PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL
FACULTY OF INFORMATICS
GRADUATE PROGRAM IN COMPUTER SCIENCE

A SOFTWARE PROCESS MODEL FOR FOLLOW THE SUN DEVELOPMENT

JOSIANE KROLL

Thesis presented as partial requirement for obtaining the Ph.D. degree in Computer Science at Pontifical Catholic University of Rio Grande do Sul.

Supervised by
Dr. Jorge Luis Nicolas Audy

Porto Alegre
2014

K93s Kroll, Josiane

A software process model for follow the sun development / Josiane Kroll. – Porto Alegre, 2014.

254 p.

Tese (Doutorado) – Fac. de Informática, PUCRS.

Orientador: Prof. Dr. Jorge Luis Nicolas Audy.

1. Informática. 2. Engenharia de Software. I. Audy, Jorge Luis Nicolas. II. Título.

CDD 005.1

Ficha Catalográfica elaborada pelo
Setor de Tratamento da Informação da BC-PUCRS

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE TESE DE DOUTORADO

Tese intitulada *"A Software Process Model for Follow The Sun Development"*, apresentada por Josiane Kroll, como parte dos requisitos para obtenção do grau de Doutora em Ciência da Computação, aprovada em 13/10/2014 pela Comissão Examinadora:

Prof. Dr. Jorge Luis Nicolas Audy - Orientador PPGCC/PUCRS

Prof. Dr. Rafael Prikladnicki - PPGCC/PUCRS

Profa. Dra. Tayana Uchôa Conte - UFAM

Prof. Dr. Cleidson Ronald Botelho de Souza UFPA

Homologada em 11/01/2015 conforme Ata No. 001 pela Comissão Coordenadora.

Prof. Dr. Luiz Gustavo Leão Fernandes
Coordenador.

To my family,

for all their support.

ACKNOWLEDGEMENTS

I would like first to thank my supervision, Dr. Jorge Luis Nicolas Audy and Dr. Ita Richardson, for all their support, advice and encouragement during my research. I also appreciate the contributions given by Dr. Rafael Prikladnicki and Dr. Sabrina Marczak, who provided valuable suggestions and recommendations for the improvement of this research.

I would like to thank to my committee members: Dr. Tayana Conte, Dr. Cleidson Souza and again to Dr. Rafael Prikladnicki for their time, interest and helpful comments.

I am grateful to all the people in, and associated with, the Systems Engineering Research Center (Centro de Pesquisa em Engenharia de Sistemas – CePES) and the PDTI program, financed by Dell Computers of Brazil Ltd. (Law 8.248/91).

I would like also to offer special thanks to the people from Lero - The Irish Software Engineering Research Centre, particularly to Sarah Beecham and John Noll for their valuable contributions to this work.

I am grateful to all the people that have participated in the various studies that I conducted for this research. Thanks to all people from Infosys Technologies Company for all their support and contributions for this research. Thanks to all experts who shared their experience and knowledge during interviews conducted in this study.

Finally to my colleagues and friends, thank you for encouraging me in all of my pursuits and inspiring me to follow my dreams. I am especially grateful to Daniela Oliveira Ferreira Do Amaral, Priscila Guarienti, Valéria Dutra, Alan Santos, and Bernardo Jose da Silva Estacio.

UM MODELO DE PROCESSO DE SOFTWARE PARA O DESENVOLVIMENTO FOLLOW THE SUN

RESUMO

Muitas organizações conduzem projetos de Desenvolvimento Global de Software para se beneficiar de sistemas, produtos e serviços de desenvolvimento mais baratos, rápidos e melhores. Organizações também buscam obter vantagens de tempo, experiência de trabalho e disponibilidade de pessoas qualificadas, onde elas estiverem localizadas ao redor do mundo. Organizações estão reestruturando as suas áreas de TI, estendendo operações para centros de desenvolvimento de software offshore. Assim, o desenvolvimento Follow the Sun é visto como uma potencial estratégia para essas organizações. O Follow the Sun visa reduzir a duração do ciclo de desenvolvimento do software ou time-to-market. Entretanto, enquanto o conceito Follow the Sun parece ser promissor na teoria, ele é difícil de ser colocado em prática. Muitas organizações de software tentaram implementar o FTS, mas abandonaram depois pela dificuldade de colocá-lo em prática. A falta de práticas e processos de software para preencher a lacuna entre a teoria e a prática é observada como uma das principais barreiras para a evolução do desenvolvimento Follow the Sun na Engenharia de Software e na indústria. Dessa forma, o principal objetivo dessa tese é desenvolver um modelo de processo de software para a adoção do desenvolvimento Follow the Sun em projetos de Desenvolvimento Global de Software. O trabalho foi dividido em três fases de pesquisa: Exploratória, Desenvolvimento e Avaliação e Evolução. Na fase Exploratória, boas práticas da literatura e lições aprendidas com a condução de um estudo de caso foram identificadas para o desenvolvimento Follow the Sun. Baseado nesses resultados, um modelo de processo de software preliminar foi construído na fase Desenvolvimento. O método validação de design e um painel com especialistas foi conduzido para avaliar o modelo preliminar na fase de Avaliação e Evolução. Como resultado, foi proposto o modelo FTS-SPM: Follow the Sun Software Process Model. O FTS-SPM compreende seis sub-processos e vinte e uma boas práticas de software. A sua adoção contribui para aumentar a probabilidade de sucesso das organizações com a implementação do desenvolvimento Follow the Sun e também para enfrentar os diferentes desafios do desenvolvimento global de software.

Palavras-chave: Desenvolvimento Global de Software; Modelo de processo de software; Follow the Sun; Gerenciamento de fuso horário; Time-to-market.

A SOFTWARE PROCESS MODEL FOR FOLLOW THE SUN DEVELOPMENT

ABSTRACT

Many companies have conducted Global Software Development projects to benefit from cheaper, faster and better software systems, products and services development. Companies also want to take advantage of time, expertise and talent pools, wherever they may be located in the world. Companies restructure their IT area by extending operations to offshore software development centers. Thus, Follow the Sun development is seen as a potential software development strategy for these companies. Follow the Sun can help with reducing the software development life cycle duration or time-to-market. However, while Follow the Sun concept looks promising in theory, it appears to be difficult to put into practice. Many software companies have tried to implement FTS, but have abandoned it after a while because of this difficulty to put it into practice. The lack of software practices and processes to close the gap between theory and practice is observed as the main barrier to the FTS evolution in Software Engineering and in the software industry. Thus, the goal of this thesis is to develop a software process model for Follow the Sun adoption in Global Software Development projects. The work was divided into three research phases: Exploratory, Development, and Evaluation and Evolution. In the Exploratory phase, best practices from the literature and lessons learned from a case study were identified for FTS development. From these results, a preliminary FTS software process model was built in the Development phase. A design validation method and an expert panel were conducted to evaluate the preliminary model in the Evaluation and Evolution phase. As a result of this process, the FTS-SPM: Follow the Sun Software Process Model has been proposed. The FTS-SPM comprises of six sub-processes and twenty-one best practices. It adoption contributes in increasing the probability of companies successfully implementing Follow the Sun and coping with the different challenges of Global Software Development.

Keywords: Global software development; Software process model; Follow the sun; Time zone management; Time-to-market.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

24HrKF	24-Hour Knowledge Factory Paradigm
BP	Best Practice
CP	Composite Persona
CVS	Concurrent Versions System
DSD	Distributed Software Development
FTS	Follow the Sun
FTS-SPM	Follow the Sun Software Process Model
GSD	Global Software Development
GSE	Global Software Engineering
IT	Information Technology
KA	Knowledge Area
PM	Project Manager
PMBOK	Project Management Body of Knowledge
SDLC	Software Development Life Cycle
SLR	Systematic Literature Review
SP	Sub-process
SWEBOK	Guide to the Software Engineering Body of Knowledge
TDD	Test Driven Development
UL	University of Limerick
XP	Extreme Programming

TABLE OF CONTENTS

1. INTRODUCTION

In the past few years, new technologies, solutions, and theories have been developed in the software engineering field. They have provided significant advances in terms of how teams should develop software. Nowadays, software is increasingly being developed by global teams. This is a new trend of producing software [RCS14]. In companies of all sizes, projects are being set up across several development sites. These sites are separated by distance, time zones, and cultural differences [DIN14]. Such a type of software development is called Global Software Development (GSD).

GSD focus on studying aspects of software development of global scale. The first report of research in the area was published around the '90s [PAS10]. Since that time, studies have been offering an understanding about GSD, but it is still considered an immature research area [DEI11].

In the software industry, many companies run globally distributed projects to benefit from cheaper, faster, and better development of software systems, products, and services [SMI10]. Furthermore, companies are implementing GSD to remain competitive in the software industry [DES10] [PRI08]. Large as small software companies are putting a lot of effort to successfully implement GSD projects [RIC10]. Several factors have contributed to the growth of this phenomenon, such as, to take advantage of time-to-market, expertise and talent pools, wherever they may be located in the world [TAN11].

