. Our results confirm that one of most important criteria to be considered when evaluating OSS component is still requirement compliance, rather than architecture compliance proposed by [12]. The authors of [10] proposed that

components with more and better comments in the community or marketplace bulletin had a good chance to be selected, because they were assumed to be better tested with generally good qualities. Our data can give that conclusion further support. Although previous studies claimed that technical support was very important to ensure the success of OSS-based systems [5, 20], our data show that the possible support from the OSS community was not considered as very important during component evaluation.

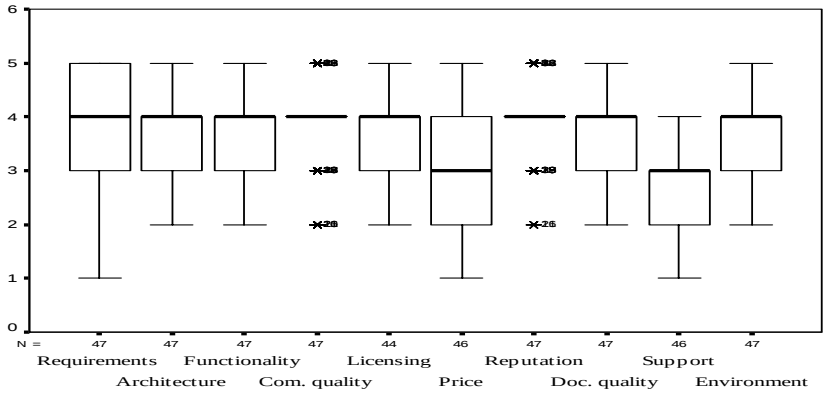

Fig. 3. Distribution of assumed importance of the evaluation criteria

4.3 Investigating RQ2: How the licensing terms were complied

Results of RQ2.1-RQ2.3. Questions related to **RQ2.1 to RQ2.3** and corresponding answers are in Table 1**. RQ2.1** and **RQ2.3** were used the same measurement as **RQ1.2**. With respect to **RQ2.2**, respondents were asked to answer "don't agree at all", "hardly agree", "agree somewhat", "mostly agree", "strongly agree", or "do not know". We assign an ordinal number from 1 to 5 (5 meaning strongly agree) to these alternatives.

Table 1. Results of **RQ2.1-RQ2.3**

RQs	Questions in the questionnaire	Results
RQ2.1	What was the extent of your understanding of OSS license?	The results show that most respondents did not understand OSS licenses very well.
RQ2.2	Have you read all licensing terms of the OSS component that you are using?	The results show that respondents have only partly read OSS licensing terms.
RQ2.3	Have you encountered OSS license related troubles?	21% of the respondents never encountered OSS license related troubles. The remaining respondents rarely encountered license related troubles.

Since the respondents' understanding and correct use of OSS licenses may be affected by their emphasis on licensing issues in the selection phase, we wonder whether the more the developers considered licensing terms in the selection phase, the better they understood the licensing terms. To investigate this question, we calculated the correlations between the respondents' emphasis of license criteria in the selection phase and answers of the above three questions with a *Spearman rank correlations* in SPSS 11.0. The results are shown in Table 2.

Table 2. Correlation between respondents' emphasis of OSS licensing term in the selection phases and the results of their understanding and using OSS license

	Correlation coefficient
Respondents' emphasis on licensing terms in the selection phase vs. their actual understandings on the licensing terms	.243
Respondents' emphasis on licensing terms in the selection phase vs. their effort used to read OSS licensing terms	.243
Respondents' emphasis on licensing terms in the selection phase vs. the occurrences of OSS license related trouble	.376*

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

Discussion of RQ2.1-RQ2.3. Results show that there are no **significant** correlations between the respondents' emphasis on licensing terms in selection phase and their knowledge and effort used to read these licensing terms. Surprisingly, the more developers emphasized the OSS licensing terms, the more frequently they encountered license related troubles. The possible explanation is that people did not understand licensing terms and did not take proper action to avoid possible troubles, even though they considered licensing terms as an important issue.

Results of RQ2.4. RQ2.4 deal with what actions have been used to avoid possible license related troubles. From the literature [2, 12, 16, 17], we have summarized possible strategies as following:

- Use other components without licensing constraints.
- Consult legal experts for help.
- Develop modules containing GPL-based components and with APIs exposing them, in order to avoid GPL restrictions.
- Package the proprietary code separately to avoid GPL restriction.
- Contact the OSS license's "owner" and agree on a certain license to avoid the licensing impacts.
- Place all the "derived programs" which relate to licensing issues, back to the OSS community.

