Proceedings of the Pakistan Academy of Sciences:
A. Physical and Computational Sciences 53 (3): 239–253 (2016)
Copyright © Pakistan Academy of Sciences
ISSN: 2518-4245 (print), 2518-4253 (online)

Pakistan Academy of Sciences

Research Article

An Exploratory Study of Success Factors in Software Integration for Global Software Development Vendors

Muhammad Ilyas*, and Siffat Ullah Khan

Software Engineering Research Group (SERG_UOM),
Department of Computer Science & IT, University of Malakand, KPK, Pakistan
milyasmkd@gmail.com*, siffatullah@uom.edu.pk

Abstract: The trend of software development has changed from local to global software development (GSD) with acceleration in information and communication technologies. GSD offers certain benefits like reduced cost, high quality, availing skilled human resource and latest technology etc. It also faces a lot of challenges like communication coordination issues, cultural, language and temporal challenges along with the technical challenge of software integration. The objective of the current study is to identify success factors for software integration to assist GSD vendors in integrating the software components into a final working product. We have used the systematic literature review (SLR) process by following standard SLR guidelines. We have identified a list of 14 success factors by extracting data from 89 selected papers. Out of these, 9 factors were ranked as critical success factors (CSFs). Some of the top ranked CSFs are "Consistency in Requirements and Architecture Design", "Intra and inter team Communication and Coordination" and "Component/Unit Testing prior to integration". We have also analyzed these CSFs on the basis of period published, project size and methodology used. We identified 9 CSFs which may assist GSD vendors in almost all size of projects at different phases of the software integration process.

Keywords: software integration, systematic literature review, global software development, vendors, success factors

1. INTRODUCTION

The trend of software development has been changed from local to global software development (GSD) with the rapid acceleration in information and communication technologies (ICTs) from the last decade. The ICTs have connected the software development organizations and the development teams to perform software development activities across national boundaries. The reason for the changing trend is to develop high quality software with low cost in minimum time by working 24-hours around the clock. GSD has not only solved the problem of finding skilled human resource but also provide the vendors to avail latest infrastructure and show their presence locally at different locations of the globe [1, 2]. Like any other field GSD will also take its time to grown-up and overcome the challenges like temporal, cultural and language differences, communication and coordination problems and deprived contract, knowledge and relationship management, etc. [3-6]. Alongside the above non-technical issues the GSD teams also face technical issues like problems in the architecture of the software components, their version management, configuration management and integration of the software components developed in isolation by GSD teams with inadequate communication [7, 8]. The integration phase uncover many of the problems that remain hidden in the previous phases of software integration [9].

Almost all types and size of software are composed from more than one software components/ modules. These components may be developed in-house or may be outsourced offshore or onshore.

Received, October 2015; Accepted, July 2016
Corresponding author: Muhammad Ilyas; Email: milyasmkd@gmail.com, siffatullah@uom.edu.pk

Similarly the components may be purchased from the market as a "commercial off the shelf (COTS)" component or from the large pole of open source community as a "off the shelf (OTS)" component. There is a need to identify the positive factors that may ease the integration process and the barriers that hurdle it in the above different scenarios [7, 8].

Integration is one of the most critical phases but its importance is not fully understood all the time. Enough resources and proper integration plan is crucial for this phase because this phase will assemble all the parts developed independently by GSD teams[10]. Regardless the importance of the software integration phase, enough empirical research has not been done for the identification of software integration success factors and their practices/solutions [9, 11]. We have planned to assist GSD vendors in the software integration process by developing a software integration model (SIM) [12].

For bridging the gap and easing the integration process for GSD vendors we have designed the following research questions.

RQ1: What are the success factors, as identified in the literature, to be adopted by GSD vendors software at various stages of the product integration, i.e. before, during and after the integration process in GSD environment?

RQ1-a Do the identified factors vary from decade to decade?

RQ1-b Do the identified factors vary from project to project?

RQ1-c What methodology/study strategy has been used in the selected papers as identified by the SLR?

This research paper is the extended version of our earlier published paper in IEEE SNPD 2015 Conference at Japan [13]. Because of the page limitations of the conference we were unable to report all findings/analysis of our SLR study. In the current paper we have reported findings of RQ1 supplemented by RQ1-a, RQ1-b and RQ1-c.

The remaining paper is structured as follows. In Section 2 we have presented background/related work. In Section 3, the methodology is presented. In Section 4, results of the SLR related to the above mentioned research questions are reported. In section 5 we present summary and discussion about the current study and in Section 6 we present the limitations of the study. The last section, i.e., section 7 present conclusions and future work.

2. BACKGROUND

The integration process require proper attention of the developers because many of the software projects are delayed during testing due to the complexities and incompatibilities found between software components in the integration phase [9]. The vendors need to properly plan the integration strategy while boarding into the software product development before assembling the sub components developed [13].

McConnel [14] describes integration as an activity of software development in which the isolated software components are combined in a single unit. Herbsleb et al. [15], in a case study reported that integration phase is one of the most complicated phases of software projects in GSD environment. They found that the loss of communication and coordination hurdle the integration phase in multisite development. Similarly the components required for assembling the final product may be unavailable according to the schedule expected and the components available may have fault in their interfaces.

In an exploratory study, Van Moll et al. [16] reported that the software integration phases is one of the complex and challenging phase in GSD environment for more than 50% of the software projects. They recommended considering the integration and software testing phase as a separate process. Guimaraes and Silva [17] have proposed a solution for continuous integration that can automatically integrate the committed and uncommitted code in the background. Thus it can automatically detect any conflict that may occur during programming in multiples teams of developers. Stayhl and Bosch [18] also performed a systematic review of literature about the practices of continuous integration. They reported that no uniform practices exist for continuous integration in all development environments. They further reported that variations exist from environment to environment and different environment may have different set of practices for continuous software integration. Tekumalla [7] from a systematic review reported that some topics of components based software engineering (CBSE) like implementation, selection and components quality are populated while other topics like components integration, testing and their storage need to be empirically researched through experiments and case studies in industry.

Similarly S. Schneider et al [8] from an SLR reported that certain areas of software engineering like requirements engineering and project management are very populated while project monitoring and control and software product integration are narrowly populated indicating the need for research in the area.

It is evident from the literature that there is no study which has considered integration problems deeply on the basis of software project size and types of software products. There is no categorization of factors in literature that the vendors need to consider at all three stages of software integration process, i.e., before integration, during integration and after integration. In the current paper, through the SLR process, we have reported a list of positive factors for each stage of the software integration process. We also examined that how these success factors differs in their significance on the basis of software project size. We have further analyzed how these success factors change from decade to decade (1994-2003, 2004-2013) due to maturity in the integration phase of GSD products or due to technological changes. We have also analyzed the methodologies used in the studies identified through SLR.