The globalization of the software development enables companies to create new strategies for software development. These strategies explore the main characteristics of GSD. Temporal distance between development sites in GSD offers the opportunity to implement Follow the Sun (FTS) development [KOR14].

FTS is a special case of GSD where software development is distributed over a twenty-four hour working day [CAR11]. The FTS concept can be described as a set of two or more geographically and temporally dispersed software development teams, all working on the same phase of a project, during their appropriate day hours pertaining to their time zone. FTS main goal is to reduce the overall development time or time-to-market [CAR11]. It is an alternative for GSD projects when trying to manage temporal distances between sites.

In literature, FTS is also referenced as around-the-clock. Although these terms are used in a similar way, their definitions are different. FTS is about speeding up and cutting down project duration, while around-the-clock and others are about 24 hours coverage and running an operation in all shifts. Both of these concepts use time zone differences to design shifts, but for different purposes and with different kinds of tasks [CAR11].

Studies performed by Carmel and Espinosa [CAR11], Conchuir et al. [OCO06] and Gorton and Motwani [GOR96] discuss the FTS benefits. However, while FTS concept looks promising in theory [CAR10], it appears to be difficult in practice. Endres and Rombach [END03] argue that developing theories without sufficient evidences to their usefulness in practice do not contribute to solve problems.

Many software companies have attempted to implement FTS but they have abandoned it after a while because of their difficulty to put it into practice [VIS09]. Setamanit, Wakeland, and Raffo [SET07] argue that FTS development requires much more communication and coordination between sites. Thus, few GSD projects are able to realize the full (theoretical) benefits of FTS. In addition, if FTS is not incorrectly applied on GSD projects, it may result in failures and increase the project costs [OCO06].

It was observed that there is a great interest from the software industry in practicing FTS. Prikladnicki and Carmel [PRI13] argue that software companies are eager to adopt FTS. However, the lack of theoretical studies combined with software practices, models and process definition make its implementation difficult [CAR11] [CZE11]. Many companies have considered FTS concept to maximize development speed, but there is no rigorous empirical support for this practice [COL10]. Recently, agile methodologies have been discussed as a promise way to develop [GUP12] [CAR10]. Agile methodologies focus on simplicity and speed [ABR02]. It promotes benefits such as, increase of productivity, innovation and professional satisfaction between teams. However, a little research has been performed on it.

In Software Engineering, new approaches to close the gap between theory and practice are needed [END03]. Thus, it is necessary to define how FTS should be practiced in GSD. In this context, the research question that guides this thesis is: *"How can software be developed using Follow the Sun development in Global Software Development projects?".*

1.1 Research Goal

In this thesis, I focus on global software development projects and the issue of how to develop software adopting the FTS development. Thus, the primary goal of this thesis is to develop a software process model for the adoption of Follow the Sun (FTS) development in GSD projects.

In my research, I adopted the Sommerville [SOM11] definition for software process model, which a software process model is defined as a standardized format for planning, organizing, and running a development project. A software process model guides actions, allowing to examine, to understand, to control, and to enhance the activities that comprise the process itself [PFL04]. A software process model for Follow the Sun (FTS) will contribute to develop theories for FTS and will support software companies interested in the FTS adoption for GSD projects.

In order to achieve my primary goal, the following objectives were defined:

- To further theoretical knowledge about GSD, FTS, around-the-clock, and agile software development.
- To analyze best practices for software development in the GSD contexts.
- To identify best practices for FTS development in GSD environments.
- To identify best practices for FTS and around-the-clock development in the literature.

1.2 Motivation and Relevance

Software Engineering is concerned with developing and maintaining software systems that are reliable and efficient. Thus, concepts, strategies and practices to avoid conflicts and to improve the software development process are needed [MIS11].

A problem to be investigated and analyzed is broken into pieces to make easier to deal with and to be understood. Once a problem is analyzed, a solution is synthesized based on the analysis of the pieces. Methods, models, processes and practices are proposed to help in solving problems [PRE10].

In Software Engineering, descriptions of models, processes, and practices are important because they are prescriptions (the way software development should progress) or descriptions (the way software development is done in practice). Building a process and discussing its sub-processes helps the team to understand the gap between practice and theory [PRE10].

Software companies require support in the implementation of their GSD projects [RIC10]. Thus, GSD as a research area from Software Engineering also uses methods, tools, processes and practices to solve problems and support software projects.

In Software Engineering, the gap between theory and practice is an opportunity to propose solutions [END03]. Many of those solutions have been adopted by software companies to gain and maintain competitive advantage in the software industry [CAS09].

The main relevance of this thesis for the Software Engineering research area is a definition of a software process model for the FTS adoption in GSD environments. A software process model helps to keep a level of consistency and quality in products and services produced by different people [PFL04]. A software process model can collaborate for successful usage of the FTS approach.

In the literature, studies report little evidence of the successful implementation of the FTS in software projects. Solingen and Valkema [SOL10] argue that FTS is a promising way for software development, however well founded knowledge of its success is rare. Improving global software engineering thought improves processes and practices can ensure successful development and implementation of software projects [RIC10].

From the academic perspective, the main relevance of this study is to define a software process model with sub-processes and best practices to support FTS implementation in GSD environments. The definition of a software process model for FTS contributes to develop new theories for FTS. New studies about FTS may be developed from this study.

From the industry perspective, software companies interested in adopting FTS to develop GSD projects can use the proposed software process model. Companies can also benefit from the results this thesis by getting more information about FTS. At the end of this research, a set of best practices mapped into sub-processes will provide evidence of solutions for FTS development.

1.3 Structure of the Thesis

The remainder of the thesis is structured as outlined below. Chapter 2 presents the theoretical background of the main topics discussed this thesis. Chapter 3 introduces the research methodology followed in this thesis. Chapter 4 and 5 present results from the Exploratory phase, which is comprised of a systematic literature review in FTS (Chapter 4) and a case study (Chapter 5). Chapter 6 presents the preliminary proposed software process model for FTS development. Chapter 7 describes the results from the Evaluation and evolution phase which aimed to preliminary evaluated the proposed model. Chapter 8 presents the proposed software process model for FTS and considerations about it. Chapter 9 discusses the research objectives, the main contributions of this thesis, limitations this research, and future studies.

The six appendices in this thesis present background material on: a guideline for FTS teams applied in the Case Study (Appendix A), details of the software application developed by the FTS team in the Case Study (Appendix B), questionnaire applied in the Case Study (Appendix C), questionnaire applied in the Expert Panel (Appendix D), details and results from a SLR on GSD (Appendix E), and publications from the thesis (Appendix F).

2. THEORETICAL BACKGROUND

In this Chapter, I present the relevant background related to this thesis. Section 2.1 introduces the concept of Global Software Development (GDS). Section 2.2 presents the Follow the Sun (FTS) development concept. Section 2.3 describes around-the-clock development. Section 2.4 introduces agile software development. Section 2.5 describes key terms and definitions adopted in this thesis. Section 2.6 presents related work to this thesis. Section 2.7 summarizes this chapter.

The results of this chapter have been published in the papers: "*Researching into Follow-the-Sun Software Development: Challenges and Opportunities*" in the 6th International Conference on Global Software Engineering (ICGSE)[1] and "Desmitificando o Desenvolvimento de Software Follow the Sun: Caracterização e Lições Aprendidas" *(in Portuguese)* in the Workshop de Desenvolvimento Distribuído de Software (WDDS)[2], collocated with 2nd Brazilian Conference on Software: Theory and Practice (CBSoft).

2.1 Global Software Development

FTS development is a research topic inserted in the GSD area and it is the main topic discussed in this thesis. In order to better understand the context of this research, this section introduces the concept of GSD, its challenges and business models.

GSD has been studied since 1990 when software development starts to become global in consequence of the PC (Personal Computer) revolution [SMI10]. Since then, GSD has been cited as an important research area in Software Engineering.

In literature, GSD is referred as outsourcing, offshore, multi-site development, Global Software Engineering (GSE) or Distributed Software Development (DSD). All these terms are used as synonyms.

GSD is defined as software development with team members from multiple geographic locations [HOL06]. Team members are distributed in different places, countries or even continents. In some cases, these teams may be from the same organization. In other cases, they are formed from different organizations [SAN07].

[1] The 6th International Conference on Global Software Engineering (ICGSE) was held in Helsinki, Finland from August 15th to 18th, 2011. All papers presented at the conference venue were published in the conference proceedings by IEEE CS Press and be available in the IEEE CS Digital Library.

[2] The Workshop de Desenvolvimento Distribuído de Software (WDDS) was held in São Paulo, Brazil on September 29th, 2011. The paper is available at www.lbd.dcc.ufmg.br/colecoes/wdds/2011/003.pdf.

