International Journal of the Physical Sciences Vol. 6(5), pp. 1213-1232, 4 March, 2011
Available online at http://www.academicjournals.org/IJPS
DOI: 10.5897/IJPS11.092
ISSN 1992 - 1950 ©2011 Academic Journals

Full Length Research Paper

Addressing a critical success factor for software projects: A multi-round Delphi study of TSP

Mohd Hairul Nizam Md Nasir[1]* and Shamsul Sahibuddin[2]

[1]Faculty of Computer Science and Information Technology, University of Malaya, Malaysia.
[2]Advanced Informatics School, Universiti Teknologi Malaysia, Malaysia.

Accepted 25 February, 2011

A five-round Delphi study was conducted to determine the degree to which the Team Software Process (TSP) can address the identified critical success factors for software projects. Three high-reputation and high-calibre experts participated in this study. Our results found agreement among the experts that the TSP provided the 'Best Practises' for addressing 14 critical success factors. The experts also agreed that the TSP provided a 'Very Good' framework for addressing 4 critical success factors. Our findings further suggested that 6 critical success factors were addressed by the TSP at a 'Good' level; only 1 critical success factor was addressed to a limited degree and none of the critical success factors were addressed at the 'Fair' level. The only critical success factor not addressed by the TSP was the 'good performance by vendors/contractors/consultants' factor. From an expert's perspective, the TSP provides an operational framework that addresses 21 critical success factors software projects. We believe that each framework or method on its own cannot address all the identified critical success factors. However, by combining a software process improvement and project management framework or other excellent software development process models, all of the critical success factors can be better addressed.

Key words: Team Software Process (TSP), critical success factors, Delphi study, software process improvement, software development, project management.

INTRODUCTION

Software development projects are known for being completed far over budget and behind schedule (Gray and Larson, 2008). In the United States, a survey conducted by the Standish Group (1995) in 1994 reported that data from several thousand information technology (IT) projects revealed a software project success rate of only 16%. Meanwhile, 31% of projects failed, and the remaining 53% had cost overruns, time overruns and impaired functionality. Of these projects, the average cost overrun was 189% and the average time overrun was 222%. Another recent report by the Standish Group (2009) showed a slight improvement in the year 2008; however, the figures remained troublesome, with a success rate of less than 40%.

Despite the widespread use of sound project management practises and process improvement models over the last several years, the failure of software projects remains a challenge to organisations. In striving to address software industry challenges, several models, frameworks and methods have been developed to improve software processes to produce quality software on time, under budget and within pre-agreed functionalities. One of the most widely practised methods is the Team Software Process (TSP), which has been implemented in a wide range of organisations worldwide and has afforded positive results (Davis and Mullaney, 2003).

Several published studies have reported that TSP teams are delivering essentially defect-free software on

*Corresponding author. E-mail: hairulnizam@um.edu.my. shamsul@utm.my.

schedule while improving productivity. To highlight a few recent results, Davis and Barbara (2009) reported the TSP team at Adobe produced 5 million lines of code that was 20 times better than the industry average, as measured in terms of system test and test density. Wilson (2010) reported that a software trouble report at the final product evaluation test found no problems after the United States Naval Air Systems Command (NAVAIR) system engineering team adopted TSP. Another study reported by Battle (2009) found that the system test-delivered defects averaged 0.9 per kilo lines of code, and customer-delivered defects averaged less than 0.5 per kilo lines of code after the United States Naval Oceanographic Office (NAVO) adopted TSP.

Although, TSP was designed to provide an operational framework for establishing an effective team environment and guiding engineering teams in their work, we believe that there are several components and processes in the TSP (e.g., project planning, information distribution, realistic budget and schedule and leadership) that significantly contribute to software project success. Each framework or method on its own cannot address all of the identified critical success factors. Thus, we predicted that the TSP would contribute to addressing the identified critical success factors.

A primary aim of this article is to determine to what degree the TSP addresses the critical success factors for software projects. Critical success factors are factors that, if addressed appropriately, will significantly improve the chances of project success (Pinto and Rouhiainen, 2001). Thus, in the first phase of our study, we attempted to identify the critical factors affecting the success of software projects by conducting an extensive literature search. In the second phase, a multi-round Delphi study was conducted to determine the degree to which the TSP could address all the critical success factors. Two research questions motivated the investigation reported here:

1. Research question 1 (RQ1): What factors, as identified in the study, have a positive impact on determining software project success?
2. Research question 2 (RQ2): To what degree, as agreed upon by the experts, can the TSP address critical factors, given a particular set of critical success factors for software projects?

TEAM SOFTWARE PROCESS (TSP)

The TSP is a prescriptive process for projects consisting of a set of process scripts, forms, standards, procedures, methods and tools for project teams to produce high-quality software products on schedule and within pre-agreed budget constraints (Humphrey, 2000, 2002, 2006). The TSP provides clear and concise guidance on software development processes, with emphasis on mutual support and leadership among software project team members. The purpose is to build effective teamwork through collaborative and disciplined work within productive team working environments, where everyone knows exactly what they are supposed to do and where roles and responsibilities are clearly defined. The outcome is a well-defined and well-planned work process. The operational processes in TSP are presented in the form of scripts, supplemented with specific forms to guide the team members throughout the project implementation.

The TSP is built upon the Personal Software Process (PSP), which acts as a set of operational procedures and a process structure to build and guide software engineering teams in developing software-intensive products. If the PSP deals with methods and guidance on how software engineers can continually improve their performance at the individual level, the TSP is designed to help PSP-trained engineers build self-directed teams of 3 to 20 members capable of planning and tracking their work, establishing goals, and taking ownership of their processes and plans (Humphrey, 1998, 2000). The process scales up by using the TSP Multi-team (TSPm) process, which allows multiple teams of 3 to 15 members using the TSP to work together on larger projects. A TSPm project uses special processes designed to address the additional complexity and communication issues related to large teams. Meanwhile, functional TSP (TSPf) is for teams in which each member usually works independently. Maintenance project teams are a good example of where TSPf could be adopted, as each member normally handles separate features of a product enhancement that require them to work on their own.

The TSP can be fit to all phases in the software development life cycle, including requirements elicitation, design, implementation and coding, testing and maintenance. It can also be used to develop various kinds of software products, ranging from real-time embedded control systems to commercial desktop client-server applications (Davis and Mullaney, 2003). The scaled-down academic version of TSP, TSPi, is intended for use in undergraduate or postgraduate courses or software projects.

The strategy of TSP is an iterative and evolving process that is continuous until the intended finished product is delivered. Each cycle starts with a TSP Launch that is conducted as a series of nine meetings over a 3- to 4-day period. The launch process is defined in eleven process scripts, which include the overall launch script, a script for each of the nine launch meetings, and a script for the launch post-mortem. In the first cycle of the launch, the team works together to create an overall estimate for the full project plan and a detailed plan covering the next 3 months. The team also decides on appropriate resource utilisation.

After the launch, the team works on the planned activities. The team holds weekly status meetings that help them to track, control, and manage the plan and the project. Because the roles and responsibilities of project management are distributed among the team members, monitoring and control activities are easier and more manageable. Every manager can focus on their main controlling and monitoring tasks. In addition, the Planning Manager assists the TSP team by aggregating team data to track project progress against the plan and reports and reviews the project status with the team weekly. The next Launch in the following cycle is called the Re-Launch. During Re-Launches, a re-planning may occur if the team notices any issues, such as schedule deviations, requirement changes, or lack of resources, to bring the project back on track.

CRITICAL SUCCESS FACTORS

Critical success factors are factors that, if addressed appropriately, will significantly improve the chances of project success (Pinto and Rouhiainen, 2001). Over the past several decades, numerous research studies (Pinto and Mantel, 1990; Belassi and Tukel, 1996; Tukel and Rom, 2001; White and Fortune, 2002) have been performed in the area of project management to identify critical factors that influence the success and/or failure of projects. However, the critical factors are usually identified for projects in various industries, such as engineering, manufacturing, construction and training, rather than focusing on software development or IT projects. Managing a software project is different from managing any other project due to the complexity, conformity, visibility and malleability of the software itself (Brooks, 1995; Galin, 2004; Jain, 2008) and because software development is intellectually intensive work (Fairly, 2009). Additionally, software has certain unique characteristics (Brooks, 1995; Galin, 2004; Jain, 2008) that cause software development projects to differ from other typical engineering projects. Most researchers agree that there are differences in project management among different industry types (Cooke-Davies and Arzymanow, 2002; Ibbs and Kwak, 2000; Zwikael and Globerson, 2006), and Dvir et al. (1998) suggested that project success factors are not universal to all projects. Thus, the critical success factors identified in other industries cannot be used as valid critical factors for software projects. In this research study, however, we did not differentiate between IT projects and software development projects because IT projects also involve software (Royal Academy of Engineering and the British Computer Society, 2004).

Many articles have reported on the critical success factors specific to software and IT projects (Standish Group, 2010; Sauer and Cuthbertson, 2003; Taylor, 2000). However, each of these studies is specific to one particular country. There has been no reported comprehensive study on different project sizes in various domains and in multiple countries. Such a thorough analysis is necessary for identifying factors that are critically important to software projects.

In this research study, the four well-known online journal databases: www.Sciencedirect.com, www.ieeexplore.ieee.org, www.springerlink.com and www.Emeraldinsight.com were extensively searched. As a result, we found 76 articles consisting of case studies, surveys and views of practitioners and experts. Of these 76 articles, only 43 articles were related to software projects, and 33 non-software projects were excluded. The remaining articles were analysed to develop a list of critical factors that specifically affect the success of software projects. The occurrences of each factor in the literature have been identified to determine the relative importance of each factor.