3. STUDY DESIGN

We have conducted a systematic literature review (SLR) to obtain the results of the current study by following the SLR standard guidelines [19]. The SLR process is used by many researchers as it is moderately rigorous and fabricate more reliable outcome in contrast to other methods of literature review [7, 8]. The SLR process utilizes a well defined methodology and is repeatable up to some extent. The SLR, which is a secondary study, has three phases i.e planning, conducting and reporting the review [19]. In the planning phase the protocol for the study is developed. The protocol for the current study has already been published [20], implemented and now we are reporting the results and analysis of the SLR process in this paper.

3.1 Search

The data in Table 1 list the libraries that we searched, the search string used for searching and the number of primary studies selected during the process. It is worth to be noted that some of the search engines and digital libraries, e.g., Google scholar have a limit of 256 characters on the length of search string. For this reason we divided our search string for Google scholar into four small sub strings. The selection of the primary studies was performed in two steps. Firstly the primary reviewer made the initial selection of papers by reading the title and abstract of the paper. Thus in the first step we selected 336 papers. After passing through the first step the final selection of the papers was made by reading the full text of the paper by primary reviewer. During the final selection we also followed the inclusion/exclusion and quality assessment criteria as already defined in our protocol for this study [20]. In some situations, regarding uncertainty about the inclusion/exclusion decision, the secondary reviewer was consulted for review. To perform inter-rater reliability test the secondary reviewer randomly extracted data from few papers and compared it with the data extracted by primary reviewer but there were no major discrepancies found between the data extracted. As mentioned in our protocol [20], the primary reviewer extracted 17 pieces of data from each selected paper. The secondary reviewer also extracted those pieces of data from few papers. We applied the Cohen's Kappa test for inter-rater reliability and found that its coefficient value is 0.89 which shows a perfect agreement between the primary and secondary reviewers.

We performed a systematic search in the

Table 1. Search results.

Name of database	Search string used in the corresponding library		No. of publications found	Initial selection	Final selection
Science Direct	[("Software integration" OR "Product integration" OR "Component integration" OR "system integration") AND ("Global software development" OR GSD OR "Global software engineering" OR GSE OR "Distributed software development" OR DSD OR "Distributed software engineering" OR DSE) AND (Challenge OR risk OR problem OR issue OR barrier OR trouble OR "critical factor" OR "key factor" OR "success factor")]		185	13	07
ACM			195	23	11
Springer			205	15	05
IEEE			417	34	16
Wiley Online Library			81	08	02
Google Scholar	**String01**	("Software integration" OR "Product integration" OR "Component integration" OR "System Integration") AND ("Global software development" OR "Global software engineering") AND (Challenge OR risk OR problem OR issue OR barrier OR trouble)	1031	163	88
	String02	("Software integration" OR "Product integration" OR "Component integration" OR "System Integration") AND ("Distributed software development" OR "Distributed software engineering") AND (Challenge OR risk OR problem OR issue OR barrier OR trouble)			
	String03	("Software integration" OR "Product integration" OR "Component integration" OR "System Integration") AND ("Global software development" OR GSD OR "Global software engineering" OR GSE) AND ("critical factor" OR "key factor" OR "success factor")			
	String04	("Software integration" OR "Product integration" OR "Component integration" OR "System Integration") AND ("Distributed software development" OR DSD OR "Distributed software engineering" OR DSE) AND ("critical factor" OR "key factor" OR "success factor")			
Snow Balling	Following the references of the selected papers and searching for the publications of the author of the selected papers		**80**	**80**	**43**
Total			**2194**	**336**	**172**
Duplicate papers					**67**
Finally selected papers					**105**

libraries listed in Table 1 and also performed the snowballing technique [21] by:

- following the references in the papers that were finally selected; and
- and searching by authors name found in the references of the papers that were finally selected.

The snowballing procedure increased the sample size of the finally selected papers from 63 to 105 papers after eliminating the duplicate papers as presented in Table 1. The search process is depicted in Fig. 1. The data for RQ1 and RQ1-a to RQ1-c

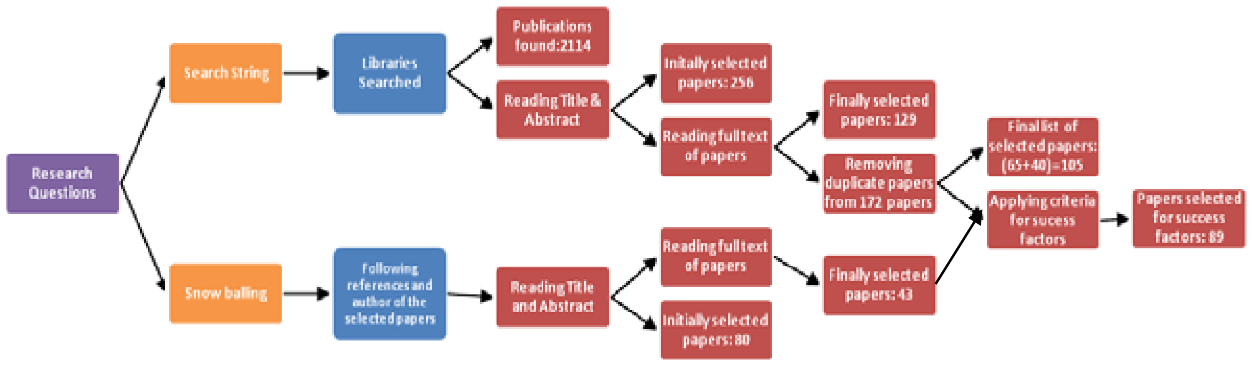
Research Questions
Search String
Snow balling
Libraries Searched
Following references and author of the selected papers
Publications found:2114
Reading Title & Abstract
Reading Title and Abstract
Initally selected papers: 256
Reading fulltext of papers
Reading fulltext of papers
Initially selected papers: 80
Finally selected papers: 129
Removing duplicate papers from 172 papers
Finally selected papers: 43
Final list of selected papers: (65+40)=105
Applying criteria for success factors
Papers selected for success factors: 89

Fig. 1 Search process.

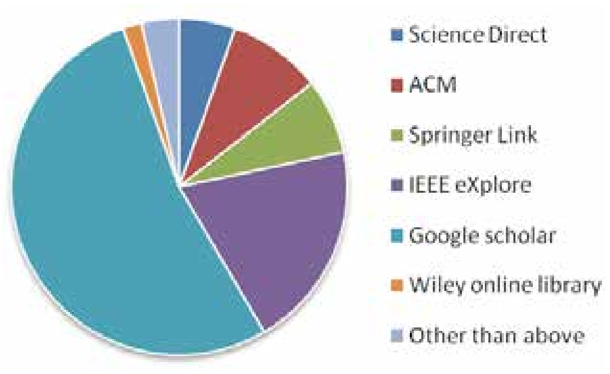

Fig. 2. Publications found in each library.

was extracted from 89 papers among the finally selected 105 papers. The remaining 16 papers were dropped due to quality assessment criteria defined in the protocol designed for this study [20]. The final list of papers found in each library through both searching and snow balling method is shown in Table 2 and graphically depicted in Fig. 2.