Any activity within the software development life cycle (SDLC) can be executed in GSD. Teams geographically dispersed can work in testing, coding, designing or any activity. There are many team settings and examples range from remote sub-teams producing specific modules of a system to teams where different functional roles such as developer or business analysis are executed at different locations [LAN08].

The adoption of GSD requires teams to collaborate over geographic and distances and also have temporal, cultural, and linguistic distances [RUA14]. Some differences are observed between GSD and traditional software development. These concepts use different strategies for software development. Some characteristics help to distinguish GSD of the traditional software development [JAB10]:

- Development sites are geographically distributed, one distant from another;
- GSD present temporal distance diversity related to different time zones between development sites;
- Socio-cultural diversity is determined by social, ethnic and cultural aspects;
- There are linguistic diversity among team members;
- Organizational diversity in terms of process maturity, work practices, and others;
- Development sites have a diversity of policies, laws and legislations.

In some cases, these characteristics may be considered challenges for GSD. On other cases, these characteristics can provide benefits for the organizations. Thus, there are many software organizations interested in taking advantages of GSD [RUA14]. Additionally, GSD seems to have become a business necessity for various reasons, including cost, scarcity of resources, and the need to locate development closer to the customers [DAM06].

2.1.1 GSD Challenges

Literature reports many challenges for GSD projects. These challenges are associated with aspects related to geographical, temporal, and socio-cultural distances [HOL06]. These aspects pose challenges mainly related to communication, coordination, and cultural differences [JAB10].

Geographical distance reduces communication frequency [JIM09]. Therefore, communication management is a critical factor for GSD projects [SHA12]. Time constraints, teams' experience, language, and cultural differences affect communication between teams. Team members may have different accents, may not respond to e-mails or they may be unavailable due to local holidays. These things make communication difficult in GSD projects. The lack of face-to-face interactions may also introduce communication issues for GSD teams [HOL06].

Different processes, time zones, and geographical locations among teams can increase coordination challenges. Team members may have different levels of experience and may have individual motivations that conflict with the goals of the project. These factors are likely to make coordination even more difficult. Casey and Richardson [CAS06] identified that coordination is negatively impacted by geographical, temporal, linguistic, and cultural distance.

Temporal, geographical and socio-cultural distances also introduce cultural-related challenges. Cultural differences become obvious when they are contrasted with different cultural norms, laws, values and assumptions as in GSD teams [RIC10]. These differences may have an effect on how people interpret a certain situation and how they react to it [HOL06]. Additionally, it becomes more challenging to develop a sense of “teamness” within a group when team members have different cultural backgrounds [RIC10].

2.1.2 Business models for GSD

There are different business models for GSD. These models differ according to the activities developed between companies and the geographical distance between them. Activities developed between companies define three business models for GSD [AUD07]:

- *Outsourcing:* a company delegates the control of one or more activities to an external company. It means that a company outsources activities or a project for an external company, which provides outsourcing services.
- *Joint-venture*: there is an agreement between two or more companies, in which through the union of resources, a new entity is created to perform one or more projects in a defined period. There is a level of control of the project and resources, reducing costs for all companies involved in the *joint venture.*
- *Wholly-owned subsidiaries or insourcing*: companies create their own software development centers. Greater control and flexibility and lower prices are the main reasons for companies to adopt this model.

Geographical distance between companies defines two business models GSD [AUD07]:

- *Offshore*: in offshore development, there is a development center that is hired by the company or the company hires services from a third company, which is located in another country and usually in another continent.

- *Onshore:* company and client are located in the same country. There are two possible configurations. Configuration first is when the entire project is developed in an office or development center from the company that is hired in the same country where the client is, but with physical distance from other clients (off-site). The second is when the project development occurs in the client site (on-site). In this case, the hired company uses its own resources to support software development activities on the client sites.

Still related to the geographical distance between companies, business models for GSD can be defined according to the demand for development in the same country (onshore) or outside of the country (offshore). The first two models presented as follow consist of the demand for development in the same country (onshore) and the last two, software development is done outside of the country (offshore) [AUD07] [CAR05] [ROB04].

- *Onshore insourcing* or *internal domestic demand*: consists in a department inside the company or a subsidiary company in the same country (onshore), which provides software development services through internal projects (insourcing);
- *Onshore outsourcing* or *outsourcing*: is based on an agreement with a third company (outsourcing) to develop a specific software product or a service. The third company is located in the same country as the main company that makes the services agreement (onshore);
- *Offshore outsourcing* or *offshoring*: provides an agreement with a third company (outsourcing) to develop specific software products or services, and the third company is located in another country from the contractor (offshore), often in another continent;
- *Offshore insourcing or captive / internal offshoring*: indicates the creation of a subsidiary of the company itself to provide software development services (insourcing). This subsidiary is located in a different country from the main company or contractor (offshore), usually in another continent.

A simplified matrix of GSD business models described above is presented in Figure 1*.*

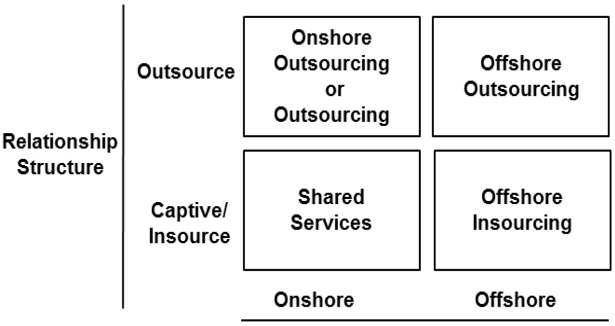


Figure 1 - Simplified matrix of GSD business models, adapted from Audy and Prikladnicki [AUD07].

This research adopts offshore insourcing model. I have chosen to address this model in order to contribute to the current demand from the software industry. It is important to highlight that characteristics of the offshore insourcing model contributes to FTS development [CAR11]. For example, internal collaboration with several sites distributed in different time zones. These sites attend needs of the company using its own resources, located in a different country. The adoption of offshore insourcing model allows companies to make business decisions such as selecting distributed sites, geographical locations, and an organizational structure.

2.2 Follow the Sun Development

FTS development is applied in the context of GSD in order to take advantage of the temporal distance between development sites located in different time zones [SOL10]. FTS is uniquely focused on speed of development. Its main purpose is the reduction of the software development life cycle duration or time-to-market [CAR11]. FTS does not offer other advantages besides decreasing the development life cycle duration. It is applied to software projects when a software product needs to be developed quickly and the cost is irrelevant to the client [CAR10].

Since team members are distributed across multiple time zones, organizations can continuously develop software twenty-four hours. Thus, the time reduction may be theoretically by 50% if there are two sites and by 67% if there are three sites [CDE09]. Solingen and Valkema [SOL10] found that when the number of sites in a daily cycle increases, on average, the overall working speed of the sites also increases.

At the beginning and at the end of each working day shift there is a handoff. *Handoff* is a term adopted in the literature to define the process transition from one site to another [CAR11]. Handoffs are performed on a daily basis to present a status update and to pass on unfinished tasks from one site to another. The next site will take these tasks in order to start its working day shift [VIS09].

Handoffs create dependencies between sites [CAR10]. The team that will be starting the working day shift depends on the status update and project source from the last production site. Handoffs' management is considered as one of the main challenges for implementing FTS projects [CAR11] [SOO08].

Carmel, Dubinsky, and Espinosa [CDE09] argue that there are many challenges to put FTS in practice. These challenges are related to communication, coordination, and cultural differences. Figure 2 shows the main challenges for FTS implementation.

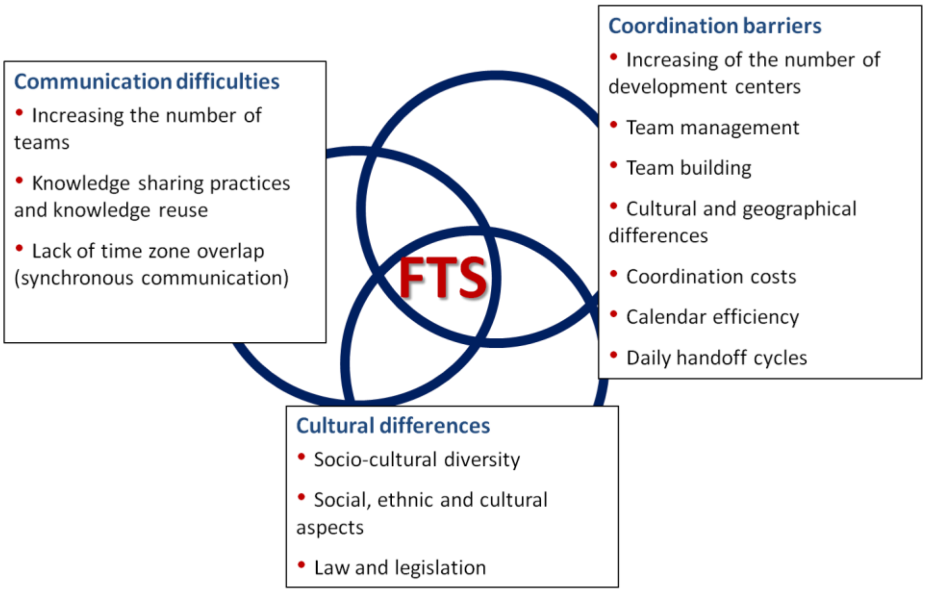

Communication difficulties
• Increasing the number of teams
• Knowledge sharing practices and knowledge reuse
• Lack of time zone overlap (synchronous communication)
FTS
Coordination barriers
• Increasing of the number of development centers
• Team management
• Team building
• Cultural and geographical differences
• Coordination costs
• Calendar efficiency
• Daily handoff cycles
Cultural differences
• Socio-cultural diversity
• Social, ethnic and cultural aspects
• Law and legislation

Figure 2 - FTS implementation challenges.