Data collection and analysis methods

In this research study, each article was carefully reviewed and a list of factors was compiled. There were three types of articles. First, in the articles describing the results of empirical studies (that is, surveys and case studies), it was easy to identify the factors because the authors often provided a summary of success or failure factors. Well-known surveys, such as reports published by the Standish Group and the British Computer Society, belong to this category. Second, there were articles in which the authors (that is experts and practitioners) described success and/or failure factors based on their wide range of experiences. Third, there were a few articles in which software or IT project failures were discussed, but the authors did not provide a summary of success or failure factors. In this case, each article was read carefully to avoid misunderstandings and misinterpretations. This research study subsequently analysed 29 published sets of empirical data from case studies, 9 published empirical data sets from surveys and 5 articles written by experts and practitioners between 1990 and 2010. Only 2 publications were found from before 1990, namely those published by Schmitt and Kozar (1978) and Wingrove (1986), and they were not included in this analysis.

To analyse and produce a list of critical success factors from the extensive literature, the content analysis method was adopted. Content analysis is an approach to the quantification of qualitative data (Holsti, 1969). Although it was originally developed for the analysis of human communication in the social sciences, several empirical software engineering studies (Rainer et al., 2003; Rainer and Hall, 2003) have adopted this method as part of their research methodology. Babbie (2010) defined content analysis as the “study of recorded human communications” including various forms of communication such as books, magazines, web pages and letters. In this research, the communications to be analysed were published articles. Seaman (1999) described this method as follows: an "analysis method based on counting the frequency of occurrence of some meaningful lexical phenomenon in a textual data set." Meanwhile, Weber (1996) described the method of measurement in content analysis as "counting the occurrences or calculating percentages of meaning units such as specific words, phrases, content categories and themes, and later transfer to control document." This method enables the application of frequency analysis by extracting quantitative data from qualitative data in an article and recording it in frequency tables for the purpose of analysis.

Prior to performing the frequency analysis, the articles were read to generate appropriate categories for responses. Different factors that contributed to the same meaning were grouped into one category. For example, focused and hardworking staff, team commitment, team morale and motivated personnel were grouped together in the 'committed and motivated team' category. This process was repeated until distinct sets of categories were obtained. Each category then represented a critical success factor for software projects.

The method of content analysis was adopted in this study rather than the data extraction method or the frequency analysis method because some of the factors described by the authors in the articles were not explicit and required careful reading to produce accurate findings. Because different authors use different terms to identify the same factors in the analysed literature, it can be quite complicated to determine to which category a given factor belongs. It is not enough to simply count the occurrence of words (factors), as is done in data extraction or frequency analysis methods. It is revealing that no study has used content analysis in articles published from 1978 to 2010. However, one article by Wateridge (1998) produced a list of success criteria using only frequency analysis, and White and Fortune (2006) produced a list of success factors using only frequency analysis.

Inter-rater reliability was verified to ensure that there was no substantial bias or subjectivity in the identification and grouping processes. Another researcher, who was not familiar with the current issues being discussed, was asked to identify factors that appeared in all the articles. The results were compared to those of the previous lists, and no major disagreements were found among the results.

To perform frequency analysis, the occurrence of each success factor in each article in the literature search was recorded. The numbers and percentages of the occurrence of each factor were then tabulated and transferred to a frequency table. By comparing the occurrences of a critical success factor in a number of articles against the occurrences of other factors in the same articles, the relative importance of each critical success factor could be calculated, and the success factors could be compared and ranked.

Critical success factors for software projects

To answer RQ1, Table 1 shows the list of critical success factors for software projects identified in 43 publications. Based on the analysis of our extensive literature search, 26 critical success factors were found to be related to project success. The total frequency of occurrences was 372. Although some of the factors had a low frequency, we decided to treat them as critical factors because the criticality depends not on the frequency but on the literature in which the critical success factors were highlighted.

The extensive literature search revealed that most of the practitioners considered clear requirements and specifications, clear objectives and goals, and a realistic schedule to be the three most critical success factors that contribute to project success. These three critical success factors could thus be considered pre-project execution aspects that need to be made clear and solidified before commencing and executing software projects. Although 88% of the publications included at least one of these three factors, only 26% cited all three. This finding was very much in line with the research studies conducted by Wateridge (1995) and Fortune and White (2006), which found that there was no broad consensus among researchers and practitioners in determining critical success factors for projects. We have performed details analysis and discussion in relation to these 26 critical success factors in Nasir and Sahibuddin (2011).

Based on these results, we asked our experts to determine the degree to which the TSP can address these critical success factors for software projects. Because we intended to gain more insight into success factors, we did not limit the list of factors that we thought were useful in a Delphi study.

HOW THE TSP ADDRESSES THE CRITICAL SUCCESS FACTORS FOR SOFTWARE PROJECTS

As discussed previously in 'critical success factors' we showed a rank-order of critical success factors for software projects. We used these findings as a baseline to conduct a multi-round Delphi study with three experts in the field who have years of experience in software industries and in-depth knowledge of the TSP. On our own, our analysis and assessment of how the TSP addresses critical success factors would have been influenced by various biases (e.g., limited knowledge and experience).

The Delphi method allowed us to capitalise on the varied experience and in-depth knowledge of the experts and to provide complete knowledge of the phenomena

Table 1. Critical success factors identified throughout 43 publications.

	Critical success factor	Literature citation	Citation count in the literature (n = 43)	
			Freq.	%
1	Clear requirements and specifications	Schmidt et al., 2001; Keil et al., 2002; Taylor, 2006, 2000; Kappelman et al., 2006; Standish Group, 1995, 2001, 1999, 2006; Whittaker, 1999; May, 1998; Yeo, 2002; Jiang and Klein, 2000; Jiang et al., 1999; Baccarini et al., 2004; Oz, 1994; Boehm, 1991; Ariane 501 Inquiry Board, 1996; Nuseibeh, 1997; Charette, 2005; Royal Academy of Engineering and the British Computer Society, 2004; Reel, 1999; Clegg et al., 1997; Oz and Sosik, 2000; Jones, 1996; Sauer and Cuthbertson, 2003.	26	60.5
2	Clear objectives and goals	Schmidt et al., 2001; Keil et al., 2002; Taylor, 2006, 2000; Sauer and Cuthbertson, 2003; Kappelman et al., 2006; Standish Group, 1995; Whittaker, 1999; Yeo, 2002; Beynon-Davies, 1999; Glaser, 2004; Ewusi-Mensah, 1997; Standing et al., 2006; Charette, 2005; Reel, 1999; Clegg et al., 1997; Standish Group, 2006, 2009; Procaccino et al., 2002; Milis and Mercken, 2002; Oz and Sosik, 2000; Humphrey, 2005; Drummond, 1998;	24	55.8
3	Realistic schedule	Schmidt et al., 2001; Taylor, 2006; Jones, 2006; Kappelman et al., 2006; Jones, 1996; Jones, 1995; Whittaker, 1999; May, 1998; Yeo, 2002; Beynon-Davies, 1999; Drummond, 1998; Ewusi-Mensah, 1997; Oz, 1994; Ewusi-Mensah and Prazasnyski, 1994; Boehm, 1991; Charette, 2005; Standish Group, 2001; Clegg et al., 1997; Standish Group, 1999; Procaccino et al., 2002; Oz and Sosik, 2000; Humphrey, 2005; Sauer and Cuthbertson, 2003.	23	53.5
4	Effective project management skills/methods (project manager)	Schmidt et al., 2001; Sauer and Cuthbertson, 2003; Kappelman et al., 2006; Standish Group, 1995; Yeo, 2002; Jiang and Klein, 2000; Perkins, 2006; Beynon-Davies, 1999; Humphrey, 2005; Oz, 1994; Nuseibeh, 1997; Charette, 2005; Clegg et al., 1997; Standish Group, 2006, 2001, 1999, 2009; Royal Academy of Engineering and the British Computer Society, 2004; Taylor, 2000; Milis and Mercken, 2002; Reel, 1999; Standing et al., 2006; Oz and Sosik, 2000.	23	53.5
5	Support from top management	Schmidt et al., 2001; Sauer and Cuthbertson, 2003; Kappelman et al., 2006; Standish Group, 1995, 2006; Whittaker, 1999; OGC, 2005; Yeo, 2002; Beynon-Davies, 1999; Baccarini et al., 2004; Glaser, 2004; Ewusi-Mensah, 1997; Standing et al., 2006; Ewusi-Mensah and Prazasnyski, 1994; Taylor, 2000; Standish Group, 2001, 1999, 2009; Procaccino et al., 2002;, Taylor, 2000; Milis and Mercken, 2002; Oz and Sosik, 2000.	22	51.2
6	User/client involvement	Schmidt et al., 2001; Keil et al., 2002; Sauer and Cuthbertson, 2003; Kappelman et al., 2006; Standish Group, 1995 2006, 2001, 1999, 2009; May, 1998; Yeo, 2002; Jiang and Klein, 2000; Jiang et al., 1999; Glaser, 2004; Standing et al., 2006; Ewusi-Mensah and Prazasnyski, 1994; Charette, 2005; Clegg et al., 1997; Milis and Mercken, 2002; Oz and Sosik, 2000.	20	46.5
7	Effective communication and feedback	Schmidt et al., 2001; Keil et al., 2002; Kappelman et al., 2006; May, 1998; OGC, 2005; Yeo, 2002; Jiang et al., 1999; Baccarini et al., 2004; Humphrey, 2005; Mahaney and Lederer, 2003; Ewusi-Mensah and Prazasnyski, 1994; Leveson, 2004; Charette, 2005; Standish Group, 2009; Royal Academy of Engineering and the British Computer Society, 2004; Procaccino et al., 2002; Taylor, 2000; Milis and Mercken, 2002; Oz and Sosik, 2000; Sauer and Cuthbertson, 2003.	20	46.5

Table 1. Contd.