3.2 Data Synthesis

We initially extracted the data from 89 finally selected papers and grouped these identified factors into 27 categories/factors. The secondary reviewer, after a thorough review, merged them into 14 categories/factors. The grouping of the factors was carried out in such a way that the similar factors were grouped under the same heading. The final list of 14 success factors is shown in Table 3 and the list of papers from which the data was extracted can be found in appendix-A. More detail can be found in our previously published papers [13, 20]

4. RESULTS

This section discusses the results and examines the identified success factors for each of the Research Questions as stated in Section 1.

Table 2. Final number of publications found in each library (searching + snowballing).

S. No	Library Name	Total public: found	% (n=105)	Public: found for Success Factors	% (n=89)
1	Science Direct	9	8.57%	5	5.60%
2	ACM	15	14.28%	4	4.50%
3	Springer Link	12	11.42%	6	6.70%
4	IEEE eXplore	33	31.42%	27	30.30%
5	Google scholar	88	83.80%	37	41.60%
6	Wiley online library	3	2.85%	2	2.20%
7	Other than above	6	5.71%	7	7.90%
Total Papers found		**156**	**100%**	**89**	**100%**
Duplicate Papers		**51**	**32.69%**	**-**	**-**
Net Total		**105**	**67.31%**	**-**	**-**

Table 3. List of success factors for software integration in GSD.

S. No	Success Factors	Freq (n=89)	%	Reference of papers in appendix A, in which the factor has been found
1	Consistency in Requirements and Architecture Design	35	39	1, 2, 5, 6, 7, 8, 10, 17, 19, 20, 22, 23, 26, 32, 35, 36, 39, 40, 41, 46, 48, 50, 51, 52, 53, 55, 60, 70, 72, 74, 75, 76, 85, 87, 88
2	Intra and inter team Communication and Coordination	34	38	1, 2, 3, 4, 6, 7, 8, 9, 11, 13, 14, 17, 20, 21, 26, 27, 30, 36, 37, 38, 39, 43, 47, 51, 53, 59, 65, 66, 67, 70, 76, 81, 82, 89
3	Component/Unit Testing prior to integration	34	38	1, 2, 3, 4, 8, 11, 14, 15, 17, 19, 22, 23, 24, 26, 27, 28, 29, 34, 35, 38, 39, 43, 44, 58, 62, 64, 66, 68, 69, 73, 76, 77, 79, 82
4	Advance & Uniform Development Environment and Training	33	37	1, 3, 6, 7, 12, 14, 16, 19, 22, 23, 24, 25, 26, 31, 32, 34, 35, 38, 39, 40, 55, 56, 58, 67, 68, 71, 74, 76, 81, 85, 86, 88
5	Efficient Incremental/Continuous integration	31	35	1, 2, 3, 6, 8, 14, 17, 18, 20, 24, 26, 27, 29, 31, 34, 35, 40, 42, 43, 44, 46, 48, 50, 63, 65, 66, 68, 80, 83, 88
6	Efficient specification for Interface Compatibility	31	35	2, 8, 10, 14, 15, 19, 20, 25, 26, 30, 32, 33, 40, 41, 44, 45, 51, 54, 55, 56, 57, 58, 64, 70, 73, 74, 76, 77, 84, 85, 86
7	Proper Documentation & Configuration Management	28	32	2, 6, 7, 8, 13, 14, 15, 17, 19, 22, 23, 26, 27, 31, 33, 38, 39, 47, 48, 55, 67, 70, 73, 76, 81, 86, 88, 89
8	Early Integration Planning and Centralized P3 management	27	30	1, 5, 6, 7, 8, 11, 13, 14, 17, 19, 20, 22, 23, 27, 30, 32, 34, 35, 38, 43, 47, 48, 67, 68, 76, 81, 85, 86
9	Careful evaluation of the COTS/OTS Components	27	30	3, 4, 8, 10, 22, 26, 30, 32, 38, 40, 45, 50, 53, 55, 60, 61, 67, 68, 69, 73, 74, 75, 77, 78, 83, 86, 87
10	Use of Standard Model for Process, Data and Product's Components	7	8	17, 26, 49, 56, 73, 76, 85
11	Use of modular approach	4	5	17, 20, 52, 76
12	Use of Efficient Metrics	2	2	13, 70,
13	Use of Quality assurance	1	1	17
14	Specific Integration Timing	1	1	47

4.1 Software Integration Success Factors

We have identified a list of 14 success factors, using SLR process to answer RQ1, as presented in Table 3. As discussed in Section 3, we have identified a total of 14 success factors in which nine success factors are significant or critical success factors (CSFs). The criterion to decide a factor is critical or not was based on the percentage of frequency with which the factor was appeared and discussed in literature. We used the threshold of 30% for deciding a factor to be a critical one. Thus if the frequency percentage of a factor was more than or equal to 30%, we marked that factor to be a critical one as did by other researchers [3, 6]. The detail about these nine CSFs can be found in our published paper [13].

4.1.1 Comparison of CSFs between Co-located and GSD Projects

The CSFs listed in Table 3 shows that most of the CSFs are also applicable to co-located projects. The CSFs "Component/Unit Testing prior to integration", "Advance & Uniform Development Environment and Training", "Efficient Incremental/ Continuous integration", "Efficient specification for Interface Compatibility", "Proper Documentation & Configuration Management" and "Careful evaluation of the COTS/OTS Components" are the factors that can be considered vital in both co-located and GSD projects for integration process. While the CSFs "Consistency in Requirements and Architecture Design", "Intra and inter team Communication and Coordination" and "Early Integration Planning and Centralized P3 management" are the factors that are important in the GSD environment. It is obvious that all the GSD teams must have a consistent definition of requirements at each site with a consistent architecture design. The GSD teams need a lot of communication and coordination to maintain the above consistency and synchronize their work. Similarly the management of GSD projects needs to properly plan the integration process in advance and centrally control the project, products and processes.

4.2 Decade-wise Comparison of the Success Factors

To answer RQ1-a, we divided the search period in two decades i.e. from 1994 to 2003 and from 2004 to 2013. It should be made explicit that we have put no date boundaries on our search process but we did not find any relevant paper before 1994. The number of publications found in each decade is presented in Table-4.

Table 4. Decade wise break up of publications.

Decade	Frequency	Percentage
1994-2003	24	27%
2004-2013	65	73%

The data in Table 4 show that integration process has got much more attention of the researchers in the second decade as compared to the first decade. One reason for this may be that GSD and CBSE started recently in the last decade with the advances in ICT technologies. The integration phase has become more critical with GSD development as the finally developed isolated components needs to be integrated into a final product.

To analyze the data of both decades we have performed multiple tests on the data. A comparison of each CSF based on the two decades is presented in Table 5. We have used linear by linear association Chi-square test for finding any significance difference in the success factors between the two decades. A linear by linear association Chi-square test is considered more powerful than Pearson ' s χ 2 test [22].

The data in Table 5 shows that there is a minor difference between the two decade for the success factors CSF1 and CSF4. It means that these factors have been considered important in both decades and have still gained the researcher attention.