FTS challenges are associated one to another as indicated in Figure 2. Thus, a challenge can increase or decrease the likelihood of happing another challenge. The coordination barriers challenges are mainly associated with the increasing of the amount of development centers [SOL10]. When more than one development center is added to the project, this increases the difficulties to coordinate aspects that involve team management, and cultural and geographical differences [CDE09]. For example, the coordination of the continuity of work involves daily handoff cycles among teams. The handoffs are hard to coordinate due to the difficulty of resolving task issues across sites/shifts, and the cross-site coordination cost will most likely be positive and nontrivial.

The increase of the amount of development sites also adds difficulties to communication. These difficulties occur due to the increasing of the number of teams allocated to the project and consequently loss of communication richness [CDE09]. The communication challenges in FTS are associated mainly to the lack of synchronous communication between distributed teams [SET07].

Cultural differences challenges are associated to the socio-cultural diversity present in FTS projects. Such differences include social, ethnic and cultural aspects. Holmstrom et al. [HOL06] argue that when constraints, such as temporal, geographical and socio-cultural distance are identified in the scope of organizational operations, these constraints result in challenges for FTS. For example, the usual problem of supporting collaboration includes language and cultural diversity. If one culture has more emphasis on self-sufficiency, team members tend not to ask for help when problems come up. Yap [YAP05] argues that cultural differences often created misunderstandings and lead to frustration and conflicts between teams.

2.2.1 Studies on FTS

FTS studies have conducted following the case study and controlled experiment methods. Case studies report the use of FTS in the software industry. Controlled experiments have been conducted to create FTS scenarios and to test hypothesis. Case studies performed in the industry are presented below:

- *Carmel* [CAR99]

The first report experience of FTS in the software industry was reported by Carmel [CAR99]. IBM decided to develop a project using FTS. In this project, five teams were defined and distributed into five development centers located in five countries. In this project, IBM faced many coordination issues, especially during daily handoffs. Because FTS was not bringing the expected results and several problems were being faced, those responsible for the project dropped off using FTS to accelerate the development process, keeping only the GSD setting.

- *Ramesh and Dennis* [RAM02]

This study describes a FTS software project between two teams in the United States and India. The main challenges observed in this project are related to coordination. To Ramesh and Dennis [RAM02], the person responsible for the project experienced difficulties in managing activities over 24 hours. Due to time constrains, those responsible for the project and developers did not document changes in the code and files since the last download.

Teams used phone calls and e-mails to communicate, but many issues related to lack of synchronous communication were faced during the project. Despite the challenges, FTS was considered feasible to develop this project and teams were able to finish the project in the estimated time.

- *Yap* [YAP05]

Yap [YAP05] describes the experience using XP (Extreme Programming) to develop a FTS project. In this study are presented challenges, lessons learned and solutions for global continuous integration such as, cultural differences and conflicts between sites. This study focuses mainly on XP distributed development using a FTS project scenario.

- *Holmstrom et al.* [HOL06]

This study gives details of a case study at HP (Hewlett-Packard) and describes the Intel case. To Holmstrom et al. [HOL06] HP used FTS to solve issues during the development phase of the software life cycle and to support teams between Monday and Friday. Intel never applied FTS neither had plans to apply it. To Intel experts, FTS is unfeasible for software development.

- *Treinen and Miller-Frost* [TRE06]

Treinen and Miller-Frost [TRE06] report two case studies at IBM. Their first case study described a software project involving development sites in the US and Australia. In this case study, two geographically distant development teams were merged into one cohesive team for FTS development. This project was considered a success. The second case study involved three distinct projects with sites in the US and India. In these projects, teams worked on the same code base. Due to team’s inexperience, time constraints and project budget, the projects were dropped out. Several challenges related to time zone issues, different development configurations, project cost estimates, and team's cultural differences contributed to the failure of these three projects.

- *Carmel* [CAR06]

In 2006 Carmel reported a study of FTS by Infosys. However, it was not considered a real case by Infosys. Infosys uses GSD concept, but according to his managers never developed projects using FTS.

- *Other studies*

Other studies [TAN11] [GOR06] [TAW06] [GOR97] present case studies discussing FTS characteristics. These studies discuss mainly models, tools and time zone issues.

Controlled experiment studies have been explored the main FTS characteristics. These studies are described as follows.

- *Hess and Audy* [HES12]

This study proposes a process for daily handoffs. The handoff process aims to alleviate difficulties faced during the development phase of FTS software projects. The proposed handoff process is based on Composite Persona (CP) and 24hr Design and Development concepts [DEN08]. In addition, it uses Test-driven development (TDD) technique and stand-up meeting guidelines from Scrum method to perform meetings.

A controlled experiment was performed to evaluate the process efficiency. Results show that is possible to reduce development difficulties in FTS using the proposed process.

- *Solingen and Valkema* [SOL10]

Solingen and Valkema [SOL10] conducted a controlled experiment to measure the impact of the number of sites in a daily cycle in terms of overall working speed, individual working speed and working accuracy (defects). Results indicated that when the number of sites in a daily cycle increases, also on average the overall working speed of sites increase, but it becomes harder to measure the working accuracy

- *Carmel, Dubinsky, and Espinosa* [CDE09]

In 2009, Carmel, Dubinsky and Espinosa performed a quasi-experiment aiming to measure working speed in FTS between collocated and distributed teams. This study followed practices from agile methodologies [CDE09]. Findings showed that teams using FTS approach are faster than collocated teams. In this study the time spent with the development increased by 10%, but for the authors, the time development increase could be even higher.

- *Espinosa, Nan, and Carmel* [ESP07]

Espinosa, Nan and Carmel [ESP07] conducted a controlled experiment to investigate the impact of time zone overlap on speed and accuracy of software development. In this study, 42 pairs of students were selected to simulate a FTS project. This study showed that development speed is higher at both full and zero overlap, but not in different fractions of overlap between sites.

Case studies report many difficulties to implement FTS in GSD projects. These difficulties are mainly related to communication and coordination issues. I observed the lack of documented processes and best practices for the FTS adoption. Controlled experiment studies investigate daily handoffs, number of sites in a daily cycle, working speed, and the impact of time zone overlap on speed and accuracy in software development. These studies investigate the main FTS characteristics and project settings.

2.3 Around-the-Clock Development

Around-the-clock development offers the promise of reducing development cycles by increasing the amount of time during the day that software is being developed [HER00]. As teams are distributed in different development sites there are cost advantages. Companies can profit from the low labor rates by adopting around-the-clock [LAM09].

The around-the-clock concept is recommended for all SLCD phases while FTS is only for the development and the testing phases [CDE09]. There are some different features that help to distinguish around-the-clock of FTS. Table 1 summarizes these differences.

Table 1 - Around-the-clock and FTS comparative table.

Features	FTS	Around-the-clock
The main goal is the reduction of the software development life cycle duration or time-to-market	[CAR10]	None
Each site works in the same software development phases and tasks	[CAR11]	None
Geographically distributed teams are located in different time-zones and production sites	[VIS09]	[MAT09]
The project settings enable development 24 hours per day	[CAR10] [SOO08]	[MAT09]
Handoffs occur on a daily basis and in a standardized way	[CAR10]	[CAR10]
Handoffs creates dependencies between production sites	[CAR10] [SOL09]	None
Coupling level: Interdependency between tasks (*shifts*)	(High) [CAR10]	(Low) [WAK08]
Any point of time only one site has the product	[CAR10]	None
The knowledge obtained during the product construction belongs to the last production site	None	[CAR10]
Its features enable its application in all phases of the SDLC	None	[WAK08]

There are six main differences between around-the-clock and FTS, as shown in Table 1. The first difference is related to the purpose of each concept. As described in section 2.2, FTS is about speed and it does not offer other advantages, while around-the-clock also provides the opportunities to reduce costs of the project. The second and the most significant difference between these two concepts is that in FTS each site works in the same software development phases and tasks, while in around-the clock each site works in different software development phases and tasks. Thus, handoffs do not create dependencies between production sites in around-the-clock, (third and fourth difference). Finally, the fifth and sixth difference is related to teams' knowledge and software phases of the SDLC. Unlike FTS, around-the-clock' teams develop individual tasks and not sharing it with other sites. Thus, the knowledge obtained during the product construction belongs to only one production site. Therefore, around-the-clock is recommended for all phases of the SDLC, whereas FTS is recommended only for coding and testing. FTS can be applied to other phases, but it may be not result in time reduction or its application becomes difficult.

