A State-of-the-Practice Survey on Risk Management in Development with Off-The-Shelf Software Components

Jingyue LI, Reidar CONRADI, Odd Petter N. SLYNGSTAD, Marco TORCHIANO, Maurizio MORISIO, Christian BUNSE

Abstract**-- An international survey on risk management in software development with OTS (Of-The-Shelf) components is reported upon and discussed. Most studies on project risk management in this area have been limited to theoretical proposals or case studies in specific project contexts. Without validation from representative, large-scale industrial projects, it is difficult for project managers to use the proposed risk-management guidelines properly and efficiently on new projects. The survey investigated actual risk-management activities and their correlations with occurrences of typical risks in OTS component-based development. Data from 133 software projects in Norway, Italy and Germany were collected using a stratified random sample of IT companies. Results show that OTS components normally do not contribute negatively to the quality of the software system as a whole, as commonly believed. However, issues such as underestimation of integration effort and inefficient debugging remain problematic and require further investigation. Results also illustrate several promising effective risk reduction activities, such as: putting more effort into learning relevant OTS components, performing integration testing early and incrementally, evaluating the quality of candidate OTS components thoroughly, and regularly monitoring the support capability of OTS providers. Five hypotheses have been proposed regarding these risk reduction activities and deserve further confirmative studies.**

Index Terms**--D.2.13. Software Engineering/Reusable Software, D.2.9. Software Engineering/Management, D.2.18. Software Engineering/Software Engineering Process**

I. INTRODUCTION

Long-term success in commercial software development is becoming increasingly challenging. Clients of development companies expect software systems that follow industry standards, are of high quality and reasonable priced, and have a short time to market. Hence, software companies are finding it increasingly difficult to protect their technology investments and maintain productivity levels. As a result, the software industry is turning to approaches that promise a larger than normal return on investment (e.g., concerning software development effort). Component-based software development (CBSD) [1] is believed to be one such approach, since it is based on the idea of develop/buy once and use often. By reusing

Jingyue Li is with the Norwegian University of Science and Technology (NTNU), Trondheim, NO-7491 Norway. Email: jingyue@idi.ntnu.no
Reidar Conradi is with NTNU, Trondheim, NO-7491 Norway, Email: conradi@idi.ntnu.no
Odd Petter N. Slyngstad is with NTNU, Trondheim, NO-7491 Norway, Email: oslyngst@idi.ntnu.no.
Marco Torchiano is with Politecnico di Torino, I-10129 Torino, Italy, Email: torchiano@polito.it
Maurizio Morisio is with Politecnico di Torino, I-10129 Torino, Italy, Email: morisio@polito.it
Christian Bunse is with the International University in Germany, D-76646 Bruchsal, Germany, Email: Christian.Bunse@i-u.de

Commercial-Off-The-Shelf (COTS) and Open Source Software (OSS) components (collectively called *OTS components)*, in addition to components made in-house, the expected benefits increase even more. Unfortunately, using such external components introduces a number of additional risks. One is the fact that the functionality and quality of a candidate component may be unknown. Another is the market-related instability of a component provider, who may terminate maintenance support [2] [3] [4] [5]. Thus, project managers have to identify and evaluate possible risks before deciding to acquire an external component instead of developing and reusing an in-house component. Several risks and risk-reduction activities in OTS-based development have been identified by previous case studies [2]-[12]. (A risk-reduction activity is an activity that is designed to minimize a particular risk or group of risks, i.e., to minimize the probability that a problem corresponding to the risk(s) will occur.) However, few subsequent, large-scale empirical studies have confirmed the conclusions of these studies. As a result, software project managers have only a few effective and well-proven guidelines for assessing the various risks and for deciding upon effective methods of reducing them. Our study is the first step to empirically validate effectiveness of the proposed risk-reduction activities, i.e. investigate state-of-the-practice in large scale and formulate hypotheses. We consider a scenario concerning the (re)use of OTS components such that they will be integrated into a new software system or product. Thus, our focus is on a software developer who plays a combined role; that of a *project manager/system integrator*.

The background of the survey presented in this article was inspired by a qualitative study [13] in 2002 of COTS usage in seven IT companies in Norway and Italy. The study identified six "theses" on COTS usage, which theses partly challenge commonly held beliefs. To build upon this previous study, we conducted a qualitative prestudy [14] in 2003 by performing structured interviews. We interviewed 16 industrial developers from 13 IT companies in order to summarize lessons learned from experience of COTS-based development. The principal insights of this prestudy were (i) that COTS-related activities can be integrated successfully into most development processes (incremental and waterfall), and (ii) that components are habitually selected in an ad-hoc manner, using either in-house expertise or web-based search engines. This led us to conclude that there is a need to store component-related experience in a systematic manner.

The study presented herein is a quantitative follow-up of the qualitative prestudy [14]. It was performed from May 2004 to August 2005and addressed IT companies in Norway, Italy and Germany. The survey performed postmortem investigations on the actual problems and risk-reduction activities in 133 completed OTS component-based projects. The study addressed three research questions:

- **RQ1**: Which problems occurred more frequently than others?
- **RQ2**: Which risk-reduction activities were performed most frequently?
- **RQ3**: Which risk-reduction activities were promising for avoiding particular risks?

It was, of course, a necessary precondition for addressing RQ1 that we first identified the problems that occurred. The results showed that the most frequent problems in OTS-based development are effort underestimation and costly identification of the location of defects. It is worthy of note that problems regarding the quality of OTS components do not occur frequently, which is commonly believed. The results also illustrate several promising risk-reduction activities to ensure the success of OTS component-based development, such as: study and understand the OTS components completely, evaluate their quality thoroughly, maintain a continual watch on the reputation of providers and their ability to provide support, limit the number of different components used in one project, and manage and share the company's knowledge on OTS components. Several hypotheses have been proposed based on the effectiveness of these risk-reduction activities upon certain risk items.

The remainder of the paper is organized as follows: Section 2 presents related work. Section 3 discusses the research design and related research questions. Profiles of the collected samples are presented in Section 4. Section 5 presents the empirical results of the research questions. Section 6 contains discussions of the results. Section 7 concludes, notes unresolved issues, and states intentions for further work.

II. RELATED WORK

A. *Risk management in software development*

Risks are factors that may adversely affect a project, unless project managers take appropriate countermeasures. Risk management includes two primary steps, i.e., risk assessment and risk control, each with three subsidiary steps as described by Boehm [15] and shown in Figure 1.

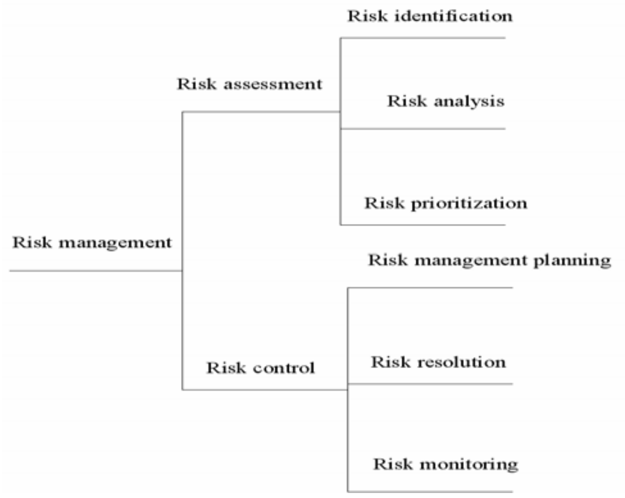

Fig.1. Risk Management Framework

Risk identification produces lists of project-specific risk items that are likely to compromise the success of a project. Several lists of possible risks in software projects have been compiled [16] [17] [18], and classified [19]-[23]. *Risk analysis* and *risk prioritization* rank the identified risk items by assessing the probability and severity of the loss for each risk item. *Risk management planning* defines how risk reduction will be conducted in a particular project by defining, among other things, risk-reduction tasks, responsibilities, activities, and budget. Risk-reduction activities must take into account viewpoints regarding process [24], organization [21], [25], and technology [26]. *Risk resolution* produces a situation in which the risk items are eliminated or otherwise resolved. *Risk monitoring* involves tracking the progress of a project towards resolving its risk items and performing corrective activities where appropriate.

Risk management is particularly important in the context of software development projects, due to the inherent uncertainties that most software projects face. Project management has to anticipate risks, understand their impact on the project, and take steps to avoid them [27, p.105]. Although previous studies have identified risk items [16]-[23], different risk taxonomies may be required by different project contexts, because building one complete and universally applicable risk taxonomy is quite unrealistic [28]. Although several risk-reduction strategies have been proposed [21] [24] [25] [26], there is a growing demand for empirical research to provide information about the efficiency of proposed risk-reduction tools and techniques [29].