While the factors CSF2, CSF5, CSF 7 and CSF 8 have got importance in second decade with the emerging trend of GSD. The factor CSF 6 has gain less researcher attention in the second decade. The possible reason may be the maturity of interface specification in CBSE. The remaining two factors CSF3 and CSF9 have significant difference between the two decades. The importance of the unit/component testing may have been increased with the advent of CBSE in the second decade.

We also performed correlation analysis, by using Spearman's rank correlation method, to find the degree of relationship between the data of two decades as shown in Table 6. We gave the rank value 1 to the highest publication frequency. Similarly the

Table 5. Comparison of critical success factors in each decade.

Critical Success Factor	Decade 1 1994-2003 (n=24)		Decade 2 2004-2013 (n=65)		Chi-square test (linear-by-linear association) $\alpha = 0.05$		
	Freq	%	Freq	%	X^2	Df	P
CSF1 Consistency in Requirements and Architecture Design	10	42	25	38	0.075	1	0.785
CSF2 Intra and inter team Communication and Coordination	6	25	28	43	2.399	1	0.121
CSF3 Component/Unit Testing prior to integration	5	21	29	45	4.152	1	0.042
CSF4 Advance & Uniform Development Environment	8	33	25	38	0.195	1	0.658
CSF5 Efficient Incremental/Continuous integration	5	21	26	40	2.805	1	0.094
CSF6 Efficient specification for Interface Compatibility	**14***	**58**	**17**	**26**	**7.906**	**1**	**0.005**
CSF7 Proper Documentation & Configuration Management	5	21	23	35	1.702	1	0.192
CSF8 Early Integration Planning and Centralized P3 management	5	21	55	85	1.389	1	0.239
CSF9 Careful evaluation of the COTS/OTS Components	**14**	**58**	**13**	**20**	**12.050**	**1**	**0.001**

*The values which have statistical significance difference (P<0.05) have been highlighted as bold.

Table 6. Spearman's rank correlation.

Critical Success Factor	Decade 1 1994-2003 (n=24)		Decade 2 2004-2013 (n=65)	
	Freq	Rank	Freq	Rank
CSF1 Consistency in Requirements and Architecture Design	10	3	25	5.5
CSF2 Intra and inter team Communication and Coordination	6	5	28	3
CSF3 Component/Unit Testing prior to integration	5	7.25	29	2
CSF4 Advance & Uniform Development Environment	8	4	25	5.5
CSF5 Efficient Incremental/Continuous integration	5	7.25	26	4
CSF6 Efficient specification for Interface Compatibility	14	1.5	17	8
CSF7 Proper Documentation & Configuration Management	5	7.25	23	7
CSF8 Early Integration Planning and Centralized P3 management	5	7.25	55	1
CSF9 Careful evaluation of the COTS/OTS Components	14	1.5	13	9

next highest value is given the rank value 2 and so on as shown for the data in the second decade. In some cases whenever two or more values were the same then we calculated the average of their ranks and gave the same average rank to each of them. For example for decade two there were two values of 38 with rank 5 and 6, we assigned the average value of 5.5 to each. The same method was applied for data in decade 1.

The correlation coefficient (ϒ) is –0.67, which shows that there is a strong negative correlation between the factors across the two decades. It means that the researcher priorities have been changed in the second decade. This has been depicted by the scatter graph in Figure 3.

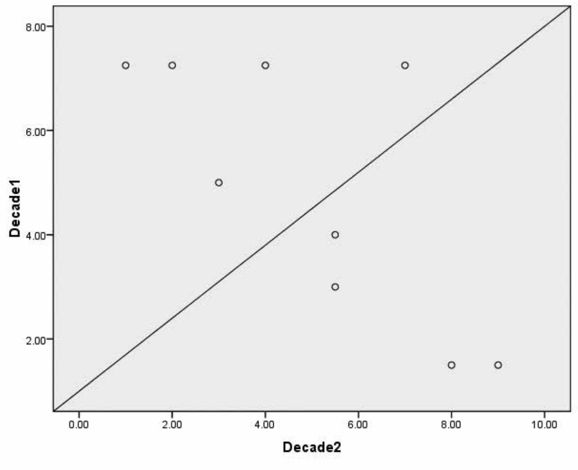

Fig. 3. Scatter graph for correlation.

4.3 Comparison on the basis of project size

To answer RQ1-b we initially divided the projects into three categories (small, medium and large) on the basis of project size as reported in the papers. Table 7 represents the frequency of papers found for each project size. The data in the table shows that most research work has been done on large size projects, while the majority of papers have not explicitly mentioned the project size.

Table 7. Project size break up of publications.

Project Size	Frequency	Percentage
Small	04	05
Medium	01	01
Large	31	35
Mixed/ Not Mentioned	53	60

We have further analyzed each CSF on the basis of project size in Table 8. Due to low frequency of success factors for small and medium projects we have combined their data into one column under the heading "Small and Medium". The data in Table 8 show that the frequency of papers for small and medium projects is five and each success factor frequency is at least two except for the factor CSF9. The factors CSF2, CSF3 and CSF5 can be considered as the most important factors for large projects. The factors CSF1, CSF4, CSF7

Table 8. Comparison of success factors based on the project size.

Critical Success Factor	Total sample size found through SLR study (n=89)						Chi-square test (linear-by-linear association) α = 0.05 Df=1	
	Small and Medium (n=5)		Large (n=31)		Not Mentioned / mixed (n=53)			
	Freq	%	Freq	%	Freq	%	X^2	P
CSF1 Consistency in Requirements and Architecture Design	2	40	11	35	22	42	0.226	0.634
CSF2 Intra and inter team Communication and Coordination	2	40	16	52	16	30	2.834	0.092
CSF3 Component/Unit Testing prior to integration	2	40	16	52	16	30	2.834	0.092
CSF4 Advance & Uniform Development Environment and Training	3	60	10	32	20	38	0.050	0.823
CSF5 Efficient Incremental/Continuous integration	2	40	14	45	15	28	1.716	0.190
CSF6 Efficient specification for Interface Compatibility	2	40	8	26	21	40	0.823	0.364
CSF7 Proper Documentation & Configuration Management	3	60	11	35	14	26	2.236	0.135
CSF8 Early Integration Planning and Centralized P3 management	2	40	12	39	14	26	1.246	0.264
CSF9 Careful evaluation of the COTS/OTS Components	**1**	**20**	**3**	**10**	**23**	**43**	**7.373**	**0.007**

The values which have statistical significance difference (P<0.05) have been highlighted as bold.

and CSF8 can be considered second in the ranked of importance for large projects. The data listed in Table 9 illustrate that there is a significant difference for only the last factor i,e CSF9. This factor has low frequency for small/medium and large size projects and has the highest frequency in the "Not Mentioned/mixed" column. The studies which have not mentioned their project size or which have mentioned all size of projects i.e. mixed category have highest frequency and hence it can be deduce that CSF9 is necessary for all size of projects to be successful in the integration process.

Table 9. Methodology wise break up of publications.