In this thesis, the main research topic is FTS development. I have followed the definition given by Carmel and Espinosa [CAR11]. However, due to some inconsistencies in the literature related to the use of terms and definitions, I also investigate around-the-clock studies to ensure that any study of interest was covered by my research.

2.4 Agile Software Development

Agile software development is discussed as a promising approach to implement FTS [GUP12] [CAR10]. Furthermore, agile software development is gaining popularity in the software industry. It is a reaction to traditional methods of developing software and acknowledge [COH04]. However, unlike traditional methods, agile methodologies employ short iterative cycle, and rely on tacit knowledge within a team as opposed to documentation [ABR02].

A core idea in agile software development is that a team can be more effective in responding to change if it can reduce the cost of moving information between people and the elapsed time between making a decision and seeing the consequences of that decision [COC01]. Agile software development is characterized by four attributes: incremental, cooperative, straightforward, and adaptive [ABR02]:

- *Incremental* refers to small software release, with rapid development cycles [ABR02]. Agile methodologies define short interactions during the working day. There is a high level of collaboration between teams, which leads to reducing the project's documentation [SMA10]. Additionally, agile methodologies enable a robust engineering from self-organizing teams [SUT07]**.**
- *Cooperative* refers to a close customer and developer interaction. In agile projects, communication performed between clients and development teams involve a high-level interaction. Face-to-face communication is considered the best way to exchange information between team members [ROB08]. Additionally, agile methodologies are focused on creating a better collaboration between clients and development teams. Its adoption helps establish deadlines without budget constraints to the project delivery.
- *Straightforward* implies that the method itself is easy to learn and to modify is sufficiently documented.
- *Adaptive* means the ability to react to changing requirements, even late in the development process.

Jalali and Zlatkovic [JAL10] also define informal communication as another characteristic of the agile software development. In order to reduce the documentation and become the software development process iterative and adaptive, agile teams perform informal communication.

There are several agile methodologies as follows: Extreme Programming (XP), Scrum, Crystal Methods, Feature Driven Development, Lean Development, and Dynamic Systems Development Methodology. These methodologies have much in common, (e.g.) what they value, but they also differ in the practices suggested.

Agile software development for GSD is called Distributed Agile Development (DAD). In the GSD literature, practices from Scrum and XP methods are the most discussed for GSD and FTS development. Scrum aims to increase the development speed, enabling to align individual and organizational goals, creating an organizational culture driven by performance, supporting stakeholders and consistent communication at all levels and individual development [SUT07]. XP provides effective communication between team members. It is recommended for software development with small teams that need to develop software quickly and in an environment with fast changing of requirements [BEC04]. Practices from XP do not include preparation of extensive requirements or design documents before the development cycle starts. However, XP defines continuous communication between stakeholders.

2.5 Key Terms and Definitions

Software process is defined as a set of tools, methods and practices, used to produce a software product [SOM11] [HUM89]. A software process is also defined by Pressman [PRE10] as a framework for a set of key-areas of processes (KPA), which could be used for an effective delivery of software engineering technologies. Key-areas from processes make a base to establish the context to apply methods, tools and techniques. All software processes have the following characteristics [PFL04]:

- Processes describe their main characteristics;
- Processes use resources and have a set of restrictions;
- Processes produce final and intermediate products;
- Processes are composed of sub-processes related to themselves. It can be considered a hierarchy in which each sub-process has its own process model;
- Each activity from a process has an input and an output criteria describing when it start and finish;
- Activities are organized in sequence to preserve relations between activities;
- Each process has a set of guidelines, that aim to explain the goals for each process activity;
- Restrictions and controls can be applied to any activity, resource or product. For example, budget or schedule may limit the time performing an activity, using a tool or defining resources to be used.

In the GSD area, terms such as process, model, method, and methodology are missed or misinterpreted or even used as synonymous. Although these terms have different definitions, they are used as the same meaning. Thus, I analyze these terms in order to justify the term used in this research.

In studies performed by Carmel, Espinosa, and Dubinsky [CAR10] and Carmel, Dubinsky, and Espinosa [CDE09], I found terms such as approach, method and structured process. The “approach” term is used to refer to agile practices. “Method” and “methodology” are used as synonymous. Carmel, Espinosa, and Dubinsky [CAR10] describe “methods” as software development methodologies. “Structured process” is defined as a practice for FTS implementation.

Richardson et al. [RIC10] developed a software process approach for GSD. In this study, the software process approach was called Global Teaming (GT). The GT has a structure similar to CMMI (Capability Maturity Model Integration) and it can be used as a support mechanism for GSD projects. It was developed based on twenty-five success factors from case studies and specifics practices and sub-practices. In this study, the authors adopted the software process definition given by Humphrey [HUM89]. To Humphrey [HUM98], software process is a set of the tools, methods and practices used to produce a software product.

Denny et al. [DEN08] also used the software process term. In this study, the authors proposed an agile software process. Based on CPro process (cooperative working model called Composite Persona (CP)), a Multimind tool was developed. Multimind is a collaborative tool with a rich semantic environment for developers collaborating with CP method. CPro is an agile software process to improve the performance of CP method [DEN08].

Richardson and Casey [RIC08] propose a structured approach for GSD. The approach term is used with the same meaning as a model. This approach is based on ten key factors from a case study and it is inspired in the IDEAL (Initiating, Diagnosing, Establishing, Acting and Learning) model.

The “approach” term is also used in studies performed by Cameron [CAM04] and Lucca, Di Penta, and Gradara [LUC02]. Cameron [CAM04] defines approach as a set of methods and techniques. On the other hand, Lucca, Di Penta, and Gradara [LUC02] use the approach term to describe how to perform the maintenance process.

In this thesis, I adopted the software process model term. I have decided to adopt this term because its definition is associated with goals and results expected from this thesis. Studies performed by Richardson et al. [RIC10], Richardson and Casey [RIC08], and Denny et al. [DEN08] contribute to justify my choice. Additionally, I have discussed the adoption of this term with research experts from Lero - The Irish Software Engineering Research Centre during my scholarship (see Section 3.3).

2.6 Related Work

There only a few studies in literature discussing FTS development. Studies are mainly focused in FTS characteristics and challenges. The lack of studies discussing practices and processes to implement FTS is one of the factors that motivated me to perform this study.

In this section, I present the related work to this thesis, both to point out the contributions of previous research and to place my contributions in the proper context. Next, I summarize the main related work.

- **Richardson et al. [RIC10]**

Richardson et al. [RIC10] conducted an extensive global software engineering research to develop a software process area, which is called Global Teaming (GT). This software process area includes specific practices and sub-practices which detail specific recommendations for organizations that are implementing GSD. Its goal is to ensure that requirements for successful global software engineering are stipulated so that organizations can ensure successful implementation of GSD.

The GT software process was developed based on results from three case studies. These case studies were undertaken over a 9-year period in the area into GSD. The authors completed a study that identified 25 factors to be taken into account when setting up virtual teams in a global environment. Based on this outcome, they developed the GT structure similar to the structure of the CMMI.

Global Teaming has two specific goals, each of which has specific practices and sub-practices. Each sub-practice was included in the GT process area based on the authors' research (case studies) and on the research of others (referenced). The GT process area presented in this study provides specific goals, specific practices, sub-practices and guidelines which can be used by industry who are implementing a GSD strategy.

- **Hess and Audy [HES12]**

Hess and Audy [HES12] proposed a software process focused on daily handoffs management, called FTSProc. The proposed process aims to alleviate difficulties faced by teams during the development phase of FTS software projects. It was built based on the Composite Persona (CP) [DEN08] and on the 24hr Design and Development [FAD00] concepts. In addition, it uses the test-driven development (TDD) technique and stand-up meetings practices from the Scrum methodology.

In order to evaluate the FTSProc's efficiency, a controlled experiment was performed. The objective of the experiment was to compare two projects, one called Adhoc and another named FTSProc. The findings from this study showed that it is possible to reduce development difficulties in FTS using the proposed process.

This study presents two main contributions. The first main contribution is for the theory area, the creation of a software process for the work handoff during the development phase. The second main contribution is for the software industry in which this research contributes to increasing the productivity gain, since the created process facilitates the use of the FTS strategy for the development phase, thus decreasing the time spent during this phase of the SDLC.