B. Specific risk management in development with OTS components

When doing research in the area of component-based development, the most prevalent issue concerns what constitutes a component. Of course, there are standard definitions, such as those provided by [30] or [31]. However, these definitions might differ from the view of component marketplaces, such as ComponentSource.com and SourceForge.net. In order to define a common basis within this study, we used the following definitions:

- **Component**: A program unit of independent production, acquisition, and deployment that can form part of a functioning system. We limit ourselves to components that it has been decided explicitly to build from scratch or acquire externally as an OTS-component. That is, such components are not shipped with the operating system, not provided by the development environment, and not included in any pre-existing platform. This means that platform ("commodity") software, e.g., Linux, DBMSes, various servers, or similar software, is not considered. Furthermore, components usually follow some component model, as expressed by the COM, CORBA, EJB, .Net, or Web Service standards, or they can be a C++/Java library.

- **OTS component**: A component provided by a commercial COTS vendor or obtained from the Open Source community. OTS components may come with certain obligations, e.g., payment or licensing terms. Further, control over features and their evolution lies mostly with the provider. The source code of an OTS component is not usually modified, even if it is available.

OTS component-based development encompasses three primary types of stakeholder: component developers, application assemblers/integrators, and clients. Different stakeholders might experience different risks and challenges [11]. This study focuses on risk-management issues in the context of OTS-based development; hence, we only investigate those risks that are relevant to the assemblers or integrators of the application. Further, since the term OTS component covers both COTS and OSS components, we have to limit ourselves to risks and risk-reduction activities that address common issues in component-based development, COTS-based development, and OSS-based development. This range of issues is shown in Figure 2.

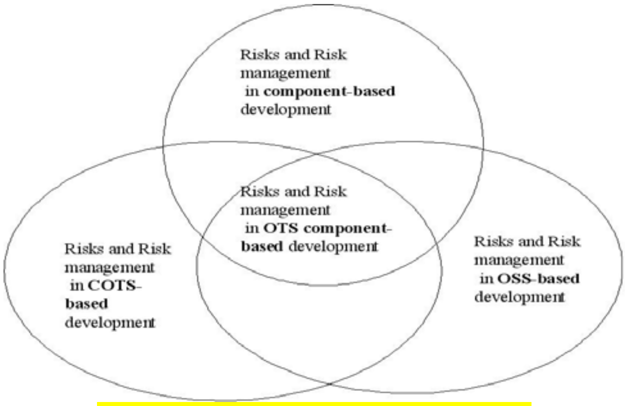
Risks and Risk management in **component-based** development
Risks and Risk management in **OTS component-based** development
Risks and Risk management in **COTS-based** development
Risks and Risk management in **OSS-based** development

Fig. 2. OTS Component-based Development Risks

Several studies have been conducted that investigate the challenges, risks, and risk-management strategies involved in developing software with OTS components (see Table 1 for a summary of their findings). An examination of their results reveals that the risk items that were identified are based on either common-sense assumptions [32] or small, non-representative case studies [8] [9] [12] [33] [34]. Without evidence from large-scale empirical studies, it is difficult to convince practitioners that the risk items identified constitute typical problems that need to be addressed prior to starting a project. Without knowing the frequency of occurrence of typical problems, it is difficult for practitioners to estimate the likelihood that identified risk items will result in corresponding problems arising in the project and to prioritize the risks. Furthermore, most of the risk-reduction strategies that have been proposed are derived from lessons learned from experience or best-practice know-how. Best practices may contribute positively to the project as a whole [14]. However, it is not clear which particular best practice might serve to reduce the probability that a problem corresponding to a particular risk item might occur during the project. Thus, further investigation is required if the preconditions and possible benefits of using best practices are to be identified. In addition, different risk-reduction strategies may have been proposed to reduce the likelihood that a risk item will occur as a problem during the project. For example, to reduce the risk of selecting an improper OTS component, components may be selected on the basis of project requirements and negotiation with clients [2] [3]. Another proposed risk-reduction strategy is to focus more on matches with the proposed architecture than requirements [13]. Without being able to consult the findings of empirical studies that compare the effectiveness of different strategies, it is difficult for practitioners to focus on the most cost-effective risk-reduction activities.

Table 1 Studies on risk and risk management in OTS component-based development

Study unit	Result/finding	Examples*
COTS in general[1]	Identified the specific challenges of using COTS software by comparing with everything built in-house [32].	Choosing the wrong COTS software may be more expensive than building it in-house.
	Summarized lessons learned from COTS-based university projects and proposed corresponding risk-management strategies [3].	Overly optimistic COTS learning curve (a more likely learning curve must be assessed during planning and scheduling).
	Summarized 10 risk factors in COTS-based development and proposed risk-reduction activities [8].	Risk factor: rapid and asynchronous changes. Risk factor: different quality practices.
	Summarized COTS lessons learned from several industrial case studies [33].	Choosing an appropriate vendor is important for success.
OSS in general[2]	Identified possible problems of development with OSS in industrial settings [9] [12] [34].	If the OSS-based system has no proper license permission, the licensor may claim damages.
CBSD [3]	Identified challenges to, and inhibitors of, component-based development [35].	Component-based development lacks engineering methods.
COTS component[4]	Identified and classified possible risks in COTS component-based development in different development phases, and proposed corresponding risk-management strategies to mitigate the risks [2].	COTS components may have asynchronous update cycles (try to involve senior analyst in analysis of the updates and derive effective field upgrade procedure and schedules).
	Summarized challenges in COTS component-based development [36].	There is insufficient information to evaluate COTS component in the market.
	Summarized problems and good practices in COTS component-based development [14].	Problem occurred: the learning effort was underestimated. Good practice: do integration testing early and incrementally.

*Examples of the proposed risk-management strategies are in brackets.

[1] COTS in general: any software bought from third-party companies.

[2] OSS in general: any software acquired from the Open Source community.

[3] CBSD: using components without differentiating between in-house built COTS, or OSS components.

[4] COTS component: software component bought from third-party companies

III. Research design

In practice, the creation of a project management plan requires project managers to first estimate the likelihood that a problem corresponding to a particular risk will arise during the project, how much effort is needed to perform a specific risk-reduction activity, and how risks might systematically be managed. In order to help project managers to predict the probability that problems that correspond to a particular risk will arise during the project and to plan risk management efficiently, it is necessary to summarize practices and results of risk-reduction activities from past projects (e.g., in the form of guidelines that may be tailored to different organizations). This being so, the motivation for the research is to perform *a posteriori* measurements and collect experience based on both actual project risks and the impact of performed risk-reduction activities. This study is designed to be exploratory; to report on the state of the practice, and to generate hypotheses for further testing.

A. Research questions

As mentioned in Section II, previous studies identified risk items on the basis of a few case studies, without knowing whether the identified risks occur frequently or happened incidentally. Our first research question **RQ1**, as shown in Figure 3, was designed to determine the relative frequency with which particular risks were present in various OTS component-based development projects. In general, the abstract OTS-based development process consists of the following phases [37]: the assessment and selection of OTS components, the tailoring and integration of OTS components, and the maintenance of OTS and non-OTS parts of the system. Each phase of OTS-based development has risks, which can be classified as relating to integrating the application (e.g., satisfying requirements), project management (e.g., quality control), and marketing (e.g., vendor relationship) [11]. To identify risks that pertain to software development with OTS components, we performed a systematic review of the literature, which resulted in our selecting a number of publications ([2] [3] [10] [11] [12] [14]). These papers listed risk items on the basis of experience or lessons learned. We believe that these items may occur frequently in practice and warrant further investigation. From the publications, we selected those risk items (see Table 2) that satisfy the following criteria:

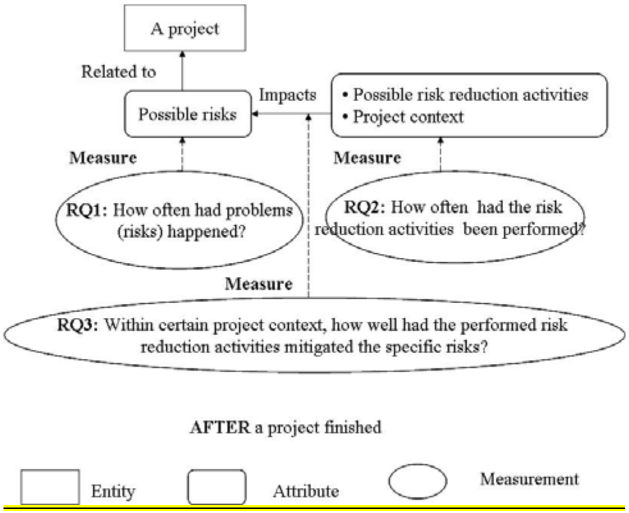
A project
Related to
Possible risks
Impacts
• Possible risk reduction activities
• Project context
Measure
Measure
RQ1: How often had problems (risks) happened?
RQ2: How often had the risk reduction activities been performed?
Measure
RQ3: Within certain project context, how well had the performed risk reduction activities mitigated the specific risks?
AFTER a project finished
Entity
Attribute
Measurement

Fig. 3. Research questions

- The risk item must be a common issue in CBSD-, COTS-, and OSS-based development. This resulted in the exclusion of a number of items. For example, risks relevant to OSS licensing [12] were excluded because they rarely occur in COTS-based development.