8	Realistic budget	Schmidt et al., 2001; Jones, 1996, 1995; Whittaker, 1999; May, 1998; OGC, 2005; Beynon-Davies, 1999; Baccarini et al., 2004; Drummond, 1998; Ewusi-Mensah, 1997; Ewusi-Mensah and Prazasnyski, 1994; Boehm, 1991; Charette, 2005; Standish Group, 2001, 2006; Clegg et al., 1997; Oz and Sosik, 2000; Oz, 1994; Sauer and Cuthbertson, 2003.	19	44.2
9	Skilled and sufficient staff	Schmidt et al., 2001; Keil et al., 2002; Sauer and Cuthbertson, 2003; Kappelman et al., 2006; Standish Group, 1995, 2001, 1999, 2009; May, 1998; Jiang and Klein, 2000; Beynon-Davies, 1999; Baccarini et al., 2004; Ewusi-Mensah, 1997; Ewusi-Mensah and Prazasnyski, 1994; Boehm, 1991; Milis and Mercken, 2002; Reel, 1999; Oz and Sosik, 2000	18	41.9
10	Frozen requirement	Schmidt et al., 2001; Jones, 1996, 1995; Yeo, 2002; Jiang and Klein, 2000; Beynon-Davies, 1999; Drummond, 1998; Oz, 1994; Jiang et al., 2001; Nuseibeh, 1997; Taylor, 2000; Oz and Sosik, 2000; Taylor, 2006; Kappelman et al., 2006; Baccarini et al., 2004; Boehm, 1991; Sauer and Cuthbertson, 2003.	17	39.5
11	Familiarity with technology/ development methods	Schmidt et al., 2001; Standish Group, 1995; Whittaker, 1999; Jiang and Klein, 2000; Beynon-Davies, 1999; Jiang et al., 1999; Baccarini et al., 2004; Drummond, 1998; Oz, 1994; Ewusi-Mensah and Prazasnyski, 1994; Jiang et al., 2001; Charette, 2005; Royal Academy of Engineering and the British Computer Society, 2004; Oz and Sosik, 2000; Sauer and Cuthbertson, 2003.	15	34.9
12	Proper planning	Schmidt et al., 2001; Jones, 2006, 1995; Kappelman et al., 2006; Standish Group, 1995, 2001, 1999; Whittaker, 1999; May, 1998; Humphrey, 2005; Oz, 1994; Taylor, 2000; Milis and Mercken, 2002; Standing et al., 2006; Sauer and Cuthbertson, 2003.	15	34.9
13	Appropriate development processes/ methods (process)	Schmidt et al., 2001; Jones, 1995; OGC, 2005; Beynon-Davies, 1999; Jiang et al., 1999; Drummond, 1998; Mahaney and Lederer, 2003; Jiang et al., 2001; Nuseibeh, 1997; Charette, 2005; Standish Group, 2009; Milis and Mercken, 2002; Oz and Sosik, 2000; Sauer and Cuthbertson, 2003.	14	32.6
14	Up-to-date progress reporting	Jones, 2006, 1995; Whittaker, 1999; May, 1998; Baccarini et al., 2004; Humphrey, 2005; Ewusi-Mensah, 1997; Oz, 1994; Charette, 2005; Reel, 1999; Oz and Sosik, 2000; Royal Academy of Engineering and the British Computer Society, 2004.	12	27.9
15	Effective monitoring and control	Schmidt et al., 2001; Jones, 1996; OGC, 2005; Beynon-Davies, 1999; Humphrey, 2005; Mahaney and Lederer, 2003; Ewusi-Mensah, 1997; Royal Academy of Engineering and the British Computer Society, 2004; Reel, 1999; Oz and Sosik, 2000; Baccarini et al., 2004; Sauer and Cuthbertson, 2003.	12	27.9
16	Adequate resources	Kappelman et al., 2006; Standish Group, 1995; Jiang and Klein, 2000; Baccarini et al., 2004; Ewusi-Mensah and Prazasnyski, 1994; Milis and Mercken, 2002; Oz and Sosik, 2000; Jones, 2006; Standish Group, 2006; Beynon-Davies, 1999; Leveson, 2004.	11	25.6
17	Good leadership	Schmidt et al., 2001; OGC, 2005; Baccarini et al., 2004; Glaser, 2004; Humphrey, 2005; Drummond, 1998; Ewusi-Mensah, 1997; Standing et al., 2006; Reel, 1999; Clegg et al., 1997; Oz and Sosik, 2000.	11	25.6

Table 1. Contd.

18	Risk management	Whittaker, 1999; OGC, 2005; Yeo, 2002; Jiang et al., 1999; Ewusi-Mensah, 1997; Leveson, 2004; Nuseibeh, 1997; Charette, 2005; Royal Academy of Engineering and the British Computer Society, 2004; Oz and Sosik, 2000.	10	23.3
19	Complexity, project size, duration, and number of organisations involved	Schmidt et al., 2001; Sauer and Cuthbertson, 2003; Yeo, 2002; Jiang and Klein, 2000; Beynon-Davies, 1999; Glaser, 2004; Humphrey, 2005; Drummond, 1998; Jiang et al., 2001; Charette, 2005.	10	23.3
20	Effective change and configuration management	Schmidt et al., 2001; Taylor, 2006; Kappelman et al., 2006; Jones, 1995; Whittaker, 1999; Baccarini et al., 2004; Royal Academy of Engineering and the British Computer Society, 2004; Taylor, 2000; Oz and Sosik, 2000; Sauer and Cuthbertson, 2003.	10	20.9
21	Supporting tools and good infrastructure	Jones, 1996, 1995; Jiang et al., 1999; Ewusi-Mensah, 1997; Ewusi-Mensah and Prazasnyski, 1994; Leveson, 2004; Standish Group, 2006, 2001, 2009.	9	23.3
22	Committed and motivated team	Standish Group, 1995; Beynon-Davies, 1999; Jiang et al., 1999; Mahaney and Lederer, 2003; Ewusi-Mensah, 1997; Oz, 1994; Standing et al., 2006; Reel, 1999; Milis and Mercken, 2002.	9	20.9
23	Good quality management	Jones, 2006, 1996, 1995; Baccarini et al., 2004; Boehm, 1991; Leveson, 2004; Ariane 501 Inquiry Board, 1996; Nuseibeh, 1997; Reel, 1999.	9	20.9
24	Clear assignment of roles and responsibilities	Schmidt et al., 2001; Keil et al., 2002; Jiang and Klein, 2000; Baccarini et al., 2004; Leveson, 2004; Milis and Mercken, 2002; Sauer and Cuthbertson, 2003.	7	16.3
25	Good performance by vendors/ contractors/ consultants	Schmidt et al., 2001; Taylor, 2006; Whittaker, 1999; Baccarini et al., 2004.	4	9.3
26	End-user training provision	Beynon-Davies, 1999; Jiang et al., 1999.	2	4.7

(Adler and Ziglio, 1996; Delbeq et al., 1975) through controlled feedback. We decided on the Delphi method for two reasons. First, prior research has not yielded a set of validated measures for the construct of interest (that is, how the TSP addresses the critical success factors for software projects). Second, we chose the Delphi method because of its ability to achieve consensus, something that was lacking in field interviews and case study methods. The Delphi method provided a good solution that allowed us to conduct our investigation with rigor and internal consistency.

The expert profiles

To ensure the reliability of the experts' opinions, the following criteria were established and used to select the experts: (1) The expert must have at least 15 years of experience in software industries, (2) The expert must have at least 10 years of experience in software project management, (3) The expert must possess knowledge of the Software Engineering Institute (SEI) Certified Team Software Process, and (4) The expert must have at least 20 publications related to software process improvement and/or TSP. The first two criteria ensured that the expert had a varied experience background, and the last two criteria ensured expertise and familiarity with the TSP and general software process improvement. We excluded experts with experience in non-TSP and non-software process improvement because our focus was on these two areas. Other important criteria that we took into account were (1) Capacity and willingness of the experts to participate, and (2) Effort and time commitment for participating in a multi-round Delphi study (Skulmoski et

Table 2. Expert profiles.

Expert	Experience in software industries (Years)	Experience in software project management (Years)	TSP coach	Software process improvement and TSP-related publication
Expert 1	16	13	Yes	24
Expert 2	27	20	Yes	20
Expert 3	27	21	Yes	22

al., 2007).

We invited nine experts to participate in this research study. Three experts responded and stated their willingness and commitment to participate. The small sample size was due to limited expertise in our country and the difficulty of finding experts who could fulfil our criteria, especially in terms of effort and time commitment. However, we had high confidence in the quality of our experts. The profiles of the three experts, as shown in Table 2, indicate that all of the experts had impressive experience in the area of software project management and software process improvement and were well qualified.

According to Hakim (1987), small samples can be used to develop and test explanations, particularly in the early stages of the work. For example, Lam et al. (2000) used three experts to develop rules for a ceramic casting process, Nambisan et al. (1999) recruited six experts to develop taxonomy of organisational mechanisms and Gustafson et al. (1973) used four experts to estimate almanac events in their investigation of Delphi accuracy. We therefore argue that the number of experts did not have a significant impact on the outcome of our research. The experts in this field provided great insight in analysing, extracting and discussing all the features that were outlined in the TSP and mapped back to the identified success factors. Thus, we decided to move our research forward utilising only three experts, and we believe that the involvement of experts of such high reputation and calibre gives weight and rigor to our results. Furthermore, to increase the reliability and accuracy of the experts' opinions, we required all of the experts to be assisted by a colleague with equivalent experience for discussion and validation of each of the opinions given in this study. This led to reduced personal bias and controlled mistakes made by single experts.