- **Denny et al. [DEN08]**

Denny et al. [DEN08] explored the utilization of agile practices for 24-Hour Knowledge Factory (24HrKF) environments. They aim to search for solutions that enable handoffs to be practiced effectively. Thus, this study describes the CPro process. The core of CPro is a model of cooperative work called the Composite Persona (CP). A CP is a highly cohesive team of developers that are distributed across the globe. When a problem is decomposed, it is decomposed horizontally as is conventional. However, subcomponent development is now assigned to a CP rather than an individual team member. The members of the CP work on the vertically decomposed subcomponent in series, each successive shift improving upon the work of the previous shift [DEN08]. This work contributed to design a tool, Multimind. The main motivation in performing the study was by the interest in new distributed and agile process, in especially for 24HrKF environments. Denny et al. [DEN08] claim similar methods for FTS in this study. Additionally, they claim that many companies have not implemented FTS due to difficulties related to coordination, communication, and cultural differences.

- **Yap [YAP05]**

Yap [YAP05] also discusses agile methodologies, but with a different purpose. This study describes the use of XP (Extreme Programming) to develop a globally distributed, around-the-clock software development project. Denny et al. [DEN08] performed their study inspired in Yap's work [YAP05]. In addition, Carmel, Espinosa, and Dubinsky [CAR10] claim the use of agile methodologies for FTS as promising practices.

In Yap' study [YAP05], a programming team was distributed across three sites (US, UK, and Asia) and they used collective ownership of code. Only one of the three sites had previous knowledge on XP. The other two were coached on XP practices prior to the collaboration. These two sites believed that the first site had an advantage due to its previous experience with XP. The team used Virtual Network Computing (VNC) and video conferencing to facilitate communication. Handoffs consisted of a daily work summary. The team faced many problems related to cultural differences, different level skills, and conflicting priorities across regions. At the end of this study, the author concludes that XP works for a globally distributed group performing around-the-clock continuous development with a shared code base.

- **Carmel, Espinosa, and Dubinsky [CAR10] and Carmel, Dubinsky, and Espinosa [CDE09]**

Studies performed by Carmel, Espinosa, and Dubinsky [CAR10] and Carmel, Dubinsky, and Espinosa [CDE09] discuss mainly FTS definition, characteristics, and challenges. The first study provides a conceptual foundation and a formal definition of FTS. The authors analyze the conditions under which FTS can be successful in reducing the duration in software development. Based on fundamental issues surrounding FTS, they developed twelve research propositions, such as: calendar efficiency, development method, product architecture and hand-off efficiency, within-site coordination, cross-site coordination, and personal productivity. This study combined the conceptual analysis with a description of a FTS exploratory comparative field study to draw out key findings and learning. The main goal this study was to address FTS issues and provide a conceptual framework to guide further studies of FTS. The second study, presents the details of the FTS concept and the outcomes of a first quasi-experiment designed to test FTS and measure the speed of software work. The main goal this study was to investigate FTS in an agile environment. Specifically, the author measured development duration in order to test the central premise of time-to-market reduction. Findings from this study showed an approximate 10% reduction in development duration rather than the theoretical 50% of FTS. This quasi-experiment is part of the research to explore FTS. Both studies argue that the agile methodologies have some characteristics that assist in structuring FTS settings.

2.7 Chapter Summary

In this chapter, I presented concepts covered in this research which includes GSD, FTS, around-the-clock, agile software development, key terms and definitions, and related work to this thesis. In the GSD literature, I found many challenges for the implementation of distributed projects. These challenges are related to communication, coordination, and cultural differences. Since FTS is a topic in the GSD area, it shares many characteristics as well challenges for its implementation. Moreover, I identified characteristics of the offshore insourcing model that contributes to FTS development. Thus, this research adopts offshore insourcing model.

Studies report that several companies are looking for increasing their efficiency in software development and FTS could be an opportunity for them. However, a few studies have been discussed FTS and presented successful cases of its implementation. In literature, agile software development is cited as a promising for FTS practice. There is still little evidence of its benefits for FTS. These findings are important for this research because I can further analyze how take advantages of agile software development for FTS.

This chapter highlighted the importance of better understanding FTS and developing solutions for its implementation. In increasing the research in this area, it might contribute to the FTS adoption by companies.

3. RESEARCH METHODOLOGY

In the previous chapter, I described the theoretical base of this research that was important to better understand FTS theory. In this chapter, I discuss aspects of the research methodology followed in this thesis. First, I provide information about the research context and research methods followed in this thesis (Section 3.1). Then, I describe the research design (Section 3.2). In section 3.3, I present details of a research visit, an internship and a scholarship. I summarize this chapter in section 3.4.

3.1 Methodological Background

This research is exploratory. Exploratory research enables to define a problem and to formulate hypotheses about the topic under study [YIN02]. It aims to examine an issue or a problem understudied, which has not been discussed previously by other studies [SAM91]. In addition, this is where a researcher has an idea or has observed something and seeks to understand more about it.

Exploratory research is important to research methods because it helps define a new problem or question. Furthermore, exploratory research enables the researcher to choose a collection of data collection techniques to perform a study.

In this research, I decided to use as main research methods: systematic literature review (SLR), case study, and expert panel. I summarize each research method adopted in this research as follows.

3.1.1 Systematic Literature Review

The Systematic Literature Review (SLR) term is used to refer to a specific methodology of research, developed in order to gather and evaluate the available evidence pertaining to a focused topic [BIO05]. A SLR (often referred to as a systematic review) is a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest [KIT07].

SLRs are one of the key building blocks of evidence-based software engineering and the interest in conducting such reviews within Software Engineering is clearly growing [SDJ07]. Kitchenham and Charters [KIT07] discuss many reasons for undertaking a SLR. The most common reasons are:

- To summarize the existing evidence concerning a treatment or technology, e.g. to summarize the empirical evidence of the benefits and limitations of a specific agile methodology.
- To identify gaps in current research in order to suggest areas for further investigation.
- To provide a framework/background in order to appropriately position new research activities.

The adoption of SLR as a research method in this study was important to identify studies performed in FTS, the state-of-art of research in FTS, research gaps and challenges for its implementation. Furthermore, the empirical evidence from a SLR provides the information about research findings and terms and definitions used in the current research area.

3.1.2 Case Study

A case study method is defined as an empirical inquiry that investigates a contemporary phenomenon within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident [YIN02]. Case study designs can be single-case or multiple-case studies, and they can involve a single unit (holistic) or multiple units (embedded) of analysis [YIN02].

In Software Engineering, case studies are important for the industrial evaluation of Software Engineering methods and tools, because they can avoid the scale-up problems that are often associated with experiments [SDJ07]. Furthermore, case studies offer in-depth understanding of how and why certain phenomena occur, and can reveal the mechanisms by which cause-effect relationships occur [EAS08].

Case studies are typically Exploratory or Confirmatory. Exploratory case studies are used as initial investigations of some phenomena to derive new hypotheses and build theories, and confirmatory case studies are used to test existing theories [EAS08].

There is a variety of different data sources used in a case study research. Qualitative data, including interviews and observation, play a central role, as they offer rich insights into the case. Quantitative data involve numbers and classes. The use of quantitative data in case studies may depend, among other things, on the phenomena under study, the research questions formulated, the type of case study, and the sources of evidence used [RUN12]. However, data collection is always performed with respect to a well-defined unit of analysis.

In Software Engineering, the unit of analysis might be a company, a project, a team, an individual developer, a particular episode or event, a specific work product, etc. [EAS08].

In the context of this research, a case study was performed during an internship at Infosys Technologies Company. Section 3.3.2 provides the information about the internship and Chapter 5 presents results from the case study.

3.1.3 Expert Panel

An expert panel is an exploratory study that looks at the strengths and weaknesses of a process, model, method, technique, or practice [BEE05]. It involves a group of experts recognized in at least one of the fields addressed by the research under evaluation. Data from an expert panel helps to construct models and to validate them [SHE01]. Expert panels are particularly appropriate for issues that require highly technical knowledge and/or are highly complex and require the synthesis of experts from many different disciplines [SLO05].

The value of expert knowledge is recognized by the capture expert judgment, the ability of experts to predict techniques to prevent requirement defects and in their analysis of the accuracy of several methods of estimating project effort [BEE05]. It reaches conclusions and recommendations through consensus [SLO05].

In this thesis, I conducted an expert panel to evaluate the proposed software process model for FTS. Similar research method has been used by Beecham and Hall [BEE03] for the validation of requirements process improvement model and by Deshpande [DES13] for the validation of a coordination model for GSD teams. I have followed the recommendations given by Slocum [SLO05] to conduct the expert panel method.

3.2 Research Design

The research design was organized in three main research phases: (1) Exploratory, (2) Development, and (3) Evaluation and Evolution. Figure 3 summarizes the selected methods and the timeline for the execution of this study. Each phase is described in details next.