We use the same measures as **RQ2.2**. The result shows that using other OSS components without license constraints was the most popularly used strategy. On the other hand, putting all "derived programs" back to OSS community was the least used strategy. The frequency of using the other strategies, such as packaging open source code with proprietary code separately and contact OSS license's "owner", was between.

Discussion of RQ2.4. From the OSS component users' perspective, the main concern on OSS licensing term is whether the system reusing OSS components is defined as a "derived programs" [2]. If so, according to many OSS licenses, the "derived work" should be published. The source code of project is a private property for business companies which hide the intellectual property (IP) from their competitors and make profits on IP investment [12]. When using OSS components, our results show that business companies would rather use components without strong licensing constraints to avoid making their code public.

4.4 Investigating RQ3: How the maintenance was performed

Results of RQ3. This research question investigates how to maintain OSS-based systems smoothly. We first investigated whether developers needed to fix bugs and to change the source code. If the answer was 'Yes', the follow up questions were what they did. Results show that 44.7% of respondents needed bug fixing and 39.3% of the developers needed to change code. When they did the fixing or changing, our results show (see Table 3) that more respondents prefer to do it themselves rather than to ask for help from the OSS community. However, respondents needed more effort (40 person-hours) on average to correct errors by themselves than by the OSS community (11 person-hours). On the other hand, respondents need less effort on average to change the code themselves (35.2 person-hours) than by the OSS community (60 person-hours).

Table 3. Results of fixing bugs and changing code

	By respondents themselves		**By the OSS community**	
	Percentage	Average effort (person-hours)	Percentage	Average effort (person-hours)
fixing bugs	40.4%	40	12.7%	11
changing code	21.3%	35.2	4.3%	60

To answer **RQ3,** we also investigated the relationship between project developers and the OSS community. We asked respondents whether there were developers (i.e., those in their projects) that have taken part in the OSS community. Only 4 respondents said 'Yes'. For the respondents with "No" answers, they were asked to select one from the following three reasons with the same measures as in **RQ2.2**.

- There was no need to take part in the community
- Do not have enough resources (such as time, human resources, etc.)
- It was difficult to take part due to the hierarchy of the OSS community.

Results illustrate that developers thought it was needed to take part in the OSS community. Due to resource limitation, such as time and cost, most of them did not join in the OSS community. However, joining in the OSS projects was not regarded as a difficult.

Discussion of RQ3. Although taking part in a corresponding community and

contributing to the OSS projects and getting contributions published may not be straightforward, it proved to be helpful [13]. Our results show that most developers thought that taking part in OSS community was needed. However, there was a lack of resource to do that. Fortunately, there are many other ways to work with the OSS community. Perhaps the simplest way is to provide feedback and to report bugs to OSS projects [7, 13]. In addition, new features and possible implementation of the features can be proposed to OSS projects [13, 20].

5 Final discussion

5.1 General discussion

This study summarized the practices of three key issues of OSS-based development in Chinese software industry, namely selecting OSS components, complying OSS licensing terms, and maintaining OSS components. Based on our results, we give three suggestions on facilitating the OSS-based development.

Improve the OSS search engine to facilitate experience sharing
Although several methods can be used to locate OSS components, our findings in **RQ1** show that two methods had been used most popularly, i.e., web search engines (e.g., Google) and OSS project portals (e.g., Sourceforge.net). The same findings have been reported in [12]. The advantage of using web search engines is that they are simple and fast. However, the disadvantage is that the search results are imprecise and possible huge. The advantage of using OSS project portals is that the OSS projects are centralized and classified. On the other hand, one OSS project portal can not include all OSS projects. People have to search in several portals to get all possible component candidates. The new ‘Google Code Search’ helps to solve the above shortcomings by combing portals of the open-source domain.

When selecting and evaluating the OSS components, experience of previous use of OSS components is valuable. Our results of **RQ1** show that, however, experience sharing was limited to internal colleagues. To facilitate experience sharing between different companies, it would have been better for ‘Google Code Search’ to include and structure the users’ experience and comments of using components, i.e., creating an OSS community for relevant components.