Data collection and analysis methods

A Delphi questionnaire was mailed to the experts to collect their input in this multi-round Delphi. We first requested that the experts review our generated list of critical success factors for software projects, as shown in Table 1. We provided a definition and description of each of the factors to ensure that they were all working from a common list of items with common definitions. The experts did not highlight any problems with the list of critical success factors provided. We also asked the experts about their ability to respond to the questions, and we confirmed that they felt qualified and able to respond to the questions. Prior to its mailing, the survey was pre-tested by five software engineering researchers for clarity and ease of understanding. No changes were found to be necessary.

In round 1, the experts were asked to rate how the TSP addressed the critical success factors for software projects and to provide descriptions to justify their rating. We also asked the experts to specifically state the TSP processes and/or components in their description so that every critical success factor was clearly addressed by the TSP processes and/or components.

A six-point classification scale was implemented as follows:

1. Best practise (5): The TSP provides a very effective framework for addressing the critical success factors and has a direct impact on the software project's success.
2. Very good (4): The TSP provides a very good framework for addressing the critical success factors and has a significant impact on the software project's success, but it may not be the most effective way of doing things.
3. Good (3): The TSP provides a good framework for addressing the critical success factors, but there are minor missing processes that may impact the software project's success.
4. Fair (2): The TSP provides a framework that addresses the critical success factors to a reasonable degree, but there are several missing processes and/or incorrect settings of priorities that impact the software project's success.
5. Weak (1): The TSP provides a framework that addresses the critical success factors to a limited degree but does not cover everything that is required.
6. Not addressed (0): The TSP does not provide any way to address the critical success factors.

Their responses were reviewed, consolidated and disseminated anonymously in the subsequent round. In the next round, we asked the experts to confirm that their ratings and descriptions were consistent with their previous responses. To achieve consensus, the experts were asked to revise, correct, add to and eventually validate their earlier input after reviewing the feedback and comments of the other experts. We measured the

Table 3. Kappa values for each of the Delphi rounds.

Expert	Kappa values(k)				
	Round 1	Round 2	Round 3	Round 4	Round 5
Expert 1 vs. Expert 2	0.640	0.774	0.774	0.829	0.829
Expert 1 vs. Expert 3	0.372	0.743	0.806	0.873	0.873
Expert 2 vs. Expert 3	0.225	0.534	0.588	0.765	0.765

degree of consensus among the experts using Cohen's kappa coefficient (*k*) for each round between each pair of experts. This coefficient reflected the extent to which the observed consensus between experts was superior to that obtained by chance (Cohen, 1960). In our study, we iterated this multi-round process until we reached a kappa value of 0.7 with $p < 0.001$, indicating an acceptable level of consensus. A kappa value of 0.85 indicates almost perfect agreement (Landis and Koch, 1977), but a kappa of 0.7 or more is usually considered an acceptable level of agreement (Cramer, 1997). We could therefore be confident of the reliability of our output by confirming a high level of agreement. This approach is consistent with the basis that the number of rounds was somewhat flexible and the Delphi iteration process stops when a reasonable level of consensus is reached (Delbeq, 1975). We also used standard deviation to observe agreement among the experts for each of the critical success factors throughout the round. A low standard deviation indicated that the ratings tended to be very close to each other, whereas a high standard deviation indicated that the ratings were spread out over a large range. During the final round, we presented the findings to the experts and asked them to review and finalise their ratings and descriptions. All the experts agreed with the final findings, and no changes were found necessary.

FINDINGS

A five-round Delphi process was used to achieve consensus among the experts as well as to finalise the findings. Table 3 reports the kappa value for each Delphi round. A kappa value from 0.40 to 0.59 was considered moderate, 0.60 to 0.79 substantial, and 0.80 outstanding (Landis and Koch, 1977). After we finalised our findings, the degree of consensus between expert 1 and expert 2 and between expert 1 and expert 3 achieved an outstanding level (0.829 and 0.873, respectively, with $p < 0.001$). The degree of consensus between expert 2 and expert 3 was at the substantial level (0.765 with $p < 0.001$). Even though we reached a kappa value of 0.7 in round 4, which indicated a reasonable level of consensus, we decided to proceed with the next round to review and finalise the findings.

Table 4 reports our findings for the final round for each of the critical success factors. In terms of agreement among the experts, the average standard deviation in round 1 was 0.861 with a total standard deviation of 22.381; this was reduced to 0.474 with a total standard deviation of 12.322 in round 2. In round 3, the average standard deviation was 0.393 with a total standard deviation of 10.217. Round 4 provided even better agreement among the experts, where the average standard deviation was further reduced to 0.194 with a total standard deviation of only 5.039. These figures remained stable until the final round. The final results showed that the agreement among experts for each critical factor was quite high. Specifically, the standard deviation for each was equal to or less than 1.000, and the standard deviation for over 69% of the factors was 0.000. These results also suggested that 18 of the 26 critical factors gained an outright consensus from the experts. The remaining 7 critical factors showed slight differences with a standard deviation of 0.577, and only one critical factor had a standard deviation of 1.000.

As shown in Table 5, the experts agreed that the TSP provided a very effective framework for addressing 14 (53.85%) critical success factors. The experts also agreed that the TSP provided a very good framework for addressing 4 critical success factors (15.38%). Our findings further suggested that 6 critical success factors (23.07%) were addressed by the TSP at a 'Good' level. In addition, only 1 critical success factor (3.85%) was addressed to a limited degree, and none of the critical success factors were addressed at a 'Fair' level. Finally, only 1 critical success factor (3.85%) was not addressed by the TSP ('good performance by vendors/contractors/consultants'). Tables 4 and 5 address our second research question (RQ2).

Table 6 summarises the opinions of the experts on how the TSP addressed the critical success factors for software projects. It also states the TSP scripts and processes for each of the critical success factors to justify the degree ratings. As a result, we can see why the TSP cannot be rated up to 'Best Practise' in addressing several identified critical success factors.

The summary of expert opinions in Table 6 shows that from an expert's perspective, the TSP provides an operational framework aimed at addressing 21 of the most critical success factors. These 21 critical success factors exclude user/client involvement, frozen requirement, complexity, project size, duration and number of organisations involved, good performance by vendors/contractors/consultants and end-user training

Table 4. Experts' ratings of how TSP addressed critical success factors: Final round.

	Critical success factor	Expert ratings on TSP			Std. deviation
		E1	E2	E3	
1	Clear requirements and specifications	5	4	5	0.577
2	Clear objectives and goals	5	5	5	0.000
3	Realistic schedule	5	5	5	0.000
4	Effective project management skills/methods (project manager)	5	5	5	0.000
5	Support from top management	5	5	5	0.000
6	User/client involvement	3	3	4	0.577
7	Effective communication and feedback	3	3	4	0.577
8	Realistic budget	5	5	5	0.000
9	Skilled and sufficient staff	3	3	3	0.000
10	Frozen requirement	3	2	3	0.577
11	Familiar with technology/development methods	3	3	3	0.000
12	Proper planning	5	5	5	0.000
13	Appropriate development processes/methods (process)	5	4	5	0.577
14	Up-to-date progress reporting	5	5	5	0.000
15	Effective monitoring and control	5	5	5	0.000
16	Adequate resources	4	4	5	0.577
17	Good leadership	4	4	4	0.000
18	Risk management	5	5	5	0.000
19	Complexity, project size, duration, and number of organisations involved	3	2	3	0.577
20	Effective change and configuration management	4	3	5	1.000
21	Supporting tools and good infrastructure	4	4	4	0.000
22	Committed and motivated team	5	5	5	0.000
23	Good quality management	5	5	5	0.000
24	Clear assignment of roles and responsibilities	5	5	5	0.000
25	Good performance by vendors/contractors/consultants	0	0	0	0.000
26	End-user training provision	1	1	1	0.000

Table 5. Degree to which TSP addressed the critical success factors in software projects.

Classification degree/level	Critical success factors
Best practise (5): The TSP provides a very effective framework for addressing the critical factor and has a direct impact on the software project's success.	Clear requirements and specifications. Clear objectives and goals. Realistic schedule. Effective project management skills/practises (project manager). Support from top management. Realistic budget. Proper planning. Appropriate development processes/methods (process). Up-to-date progress reporting. Effective monitoring and control. Risk management. Committed and motivated team. Good quality management. Clear assignment of roles and responsibilities.
Very good (4): The TSP provides a very good framework for addressing the critical factors and has a significant impact on the software project's success, but it may not be the most effective way of doing things.	Adequate resources. Good leadership. Effective change and configuration management. Supporting tools and good infrastructure.

Table 5. Contd.

Good (3): The TSP provides a good framework for addressing the critical factor, but there are minor missing activities that may impact the software project's success.	User/client involvement. Effective communication and feedback. Skilled and sufficient staff. Frozen requirement. Familiar with technology/development methods. Complexity, project size, duration, and number of organisations involved.
Fair (2): The TSP provides a framework that addresses the critical factor to a reasonable degree, but there are several missing activities and/or incorrect settings of priorities that impact the software project's success.	None
Weak (1): The TSP provides a framework that addresses the critical factor to a limited degree but does not cover everything that is required.	End-user training provision.
Not addressed (0): The TSP does not provide any way to address the critical factor.	Good performance by vendors/contractors/consultants.

provisions. This indicates that TSP processes cover many of the software project management aspects. In addition, the operational framework in TSP is centralised on the TSP scripts, forms, and clearly defined roles and responsibilities as well as training for all levels of management, from senior executives to middle management to team leaders to the software engineer.