Methodology	Frequency	Percentage
Case study	38	43
Technical reports	12	14
Interview	8	9
Literature review	7	8
SLR	7	8
Experience report	4	5
Questionnaire survey	3	3
Experiment	3	3
Field study	3	3

4.4 Methodology used in the selected papers

As discussed in section 3, the data for CSF was extracted from 89 papers. To answer RQ1-d we have analyzed the methodologies used in these papers in Table 9. The most dominating methodology used in these papers is "Case Study" with 43%. Technical reports have 14%, interview has 8%, while literature review and SLR has 7%. Similarly experience report has 4% while questionnaire survey, experiment and field study has 3% each.

We have further compared and examined each factor on the basis of the study strategy used as shown in Table 10. In this table we have combined experience report, questionnaire survey, experiments and field studies into one group named "Other" because of their low frequencies. We also performed linear-by-linear Chi-square test for each factor based on the methodology used to find any significance difference among the methods if they have. From Table 10 we can see that the only factor which has significant difference is the CSF8. This factor has a 21% frequency for case studies, 25% for technical report and 50% for interview. It has no occurrence in the ordinary literature review while 57% frequency in the SLR. Similarly it has

Table 10. Comparison of success factors based on methodology used.

Critical Success Factor	Total sample size found through SLR study (n=89)												Chi-square test α = 0.05 Df=1	
	Case Studies (n=38)		Technical Reports (n=12)		Interview (n=8)		Ordinary Review (n=7)		SLR (n=7)		Other (n=13)			
	F	%	F	%	F	%	F	%	F	%	F	%	X^2	P
CSF1 Consistency in Requirements and Architecture Design	16	42	5	42	3	38	1	14	3	43	7	54	0.001	0.970
CSF2 Intra and inter team Communication and Coordination	17	45	2	17	4	50	0	0	4	57	7	54	0.172	0.679
CSF3 Component/Unit Testing prior to integration	14	37	2	17	4	50	3	43	4	57	7	54	0.087	0.768
CSF4 Advance & Uniform Development Environment and Training	16	42	4	33	5	63	1	14	3	43	4	31	1.329	0.249
CSF5 Efficient Incremental/Continuous integration	16	42	3	25	2	25	1	14	3	43	6	46	0.242	0.623
CSF6 Efficient specification for Interface Compatibility	13	34	6	50	2	25	4	57	1	14	5	38	0.138	0.711
CSF7 Proper Documentation & Configuration Management	11	29	0	0	3	38	2	29	3	43	9	69	0.826	0.363
CSF8 Early Integration Planning and Centralized P3 management	**8**	**21**	**3**	**25**	**4**	**50**	**0**	**0**	**4**	**57**	**9**	**69**	**5.150**	**0.023**
CSF9 Careful evaluation of the COTS/OTS Components	8	21	6	50	4	50	3	43	3	43	3	23	0.592	0.442
Number of factors found	9		8		9		7		9		9			

The values which have statistical significance difference (P<0.05) have been highlighted as bold.

69% frequency in the “other category”. At the end of the table we have also listed the number of critical factors identified by each methodology._We found that case studies, interview and SLR each have identified 09 CSF while technical reports and literature review have identified 8 and 7 CSF respectively. Thus it is evident that nearly all of our identified CSFs have been reported in studies conducted through various methodologies.

The purpose of above analysis is to find out the relative weight of each factor i.e. to find out that a factor is identified by what type of methods. It is more authentic if a factor is identified by both literature and empirical methods like case studies, interviews and questionnaire survey etc. The data in Table 10 shows that out of 89 papers, 38 papers have used the case study method and all the factors are identified using case study methods in addition with other methods. This shows the importance of our identified factors because all the factors have been mentioned in the empirical studies.

5. DISCUSSIONS AND SUMMARY

Through this SLR study we identified a list of success factors for GSD vendors to assist them in successful integration of the software components. We further categorized the critical success factors of software components integration, which needs special attention of GSD vendors. The criterion we used for a success factor to be critical or not was based on its citation in the literature. The factors having a frequency percentage >=30 were marked as critical success factors. The practitioners can, however, define their own criteria for the factors in the list to be critical or not. Thus for answering RQ1 we identified a list of nine critical success factors as shown in Table 3 from serial number 1 to 9.

To answer RQ1-a, we divided the time period of publications into two decades. It was found that the research on software integration has been boosted in the second decade as compared to the first decade. Because in first decade we found 24 publications for success factors while in second decade we found 65 publications. This shows an increase in the importance of the software integration. The data in Table 5 show that all the CSFs are present in both decades i.e these CSFs have been considered important in both decades. We also categorized the above CSF on the basis of project size i.e small, medium and large for RQ1-b. As large projects are more complicated and face more problems, the literature has cited these CSF in studies conducted on large size projects. Majority of the studies have not explicitly mentioned any specific project size, so these CSF can be considered important for integration process in all size projects.

We further categorized the selected papers on the basis of methodology/ strategy used for RQ1-c. The most dominating methodology used is case study with 43% citation in the selected papers. We also compared the CSFs on the basis of study strategy used and found that case study, interview and SLR, all three method have identified all the CSFs. Technical reports missed one CSF while literature review missed two CSFs.

6. LIMITATIONS

How valid are the results of success factors for achieving successful integration? In some papers the authors have not mentioned explicitly that why they have considered a particular factor to be vital for the integration process. This may be a threat to internal validity. However it was not possible to overcome this problem independently. Similarly some studies contributed to this SLR consisted of self experiences reports and case studies which may be possibly subject to the publication bias.

Due to our limited resources we searched only six digital libraries while conducting the SLR process and may have missed some relevant studies in other libraries. According to SLR criteria this is not a systematic lapse [19].

We also used the snowballing technique for finding the relevant papers, which we may have missed during the search process, and to increase the sample size of papers for our SLR.

7. CONCLUSIONS AND FUTURE WORK

We found that the critical success factors CSF1, CSF3, CSF5, CSF6 and CSF9 can play a vital role before the integration process. It may therefore be helpful that GSD vendors give full consideration to the implementation of these success factors before the integration phase. The factor CSF4 can play a vital role during the integration process since the skilled use of modern and advance technology will ease the integration process. It is also necessary to use the same and uniform technology at all GSD sites to avoid complication during the integration process. Lastly the factors CSF2, CSF7 and CSF8 are the factors that can be considered important for all stages of software integration.

The findings of the this study were obtained from a thorough review of the literature using SLR process, which complete the 1st phase of our proposed software integration model (SIM) [12]. The findings of this study will be further validated empirically in industry in future.