Understand and comply with OSS licensing terms properly
Another important issue of reusing OSS component is OSS licensing terms [12]. It is important for companies to carefully read, understand, and comply with the license of an OSS component. Our results of **RQ2.1** and **RQ2.2** show that most developers did not read and understand OSS licensing terms properly. Although there are many OSS licenses in use (more than 50 approved by opensource.org) and the licensing terms varies, five common licenses (i.e., GPL, LGPL, BSD, AL, and MIT) [20, 21], which are simple to comply with, cover 90% of OSS projects [20]. It is may be wise for OSS users to learn and understand these most common licenses before they start to select and integrate OSS components.

Take a more active part in the OSS community
When considering maintenance of the OSS-based system, project developers may need to fix bugs of OSS components, to add or revise the components' functionalities. Our results of **RQ3** show that developers needed more effort on debugging, than what the OSS community did. A better way might be to report bugs on bulletin boards and then letting the OSS community fix them. To change the OSS component code, our results of **RQ3** show that asking the community the changes needed more effort than doing the changes locally. The possible reason is that OSS community needs a long time to accept suggested changes.

As indicated in previous studies, one of the solutions to the maintenance of OSS-based system is to take part in OSS community [7, 16]. Some previous data show that 83% of community participants live in the Western countries and 55% of them contribute to OSS projects during working hours [24]. In contrast, our results from Chinese software industry show that only 9% of the investigated projects had dedicated developers take part in the OSS community. Thus, one of the primary tasks of the Chinese OSS movement is to mingle with the OSS community [19].

5.2 Threats to validity

Construct validity. In this study, most variables and alternatives are taken directly, or with little modification from existing literature. We did a pre-study to ensure the quality of questionnaire, and nearly 15% of the questions and alternatives in the final questionnaire were revised based on the pre-study.
Internal validity. We promised respondents in this study a final report or the annual membership of the CSO which worth of 500 Chinese Yuan. Most respondents took part in this survey as volunteers and selected the report as the reward. We therefore think that the respondents answered the questionnaire truthfully. However, our unit of study was a finished project. So a possible threat is that the respondents have failing memory on past projects.
External validity. There were more than 11,550 software companies registered in China in 2005 [23]. The CSO database contained only less than a half of them. Although we have put much effort on collecting data, we only got data from 43 companies out of our initial contact list of 2000 companies. For the remaining companies, we do not know their reasons for not participating. The respondents answered the questionnaires based on finished projects which were selected based on convenience by respondents. All the above issues may bring external threats to the conclusion of this study.

6 Conclusions and future work

More and more software companies are reusing OSS components in their software development projects, in China and elsewhere. Such companies need empirically-based guidelines for OSS-based development. The main conclusions of our survey are:

- Selection of OSS components is mainly based on existing web search engines, followed by local expertise for evaluation, e.g., requirements compliance and

assumed component quality. The new Google code search engine (http://labs.google.com) illustrates the need for improved search support.

- OSS licensing terms are not a barrier to software companies, when reusing OSS components in system development.
- System maintenance leads in 84% of the development projects to bug fixing or other code changes in the selected OSS components, and involves the OSS community on a case-to-case basis. We recommend that the experience and knowledge around relevant OSS components is handled by an internal "component uncle", and by a more active participation with the OSS community. The latter is also expressed by the developers themselves, but not followed up - perhaps for cultural and organizational reasons?
- Finally, since China has no comprehensive, national database of software companies, it is difficult to select a random sample of participants in such surveys, even if the present one is maybe as good as we can get.

In Europe 2005, over 50% of the software companies report that they are using OSS components in own software development [4]. We do not know a similar figure for China, but have a feeling that it is lower. We therefore need further studies of the extent, challenges, problems and cost/benefits of OSS-based software development in China and elsewhere. We also need to study in what ways the use of OSS affect the software projects.

Acknowledgements

This study was a joint research effort between BJUT and NTNU, partially funded by the Norwegian SEVO project with grant 159916/V30. We thank the CSO for data sampling and questionnaire collection. We also thank our colleagues and all participants in the survey.