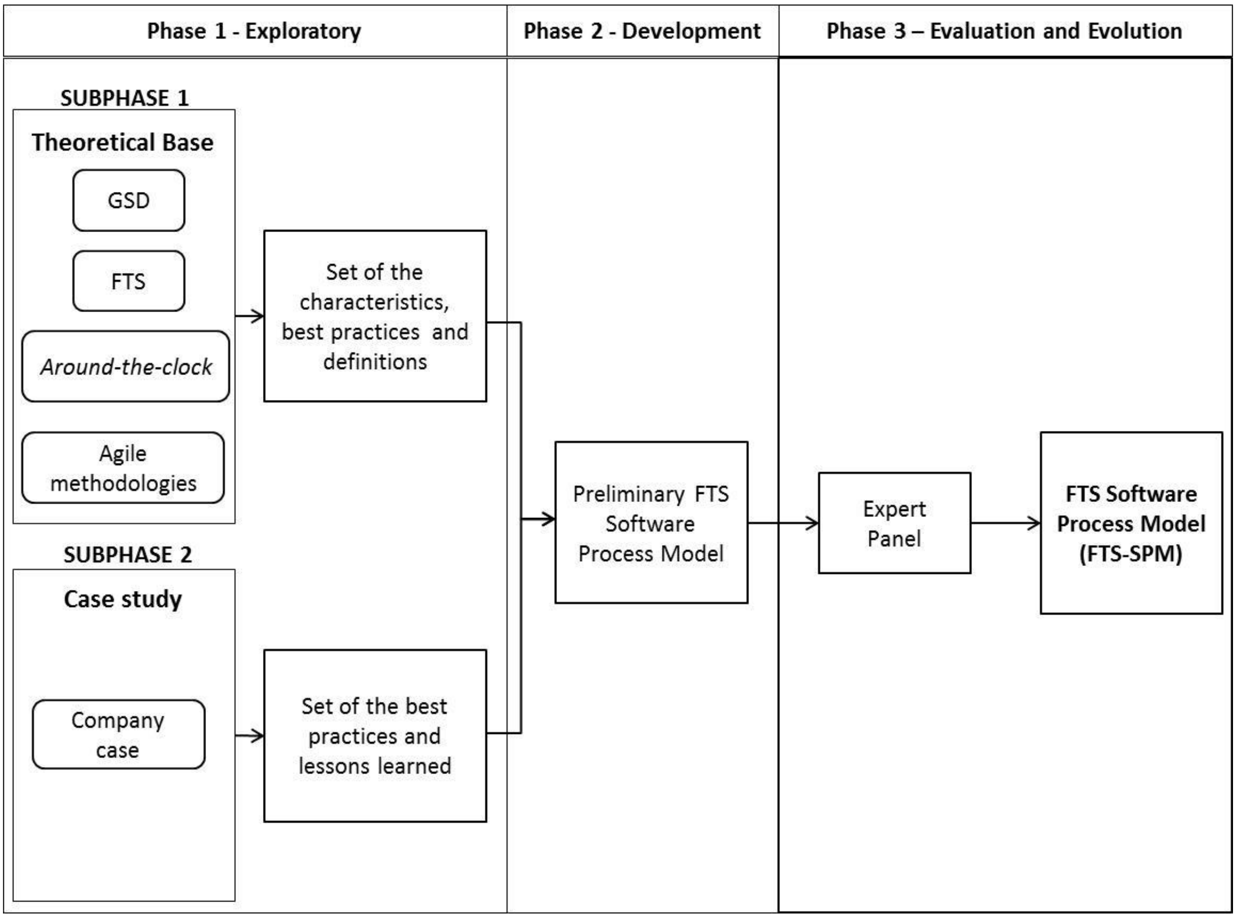

Phase 1 - Exploratory
Phase 2 - Development
Phase 3 – Evaluation and Evolution
SUBPHASE 1
Theoretical Base
GSD
FTS
Around-the-clock
Agile methodologies
Set of the characteristics, best practices and definitions
SUBPHASE 2
Case study
Company case
Set of the best practices and lessons learned
Preliminary FTS Software Process Model
Expert Panel
FTS Software Process Model (FTS-SPM)

Figure 3 - Research design.

Phase 1- Exploratory: This phase was divided in two subphases as shown in Figure 3. I describe each of these subphases next.

SUBPHASE 1*:* In this subphase, I conduct a review of the theoretical basis, which involved conducting a SLR on FTS. The SLR method was previously described in section 3.1.1. The purpose of this subphase was to build a set of characteristics, best practices, and definitions for the preparation of a process model for FTS.

I have followed the recommendations provided by Kitchenham and Charters [KIT07] to conduct the SLR. This study aimed to identify best practices for FTS development. Chapter 4 presents results from this subphase.

In this subphase, I also had the opportunity to discuss this research during a research visit to two research groups of the study area led by Professor Erran Carmel (American University) and Professor Amar Gupta (University of Arizona). The purpose the research visits was to get a further understanding about the FTS research area. I describe it more in details in section 3.3.1.

SUBPHASE 2: During the second subphase, I conducted a case study at Infosys Technologies in Bangalore, India. Infosys has been conducted research in FTS to improve the efficiency of its application management process. The case study was approved to be developed during the Infosys' internship called InStep program (Infosys' Global Internship Program). This program provides opportunities for students to work on a real software project as an intern and a platform to interact with Infosys' experts. The case study at Infosys was developed from May 28th to August 20th, 2012. The Infosys Company provided all facilities and resources to develop this study.

I rigorously followed the recommendations defined by Yin [YIN02] to perform this study. I present the results of this subphase in Chapter 5.

Phase 2 - Development: In this research phase, I proposed a preliminary software process model for FTS. Studies conducted during the Phase 1 provided an understanding of how to build a software process model for FTS. Furthermore, the results obtained in subphases 1 and 2 (see Figure 3), provided benefits in terms of software practices and lessons learned for the construction of the software process model. I conducted this phase during my scholarship at Lero - The Irish Software Engineering Research Centre (University of Limerick). I describe details the scholarship at Lero in Section 3.3.3.

Phase 3 – Evaluation and Evolution: The third and final stage of this research included the evaluation and evolution phase. In the evaluation and evolution phase, I conducted an expert panel. The expert panel is a recognized way to perform an initial evaluation [SHE01] as it allows an interviewer to examine the answers from the interviewees by being less rigid. The adoption of the expert panel method in this research had the follow goals:

- To gather the view of experts about the applicability of best practices included into sub-processes for FTS projects;
- To gain an understanding on how best practices included into sub-processes can support FTS projects.

Experts in the evaluation process helped to refine the software process model to make it applicable in the software industry. I selected 20 experts to evaluate the proposed model. To Beecham et al. [BEE05], small samples can be used to develop and test explanations, particularly to gain expert feedback to evaluate and support model development. The evaluation questionnaire and a summary of questions by experts to conduct the expert panel are presented in Appendix D. I conducted a pilot questionnaire with some researchers from Lero to validate the questions.

In the evaluation and evolution phase, I also performed the design validation to improve the initial software process design. It was defined during my scholarship following recommendations from research experts from Lero. I provide more details and its results in Chapter 7.

3.2.1 Considerations about the research design

The initial research design of this thesis was proposed in 2010 when I started this research. As the research furthered, I observed the need to improve or to adapt the research design. I have presented partial results of my research during the workshops at Lero. It gave me the opportunity to discuss my research with other researchers. Results from these interactions with Lero group and interactions with MuNDDos group motivated two main changes in the initial research design. Changes were made mainly due to further discussions with researchers from Lero.

The first change in the research design was made in the Subphase 2. I planned to conduct multiple case studies. However, I had the opportunity to conduct a case study at Infosys Company. In this case study, a software application was developed adopting FTS software development. It was an opportunity to combine theory and practice. Infosys is the third-largest India-based IT services company. The company has 87 global software development centers of which 32 are in India and 55 are outside India. Moreover, Infosys is a global software company leader in technologies and innovation.

The results of this case study at Infosys were discussed with researchers from Lero. As a result, these discussions, I decided to have only one case study in the Subphase 2. The main motivation to make this change was the significant contribution obtained in this study.

The second change in the research design was made in the Evaluation and Evolution phase. In this phase, I planned to execute two research methods for evaluating the proposed software process model. I had planned to conduct another SLR to complement the SLR developed in Phase 1 and a controlled experiment to evaluate and improve the model. Once I observed few publications about FTS in the last years, I decided to keep the theoretical background updated, but not using an SLR as a method. I have been searching periodically new publications about FTS in all renowned digital libraries. I also defined to carry out the design validation and an expert panel. The design validation would help to make improvements in the preliminary software process model design. The expert panel method was adopted as a research tool instead of a controlled experiment. Controlled experiments in software engineering often involve students as subjects, which might make it difficult to generalize its results to setting with various kinds of professionals [SJO03]. In this research, I performed a case study applying FTS to develop an application in a real software company. Thus, a controlled experiment as a research tool became inappropriate for the context of this study.

All decisions relate to the research design were supported by Lero and MuNDDos research group. Thus, this study adopts the similar research design adopted in researches developed by Lero group. Some examples are Deshpande [DES13] and McLoughlin [MCL10].