- The risk item should be relevant only to the assemblers or integrators of the application. This also resulted in a number of items being excluded; for example, risks related to the producers of components in [11].

TABLE 2 TYPICAL RISKS IN OTS COMPONENT-BASED DEVELOPMENT

Stage	Issue	Risk item	Description
Selection and integration	Project manag.	Selection effort	Effort to select OTS components is not estimated satisfactorily [3].
	Project manag.	Integration effort	Effort to integrate OTS components is not estimated satisfactorily [2].
	Integration	Negative reliability	OTS components affect system reliability negatively [10] [11] 12].
	Integration	Negative security	OTS components affect system security negatively [10] [11] [12].
	Integration	Negative perform.	OTS components affect system performance negatively [10] [11] [12].
	Integration	Follow req. changes	OTS components cannot be adapted sufficiently to changing requirements [3].
	Integration	Negotiate req.	It is not possible to (re)negotiate requirements with the client, if OTS components do not satisfy all requirements [14].
	Integration	Locate defect	It is difficult to identify whether defects lie inside or outside the OTS components [3].
	Integration	Incompatible deploy	OTS components are not sufficiently compatible with the production environment when the system is deployed [11].
Maintenance	Project manag.	Plan maintenance	It is difficult to plan system maintenance, because different OTS components have asynchronous release cycles [2].
	Project manag.	Update comp.	It is difficult to upgrade the system with the last OTS component version [2].
	Vendor relation	Lack provider info.	Information about the reputation and technical support ability of the provider is inadequate [2] [9].
	Vendor relation	Lack support	The provider does not provide sufficient technical support/ training [2] [9].

As discussed in Section II, the majority of published risk-reduction strategies are based on a few case studies with specific contexts. To convince practitioners to apply these strategies, it is important to demonstrate that the strategies have been used successfully with positive results. Therefore, the purpose of our second research question **RQ2** (see Figure 3) was to investigate the application of risk-reduction activities in actual industrial projects. We selected a set of risk-reduction strategies (see Table 3) using the process and criteria as for the risk items from the literature [2] [3] [8] [13] [14].

The purpose of the third research question **RQ3** was to examine the correlation between the level of risk presence and the degree to which risk-reduction activities were adopted. If we found a significant negative correlation between the performed risk-reduction activity and the actual occurrence of a certain risk, we assumed that the risk-reduction activity may have helped to reduce the likelihood that the risk item would occur as a problem for the project. For a specific risk, we were interested in the most effective reduction strategy. In addition, we hypothesized that the capability to cope with the project risks is also contingent upon several factors that pertain to the context of a project. These include business characteristics (time-to-market pressure and project budget) and technology characteristics (project complexity and quality requirements). We also expected that individual project members' experience of OTS-based development is an important predictor for managing risks successfully. In addition to investigating the direct correlations between risk occurrences and performed risk reduction activities, we were also interested in determining the impact on these direct correlations of the factors that pertain to the context of a project.

TABLE 3 TYPICAL RISK-REDUCTION ACTIVITIES IN OTS COMPONENT-BASED DEVELOPMENT

Stage	Risk management activity	Description
Selection and integration	Client attend decision	Client was actively involved in the "acquire" vs. "build" decision of OTS components [8] [14].
	Client attend selection	Client was actively involved in the selection of OTS components [2] [3] [8].
	Architecture based selection	OTS components were selected mainly on the basis of architecture and compliance with standards, instead of expected functionality [13].
	Good quality evaluation	The characteristics of OTS components with respect to quality (reliability, security etc.) were considered seriously in the selection process [2] [3] [14].
	Add learning effort	Effort required to learn OTS components was considered seriously in effort estimation [3].
	Add testing effort	Effort expended in black-box testing of OTS components was considered seriously in effort estimation [3] [14].
	First integrate unfamiliar comp.	Unfamiliar OTS components were integrated first [14].
	Incremental testing	Integration testing was done incrementally (after each OTS component was integrated) [14].
Maintenance	Follow comp. update	Local OTS experts actively followed updates to, and possible consequences of integrating, OTS components [14].
	Track substitute comp.	Maintained a continual watch on the market and looked for possible substitute components [14].
	Track provider	Maintained a continual watch on the reputation of providers and their ability to provide support [2].

B. *Survey design*

We have used a structured questionnaire to collect data. The questionnaire had five major parts. The first part was introductory. It defined our unit of study as a completed software development project that uses one or more OTS components. The participants were asked to select one such project that they were familiar with as a basis for completing the questionnaire.

The second part provided definitions of key concepts used in the questionnaire, such as the definition of an OTS component. The third part was aimed at collecting background information about the company, actual ("chosen") project, and respondents. The fourth part contained detailed questions, especially on risk management and process improvement. The last part was aimed at collecting more detailed information about one particular component used in the project. Information obtained from the last part is not reported herein. Data was collected by asking participants to respond to the following questions that were presented, in most cases, using a five-point Likert scale [38].

- The first question concerned the risk variables. Participants were asked "Did risk factor x occur?" The alternatives provided to the participants were "don't agree at all", "hardly agree", "agree somewhat", "agree mostly", "strongly agree", or "don't know", with 1 being assigned to "don't agree at all and 5 being assigned to "strongly agree". Table 2 lists the examined risk variables and defines what they measure.
- To collect data concerning the risk-reduction activity variables, we listed possible risk-reduction activities, as shown in Table 3, and asked participants whether such activities had been performed, providing the same alternatives as used for the risk variables.
- To measure the respondents' views regarding the business characteristics of the project, we asked participants about the time to market and cost/effort emphasis of the project. The alternatives provided were "very low", "low", "medium", "high", "very high", or "don't know", with 1 being assigned to "very low" and 5 being assigned to "very high".
- To measure respondents' views regarding the technology characteristics of the selected project, we used the same alternatives as for business characteristics and asked the

participants to rate the reliability, security, and performance of the final system/product. Since the number of different COTS packages used in one project may influence the complexity of the project dramatically [39], we also asked the respondents to state the number of different OTS components that were used.

- We asked participants to state what percentage of project members had previous experience with OTS-based development in general.

C. Data collection

We used random sampling to select companies and convenience sampling to select projects inside a company. The survey was performed in Norway, Italy and Germany.

- In Norway, we gathered a company list from the Norwegian Census Bureau (SSB) [40] containing large (100 or more employees), medium (20-99 employees), and small (5-19 employees) IT companies. Using the number of employees as the criterion of size, we selected the 115 largest software-intensive companies (100 IT companies and 15 IT departments in the largest three companies in five other sectors), 200 medium-sized software companies (20-99 employees), and 100 small companies (5-19 employees) as the original contact list (415 companies in all).
- In Italy, we first identified 43,580 software companies from the Yellow Pages and compiled a numbered list. We then selected companies from this list at random by using a random number generator. We visited the website of these randomly selected companies to confirm that they really were software companies. We verified that 196 of the companies were software companies, and included them in the original contact list.
- In Germany, we composed a list of companies using names from an organization

similar to the Norwegian Census Bureau and selected 476 of them in a manner similar to that in which we selected companies in Norway. We then used an existing client database to get contact information of these companies.