Reference

1. Briand, L. C.: COTS Evaluation and Selection. Proc. of International Conference on Software Maintenance, Bethesda, Maryland (1998) 222-223.
2. Brown, A. W., and Booch, G.: Reusing Open-Source Software and Practices: The Impact of Open-Source on Commercial Vendors. Proc. of the 7th International Conference on Software Reuse (ICSR-7). Austin, TX, USA, April 15-19, 2002, Springer Verlag LNCS, Vol. 2319, 123-136.
3. Dagdeviren, H., Juric, R., and Kassana, T. A.: An Exploratory Study for Effective COTS and OSS Product Marketing. Proc. of the 27th International Conference on Information Technology Interfaces, Cavtat, Croatia (2005) 644-649.
4. Evans Data Corporation, "Open Source/Linux Development Survey", 2006, http://www.evansdata.com/survey_linux_topical.shtml.
5. Fitzgerald, B., and Kenny, T.: Developing an Information Systems Infrastructure with Open Source Software. IEEE Software, January-February (2004), 21(1):50-55.
6. Giacomo, P. D.: COTS and Open Source Components: Are They Really Different on the Battlefield? Proc. of the 4th International Conference on COTS-Based Software Systems.

Bilbao, Spain, February 2005, Springer Verlag, LNCS, Vol. 3412, 301-310.
7. Holck, J., Larsen, M. H., and Pedersen, M. K.: Managerial and Technical Barriers to the Adoption of Open Source Software. Proc. of the 4th International Conference on COTS-Based Software Systems. Bilbao, Spain, February, 2005, Springer Verlag, LNCS, Vol. 3412, 289-300.
8. Kshetri, N.: Structural Shifts in the Chinese Software Industry. IEEE Software, July-August (2005), 22(4):86-93.
9. Open Source Initiative, 2005, available at http://www.opensource.org/index.php.
10. Li, J., Bjørnson, F. O., Conradi, R., and Kampenes, V. B.: An Empirical Study of COTS Component Selection Processes in Norwegian IT companies. Proc. of the Int'l Workshop on Models and Processes for the Evaluation of COTS Components (MPEC'04 Arranged in co-location with ICSE'04), Edinburgh, Scotland. May 2004, 27-30.
11. Li, J., Conradi, R., Slyngstad, O. P. N., et al.: An Empirical Study on Off-the-Shelf Component Usage in Industrial Projects. Proc. of the 6th Intl. Conf. on Product Focused Software Process Improvement, Oulu, Finland, Jun. 2005, Springer Verlag, LNCS Vol. 3547, 54-68.
12. Mandanmohan, T. M., and Rahul De': Open Source Reuse in Commercial Firms. IEEE Software, November-December (2004), 21(6):62-69.
13. Merilinna, J., and Matinlassi, M.: State of the Art and Practice of Open Source Component Integration. Proc. of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications, Cavtat/Dubrovnik, Croatia (2006) 170-177.
14. Morad, S., and Kuflik, T.: Conventional and Open Source Software Reuse at Orbotech – an Industrial Experience. Proc. Of the IEEE International Conference on Software – Science, Technology & Engineering (SwSTE'05), (2005) 110–117.
15. Ncube, C., and Maiden, N.: Selecting COTS Anti-Virus Software for an International Bank: Some Lessons Learned! Proc. of the 26th International Conference on Software Engineering MPEC 2004, Edinburgh, Scotland, UK. (2004) 17-21.
16. Norris, J. S.: Mission-Critical Development with Open Source Software. IEEE Software, January-February (2004), 21(1):42-49.
17. Ruffin, M., and Ebert, C.: Using Open Source Software in Product Development: A Primer. IEEE Software, January-February (2004), 21(1):82-86.
18. Spinellis, D., and Szyperski, C.: How is Open Source Affecting Software Development? IEEE Software, January-February (2004), 21(1): 28-33.
19. Wang, G., and Zhang, X.: Chinese Linux Open Source Encounters Close. IT Time Weekly, Volume 22, (2004) 20-21.
20. Tuma, D.: Open Source Software: Opportunities and Challenges. Journal of Defence Software Engineering, January (2005) 6-10.
21. Ueda, M.: Licenses of Open Source Software and Their Economic Values. Proc. of the 2005 Symposium on Applications and the Internet Workshops (SAINT-W'05), January (2005) 381-383.
22. Xu, J., Gao, Y., Christley, S., and Madey, G.: A Topological Analysis of the Open Source Software Development Community. Proc. of the 38th Annual Hawaii International Conference on System Science. Hawii, Jan. (2005) 198a - 198a.
23. Ministry of Information of the People's Republic of China & Chinese Software Industry Association: Annual Report of China Software Industry, (2006): http://www.soft6.com/news/detail.asp?id=15759.
24. Lakhani, K. and Wolf, R. G.: Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects, In: Feller, J., B. Fitzgerald, S. Hissam, K. Lakhani (eds.), Perspectives on Free and Open Source Software, MIT Press, Cambridge. (2005) 3-22.