In the case of clear requirements and specifications, for example, the TSP addresses requirements and specifications via the requirements process script. The purpose of the script is to produce a complete, valid, and accurate system requirements specification (SRS) and hardware or software engineering requirement specification (ERS). The process steps in the requirements process script include Market Requirements Study, Requirements Elicitation, Requirements Prototypes, SRS, User Manual Draft, System Test Plan, SRS Inspection, ERS, ERS Inspection, ERS Baseline, and requirements process post-mortem. The TSP establishes a Customer Interface Manager as one of the standard roles to be assigned to the team members. One of the main roles of the Customer Interface Manager is to establish team standards and procedures for documenting and reviewing the product requirements. This task includes (1) Leading the team in development, review and verification of the product requirements, (2) Ensuring all the product requirement assumptions are identified, documented, tracked and verified with the customer, and (3) Ensuring that the customer agrees with the requirements. In addition, during TSP launch meeting 4, the first planning step led by the Design Manager is to assess the completeness and correctness of the requirement documentation. If all these activities are performed and managed successfully by the Customer Interface Manager and the Design Manager, it will lead to clearer product requirements and specifications. These observations indicate that the TSP provides quality guidance for the team to follow.

Our study also found that the TSP did not provide any operational framework for 5 critical success factors: user/client involvement, frozen requirement, complexity, project size, duration and number of organisations involved, good performance by vendors/contractors/consultants and end-user training provision. This does not mean that the TSP ignores these critical aspects, but they are beyond the scope of the TSP.

For critical success factor number 6 (user/client involvement), there is no direct involvement from the user/client during the TSP Launch. Customer representation is generally absent from the launch itself, so it does not have an opportunity to influence the planning during the launch. The team and the coach are therefore on their own when ensuring the high involvement of the user/client in the project. The experts suggest that this can be corrected by including clients/end user representatives during the launch. If this is done, the mechanisms for ensuring their involvement are essentially the same for top management.

In respect to the tenth critical success factor (frozen requirement), the TSP does not provide any mechanism to ensure requirement freeze because it is nearly impossible to avoid requirement changes in most software projects. However, the TSP handles requirement changes very well. According to the TSP,

Table 6. Summary of expert opinions on how the TSP addressed the critical success factors.

Critical success factor	Experts' opinions	Coverage level
Clear requirements and specifications	The requirements process script (Script REQ) is used to produce a complete, valid, and accurate system requirements specification (SRS) and hardware/software engineering requirement specification (ERS). Additionally, the customer interface manager leads the team in requirement development, review and verification, and ensures that the customer agrees with the requirements. Furthermore, during Meeting 4 of the TSP launch, the first planning step led by the design manager is to assess the completeness and correctness of the requirement documentation. Scripts DEV, MAINT, and ANA also provide guidance for requirements specifications.	Best practise (5)
Clear objectives and goals	In Meeting 1 of the TSP launch, project goals and constraints are briefed to the entire team, clarified, revised, and agreed. Meeting 1 provides guidance to senior management and marketing representatives for preparing goals and objectives for presentation and discussion with the team, including answering team members' questions on goals and objectives. The entire team is made aware of the goals and constraints, producing better alignment, a more realistic assessment of the feasibility of meeting the goals and constraints, a broader sense of ownership, and ultimately a better plan. Goals are then re-stated in quantitative form. In meeting 2, the team further reviews management's goals and objectives and any future revision to the plan against the goals to ensure compliance and to eliminate unnecessary steps.	Best practise (5)
Realistic schedule	During the TSP launch, the team follows a defined estimation process to produce project estimates and considers a development strategy. Next, the team produces a quality plan to ensure that poor quality does not impact schedule and the load balances the work amongst team members, ensuring all team members contribute to meeting the schedule. If the project goals cannot be met within the schedule, feature, quality, and resource constraints are provided by management, and the team comes up with alternate plans to present to management asking for a more realistic schedule, more resources, and/or reduction in project scope. Any negotiations on scope, budget, staffing, etc. are performed in public. Commitments are documented in the minutes of the meeting to prevent the project manager from privately acquiescing to unrealistic goals or constraints under pressure from management or the client.	Best practise (5)
Effective project management skills/ practises (project manager)	TSP roles provide a framework for the team in which management responsibilities are distributed across the entire team and the project manager primarily serves as team coach or team leader. The team leader attends a mandatory three-day class called *Leading Development Teams* to teach team leaders how to lead using the TSP. Additionally, team members help with different management aspects via the eight defined team roles on a TSP team. All team members attend TSP training prior to participating in TSP teams. Developers complete a minimum of 5 days of training, titled *Personal Software Process (PSP) for Engineers*. Non-developers attend the 2-day *TSP Team Member Training.* All of this training addresses fundamental project management skills.	Best practise (5)
Support from top management	The TSP introduction strategy includes training for all levels of management, from senior executives to middle management to team leaders. The first level of support required from top management is to attend the one-day TSP Executive Strategy Seminar. Top management is trained in the concepts of the TSP and agrees to sponsor TSP by providing resources to train all levels of management as well as TSP team members. Management also provides TSP coaching services for TSP teams. Top management has to agree to meet with the teams to provide business and product goals in launch meeting 1, to be present when teams present their plan to management during launch meeting 9, to be involved in status review meetings at the end of each development cycle, and to be involved in periodic status review meetings.	Best practise (5)
User/client involvement	In the TSP, the marketing manager role represents the user/client to the team. The product manager meets with the development team to present the product goals,	Good (3)

Table 6. Contd.

	including user needs, during launch meeting 1 (Script LAU1). The customer interface manager role also focuses on the user/client, so he/she needs to understand the customer's wants and needs and to lead the team in providing a product that satisfies the customer. However, there is no direct involvement from the user/client in the project.	
Effective communication and feedback	Communication amongst team members takes place during the launches, re-launches, cycle post-mortems, project post-mortems, and weekly status review meetings. Role managers communicate by presenting role status during the team weekly meetings. Communication amongst team members and the team coach occurs on an as-needed basis and during TSP checkpoints. Communication with management happens during launches and re-launches and via status reporting (Specification STATUS). TSP is excellent at the distribution of information within the project team and between the team and sponsoring management, but there is nothing explicitly in TSP about communicating with the organisation as a whole. This is a weakness, particularly for larger projects, and TSP is very limited with respect to communications planning as well.	Good (3)
Realistic budget	In the TSP, the principal budget measure is effort hours. When management forms a project team, the number and availability of the team members represents the initial budget. Once the project is initiated via a TSP Launch, the team determines the resources it needs to do the work to meet management's business and product goals. The team uses a defined estimation process that uses historical data from the team, from the organisation, or industry benchmarks. If the resources provided by management are not adequate, the team asks management for more resources or an adjustment in the scope and/or schedule.	Best practise (5)
Skilled and sufficient staff	The TSP introduction strategy includes in-depth process training for all team members, thus ensuring some degree of process skill. Team formation guidance always specifies the formation of teams with technically skilled team members; however, this is not addressed in the TSP process scripts. In launch meetings 3 and 4, the team estimates the work to be done and determines resource availability. If sufficient resources are not available, the team produces alternate plans that adjust schedule, scope, or resources and presents the alternate plans to management. Even though the process training is very good, it does nothing to address requirements analysis, architectural design, and high-level testing. TSP also does very little to address staff acquisition and does not actually produce a staffing plan, nor does it ensure that the project manager actually knows how to staff the project. Moreover, there is no mention of planning for staffing lead time, ongoing attrition rate, or tradeoffs between overtime and additional staff.	Good (3)
Frozen requirement	The support manager is responsible for change management and configuration management issues. At the beginning of the project, the agreed requirements should be baselined and retained. If any changes are needed, they go through the Change Control Board (CCB) for review and approval. Configuration management, on the other hand, allows only authorised changes to the baselined products and makes only approved changes to the controlled version of the configuration items. However, it is difficult to avoid requirement changes in TSP, so the support manager ensures all the changes are controlled, monitored and tracked.	Good (3)
Familiar with technology/ development methodology	The TSP ensures that all team members are trained and familiar with the development methodology because TSP training is a pre-requisite for membership on a TSP team. The developers attend a minimum of 5 days of training, and non-developers attend 2 days. The TSP does nothing to address familiarity with technology, but most teams identify training needs during a launch.	Good (3)
Proper planning	The heart of the planning process is the TSP Launch, a 3- to 4-day defined planning process captured in eleven process scripts (that is, Script LAU, Scripts LAU1-LAU9, and Script LAUPM). TSP teams create several plans: the product plan is a high-level plan for the entire project, the period plan is a detailed plan for the next few weeks or	Best practise (5)

Table 6. Contd.