To avoid a bias in the process of data collection, we constructed the following guidelines for contacting potential respondents before the survey started. We first contacted them by telephone. If the potential respondents had undertaken suitable OTS-based projects and wished to participate in our study, we sent them a username and password for the web-based Simula Experiment Support Environment (SESE) [41] and an electronic version of the questionnaire in MS Word format. The respondents could use either the SESE web tool or the electronic version to complete the questionnaire. We also registered those potential respondents who did not want to complete the questionnaire. We logged the main reasons for not wanting to respond, such as “no software development”, “no OTS-based projects”, and “busy”. In total, we contacted (by phone) 1087 companies in three countries. Four hundred and twenty-five of these 1087 companies were software companies and had experience with OTS component-based development. Of the 425, 234 companies declared willingness to participate the study, and the other companies claimed that they were too busy to attend the study. We sent questionnaire to these 234 companies. However, we managed to collect data from only 127 companies. The remaining 107 companies, which claimed willingness without answering the questionnaire, mainly gave excuse as “no time”. This study is probably the first major software engineering survey to use census-type data. We found that the entire sampling and contact process can be unexpectedly expensive. Given a person-year of total effort just in Norway, the cost of a single completed questionnaire is about one person-week. The study reveals that, at best, we can achieve a stratified-random sample of IT companies, followed by a convenience sample of relevant projects. All respondents in Norway and Italy used the SESE tool to complete the questionnaire. A number of questionnaires were filled in by the researcher in Germany through telephone interviews, because of the concerns of some companies regarding confidentiality. Detailed discussions of sample selection, contact process, and response rate are reported in [42].

D. Data analysis

Although the variables in this study were measured using a five-point Likert scale, we treated them as interval variables. Spector [43] has shown that people tend to treat categories in Likert scales as equidistant, regardless of the specific categories employed. In addition, using parametric tests for scales that do not contain strictly equal intervals does not lead, except in extreme cases, to incorrect statistical conclusions [44]. Hand concluded that restrictions on statistical operations that arise from scale type are more important in contexts pertaining to model fitting and hypothesis testing than in contexts pertaining to model generation or hypothesis generation. In the latter contexts, in principle, anything is legitimate in the initial search for potentially interesting relationships [45].

For research questions **RQ1** and **RQ2**, we analyzed the level of presence of risks and level of adoption of risk-reduction activity using the *median* and *mode*. In addition, we have analyzed the inter-correlations between occurrences of risk items and risk-reduction activities respectively.

For research question **RQ3**, we viewed the level of presence of risks as a dependent (“success”) variable, and the various risk-reduction activities as independent variables. Hence, for each risk item shown in Table 2, we were interested in the relative predictive importance of the risk-reduction activities displayed in Table 3. However, the purpose of our study was to generate hypotheses and propose possible effective risk-reduction activities for reducing the probability that a problem corresponding to a risk will arise during the project.

The risk items and risk-reduction activities in Table 3 are categorized as belonging to either the selection and integration phase, or the maintenance phase. For the risk items and risk-reduction activities of each phase, we first used Pearson’s correlation analysis (excluding missing data pairwise) [46, p. 242] to identify those risk-reduction activities that have significant (with p-value <.05) negative correlations with a certain risk item. If a risk item was correlated significantly with more than one risk-reduction activity, we used stepwise multiple regressions [46, p. 532] to compare the effectiveness of those risk-reduction activities respectively.

With respect to research question **RQ3**, further analysis is needed to identify the possible effects of contextual factors on the correlations between risk items and the most effective risk-reduction activities. Since we were concerned that the correlation between the risk items and the risk-reduction activities may be partially confounded by different project contexts, we asked the question “*What is the correlation between risk items and the risk reduction activities when the project context variables are controlled?*” We used hierarchical multiple regression [46, p. 523] to examine the effects of each project context variable. For each risk item, a project context variable was entered in the first block of the hierarchical multiple regression analysis. The corresponding most efficient risk-reduction activity (i.e., the result of stepwise multiple regression) was entered in the second block of the hierarchical multiple regression analysis. The results for the second block told us how much additional variation in the risk item is accounted for by the corresponding risk-reduction activity, once the effect of the project context has been accounted for.

IV. COLLECTED DATA

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

The respondents came from 29% small, 31% medium and 40% large companies. The main business areas of the sampled companies were as follows: software house (50%), IT consulting company (32%), IT department of traditional industry (16%), and telecommunication industry (2%). The final systems produced by these 133 projects are deployed in various business domains: public sector (25%), banking or finance (16%), traditional industry (31%), ICT sector (9%), and others (19%).

In the selected 133 projects, 83 used only COTS components, 44 used only OSS components, and six used both. The mean value of the number of different OTS components used in a project was 3 (median value is 2). The mean value of the effort used before the first delivery of the project was 86 person-months, and the median value was 32 person-months.

Most respondents of the 133 projects had a solid IT background. More than 90% of them were IT managers, project managers, or software architects. Most of them had more than two years' experience with OTS-based development. All of them held at least a Bachelor's degree in computer science or telecommunication.

The OTS components used in the investigated projects include components that follow the COM, DCOM, and EJB models. There were also some components in the format of C++ or Java libraries. Examples of the selected components are listed in Table 4.

TABLE 4 EXAMPLES OF OTS COMPONENTS

Name of OTS component	OSS/ COTS	Functionality	Component model / technique
Log4j	OSS	Logging and tracing	Library
Snoopy	OSS	RSS parser	PHP Library
Crystal Report	COTS	Report design and development	J2EE, .NET, COM
MapObjects	COTS	Collection of embeddable mapping and GIS components	Library

V. Results for the research questions

A. RQ1: Occurrences of risks in OTS component-based projects

The median and mode value of the risk occurrences (excluding missing data) are shown in Table 5 and correlations between them are in Table 6. Data in table 5 illustrate that 11 out of 13 risks reported to occur infrequently. Among them, the least frequent risks are the possible negative effects of OTS components on the quality of the whole system, such as reliability, security, and performance. Surprisingly, 26% of the answers to the risks related to security are missing. This probably indicates that few respondents really care about the security effects of the OTS components on the whole system. Two risks, i.e., underestimation of integration effort and inefficient defect localization, occurred more frequently than others. Data in Table 6 illustrate correlations between occurrences of different risk and show that unsatisfied effort estimation on selection and integration are highly correlated. In addition, occurrences of quality related risks and maintenance related risks are highly inter-correlated respectively. Moreover, the difficulties of following requirements changes are highly correlated with the occurrence of unsatisfied selection effort estimation and the bad quality system. Inefficient debugging is strong correlated with unsatisfied integration effort estimation and inefficient system deployments.

TABLE. 5. RISK OCCURRENCES

Risk item	**Number of answers (valid/missing)**	**Median**	**Mode**
Selection effort	128/5	2	2
Integration effort	130/3	3	3
Negative reliability	117/16	1	1
Negative security	98/35	1	1
Negative perform.	121/12	1	1
Follow req. changes	131/2	2	2
Negotiate req.	123/10	2	1
Locate defect	129/4	3	3
Incompatible deploy	123/10	2	2
Plan maintenance	123/10	2	1
Update comp.	129/4	2	1
Lack provider info.	119/14	2	1
Lack support	112/21	2	1 and 2

TABLE. 6. CORRELATIONS BETWEEN RISK OCCURRENCES

Risk Items	Integration effort	Negative security	Negative perform.	Follow req. changes	Negotiate req.	Locate defect	Incompatible deploy	Update comp.	Lack provider info.	Lack support
Selection effort	.614 (.000) [128]			.370 (.000) [127]			.196 (.033) [119]			
Integration effort				.286 (.001) [129]		.336 (.000) [126]				
Negative reliability		.581 (.000) [97]	.559 (.000) [114]	.430 (.000) [115]	.297 (.002) [107]		.295 (.002) [108]			
Negative security			.424 (.000) [96]	.342 (.001) [97]	.226 (.033) [89]					
Negative perform.				.239 (.009) [119]						
Follow req. changes					.209 (.021) [122]	.201 (.023) [128]	.292 (.001) [122]			
Locate defect							.446 (.000) [120]			
Plan maintenance								.312 (.000) 120	.227 (.010) [112]	.227 (.020) [105]
Update comp.									.371 (.000) [116]	.180 (.061) [109]
Lack provider info.										.306 (.002) [105]

Numbers in () are p-value
Numbers in [] are numbers of cases

B. *RQ2: Performed risk-reduction activities*

The median and mode of the performed risk-reduction activities (excluding missing data) are shown in Table 7. The correlations between occurrences of risk-reduction activities are shown in Table 8. Data in Table 7 illustrate that 7 out of 11 risk-reduction activities were infrequently performed. Activities relevant to the clients, such as involving the client in the "acquire vs. build" decision, or in the selection of OTS components, were rarely performed. Some risk-reduction activities were performed more frequently, for example, developers usually evaluated the quality of the OTS components seriously during the selection phase; some developers tried to performed integration testing as early as possible, and integrated the unfamiliar components first. To maintain the system successfully, some developers also consulted internal experts to follow the updates of an OTS component. Data in Table 8 illustrate that clients will be involved in selecting OTS components if they attend the decision of using OTS components. In addition, the learning and testing effort is granted if the quality of components will be evaluated carefully. Furthermore, developers usually catch the information of the component itself and its provider meanwhile.