8. REFERENCES

1. Ray, D.M. & P. Samuel. Improving the productivity in global software development. In: *Innovations in Bio-inspired Computing and Applications*. Springer, p. 175-185 (2016).
2. Šmite, Darja, F. Calefato & C. Wohlin. Cost-savings in global software engineering: Where's the evidence," *IEEE Software* 32: 26-32 (2015).
3. Khan, A.W. & S.U Khan. Solutions for critical challenges in offshore software outsourcing contract. *Pakistan Academy of Sciences* 52: 331-344 (2015).
4. Nguyen-Duc, A., Daniela S. Cruzes & Reidar, Conradi. The impact of global dispersion on coordination, team performance and software quality–a systematic literature review. *Information and Software Technology* 57: 277-294 (2015).
5. Ghobadi, Shahla. What drives knowledge sharing in software development teams: A literature review and classification framework. *Information & Management* 52: 82-97 (2015).
6. Khan, Rafiq Ahmad, Siffat Ullah Khan & Mahmood, Niazi. Communication and coordination challenges mitigation in offshore software development outsourcing relationships: Findings from systematic literature review. In: *The Tenth International Conference on Software Engineering Advances (ICSEA 2015)*, Barcelona, Spain, p. 45-51 (2015).
7. Tekumalla, Bharath. Status of empirical research in component based software engineering. Master of Science Thesis, Department of Computer Science and Engineering, University of Gothenburg, Sweden (2012).
8. Schneider, Stefan, Richard, Torkar & Tony, Gorschek. Solutions in global software engineering: A systematic literature review. *International Journal*

of Information Management 33: 119-132, (2013).

9. Adams, Bram, Ryan Kavanagh, Ahmed E. Hassan & Daniel M. German. An empirical study of integration activities in distributions of open source software. *Empirical Software Engineering,* p. 1-42 (2015).
10. Hintikka, Anssi & Jarkko Hyysalo. Rooster project task 2: Cluster processes. ROOSTER SMARTPHONE INNOVATION CLUSTER, Helsinki, Finland (2007).
11. Groen, Derek. Xiaohu Guo, James A. Grogan, Ulf D. Schiller & James M. Osborne. Software development practices in academia: A case study comparison. *arXiv preprint arXiv:1506.05272,* (2015).
12. Ilyas, Muhammad & Siffat Ullah Khan. Software integration model for global software development. In: *15th International Multitopic Conference (INMIC)*, Islamabad, Pakistan, p. 452-457, (2012).
13. Ilyas, Muhammad & Siffat Ullah Khan. Software integration in global software development: Success factors for gsd vendors. In: *16th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/ Distributed Computing (SNPD 2015)*, Takamatsu, Japan, p. 119-124 (2015).
14. McConnel, S. *Code complete 2*, 2nd ed.: Microsoft Press Redmond, WA, USA (2004).
15. Herbsleb, J.D. & R.E. Grinter. Splitting the organization and integrating the code: Conway's law revisited. In: *Proceedings of the 21st International Conference on Software Engineering (ICSE)*, NY USA, p. 85-95 (1999).
16. Van Moll, J.H. & R.W.M. Ammerlaan. Identifying pitfalls of system integration -- an exploratory study. In: *IEEE International Conference on Software Testing Verification and Validation Workshop*, p. 331-338 (2008).
17. Guimaraes, Mario Luis & Antonio Rito, Silva. Making software integration really continuous. In: *Fundamental Approaches to Software Engineering*, ed: Springer, pp. 332-346 (2012).
18. Stayhl, Daniel & Bosch, Jan. Modeling continuous integration practice differences in industry software development. *Journal of Systems and Software* 87: 48-59 (2013).
19. Kitchenham B. & S. Charters, Guidelines for performing systematic literature reviews in software engineering. Keele University, UK, (2007).
20. Ilyas, Muhammad & Siffat Ullah Khan. Software integration challenges in global software development environment: A systematic literature review protocol, *IOSR Journal of Computer Engineering* 1: 29-38 (2012).
21. Kitchenham, Barbara & Pearl, Brereton. A systematic review of systematic review process research in software engineering, *Information and Software Technology (IST)* 55: 2049–2075 (2013).
22. Bland, Martin, *An Introduction to Medical Statistics*, 3 ed: Oxford University Press (2000).

Appendix-A List of selected papers for data extraction.

1. Land, R. and I. Crnkovic, *Software systems in-house integration: Architecture, process practices, and strategy selection.* Information and Software Technology, 2006. **49**(5): p. 419-444.
2. Bosch, J. and P. Bosch-Sijtsema, *From integration to composition: On the impact of software product lines, global development and ecosystems.* Journal of Systems and Software, 2010. **83**(1): p. 67-76.
3. Kommeren, R. and P.i. Parviainen, *Philips experiences in global distributed software development.* Empirical Software Engineering, 2007. **12**(6): p. 647-660.
4. Jimenez, M., M. Piattini, and A. Vizcaino, *Challenges and improvements in distributed software development: A systematic review.* Advances in Software Engineering, 2009. **2009**: p. 1-14.
5. Prikladnicki, R., et al. *Distributed Software Development: Practices and challenges in different business strategies of offshoring and onshoring.* in *Second IEEE International Conference on Global Software Engineering (ICGSE)*. 2007. Munich, Germany: IEEE.
6. Gotel, O., et al. *Integration starts on day one in global software development projects.* in *IEEE International Conference on Global Software Engineering (ICGSE)*. 2008. Bangalore: IEEE.
7. Avritzer, A., W. Hasling, and D. Paulish. *Process investigations for the global studio project version 3.0.* in *Second IEEE International Conference on Global Software Engineering (ICGSE)*. 2007. Munich: IEEE.
8. Leszak, M. and M. Meier. *Successful global development of a large-scale embedded telecommunications product.* in *Second IEEE International Conference on Global Software Engineering (ICGSE)*. 2007. Munich: IEEE.
9. Wolf, T., et al. *Predicting build failures using social network analysis on developer communication.* in *Proceedings of the 31st International Conference on Software Engineering (ICSE)*. 2009: IEEE Computer Society.
10. Yau, S.S. and F. Karim. *Component customization for object-oriented distributed real-time software development.* in *Proceedings of Third IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC 2000)*. 2000. Newport, CA: IEEE.
11. Sangwan, R.S. and P.A. Laplante, *Test-driven development in large projects.* IT Professional, 2006. **8**(5): p. 25-29.
12. Biehl, M. and M. Torngren. *A Cost-Efficiency Model for Tool Chains.* in *Seventh International Conference on Global Software Engineering Workshops (ICGSEW), 2012 IEEE*. 2012. Porto Alegre: IEEE.
13. Sarma, A. and A. Van Der Hoek. *Towards awareness in the large.* in *International Conference onGlobal Software Engineering (ICGSE)*. 2006. Florianopolis: IEEE.
14. Koroorian, S. and M. Kajko-Mattsson. *A Tale of Two Daily Build Projects.* in *The Third International Conference on Software Engineering Advances (ICSEA)*. 2008. Sliema: IEEE.
15. Kajko-Mattsson, M. and T. Bjornsson. *Outlining developers' testing process model.* in *Proceedings of the 33rd EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*. 2007. Lubeck, Germany.
16. Yau, S.S. and B. Xia. *A Component-Based Approach to Object-Oriented Distributed Application Software Development.* in *The Twenty-Second Annual International Computer Software and Applications Conference (COMPSAC '98)*. 1998. Vienna: IEEE.
17. Zafar, A., S. Ali, and R.K. Shahzad. *Investigating integration challenges and solutions in global software development.* in *Frontiers of Information Technology (FIT)*. 2011. Islamabad: IEEE.
18. Hyysalo, J., P. Parviainen, and M. Tihinen. *Collaborative development: Industrial experiences and needs for further research.* in *Profes2005 workshop*. 2005.
19. Avritzer, A. and J.P. Ros, *Test Driven Architecture Enabled Process For Open Collaboration in Global.* 2008, Google Patents.
20. Paloheimo, H., *Feasibility of SW Architecture Integration in a Distributed R&D Environment.* Distributed and Inter-organizational Software Development, 2003: p. 1-8.
21. Redmiles, D., et al., *Continuous Coordination-A New Paradigm to Support Globally Distributed Software Development Projects.* Wirtschafts Informatik, 2007. **49**(1): p. 28.
22. Morisio, M., et al. *Investigating and improving a COTS-based software development.* in *Proceedings of the 22nd international conference on Software Engineering (ICSE)*. 2000. Limerick, Ireland: ACM.
23. Ramasubbu, N., A. Mehra, and V.S. Mookerjee. *Modeling Coordination in Offshore Software Development.* in *Pacifc Asia Conference on Information Systems (PACIS)*. 2009. Hyderabad, India.
24. Holck, J. and N. Jorgensen, *Continuous Integration and Quality Assurance: a case study of two open source projects.* Australasian Journal of Information Systems, 2007. **11**(1): p. 40-53.
25. Ramasubbu, N., et al. *Effect of Quality Management Practices in Distributed Offshore Software Development.* in *Proceedings of the Academy of Management (AOM) Annual Meeting*. 2004. New Orleans: Citeseer.
26. Larsson, S., *Key Elements of the Product Integration Process.* 2007, Malardalen University Sweden. p. 78.
27. Begin, M.-E. and L. Merifield, *Software Integration Final Report.* 2012, Members of the StratusLab collaboration: Centre National de la Recherche Scientique, Universidad Complutense de Madrid, Greek Research and Technology Network S.A: Brussels, Belgium. p. 27.
28. Cao, D. *Oil-Field Services Data Acquisition System - A Globally Distributed Development.* in *32nd Annual IEEE Internationa Computer Software and Applications, 2008 (COMPSAC'08)*. 2008. Turku: IEEE.
29. Yu-Hong, W., et al. *A DDS based framework for remote integration over the Internet.* in *7th Annual Conference on Systems Engineering Research 2009 (CSER 2009)*. 2009. Loughborough University.