	months of work, and an individual plan is a detailed plan for a team member's near-term work. Teams also create a quality plan (Script LAU5), a process plan (Script LAU3), a support plan (Script LAU3), and a risk management plan (Script LAU7). Planning is an ongoing process: Small adjustments to the plan are made as needed, and major changes trigger a re-launch (Script REL). A re-launch is also scheduled at the end of each development cycle, where the team plans out the next period of work in detail.	
Appropriate development processes/ methods (process)	The TSP is a development process that has been used to develop all kinds of software: Shrink wrap, embedded, IT, web applications, commercial, defence, government, financial, game, and many other types. It is appropriate for most (if not all) development projects involving single-teams of 3 to 15 team members and multi-team projects with several hundred team members. This is augmented by scripts REQ and ANA (Requirements Management and Requirements Analysis, respectively), scripts HLD (High Level Design), IMP6 (Unit Test and Test Development), INS (Inspections), PMTD (TSP Post-mortem Test Defects), TEST (TSP Release Test), TEST1 (TSP Product Build), TEST2 (TSP Integration), TEST3 (TSP System Test), and TESTD (TSP Test Defect Handling).	Best practise (5)
Up-to-date progress reporting	The TSP employs several methods and levels for progress reporting. Schedule status is planned, tracked, and reported via a simplified earned value reporting system. Cost status is planned, tracked, and reported via effort hours. Product quality status is reported by tracking planned vs. actual defect injection and removal rates as well as other quality measures, such as defect density. Process quality is reported via measures such as time-in-phase ratios and process quality indexes. Feature status is reported via feature completion status. Status is reported at least once a week. Progress reporting is easier if the organisation adopts PSP/TSP automated tools to automatically track and analyse the project progress.	Best practise (5)
Effective monitoring and control	The team leader, the team coach, and the eight role managers monitor and control all aspects of the project (schedule, quality, cost, scope, features, and processes). Individual team members monitor and control their own work, treating individual plans as mini-projects. Data is then used to plan and track all work at all levels: individual, team, and multi-team. Senior management monitors the project less frequently than the team, on a periodic basis at the end and beginning of each project cycle and via a weekly, monthly, or quarterly status report. Overall, TSP techniques for quality control are strong, particularly in quality planning and quality management, and they explicitly provide a more structured process for status monitoring.	Best practise (5)
Adequate resources	During launch meetings 2 to 8, the team develops a plan and determines the resources it needs to do the work to meet management goals and constraints. If the resources provided by management are not adequate, the team asks management in launch meeting 9 for more resources or an adjustment in the scope and/or schedule. The team continues to monitor actual resource usage. If the team determines that they underestimated resources needed during the launch, the team re-estimates resources needed for the remaining work based on the rate of to-date completion and then asks management for further adjustments to resources, scope, and/or schedule.	Very good (4)
Good leadership	The TSP encourages the use of the word leader instead of manager. In fact, the TSP manager training course is called *Leading Development Teams.* The project manager of the team is therefore called the team leader. The TSP sets up the conditions for good leaders to thrive. It provides training as well as team leader role guidance (Specification Team Leader). The TSP teaches project managers and senior managers how to transform from managers into leaders and coaches. This model enables team leaders to lead the team in the right direction, maintain a clear and continuous focus on the team's project goals, and motivate, coach and support the team while dealing with management. The TSP places a tremendous amount of importance on leadership.	Very good (4)

Table 6. Contd.

Risk management	Risk management involves risk identification, risk categorisation, risk tracking, and risk mitigation. The TSP team assesses risks during the TSP Launch in launch meeting 7, guided by a process script (Script LAU7). Following the launch, risks are tracked during the TSP weekly status team meeting. In TSP, tracking of specific risks is assigned to individual team members based on project roles and responsibilities. The weekly meeting is guided by a process script (Script WEEK), which includes a step for risk reports from team members who are tracking project risks. The status of risks is also communicated to management as an item in regular project status reports (Specification STATUS). During re-launches, the team identifies new risks, which are in turn mitigated and tracked as needed.	Best practise (5)
Complexity, project size, duration, and number of organisations involved	TSP is most effective for teams of 3 to 12 engineers. To cater to larger projects, TSPm (Multiple TSP process) is needed, which allows multiple teams of 3 to 15 people using the TSP to work together on larger projects. A TSPm project uses special processes designed to address the additional complexity and communication issues related to larger teams. As the project grows and the number of organisations involved increases, sub-contract and procurement issues become important, but these elements are not addressed by TSP.	Good (3)
Effective change and configuration management	In TSP, the support manager is responsible for handling change in management issues. At the beginning of the project, the agreed requirements should be baselined and retained. If any changes are needed, they go through the Change Control Board (CCB) for review and approval. During the TSP Launch in meeting 3, the CCB membership is discussed and the procedure for change management is finalised at the beginning of the project. This step will ensure that the process is in place and any changes go through the right channel for approval before they are adopted and implemented.	Very good (4)
Supporting tools and good infrastructure	In the TSP launch meeting 3 (Script LAU3), the Support Manager leads a short meeting to develop a Support Plan to cater to the team’s needs. The Support Manager is responsible for overall supporting tools that are needed by the project team and ensures the team has necessary knowledge and training to use the tools. The support manager is also responsible for handling the team's configuration management and change control functions and acting as the team's reuse advocate.	Very good (4)
Committed and motivated team	TSP has been designed to establish the conditions that characterise an effective team through team-building principles. TSP facilitates team-building through a well-defined TSP launch process that has built-in activities and guidelines that can assist the team leader in effectively bringing together all the team members to begin working on the project in the most effective way. When management clearly tells the team what they want the team to do but then lets the team determine *how* they will do their work and *how long* the work will take, the result is motivated and committed teams.	Best practise (5)
Good quality management	During meeting 5 in the TSP launch, the quality manager leads the team in developing the project quality plan for a product and covers the details of how it will achieve its product quality goals. The quality manager also ensures that the team plans for defect injection and detection based on historical data, TSP quality planning guidelines or industry-published data. The plan includes specifics on where defects will be injected (defect density), what phases will catch these defects and the estimated final quality of the product. This provides good understanding for the team members on how to locate the number of defects they are injecting and finding in each phase. The developed quality plan is then followed and tracked by the team by comparing the data for any module with the quality plan. If a phase is likely to have a quality problem (that is, higher defect density), several corrective actions can be taken. In addition, every team member’s work product is reviewed and inspected by a qualified effective moderator.	Best practise (5)
Clear assignment of roles and responsibilities	During TSP launch meeting 2, the roles and responsibilities are formally defined and assigned among the team members. The team is built as a self-directed team, and they produce their own defined work processes in accordance with the established team goals. There are no clarity issues regarding who is doing what or conflicts due to overlapping responsibilities. This meeting takes particular care in distributing the roles and responsibilities across the entire team to avoid bottlenecks and promote career development.	Best practise (5)

Table 6. Contd.

Good performance by vendors/contractors /consultants	TSP does nothing in selecting, procuring and measuring vendor/contractor/consultant performance.	Not addressed (0)
End-user training provision	Although not directly addressed, the TSP addresses end-user training via team goals in which the team plans for a business or product goal that includes end-user training. During TSP Launch meeting 3, the team leader leads the team in defining a set of comprehensive work products to be produced for the project and for each of the project phases. If end-user literacy is critical in determining project success, work products, such as system prototypes, user manuals, training, and installation guides, can be listed as critical components to be delivered to the customer/end-user to ensure project success.	Limited (1)

once the requirements have been baselined, changes occur only after the impact of the change has been assessed, approved, and reflected upon in updated plans. Specifically, minor changes are handled by the team as part of their day-to-day work in maintaining their plan, and major changes in requirements trigger a re-launch, where the team systematically follows a defined process to assess the impact of the change on their plan, re-plan, and re-commit.

For the nineteenth and twenty-fifth factors, the experts highlighted that the TSP did not address the issues of sub-contracts and procurement. If there are any sizable procurements of materials, any subcontracts, or any significant outsourcing, this is a critical omission that could lead to project failure. On occasion, a significant portion of project management activities centre on managing subcontracted work, so this is by no means a minor omission. TSP assumes that the project management team has these skills and does nothing to provide them in its manager's training. If TSP is being implemented in the context of a Capability Maturity Model Integration (CMMI) based process improvement initiative, this is not a severe deficiency because project procurement management is covered by the CMMI maturity level 2 supplier agreement management process area. For organisations that are not using TSP within the context of CMMI, Project Management Professional (PMP) certification addresses this issue with the Project Procurement Management knowledge area.

We are also interested in pointing out the expert views of why the TSP could not be rated to ‘Best Practise’ in several critical success factors. Among these issues were external communications between the teams and the stakeholders, staffing plans, technology familiarity, and sub-contract and procurement issues. All these issues are part of the organisation level that is not part of the TSP. In the staffing plan, for example, TSP includes the development of a staffing profile, an inventory of skills, assignment of roles and responsibilities, and development of a training plan. These are the key inputs to a staffing plan, but they do not actually produce a staffing plan and do nothing to ensure that the project manager actually knows how to staff the project. Failure to do this effectively is an obvious cause of project failure. At the organisational level, this responsibility belongs to the human resources management area.

Similarly, with the issues of sub-contracting and procurement, TSP does nothing to specifically address selecting or managing subcontractors and vendors. However, TSP is relatively strong in addressing post mortem analysis, but it totally misses the administrative work related to closing out a contract. Both sub-contract and procurement issues belong to the procurement management area. The situation is similar for communications planning. TSP is excellent at the distribution of information within the project team and between the team and sponsoring management, but it has little to offer a project with a broad array of stakeholders. This defect is addressed to some degree by CMMI with the maturity level 2 Project Management and Control Practise area. In terms of familiarity with technology, TSP does nothing to address this issue except in the context of risk identification and mitigation planning, as most teams identify training needs during a launch.

An interesting point that is worth considering, as highlighted by one of the experts, is that a framework or a model should gain wide acceptance by the customer community. If a framework or a model cannot gain traction with the user community, it cannot have much impact, regardless of how capable or perfect it is.

Throughout this round, we noticed that the Delphi study provided good commentary and discussion channels. Although many of the same issues emerged, it was clear that the experts often focused on different angles when discussing the same critical success factors.