TABLE. 7. OCCURRENCES OF RISK-REDUCTION ACTIVITIES

Risk management activity	**Number of answers (valid/missing)**	**Median**	**Mode**
Client attend decision	131/2	1	1
Client attend selection	129/4	1	1
Architecture based selection	131/2	2	2
Good quality evaluation	130/3	3	5
Add learning effort	129/4	3	2
Add testing effort	124/9	2	2
First integrate unfamiliar comp.	115/18	3	3
Incremental testing	127/6	3	3
Follow comp. update	131/2	3	3
Track substitute comp.	130/3	2	1 and 2
Track provider	125/8	2	1

TABLE. 8. OCCURRENCES OF RISK-REDUCTION ACTIVITIES

	Client attend selection	Good quality evaluation	Add learning effort	Add testing effort	Incremental testing	Follow comp. update	Track substitute comp	Track provider
Client attend decision	.799 (.000) [127]	.179 (.043) [128]	.225 (.011) [127]	.219 (.015) [122]				
Client attend selection		.221 (.013) [126]	.182 (.042) [125]					
Good quality evaluation			.428 (.000) [128]	.259 (.004) [122]	.180 (.045) [125]			
Add learning effort				.545 .000 122	.180 (.045) [124]			
Add testing effort					.212 (.020) [119]			
Follow comp. update							.224 (.011) [129]	.368 (.000) [124]
Track provider								.545 (.000) [124]

Numbers in () are p-value
Numbers in [] are numbers of cases

C. *RQ3: Risks and risk-reduction activities*

The significant negative correlations between risk items and the risk-reduction activities by Pearson's correlation analysis are summarized in Table 9. The effectiveness of risk-reduction activities was compared with stepwise multiple regression *analysis* (significant at .05 level, including constant in equation, and excluding missing data listwise). All the regression models that we built satisfy the basic assumptions [46, p. 533] of the multiple regressions. The results of comparison are in Table 10 and show that:

- The occurrence of good quality evaluation of a component in the selection phase is negatively correlated with occurrences of most development related risks.
- Devoting a significant portion of time to learning the OTS component is negatively correlated with the occurrences of several risk items, such as unsatisfied estimation of selection and later integration effort, inefficient to follow requirements changes, and inefficient to locate defects.
- The occurrence of performing integration tests incrementally, instead of doing a complete system test after all the components have been integrated, is negatively correlated with the occurrence of inefficient requirements negotiation. However, it is positively correlated with occurrence of inefficient following up with requirement changes.
- The occurrence of monitoring support reputations of a provider is negatively correlated with the risk of lacking support from the provider.
- However, certain risks, such as negative performance effect, negative security effect, and hardness to plan maintenance of the system, have no negative correlation with occurrences of any risk-reduction activities.

The effect of the project context variable was tested using hierarchical multiple regression. The project context variables with a significant R^2 change are in Table 11, which illustrates that:

TABLE. 9. CORRELATIONS BETWEEN THE RISK ITEMS AND RISK-REDUCTION ACTIVITIES

Risk items	**Risk-reduction activities in the selection and integration phases**								**Risk-reduction activities in the maintenance phase**		
	Client attend decision	Client attend select.	Architecture based selection	Good quality evaluation	Add learning effort	Add testing effort	First integrate unfamiliar comp.	Incremental testing	Follow comp. update	Track substitute comp.	Track provider
Selection effort		-.177* 0.49 124		-.203* .023 126	-.195* .029 126						
Integration effort				-.231* .009 128	-.211* .017 127						
Negative reliability				-.273** .003 114				-.224* .018 111			
Negative security											
Negative perform.											
Follow req. changes				-.252** .004 129	-.198* .025 128		**.198*** **.034** **115**				
Negotiate req.							-.249** .009 109				
Locate defect				-.189* .033 127	-.211* .018 126						
Incompatible deploy				-.283** .002 120							
Plan maintenance											
Update comp.											
Lack provider info.											
Lack support											-.232* .016 107

** Correlation is significant at the 0.01 level (2-tailed)
*Correlation is significant at the 0.05 level (2-tailed)

TABLE 10 THE RISKS AND CORRESPONDING MOST EFFECTIVE RISK-REDUCTION ACTIVITIES

Risk items	**Risk-reduction activities**		
	Most effective risk-reduction activity	**Adjusted R^2 (df1/df2)**	**Standardized Beta**
Selection effort	Good quality evaluation	.039* (1/119)	-.217
Integration effort	Good quality evaluation	.047* (1/124)	-.234
Negative reliability	Good quality evaluation	.068** (1/107)	-.276
Negative security			
Negative perform.			
Follow req. changes	Good quality evaluation	.056** (1/127)	-.252
Follow req. changes	First integrate unfamiliar comp.	.050* (1/112)	.241
Negotiate req.	First integrate unfamiliar comp.	.053** (1/107)	-.249
Locate defect	Add learning effort	.040* (1/123)	-.218
Incompatible deploy	Good quality evaluation	.087** (1/118)	-.308
Plan maintenance			
Update comp.			
Lack provider info.			
Lack support	Track provider	.045* (1/105)	-.232

** Significance of F change is less than 0.01
* Significance of F change is less than 0.05

- The correlations between risk items and the risk-reduction activities in Table 10 are still valid when the project context variables are controlled
- The general experience of OTS component-based development of the project members is negatively correlated with unsatisfied effort estimation, and inefficient debugging, and costly system deployment. However, the number of different components used in the project is positively correlated with costly system deployment, maintenance, and debugging.
- It is not surprising that strictness of the reliability requirements of a project is positively correlated with reliability of the system. One possible explanation is that project members may put more effort into quality control when the project is reliability-critical.
- Surprisingly, the experience the people is negatively correlated with their satisfactions with the technical support from the provider. One possible explanation is that they may have higher expectations of the provider support than less experienced people.
- The R^2 of each stepwise regression was, in total, less than .015. This shows that our investigated risk-reduction activities explain less than 15% of the variance of each corresponding risk. This probably means that more efficient risk-reduction activities had not been noted in the literature, and hence were not included in our study.

VI. Discussion

A. Comparison with related studies

Both the effort expended on selecting OTS components and on integrating them are difficult to estimate [3] [47]. Basili and Boehm [48] hypothesized that COTS-based development is currently a high-risk activity, with effort and schedule overruns exceeding non-COTS-based software overruns. The results of **RQ1** show that inaccurate estimation of integration effort occurs more frequently than other problems. Kotonya and Rashid [7] point out that OTS components that offer similar functionality may have very different system resource requirements. When a problem is identified, it may not be obvious where its root cause lies. It may lie in one of the components, or in the integration code, or it could be due to a misinterpretation of the semantics behind a component interface [2]. The results of **RQ1** show that another significant problem that occurs when using OTS components is the high cost of locating defects.