30. A Satzger, G. *System Integration in Information Technology - An Intermediation Rather Than A Procurement Task?* in *European Conference on InformationSystems (ECIS)*. 1997. Cork, Ireland.
31. Bendix, L., J. Magnusson, and C. Pendleton. *Configuration Management Stories from the Distributed Software Development Trenches*. in *IEEE Seventh International Conference on Global Software Engineering (ICGSE)*. 2012. Porto Alegre: IEEE.
32. Yau, S.S. and F. Karim. *Integration of object-oriented software components for distributed application software development*. in *7th IEEE Workshop on Future Trends of Distributed Computing Systems*. 1999. Cape Town: IEEE.
33. Gill, N.S. and L.M. deCesare Sergio, *Measurement of Component-based Software: Some Important Issues*. UKAIS, 2002: p. 373-381.
34. Jorgensen, N., *Incremental and decentralized integration in FreeBSD*. Feller ed. Perspectives on Free and Open Source Software, ed. J.F.a.B.F.a.S.H.a.K. Lakhani. Vol. 112. 2005, Cambridge, Massachusetts, USA: MIT Press. 227-243.
35. Hintikka, A. and J. Hyysalo, *ROOSTER project task 2: Cluster prosesses*. 2007, ROOSTER SMARTPHONE INNOVATION CLUSTER. p. 21.
36. Bruegge, B. and A. Dutoit, *Software Metrics for Distributed Development*. 1996: Citeseer.
37. Magni, M., et al. *Improvisation and Performance in Software Development Teams: The Role of Geographic Dispersion*. in *International Conference on Information Systems (ICIS)*. 2008. Paris, France: Citeseer.
38. Tekumalla, B., *Status of Empirical Research in Component Based Software Engineering-A Systematic Literature Review of empirical studies*. 2012, University of Gothenburg: Sweden. p.52.
39. Kotlarsky, J., *Management of Globally Distributed Component-Based Software Development*. 2005, RSM Erasmus University Rotterdam, The Netherlands. p. 391.
40. Callahan, J.R. and J.M. Purtilo, *Using an architectural approach to integrate heterogeneous, distributed software components*. 1995, West Virginia University. p. 36.
41. Sosa, M.E., *A structured approach to predicting and managing technical interactions in software development*. Research in Engineering Design, 2008. **19**(1): p. 47-70.
42. Lehmusvirta, J., *Systemic Modeling of Hierarchical Software Development in a Distributed Enterprise*. 2005, HELSINKI UNIVERSITY OF TECHNOLOGY: Finland. p. 98.
43. Usta, A.S., *Tool support for distributed agile software development*. 2006, MIDDLE EAST TECHNICAL UNIVERSITY: Ankara / TURKIYE. p. 104.
44. Bourgeois, S., et al., *Architectural design of a continuous integration environment*. 2008, Ghent University: UGhent. p. 141.
45. Syvertsen, M.M., *Selection and use of third-party Software Components: Study of a IT consultancy firm*. 2011, Norwegian University of Science and Technology: Norway. p. 99.
46. Kuderli, L., *Software Architecture Design for Globally Distributed Development Teams*. 2005, Technical University of Munich. p. 127.
47. Haikonen, P., *Distributed Agile software development and the requirements for Information Technology*. 2009, HELSINKI UNIVERSITY OF TECHNOLOGY: Finland. p. 74.
48. Masticola, S. and D. Paulish, *Process for global software development*, in *US 2005/0166178 A1*. 2005, Siemens Corporation Intellectual Property Department 170 Wood Avenue South Iselin, NJ 08830 (US): US. p. 14.
49. Lee, G., J.A. Espinosa, and W. DeLone, *BALANCING RIGOR, STANDARDIZATION, AND AGILITY IN DISTRIBUTED IS DEVELOPMENT PROCESS: AN AMBIDEXTERITY PERSPECTIVE*, in *Kogod School of Business*. 2012, American University: Washington DC. p. 41.
50. Klaas-Jan, S., *Supporting Product Development with Software from the Bazaar*. 2011, University of Limerick: Limerick. p. 336.
51. Land, R., *An architectural approach to software evolution and integration*. 2003, Malardalen University: Sweden. p. 212.
52. Penzenstadler, B., *DeSyRe: Decomposition of Systems and Their Requirements*. 2011, Technical University Munich: Munich. p. 182.
53. Dunn, L., *An investigation of the factors affecting the lifecycle costs of COTS-based systems*. 2011, University of Portsmouth: Portsmouth UK. p. 453.
54. Ainslie, J., et al., *Global distributed objects--opportunities and challenges*. BT technology journal, 2000. **18**(1): p. 57-59.
55. Han, J. and P. Chen, *Architecture support for system-of-systems evolution*, in *Engineering and Deployment of Cooperative Information Systems*. 2002, Springer. p. 332-346.
56. Land, R. and I. Crnkovic. *Software systems integration and architectural analysis-a case study*. in *International Conference on Software Maintenance (ICSM)*. 2003. Netherlands: IEEE.
57. Land, R. and I. Crnkovic. *Existing Approaches to Software Integration and a Challenge for the Future*. in *Fourth Conference on Software Engineering Research and Practice in Sweden (SERPS)*. 2004. Linköping University, Sweden: Citeseer.
58. Larsson, S. and I. Crnkovic. *Case study: software product integration practices*. in *6th international conference Profes*. 2005. Oulu Finland: Springer.
59. Herbsleb, J.D., D.J. Paulish, and M. Bass. *Global software development at siemens: experience from nine projects*. in *27th International Conference on Software Engineering (ICSE)*. 2005. St. Louis, Missouri, USA: IEEE.
60. Crnkovic, I., *Component-based Software Engineering - New Challenges in Software Development*. Software Focus, 2001. **2**(4): p. 127-133.
61. Abts, C., B.W. Boehm, and E.B. Clark. *COCOTS: A COTS software integration lifecycle cost model-model overview and preliminary*