RESEARCH LIMITATIONS

The critical success factors identified in this research study were extracted from multiple empirical data sets and expert views from eight countries (Finland, the United States, the United Kingdom, Hong Kong, Singapore, Belgium, Australia, and Canada) concerning small to large software projects in various domains. However, the findings are applicable only to software projects. Although a sample of eight countries is small and generalisability to the entire software engineering community worldwide is problematic, we have high confidence in our research findings. Most of the articles included were taken from established scientific research journals and had a minimum of 11 citations, while a few of them were from well-known survey reports and journal articles written between 1990 and 2010 by experts and practitioners who had a wide range of experience in software-related industries. Often, factors reported in books were based on the previous work of others and did not cover the latest research findings, so we did not consider books in this study. We also decided not to include conference and workshop proceedings because it was difficult to determine the quality of the articles in such forums. Note that this research study was not intended to localise the findings; thus, we considered it irrelevant to conduct an empirical study in any particular country.

As with any Delphi-type technique, this research was limited by the fact that it employed only three experts. While experts were chosen for their vast experience in software development and software project management as well as their in-depth knowledge of software process improvement and TSP, we can make no claim about the representativeness of our sample set. The experts were not randomly chosen, but their selection was based on the quality and reliability of the set criteria. The profiles of the three experts (Table 2) indicated that all had impressive experience in the area of software project management and software process improvement and were well qualified. With our careful design and execution of the Delphi study, we have high confidence in the quality of the experts and the opinions they contributed. Also, to increase the reliability and accuracy of the experts' opinions, we required all the experts to be assisted by their colleagues with equivalent experience for discussion and validation of each of their opinions given in this study. This practise reduces personal bias and controls mistakes made by single experts. Despite the aforementioned limitations, we believe that the results have both informative and practical implications.

CONCLUSIONS AND FUTURE RESEARCH

We have reported our extensive literature survey of critical success factors that impact software projects. In this study, 43 articles were found to make significant contributions that could be analysed to develop a list of critical factors that specifically affected the success of software projects. These 43 articles consisted of 9 published sets of empirical data from case studies, 29 published empirical data sets from surveys and 5 articles written by experts and practitioners between 1990 and 2010. The method of content analysis was adopted in this study rather than the data extraction method or the frequency analysis method because some of the factors described by the authors in the articles were not explicitly clear and required careful reading, understanding and interpretation to produce accurate findings.

Based on this set of critical success factors, a five-round Delphi study was conducted to determine the degree to which TSP could address all the identified factors. Our results demonstrated that the experts agreed that the TSP provided a very effective framework for addressing 14 (53.85%) critical success factors. The experts also agreed that the TSP provided a very good framework for addressing 4 critical success factors (15.38%). Furthermore, our findings suggested that 6 critical success factors (23.07%) were addressed by the TSP at a 'Good' level, only 1 critical success factor (3.85%) was addressed to a limited degree and none of the critical success factors were addressed at a 'Fair' level. Moreover, only 1 critical success factor (3.85%) was not addressed by the TSP ('good performance by vendors/contractors/consultants').

From an expert's perspective, the TSP provided an operational framework for addressing 21 of the critical success factors. This indicated that TSP processes covered many important software project management aspects.

We were also interested in pointing out the expert views of why the TSP could not be rated 'Best Practise' in addressing several critical success factors. Among the issues not rated 'Best Practise' were external communication between the teams and the stakeholders, staffing plans, technology familiarity, and sub-contract and procurement issues. All these issues were on the organisational level that was not part of the TSP.

This research focused on the TSP, one of the Software Engineering Institutes' products. A similar approach could also be used in models other than TSP. For instance, Project Management Body of Knowledge (PMBOK), Project in Controlled Environment 2 (PRINCE2), Capability Maturity Model Integration (CMM-I), International Standard for Software Process Improvement and Capability Determination (SPICE), International Organisation for Standardisation (ISO) 9000, or other software development process models including agile processes, Rational Unified Process (RUP), for determining how these frameworks and methods addressed these critical factors.

Each framework or method on its own could not perfectly address all the identified critical success factors. By blending a software process improvement and project management framework (or any other excellent software development process model), we believe that all of the critical success factors can be more effectively addressed. Based on our initial research study, for example, it was found that TSP and PMBOK each contributed to addressing the identified critical success factors. To highlight ‘effective communication and feedback’ as an example, TSP was excellent at the distribution of information within the project team and between the team and sponsoring management, but it had little to offer beyond that. Basically, it only addressed planning status meetings within a team and with the sponsoring management without including a stakeholder identification process. By comparison, much more planning of the appropriate types of communication with each stakeholder is envisioned in PMBOK.

Communications planning becomes progressively more important with project size, and failure to perform it adequately can be a major source of risk in larger projects. The PMBOK, in contrast, offers a rich set of processes to ensure effective communication between all of the identified stakeholders and recommends that communication activity be considered from several potential perspectives. The PMBOK also calls for regular status meetings but provides very little guidance on content or how to make them effective, especially in the operational communication process for teams. We can integrate these two models, however, to complement each other and to more effectively address critical success factors for software projects. This approach is supported by several research efforts that sought integration between different areas to ensure better control in managing software projects (e.g., traditional project management and agile project management (Hass, 2007), CMMI for Development (CMMI-Dev) and PMBOK (von Wangenheim, 2010), PMBOK and Rational Unified Process (Callegari and Bastos, 2007), Agile and PRINCE (Nawrocki et al., 2006), CMMI and PMBOK (Jenkins, 2005) and many more).

A study conducted by Bayo et al. (2007) showed that there has been increasing demand in the knowledge and application of quality software to improve the country socio-economic growth. Thus, a comprehensive model is needed in ensuring the way to produce high quality software. It is our hope that the findings reported here will complement existing research in the area of software engineering, particularly in software process improvement and software project management, and will be investigated more thoroughly. Specifically, the findings provide an indication as to what extent TSP addresses the critical success factors for software projects.

ACKNOWLEDGEMENTS

We are extremely grateful to the three experts who participated in this study. We are also sincerely grateful to the Faculty of Computer Science and Information Technology, the University of Malaya and the Advanced Informatics School, Universiti Teknologi Malaysia, for supporting this research effort.

REFERENCES

Adler M, Ziglio E (1996). Gazing into the Oracle: The Delphi Method and its Application to Social Policy and Public Health, Jessica Kingsley Publishers

Ariane 501 Inquiry Board. (1996). Ariane 5: Flight 501 Failure. Report by the Inquiry Board. Paris: Ariane 501 Inquiry Board. Available at: http://www.di.unito.it/~damiani/ariane5rep.html.

Babbie ER (2010). The Practice of Social Research, 10th ed., Thomson Learning Inc.

Baccarini D, Salm G, Love PED (2004). Management of Risks in Information Technology Projects. Ind. Manage. Data Syst., 104(4): 286-295.

Battle ED (2009). Using TSP at the MSG Level. Proceedings of the TSP Symposium 2009, Louisiana, Software Engineering Institute.

Bayo ML, Ekene NS, Kenneth AC, Idowu SF (2007). Software development: An attainable goal for sustainable economic growth in developing nations: The Nigeria experience. Int. J. Phys. Sci., 2(12): 318-323.

Belassi W, Tukel OI (1996). A New Framework for Determining Critical Success/Failure Factors in Projects. Int. J. Proj. Manage,. 14(3): 141 -151.

Beynon-Davies P (1999). Human Error and Information Systems Failure: The Case of the London Ambulance Service Computer-Aided Despatch System Project. Interact. Comp., 11(6): 669-720.

Boehm BW (1991). Software Risk Management: Principles and Practices. IEEE Softw,. 8(1): 32-41.

Brooks FP (1995). The Mythical Man-Month: Essays on Software Engineering, Anniversary ed., Addison-Wesley.

Callegari D, Bastos R (2007). Project Management and Software Development Processes: Integrating RUP and PMBOK. Proceedings International Conference Systems Engineering and Modeling.

Charette RN (2005). Why Software Fails. IEEE Spectrum, 42(9): 42-49.

Clegg C, Axtella C, Damodaran L, Farbey B, Hull R, Llyod-Joness R, Nicholls J, Sells R, Tomlinson C (1997). Information Technology: A Study of Performance and the Role of Organizational Factors. Ergonomics, 40(9): 851-857.

Cohen J (1960). A coefficient of agreement for nominal scales. Educ. Psychol. Meas., 20(1): 37-46.

Cooke-Davies TJ, Arzymanow A (2002). The maturity of project management in different industries: An investigation into variations between project management models. Int. J. Proj. Manage., 21(6): 471–478.

Cramer D (1997). Basics Statistics for Social Research, Routledge

Davis N, Barbara S (2009). Experiences Using the Team Software Process at Adobe Systems. Proceedings of the TSP Symposium 2009, Louisiana, Software Engineering Institute.

Davis N, Mullaney J (2003). The Team Software Process in Practice: A Summary of Recent Results. Technical Report, Software Engineering Institute.

Delbecq AL, Van de Ven AH, Gustafson DH (1975). Group techniques for program planning: A guide to nominal group and Delphi processes, Scott, Foresman and Company.

Drummond H (1998). Riding a Tiger: Some Lessons of Taurus. Manage. Decis., 36(3): 141-146.

Dvir D, Lipovetsky S, Shenhar A, Tishler A (1998). In Search of Project Classification: A Non-Universal Approach to Project Success Factors. Res. Pol., 27(9): 915-935.

Ewusi-Mensah K (1997). Critical Issues in Abandoned Information Systems Development Projects. Commun. ACM. 40(9): 74-80.

Fairly ER (2009). Managing and Leading Software Projects, Wiley-IEEE Computer Society Press.