TABLE 11 THE EFFECT OF THE PROJECT CONTEXT VARIABLES

Risk items	**Project context variable**			**Risk-reduction activities**			**Total R^2 change (df1/df2)**
	Name	**Standard Beta**	**Adjusted R^2 change**	**Name**	**Standard Beta**	**Added R^2 change**	
Selection effort	Experience	-.294	.080**	Good quality evaluation	-.197	.032*	.112 (1/122)
Integration effort	Experience	-.328	.103**	Good quality evaluation	-.224	.044*	.147 (1/124)
Negative reliability	Reliability	-.162	.046*	Good quality evaluation	-.217	.034*	.080 (1/110)
Follow req. changes	Reliability	-.120	.036*	Good quality evaluation	-.209	.029*	.065 (1/124)
Follow req. change	Reliability	-2.36	.032*	First integrate unfamiliar comp.	.229	.044*	.076 (1/111)
Negotiate req.				First integrate unfamiliar comp.	-.249	.053**	.053 (1/107)
Locate defect	Experience	-.258	.039*	Add learning effort	-.262	.060*	.099 (1/122)
	Number	.164	.024*	Add learning effort	-.186	.027*	.051 (1/119)
Incompatible deploy	Experience	-.182	.033*	Good quality evaluation	-.272	.066*	.099 (1/116)
	Number	.192	.034*	Good quality evaluation	-.276	.069*	.103 (1/114)
Lack support	Experience	.226	.067*	Track provider	-.164	.016*	.083 (1/103)
	Number	.242	.056*	Track provider	-.197	.030*	.086 (1/101)

** Significance of F change is less than 0.01
* Significance of F change is less than 0.05

One proposed risk-reduction strategy with respect to selecting proper OTS components is to select components based on requirements and negotiation with clients [2] [3]. However, the results for **RQ2** show that very few projects involved their clients in the selection of OTS components. Another risk-reduction strategy that has been proposed is to focus more on architecture matches than requirements [13]. However, the results for **RQ3** do not show that this strategy has no negative correlation with any of the risk items investigated in our study.

B. Hypotheses for effectiveness of risk reduction activities

1) The possible effectiveness of adding possible efforts on learning OTS components

Boehm et al. [3] found that one common problem in OTS-based development is the overly optimistic OTS learning curve, and propose that a most likely OTS learning curve must be determined and taken into account during planning and scheduling. However, the mode value of the variable “add learning effort”in Table 7 is only 2, which shows that most respondents did not consider the possible learning curve seriously. As found by Rose [2], developers usually work with several OTS components, of which they often have only a partial understanding.

The results for **RQ3** illustrate that devoting a sufficient amount of time to learning about OTS component that are candidates for inclusion has negative correlation with occurrence of unsatisfied estimation of project effort. Again, allocating more time for developers to learn and understand OTS components has negative correlation with occurrence of inefficient debugging at a later stage. Thus, the first hypothesis is:

Hypothesis 1: *Adding sufficient learing effort during the planning phase may reduce the occurrences of unsatisified effort estimation and inefficient debugging.*

2) The possible effectiveness of evaluating the quality of OTS components carefully

To develop a high-quality system, it is important to ensure, for every component in the system, that it does not have any negative effects on the system as a whole. In the COTS component market, the salesman may make false, inaccurate or exaggerated claims about the functionality and other aspects of their products [3] [8] [47]. For OSS components, there is never enough information provided for the developers to gain a full picture of the functions and the quality of the components. Thus, the developers themselves need to test and evaluate the quality of possible OTS components in the selection phase, by either reading the source code or doing black-box testing. The results for **RQ3** show that a thorough evaluation of the quality of OTS components in the selection phase has negative correlation with occurrences of several risks, such as negative effect of the component on the reliability of the system, unsatisfied estimation, inefficient deployment, and inefficient of following requirements changes. Thus, the second hypothesis is:

Hypothesis 2: *A thorough evaluation of the quality of OTS components may reduce the occurrences of bad quality and inflexible OTS-based system.*

3) The possible effectiveness of integrating critical and high-risk components first

An important issue regarding the integration of OTS components concerns changes in requirements and mismatches with the functionality of OTS components [2] [3] [8] [47]. It is difficult to change an OTS component to accord with the clients' changes in requirements; hence, it is important for the OTS component integrator to be able to (re)negotiate requirements with the client [2] [3], and probably involve clients into the "acquire vs. build" decision and the selection of OTS components [8]. However, the results for **RQ2** show that very few integrators have actually involved clients in these two processes. It has been found in [14] that it is not easy to negotiate requirements with the client in OTS component-based development. One reason is that the client typically cares only about the final product and lacks the willingness or capability to assimilate and assess the technical details. The results for **RQ3** show the occurrence of first integrating the unfamiliar, high-risk, or critical OTS components has a negagtive correlation with the occurrence of inefficient requirment negotiations. However, it has a positive correlation with occurrence of inflexible system to follow the change requirements. The possible explaination is that integrating such components first can help in the identificaton of problems in the early phase of the project, which may give the integrator leeway to negotiate with the clients to redefine the requirements [49]. However, developers may not feel familiar with the system if it involves unfamiliar OTS components. The unconfidency may hider developers to make dramatic changes of the system to follow changed customer requirements.   Thus, the third hypothsis is:

Hypothesis 3: *Integrating unfamiliar OTS components first may have two-fold effects on requirement related risks. It may facilitate the requirement negotiation and hinder changes to follow requirments changes.*

4) The possible effectiveness of monitor OTS providers

Maintaining an OTS component-based system is difficult. The post-deployment costs may exceed the development costs [48]. Different (non-overlapping) client-vendor evolution cycles may result in uncertainty about how often the OTS components in a system may have to be replaced and about the impact of such updates on the rest of the system [2] [7] [8] [47]. Hence, it is important to receive sufficient technical support from the provider. Although the developers have access to the source code for all OSS components (and for 1/3 of the COTS ones in our investigated projects), few of them have either read or modified the code to meet their requirements [50]. It is also advisable to avoid modifying OTS components, even when it is possible [8]. COTS component users (here developers) can get support from the provider by signing a support contract. However, COTS providers may discontinue support for their products for different reasons (for example, the deployed version of a COTS component may be too old to be supported), or may even go out of business [2] [48]. With respect to OSS components, some of them are currently well-supported by volunteers in OSS projects. However, current support does not ensure that the developers will receive the same level of support in the future. In contrast to the situation with COTS components, when the developers can sign a support contract, the developers have no control over the technical support that they would like from the OSS community. Our study tried to identify common risk-reduction activities for both COTS and

OSS components; hence we did not investigate contract techniques, which are less applicable for OSS-based developers. We therefore devoted our efforts to investigating those risk-reducing activities that can be controlled by all OTS-based developers. Regarding risks related to provider support, the results for **RQ3** show that maintaining a continuous watch on the support reputation of the OTS component providers, as proposed in [8] [47], has negative correlations with the occurrence of insufficient support from providers. Unfortunately, the results for **RQ2** show that few projects actually do this. The hypothesis four is:

Hypothesis 4: *Monitoring the support reputation of an OTS provider may help to select the right provider, and therefore reduce the occurrence of insufficient support.*

C. Hypotheses for possible effects of contextual factors

Due to the complexities of development with OTS component, not all issues can be documented clearly. Personal capabilities and experience are probably important factors influencing OTS-based development [48]. Donald et al. [39] conclude that the most significant factor that influences the lifecycle cost of a COTS-based system is the number of COTS packages that must be synchronized within a release. The results for **RQ3** illustrate that several contextual factors, such as experience with OTS-based development, the reliability requirements of the system, and the number of different components used, have confounded effect on the effectiveness of risk reduction activities mentioned above. Effect of these context variables need to be adjusted before testing the hypotheses H1 to H4. Thus, the fifthe hypothsis is:

Hypothesis 5: *The effectiveness of risk reduction activities may be confounded by factors, e.g. developers' experience, number of different components, and the system quality requirements.*

D. Remaining issues of managing risks in OTS component-based development

The studied risk-reduction (independent) variables in the study explain only limited variations of the actual risk (dependent) variable. That being so, we may have to look for "non-OTS-specific" reduction activities, such as code reviews or general process improvement, to avoid some risks more effectively. In addition, some advanced technologies that help us to manage the assembly of components, for example, that allow us to make statements about components if they are merged into one system, may also help to ease the integration, debugging, and deployment of OTS components.

E. Possible threats to validity

1) Internal validity

We promised to send the respondents a final report and to hold a seminar (in Norway) to share experiences. Our reasons for doing so were to get better feedback, honest answers, and to offer a kind of payment. One positive effect of our strategy is that participants may have answered more carefully because they knew that they would get feedback. One possible negative effect is that since the participants knew that they were being observed, they may have altered their behaviour. In general, we believe the participants wanted to share their experience and to learn from others. Thus, we think that the participants answered the questionnaire truthfully. However, the projects were selected by the respondents by convenience. The respondents may have selected the most successful project to answer the questionnaire, although we asked for the most familiar one. We have controlled this possible bias by asking respondents to select a project and answer questions about the background information of a project in the early sections of the questionnaire. The questions of this paper, which may involve sensitive information, are in later sections of the questionnaire. Thus, possible side-effects of sensitive questions on the selection of projects have been minimized.

Different persons in the same project might have had different opinions about the same project, especially if it had been some time since the project had been completed. Hence, asking only one person in each project might not reveal the whole picture. However, our samples were selected randomly. In many cases, we had had no previous contact with the participants. Hence, it would have been almost impossible to get two independent persons to answer questions on the same project.