data collection findings. in *11th European Software Control and Metric Conference (ESCOM SCOPE 2000)*. 2000. Munich, Germany: Citeseer.

62. Chang, H., L. Mariani, and M. Pezze. *In-field healing of integration problems with COTS components*. in *IEEE 31st International Conference on Software Engineering (ICSE)*. 2009. Vancouver, BC: IEEE.
63. Cataldo, M., et al. *On Coordination Mechanisms in Global Software Development*. in *2nd IEEE International Conference on Global Software Engineering (ICGSE)*. 2007. Munich, Germany: Citeseer.
64. Sengupta, B., et al. *Test-driven global software development*. in *7th International Workshop on Global Software Development*. 2004. Edinburgh, Scotland: IEEE.
65. Paasivaara, M. and C. Lassenius. *Using iterative and incremental processes in global software development*. in *The 3rd International Workshop on Global Software Development/International Conference on Software engineering (ICSE)*. 2004. Edinburgh, Scotland: IET.
66. de Souza, C.R., D. Redmiles, and P. Dourish. *Breaking the code, moving between private and public work in collaborative software development*. in *Proceedings of the 2003 International ACM SIGGROUP conference on Supporting group work*. 2003. Sanibel Island, USA: ACM.
67. Li, J., et al., *Development with off-the-shelf components: 10 facts*. IEEE software, 2009. **26**(2): p. 80-87.
68. Li, J., et al., *A state-of-the-practice survey of risk management in development with off-the-shelf software components*. IEEE Transactions on Software Engineering, 2008. **34**(2): p. 271-286.
69. Crnkovic, I., M. Chaudron, and S. Larsson, *Component-based development process and component lifecycle*. Journal of Computing and Information Technology - CIT, 2005. **13**(4): p. 321-327.
70. Larsson, S., et al., *Software product integration: A case study-based synthesis of reference models*. Information and software technology, 2009. **51**(6): p. 1066-1080.
71. Land, R., et al. *Towards guidelines for a development process for component-based embedded systems*. in *International Conference on Computational Science and Its Applications (ICCSA 2009)*. 2009. Suwon (Korea): Springer Berlin Heidelberg.
72. Land, R., et al. *Software systems in-house integration strategies: merge or retire-experiences from industry*. in *Proceedings of Software Engineering Research and Practice in Sweden (SERPS)*. 2005. Sweden: Citeseer.
73. Larsson, S. *Towards an Efficient and Effective Process for Integration of Component-Based Software Systems*. in *Proceedings of the 3rd Conference on Software Engineering Research and Practice in Sweden (SERPS)*. 2003. Lund, Sweden: Citeseer.
74. Stol, K.-J., M.A. Babar, and P. Avgeriou, *The importance of architectural knowledge in integrating open source software*, in *Open Source Systems: Grounding Research*. 2011, Springer. p. 142-158.
75. Land, R., et al., *COTS selection best practices in literature and in industry*, in *High Confidence Software Reuse in Large Systems*. 2008, Springer. p. 100-111.
76. Larsson, S., *Improving software product integration*. 2005, Department of Computer Science and Electronics, Malardalen University: Sweden. p. 120.
77. Gao, J., H.-S. Tsao, and Y. Wu, *Testing and quality assurance for component-based software*. 2003, Boston, London: Artech House Publishers. 439.
78. Merilinna, J. and M. Matinlassi. *State of the Art and Practice of OpenSource Component Integration*. in *32nd EUROMICRO Conference on Software Engineering and Advanced Applications, 2006. SEAA '06*. 2006. Cavtat, Dubrovnik: IEEE.
79. van Gurp, J., C. Prehofer, and J. Bosch, *Comparing practices for reuse in integration oriented software product lines and large open source software projects*. Software: Practice and Experience, 2010. **40**(4): p. 285-312.
80. Karlsson, E.-A., L.-G. Andersson, and P. Leion. *Daily build and feature development in large distributed projects*. in *Proceedings of the 2000 International Conference on Software Engineering, 2000 (ICSE)*. 2000. Ireland: IEEE.
81. Van Moll, J. and R. Ammerlaan. *Identifying Pitfalls of System Integration-An Exploratory Study*. in *IEEE International Conference on Software Testing Verification and Validation Workshop (ICSTW)*. 2008. Lillehammer: IEEE.
82. Stayhl, D. and J. Bosch, *Modeling continuous integration practice differences in industry software development*. Journal of Systems and Software, 2013. **87**: p. 48-59.
83. Baik, J., N. Eickelmann, and C. Abts. *Empirical software simulation for COTS glue code development and integration*. in *25th Annual International Computer Software and Applications Conference (COMPSAC)*. 2001. Chicago, IL: IEEE.
84. Dogru, A.H. and M.M. Tanik, *A process model for component-oriented software engineering*. IEEE Software, 2003. **20**(2): p. 34-41.
85. Farcas, C., et al., *Addressing the Integration Challenge for Avionics and Automotive Systems From Components to Rich Services*. Proceedings of the IEEE, 2010. **98**(4): p. 562-583.
86. Herbsleb, J.D. and R.E. Grinter. *Splitting the Organization and Integrating the Code: Conway's Law Revisited*. in *Proceedings of the 21st International Conference on Software Engineering (ICSE)*. 1999. NY USA: ACM Press.
87. Yau, S.S. *Achieving Quality Software Development for Distributed Environments*. in *Proceedings of the First Asia-Pacific Conference on Quality Software (APAQS)*. 2000. Hong Kong: IEEE.
88. Sauer, J., *Architecture-Centric Development in Globally Distributed Projects*, in *Agility Across Time and Space*. 2010, Springer Berlin Heidelberg. p. 321-329.
89. Fowler, M. and M. Foemmel, *Continuous integration*. Thought-Works) http://www. thoughtworks. com/Continuous Integration. pdf, 2006.