Fortune J, White D (2006). Framing of Project Critical Success Factors by a Systems Model. Int. J. Proj. Manage., 24(1): 53-65.

Galin D (2004). Software Quality Assurance-From Theory to Implementation, Addison-Wesley.

Glaser J (2004). Management’s Role in IT Project Failures. Healthc. Finance Manage., 58(10): 90-92.

Gray CF, Larson EW (2008). Project Management: The Managerial Process, 4th ed., Irwin/McGraw-Hill.

Gustafson DH, Shukla RK, Delbecq A, Walster GW (1973). A comparison study of differences in subjective likelihood estimates made by individuals, interacting groups, Delphi groups and nominal groups. Organ. Behav. Hum. Perf., 9(2): 280–291.

Hakim C (1987). Research Design: Strategies and Choices in the

Design of Social Research, Allen and Unwin.
Hass KB (2007). The Blending of Traditional and Agile Project Management. PM World Today, 9(5): 1-8.
Holsti OR (1969). Content Analysis for the Social Sciences and Humanities, Addison-Wesley.
Humphrey WS (1998). Three Dimensions of Process Improvement. Part III: The Team Process. CrossTalk J. Defense Softw. Eng., 11(4): 14-17.
Humphrey WS (2000). Introduction to Team Software Process, Addison-Wesley.
Humphrey WS (2002). Relating the Team Software Process SM (TSP SM) to the Capability Maturity Model for the Software (SW-CMM). Technical Report, Software Engineering Institute.
Humphrey WS (2005). Why Big Software Project Fail: The 12 Key Questions. CrossTalk J. Defense Software Eng., 18(3): 25-29.
Humphrey WS (2006). TSP: Coaching Development Teams, Addison-Wesley.
Ibbs CW, Kwak YH (2000). Assessing Project Management Maturity. Proj. Manage. J., 31(1): 32–43.
Jain M (2008). Delivering Successful Projects with TSP and Six Sigma: A Practical Guide to Implementing Team Software Process, Auerbach Publications.
Jenkins M (2005). Combining Multiple Models to Develop a Software Project Management Methodology. Proceedings International Conference Software Engineering and Applications, USA. ACTA Press.
Jiang J, Klein G (2000). Software Development Risks to Project Effectiveness. J. Syst. Softw., 52(1): 3-10.
Jiang J, Klein G, Discenza R (2001). Information System Success an Impacted by Risks and Development Strategies. IEEE T. Eng. Manage., 48(1): 46-55.
Jiang JJ, Klein G, Balloun JK, Crampton SM (1999). System Analyst' Orientation and Perceptions of Systems Failure. Inf. Softw. Tech., 41(2): 101-106.
Jones C (1995). Patterns of Large Software Systems: failure and Success. Computer, 28(3): 86-87.
Jones C (1996). Our Worst Current Development Practices. IEEE Softw., 13(2): 102-104.
Jones C (2006). Social and Technical Reasons for Software Project Failures. CrossTalk J. Defense Software Eng., 19(6): 4-9.
Kappelman LA, Mckeeman R, Zhang L (2006). Early Warning Signs of IT Project Failure - The Dominant Dozen. Inf. Syst. Manage., 23(4): 31-36.
Keil M, Tiwana A, Bush A (2002). Reconciling User and Project Manager Perceptions of IT Project Risk: A Delphi Study. Inf. Syst. J., 12(2): 103-119.
Lam SSY, Petri KL, Smith AE (2000). Prediction and optimization of a ceramic casting process using a hierarchical hybrid system of neural networks and fuzzy logic. IIE Trans., 32(1): 83–92.
Landis J, Koch G (1977). Measurement of observer agreement for categorical data. Biometrics, 33: 159-174.
Leveson NG (2004). The Role of Software in Spacecraft Accidents. J. Spacecraft Rockets, 41(4): 564-575.
Lloyd S, Simpson A (2005). Project Management in Multi-Disciplinary Collaborative Research. Proceedings of the International Professional Communication Conference (IPCC 2005), Ireland. IEEE, pp. 602-611.
Mahaney RC, Lederer AL (2003). Information System Project Management: An Agency Theory Interpretation. J. Syst. Softw., 68(1): 1-9.
May LJ (1998). Major Causes of Software Project Failure. CrossTalk J. Defense Softw. Eng., 11(7): 1-7.
Nambisan S, Agarwal R, Tanniru M (1999). Organisational mechanisms for enhancing user innovation in information technology. MIS Q., 23(8): 365–395.
Nasir MHNM, Sahibuddin S (2011). Critical Success Factors for Software Projects: A Comparative Study. Sci. Res. Essays, In Press.
Nawrocki J, Olek L, Jasinski M, Paliswiat B, Walter B, Piertrzak B, Godek P (2006). Balancing Agility and Discipline with XPrince. Lecture Notes in Comput. Sci., 3943: 266-277.
Nuseibeh B (1997). Ariane 5- Who Dunnit? IEEE Softw., 14(3): 15-16.
OGC Best Practice. (2005). Common Causes of Project Failure.
Oz E (1994). When Professional Standards are Lax, The CONFIRM Failure and its Lessons. Commun. ACM, 37(10): 29-36.
Oz E, Sosik JJ (2000). Why Information Systems Projects are Abandoned: A Leadership and Communication Theory and Exploratory Study. J. Comp. Inf. Syst., 41(1): 66-77.
Perkins TK (2006). Knowledge: The Core Problem of Project Failure. CrossTalk J. Defense Softw. Eng., 19(6): 13-15.
Pinto JK, Mantel SJ (1990). The Causes of Project Failure. IEEE T. Eng. Manage., 34(7): 305-327.
Pinto JK, Rouhiainen PJ (2001). Building Customer-Based Project Organizations, John Wiley and Sons.
Procaccino JD, Verner JM, Overmyer S, Darter ME (2002). Case Study: Factors for Early Prediction of Software Development Success. Inf. Softw. Tech., 44(1): 53-62.
Project Management Institute (2008). PMBOK: A Guide to the Project Management Body of Knowledge. 4th ed., Project Management Institute.
Project Management Institute (2009). The Growing Gap between Project Manager Supply and Demand. PMI Today Supplement, June, 2.
Project Management Institute (2010). PMI Fact File. PMI Today, May, 16.
Rainer A, Hall T (2003). A quantitative and qualitative analysis of factors affecting software processes. J. Syst. Softw., 66(1): 7-21.
Sauer C, Cuthbertson C (2003). The State of IT Project Management in the UK 2002-2003. Comput. Wkly., 15 April.
Schmidt R, Lyytinen K, Keil M, Cule P (2001). Identifying Software Project Risks: An International Delphi Study. J. Manage. Inf. Syst., 17(4): 5-36.
Schmitt JW, Kozar KA (1978). Management's Role in Information System Development Failures: A Case Study. MIS Q., 2(2): 7-16.
Seaman CB (1999). Qualitative Methods in Empirical Studies of Software Engineering. IEEE T. Softw. Eng., 25(4): 557-572.
Skulmoski GJ, Hartman FT, Krahn J (2007). The Delphi Method for Graduate Research. J. Inf. Technol. Educ., 6: 1-21.
Standing C, Gulfoyle G, Lin V, Love PED (2006). The Attribution of Success and Failure in IT Projects. Ind. Manage. Data Syst., 106(8): 1148-1165.
Standish Group International (1995). Chaos. Technical Report
Standish Group International (1999). Chaos: A Recipe for Success. Technical Report.
Standish Group International (2001). Extreme Chaos. Technical Report.
Standish Group International (2006). Chaos. Technical Report.
Standish Group International (2009). Chaos Summary 2009: 10 Laws of CHAOS. Technical Report.
Standish Group International (2010). Chaos Summary for 2010. Technical Report.
Taylor A (2000). IT Projects: Sink or Swim. Comput. Bull., 42(1): 24-26.
Taylor H (2006). Critical Risks in Outsourced IT Projects: The Intractable and the Unforeseen. Commun. ACM, 49(11): 75-59.
Tukel OI, Rom WO (2001). An Empirical Investigation of Project Evaluation Criteria. Int. J. Oper. Prod. Man., 21(3): 400–416.
von Wangenheim CG, Silva DAd, Buglione L, Scheidt R, Prikladnicki R (2010). Best Practice Fusion of CMMI-DEV v1.2 (PP, PMC, SAM) and PMBOK 2008. Inf. Softw. Tech., 52(7): 749-757.
Wateridge J (1995). IT projects: A Basis for Success. Int. J. Proj. Manage., 13(3): 169-172.
Wateridge J (1998). How Can IS/IT Projects be Measured for Success. Int. J. Proj. Manage., 16(1): 59–63.
Weber RP (1996). Basic Content Analysis. 2nd ed., SAGE Publications.
White D, Fortune J (2002). Current Practice in Project Management – An Empirical Study. Int. J. Proj. Manage., 20(1): 1-11.
Whittaker B (1999). What Went Wrong? Unsuccessful Information Technology Projects. Inf. Manage. Comput. Security, 7(1): 23-29.
Wilson D (2010). New TSP Paths: System Engineering Team uses TSP to Manage Software Test Product Development. Proceedings of the TSP Symposium 2010, Pittsburgh, PA, Software Engineering Institute.
Wingrove A (1986). The Problems of Managing Software Projects. Softw. Eng. J., 1(1): 3-6.
Yeo KT (2002). Critical Failure Factors in Information Systems Projects. Int. J. Proj. Manage., 20(3): 241-246.

Zwikael O, Globerson S (2006). From Critical Success Factors to Critical Success Processes. Int. J. Prod. Res., 44(17): 3433–3449.