Although all questionnaires were completed by using the web-based tool in Norway and Italy, for Germany, most of the questionnaires were completed by the researcher during telephone interviews. Such different methods of completing the questionnaire may have biased the results of the study.

2) *External validity*

Although we used different techniques to select samples in different countries, our study was the first survey on component-based software engineering (and, indeed, software engineering in general), to use stratified random sampling of IT companies in several countries. However, our study focused on OTS components that satisfy our definition of a component that we presented in Section II. Conclusions may be different for projects that use complex and large OTS packages, such as ERP, CRM, or content management system solutions.

We primarily present overall results from the aggregated data set over three countries. In addition, we analyzed possible diverging results using nationality as a grouping criterion. Through dummy-variable coding [51, p.303], we have transferred the categorical variable "country" into two binary variables v1 and v2 (v1=1 and v2=1 for Norway, v1=0 and v2=1 for Italy, and v1=0 and v2=0 for Germany). We have used the *hierarchical multiple regression* as shown in Section III to verify nationality effect on the risk and risk management correlations. The country variables v1 and v2 are entered into the first block of the hierarchical multiple regression analysis and the most effective risk-reduction activity is entered in the second block. The results are summarized in Table 12 and show that most of our findings (shown in Table 10), except the risk-reduction activity related to inefficient defect locating, are still valid when the country variable is controlled. A further ANOVA analysis [46, p. 324] on the variables "inefficient defect locating" and "add learning effort" illustrates that German respondents have significantly devoted more learning effort and have met significantly fewer occurrences of the costly debugging problem than what is the case for Italian and Norwegian respondents. Results in Table 12 also illustrate that the country variables v1 and v2 have covariance with several risk items. It reveals that some "cultural" differences between countries need to be considered regarding risk management. Therefore, the generalization of our conclusions may need to be verified country by country.

TABLE 12 THE EFFECT OF COUNTRY DIFFERENCES

Risk items	**Country code variable**	**Risk-reduction activities**	
	Adjusted R^2 change	**Name**	**Added adjusted R^2 change (df1/df2)**
Selection effort		Good quality evaluation	.034* (1/124)
Integration effort		Good quality evaluation	.100** (1/126)
Negative reliability	.080**	Good quality evaluation	.059** (1/111)
Follow req. changes		Good quality evaluation	.056** (1/127)
Follow req. changes		First integrate unfamiliar comp.	.031**(1/113)
Negotiate req.		First integrate unfamiliar comp.	.053** (1/107)
Locate defect	.454**	Add learning effort	.020* (1/123)
Incompatible deploy	.051**	Good quality evaluation	.043** (1/117)
Lack support		Track provider	.036* (1/105)

** Significance of F change is less than 0.01
*Significance of F change is less than 0.05

3) Construct validity

In our study, most variables and alternatives were taken directly, or with little modification, from the existing literature. However, we may have failed to consider some risks and risk-reduction activities, because they were not mentioned in the literature. Due to the exploratory nature of this study, we did not measure the risk item and risk-reduction activities with subitems. Hence, a single-item bias may bring treats to construct validity of our results. However, the questionnaire was pre-tested using a paper version by 10 internal experts and eight industrial respondents before being published on the SESE tool [41]. About 15% of the questions were revised on the basis of the pretest results. Another possible threat is that only self-reported questionnaire was used to collect data in our study. That may bring mono-method bias [52].

4) *Conclusion validity*

We observed what had happened without being able to control any variables. We assumed that the *a priori* probabilities of typical risks are similar in most OTS component-based projects and examined only the relations between the performed risk-reduction activities and the actual incidences of problems corresponding to the risks that arise during the projects. Therefore, we cannot exclude the possibilities that:

- Projects that were less likely to realize a particular risk did not take any preventive action, yet the risk did not materialize because it was unlikely.
- Projects that were more likely to realize a particular risk recognized that and employed relevant activities, yet the risk materialized anyway.

Given that different projects might have slightly different *a priori* probabilities of typical risks, our assumption might have biased our conclusions. To answer **RQ3**, we have analyzed correlations between 11 risk reduction activities and 13 risk items, which include 143 comparisons. A possible threat to conclusion validity of our study is the issues of multiple comparisons [53]. However, this study is designed as an exploratory study. Having significant correlations is not essential and smaller p-values were used to formulate the hypotheses.

VII. CONCLUSION AND FUTURE WORK

OTS component-based development brings both benefits and risks. Although several studies have investigated and proposed risk-reduction activities in OTS component-based development, few studies have validated and compared the effects of the risk-reduction activities on the corresponding risks. Our study explored the occurrences of typical risks in OTS-based projects and compared the effectiveness of performed risk-reduction activities. The data was collected from 133 OTS component-based projects in Norway, Italy and Germany.

The results show that two of the most frequent problems in OTS-based projects are the incorrect estimation of the integration effort and costly debugging. The least frequent problems are the possible negative effects of OTS components on the quality of the system as a whole. We also illustrate several promising risk reduction activities, such as adding sufficient learning effort in the plan phase, evaluating the quality of OTS component carefully in selection phase, integrating unfamiliar components first, and monitoring the support reputation of component provider, and formulate hypotheses based on the effectiveness these activities on certain risk items. In addition, several context variables, such as experience of the developer, number of components, and quality requirements of the system, have been discovered as confound factors on the relationships between risk reduction activities and the occurrence of certain risk items.

This study was an exploratory one to identify valid (or significant) patterns in the collected data material. Statistical correlation analysis can only reveal similar data patterns based on co-variations in the given variables, in spite of all the world's "significant" p-values. That is, no causal effects can be discovered, or hardly validated, based on statistics alone. Later causality tests will require theory building, hypothesis formulation, design of further studies, and further analysis of new data set.

We have assumed that the a priori probabilities of typical risks in the investigated projects are similar. Hence, it is necessary to follow several OTS component-based projects from start to finish by realistic case studies. We intend to examine several projects by estimating the risk probabilities before the project starts, follow the execution of project, and measure the occurrence of problems corresponding to the risks after the project has been completed. The purpose is to investigate the causal efficacy of the risk-reduction activities on reducing the occurrence of problems that correspond to the risks that are addressed by the risk-reduction activities. Due to the possible single-item problem of this study, both the risk items and risk-reduction activities will be measured with subitems in future studies. The reliability of the measurements will also be tested.

Results of this study reveal that the country culture may require investigation when talking about risk management. We are in the process of repeating the same study in China, which probably has large cultural differences with “western” countries, to further investigate and verify our findings.

The small variations of risk items explained by our investigated risk-reduction activities illustrate that there are several other risk-reduction activities and general context variables (e.g., application domain, development processes, and integration technology) that need to be investigated by future qualitative and quantitative studies.

REFERENCES

[1] C. Szyperski, D. Gruntz, and S. Murer, Component Software – Beyond Object-Oriented Programming. Addison-Wesley, 2002.

[2] L. C. Rose, “Risk Management of COTS Based System Development,” Component-Based Software Quality - Methods and Techniques, Springer – Verlag LNCS vol. 2693, pp. 352-373, 2003.

[3] B. W. Boehm, D. Port, Y. Yang, and J. Bhuta, “Not All CBS Are Created Equally COTS-intensive Project Types,” Proc. Second Intl. Conf. on COTS-Based Software Systems, pp. 36-50, 2003.

[4] J. Voas, “COTS Software – the Economical Choice?” IEEE Software, vol. 15, no. 2, pp. 16-19, March/April 1998.

[5] J. Voas, “The challenges of Using COTS Software in Component-Based Development,” IEEE Computer, vol. 31, no. 6, pp. 44-45, June 1998.

[6] C. Abts, B. W. Boehm, and E. B. Clark, “COCOTS: A COTS Software Integration Lifecycle Cost Model - Model Overview and Preliminary Data Collection Findings,” Technical report USC-CSE-2000-501, USC Center for Software Engineering, http://sunset.usc.edu/publications/TECHRPTS/2000/usccse2000-501/usccse2000-501.pdf, 2000.

[7] G. Kotonya and A. Rashid, “A Strategy for Managing Risk in Component-based Software Development,” Proc. 27th EUROMICRO Conf, pp. 12-21, 2001.

[8] COTS risk factor, available at http://www.faa.gov/aua/resources/cots/Guide/CRMG.htm, 2003.

[9] B. Fitzgerald, “A Critical Look at Open Source,” IEEE Computer, vol. 37, no. 7. pp. 92-94, July 2004.

[10] G. Lawton, “Open Source Security: Opportunity or Oxymoron?” IEEE Computer, vol. 35, no. 3, pp. 18-21, March 2002.

[11] P. Vitharana, “Risks and Challenges of Component-Based Software Development,” Communications of the ACM, vol. 46, no. 8, pp. 67-72, August 2003.

[12] M. Ruffin and C. Ebert, “Using Open Source Software in Product Development: A Primer,” IEEE Software, vol. 21, no. 1, pp. 82-86. January/February 2004.

[13] M. Torchiano and M. Morisio, “Overlooked Facts on COTS-based Development,” IEEE Software, vol. 21, no. 2, pp. 88-93, March/April 2004.

[14] J. Li, F. O. Bjørnson, R. Conradi, and V. B. Kampenes, “An Empirical Study of Variations in COTS-based Software Development Processes in Norwegian IT Industry,” Journal of Empirical Software Engineering, vol. 11, no. 3, Sept. 2006. pp. 433-461

[15] B. W. Boehm, “Software Risk management: Principles and Practices,” IEEE Software, vol. 8, no. 1, pp. 32-41, Jan.1991.

[16] B. W. Boehm, Software Risk Management, Tutorial, IEEE CS Press, 1989.

[17] H. Barki, S.Rivard, and J. Talbot, "Toward an Assessment of Software Development Risk," J. Management Information Technology, vol. 22, no. 2, pp. 359-371, Dec. 1993.

[18] M. Carr, S. Kondra, I. Monarch, F. Ulrich, and C. Walker, "Taxonomy-Based Risk Identification," Technical Report SEI-93-TR-006, SEI, Pittsburgh, USA, 1993.

[19] S. A. Sherer, "The Three dimensions of Software Risk: Technical, Organizational, and Environmental," Proc. 28th Hawaii International Conference on System Sciences, pp. 369-378, 1995.

[20] C. G. Chittister and Y. Y. Haimes, "System Integration via Software Risk Management," IEEE Trans Systems, Man, and Cybernetics, vol. 26, no. 5, pp. 521-532, Sep. 1996.

[21] J. Ropponen and K. Lyytinen, "Components of Software Development Risk: How to Address Them? A Project Manager Survey," IEEE Trans. Software Engineering, vol. 26, no. 2, pp. 98-112, February 2000.

[22] M. Keil, P.E. Cule, K. Lyytinen, and R.C. Schmidt, "A Framework for Identifying Software Project Risks," Communications of the ACM, vol. 4, no. 11, pp. 76-83, Nov. 1998.

[23] L. Wallace and M. Keil, "Software Project Risks and Their Effect on Outcomes," Communications of the ACM, vol. 47, no. 4, pp. 68-73, April 2004.

[24] B. W. Boehm, "A Spiral Model of Software Development and Enhancement," IEEE Computer, vol. 21, no. 5, pp. 61-72, May 1988.

[25] A. Gemmer, "Risk Management: Moving Beyond Process," IEEE Computer, vol. 30, no. 5, pp. 33-41, May, 1997.

[26] H. Hecht, Systems Reliability and Failure Prevention. Artech House Publishers, 2003.

[27] I. Sommerville, "Software Engineering", 7th edition, Addison Wesley, 2004.

[28] T. Moynihan, "How Experienced Project Managers Assess Risk," IEEE Software, vol. 14, no. 3, May/June 1997.

[29] R. L. Glass, "Frequently Forgotten Fundamental Facts about Software Engineering", IEEE Software, vol. 18, no. 3, pp. 110-112, May/June, 2001.

[30] I. Crnkovic and M. Larsson, "Building Reliable Component-Based System," Artech House, 2002.

[31] C. Szyperski, "Component Software: Beyond Object-Oriented Programming", 2nd edition, Addison Wesley, 2002.

[32] M. Vigder, M. Gentleman, and J. Dean, "COTS Software Integration: State of the Art", Technical Report NRC No. 39190, 1996.

[33] V. R. Basili, M. Lindvall, I. Rus, C. Seaman, and B. W. Boehm, "Lessons-Learned Repository for COTS-Based SW Development," Software Technology Newsletter, vol. 5, no. 3, pp. 4-7, available at: http://fc-md.umd.edu/ll/index.asp, 2002.

[34] T.R. Madanmohan and R. De', "Open Source Reuse in Commercial Firms," IEEE Software, vol. 21, no. 1, pp. 62-69, January/February, 2004.

[35] L. Bass, C. Buhman, S. Comella-Dorda, F. Long, J. Robert, R. Seacord, and K. Wallnau, "Volume I: Market Assessment of Component-based Software Engineering," SEI Technical Report number CMU/SEI-2001-TN-007, available at: http://www.sei.cmu.edu/, 2001.

[36] Component-based Software Engineering Network. CBSEnet, available at http://www.cbsenet.org, 2004.

[37] C. Abts, B. W. Boehm, and E. B. Clark, "COCOTS: A COTS Software Integration Cost Model - Model Overview and Preliminary Data Findings", Proc. of the 11th ESCOM Conf, Munich, Germany, pp. 325-333, April 2000.

[38] R. Likert: "A Technique for the Measurement of Attitudes", Archives of Psychology, no. 140, pp. 5-55. 1932.

[39] J.R. Donald, V. Basili, B. Boehm, and B. Clark, "Eight Lessons Learned during COTS-Based Systems Maintenance", IEEE Software, vol. 20, no., 5, pp.94-96, Sep. /Oct. 2003.

[40] Norwegian Census Bureau (SSB), Oslo, ICT company data, available at http://www.ssb.no/emner/10 /03/ikt/, 2002.

[41] E. Arisholm, D. I. K. Sjøberg, G. J. Carelius, and Y. Lindsjørn: "SESE – an Experiment Support Environment for Evaluating Software Engineering Technologies", Proc. of the 10th Nordic Workshop on Programming and Software Development Tools and Techniques, Copenhagen, Denmark, pp. 81-98, Aug. 2002.

[42] R. Conradi, J. Li, O. P. N. Slyngstad, C. Bunse, M. Torchiano, and M. Morisio, "Reflections on Conducting an International CBSE Survey in ICT Industry," Proc. Forth IEEE Intl. Symposium on Empirical Software Engineering , pp. 214-223, 2005.

[43] P. Spector: "Ratings of Equal and Unequal Response Choice Intervals", Journal of Social Psychology, vol. 112, pp. 115-119, 1980.

[44] P. F. Velleman and L. Wilkinson: "Nominal, Ordinal, Interval, and Ratio Typologies Are Misleading", Journal of the American Statistician, vol. 47, no. 1, pp. 65-72, February 1993.

[45] D. J. Hand: "Statistics and Theory of Measurement", Journal of the Royal Statistical Society: Series A (Statistics in Society), vol. 159, no. 3, pp. 445-492, 1996.

[46] B. H. Cohen, "Explaining Psychological Statistics", 2nd edition, Wiley, 2000.

[47] Common Risks and Risk Reduction Actions for a COTS-based System, www.mitre.org/work/sepo/toolkits/risk/taxonomies/files/CommonRisksCOTS.doc, 2005.

[48] V. R. Basili, and B. W. Boehm, "COTS-Based Systems Top 10 List," IEEE Computer, vol. 34, no. 5, pp. 91-93, May 2001.

[49] S. Lauesen, "COTS Tenders and Integration Requirements," J. Requirements Engineering, vol. 11, no. 2, pp. 111-122, 2006.

[50] J. Li, R. Conradi, O. P. N. Slyngstad, C. Bunse, U. Khan, M. Torchiano, and M. Morisio, "Validation of New Theses on Off-The-Shelf Component Based Development," Proc. 11th IEEE Intl. Metrics Symp., pp. 26 (abstract), Sept. 2005.

[51] J. Cohen, P. Cohen, S. G. West, and L. S. Aiken, "Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences", Lawrence Erlbaum Associates, 3rd edition, 2002.

[52] P. M. Podsakoof and D. W. Organ, "Self-Reports in Organizational Research: Problems and Prosects", J. Management, vol. 12, no. 4, pp. 531-544, 1986.

[53] S. J. Pocock, M. D. Hughes, and R. J. Lee, "Statistical Problems in Reporting of Clinical Trials: A Survey of Three Medical Journals", the New England Journal of Medicine, vol. 317, no. 7, pp. 426-432, 1987.