Deshpande [DES13] propose the GSD-COORD Model. This model aims to provide specific process, strategies and practices to both client and vendor companies to manage the GSD projects. The research design for this study was organized in four stages. The first stage of the research design was to conduct a literature review. The second stage of the research was to perform multiple case studies which included empirical research with grounded approach. The third stage of research design adopted the focus group method to compare GSD with the recognized software development processes and project management standards that are applicable to the software industry. The fourth and final stage of the research included the evaluation and evolution phase. The expert panel approach was followed in the evaluation process of the GSD-COORD. The experts in the evaluation process helped to refine the model to make it applicable in the software industry. Fourteen experts were selected to validate the GSD-COORD Model.

McLoughlin [MCL10] created the Rosetta Stone Methodology and the RS-ICMMI instance mapping. Both the generic methodology and the concrete mapping developed during his research were evaluated and validated through interviews. Interviews were conducted with various practitioners, authors, and academics. The evaluation process was carried out using the expert panel method. The author decided to use the expert panel method to elicit feedback on the proposed model.

3.3 Research Visits, Internship, and Scholarship

During this study, I had the opportunity to do two research visits, an internship and a scholarship. It was important to support this research and my decisions along of this study. I had mainly the opportunity to learn more about the research topic, research methods adopted by other research groups, and interact with other researchers. First, I describe the details of a research visit did to research groups from American University and University of

Arizona. As follows, I describe the internship at Infosys Technologies in Bangalore (India). Finally, I describe the scholarship at the University of Limerick in Ireland.

3.3.1 Research visit to American University and University of Arizona

During the subphase 1 (see Figure 3), I did a research visit to interact with researchers from American University and University of Arizona, in special with Professor Erran Carmel and Professor Amar Gupta. Professor Erran Carmel develops research in FTS and the globalization of technology work, which involves global teams and global sourcing. Professor Amar Gupta develops research on 24 Hour Knowledge Factory (24HKF) concept mainly in around-the-clock environments. With Professor Carmel was possible to obtain a deeper theoretical basis about FTS, discuss results found in studies and possible solutions for the area. Moreover, with him was possible to identify in which direction from the management perspective the research in FTS is taking. With Professor Amar Gupta was possible to deeper in theoretical basis about 24HKF, to identify the main studies in the area and its results.

I observed that FTS and 24HKF have many differences. From analysis these differences it was possible to understand the point of view of each researcher. The concept of 24HKF can be practiced in distributed or co-localized environments. In addition, 24HKF focus on finding an optimal solution to share knowledge between teams [GUP09]. It can be applied for FTS and around-the-clock environments. However, I have not found studies about 24HKF applied to FTS scenarios in studies performed by the research group from University of Arizona. All studies in 24HKF use around-the-clock scenarios. The concept of 24HKF concept can be adopted in GSD, but it is not specific for this type of software environment. A company can decide the best way to use 24HKF.

On the other hand, FTS is applied in software development scenarios with teams distributed in different sites and time zones [CAR10]. Studies in FTS report many research opportunities, in special the adoption of agile methodologies for FTS scenarios [CAR10] [CDE09]. To Professor Erran Carmel, agile methodologies are the most promising for FTS practice. Professor Gupta also considers agile methodologies as a tendency for the research area [DEN09].

I also observed that FTS and 24HKF are recommended for different software development phases. 24HKF concept is indicated for all software development phases. On the other hand, only testing and development are recommended for FTS [CDE09]. These characteristics show specific details about each concept.

My research visit to American University and University of Arizona contributed mainly in the follow aspects of this study:

- To observe how each group has been developed research in each concept;
- To identify the current state of the art of the FTS and 24HKF;
- To gain a better understanding of each research area;
- To develop a database of each concept.

3.3.2 Internship at Infosys Technologies Company

The empirical data in this thesis were collected from a case study performed during an internship at Infosys Technologies, Bangalore, India. Infosys is a global leader in consulting, technology and outsourcing solutions. The company is present in more than 30 countries, with more than 155,000 employees worldwide, across 73 offices and 90 development centers in the United States, India, China, Australia, Japan, Middle East, and Europe (http://www.infosys.com/).

My project at Infosys aimed to investigate FTS development. I conducted a case study to examine the feasibility and outcomes of FTS software development. A project mentor and a project student were assigned to give me support during the internship . During this project, I collected data for this research. Chapter 5 describes the details of the case study.

3.3.3 Scholarship at the University of Limerick

The scholarship at the University of Limerick (UL) took place from September 13rd, 2012 until September 9th, 2013. The main goal of this scholarship was to interact with Lero - Irish Software Engineering Research Centre group (http://www.lero.ie/), in special with Professor Ita Richardson.

During my time at Lero, discussions were performed to plan and to define further research steps. It was possible to discuss my research with other researchers from Lero, particularly those belonging to the Process Quality Research Group led by professor Richardson. Meetings were performed every week to discuss the research status. In addition to spending time within Lero, it was possible to attend workshops and give presentations about the research status. Some activities performed during the scholarship at UL were:

- Reviewing of the research design

- Presentation of the research status in workshops to discuss results, research contributions and research methodology
- Building a preliminary software process model for FTS
- Reviewing of the software process model for FTS with experts from Lero
- Planning of the evaluation and evolution phase
- Partial execution of the evaluation and evolution phase
- Collaboration on research papers, some of which have already been accepted for publication.

This scholarship was funded by the program Science without Borders (*CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior*). During the scholarship at Lero, the initial proposal this thesis was presented at the Annual NUIG-UL (National University of Ireland, Galway/University of Limerick) Research Day held in Galway (Ireland). It was awarded 2nd place in the PhD' poster competition.

3.4 Chapter Summary

This chapter presented the research methodology adopted in this thesis. I discussed the research design, including the research methods and how these methods were executed through each stage of the research. This research is exploratory and it adopted as main research methods, SLR, case study, and expert panel.

At the beginning of this study, I had planned to conduct multiple case studies and a controlled experiment. However, some changes were made in the initial research design. As the research further, I observed the need to improve or to adapt the research design. Changes in the research design were motivated by the results obtained and some by research constraints. Thus, I proposed some changes in the initial research design during my time at Lero.

During this research, I did a research visit, an internship and a scholarship. I visited two renowned researchers, Professor Erran Carmel (American University) and Professor Amar Gupta (University of Arizona) and their research groups. I did an internship at Infosys Technologies in Bangalore (India) and a scholarship at the University of Limerick in Ireland. The main contribution of the research visit, internship and scholarship was the opportunity to discuss my research with other researchers getting feedback about research in terms of research methodology, results, and data collection. That was important to improve methodological aspects and review obtained data. Furthermore, the internship at Infosys Technologies allowed collecting data for this research.

The research methodology followed in this thesis adopts research methods adopted in studies performed by researchers from Lero and MuNDDoS research groups. The following chapters in the thesis report how research methods introduced in this section are used to build a software process model for FTS.

4. BEST PRACTICES AND CHALLENGES IN FOLLOW THE SUN SOFTWARE DEVELOPMENT

In this chapter, I present results obtained in the Subphase 1 from the research design (see Figure 3). In this subphase, I conducted a SLR in FTS. The main goal of this SLR was to identify best practices and challenges for FTS development in the literature. This chapter is organized as follows. In section 4.1, I describe the research method. In section 4.2, I present results from this study. In section 4.3, I summarize this chapter and its main contributions.

The results of this chapter have been published in the papers: "*Mapping Global Software Development Practices for Follow-the-Sun Process*", in the 7th International Conference on Global Software Engineering (ICGSE)[3] and "*A Systematic Literature Review of Best Practices and Challenges in Follow-the-Sun Software Development",* in the PARIS workshop[4] collocated with ICGSE 2013.

4.1 Research Method

This study was performed using a Systematic Literature Review (SLR) as a research method. As described in section 3.1.1, a SLR is developed to gather and evaluate the available evidence pertaining to a focused topic. I followed the guidelines defined by Kitchenham and Charters [KIT07] to conduct this study. As described by the authors, the first step to perform an SLR is to define a research protocol which is described in the following sections.

4.1.1 Research Questions

I defined two research questions (RQ) for this study:

- RQ1: What FTS challenges are reported in the literature?
- RQ2: What are the best practices recommended for FTS?

[3] The 7th International Conference on Global Software Engineering (ICGSE) was held in Porto Alegre, Brazil from August 27th to 30th, 2012. All papers presented at the conference venue were published in the conference proceedings by IEEE CS Press and be available in the IEEE CS Digital Library.

[4] The PARIS: Methods and Tools for Project/Architecture/Risk Management in Globally Distributed Software Development Projects is a workshop collocated with ICGSE. The PARIS workshop was held during the 8th International Conference on Global Software Engineering (ICGSE) in Bari, Italy from August 26 - 29th, 2013. All papers presented at the conference venue were indexed by IEEE Computer Society (CSDL) digital libraries.

While challenges identified by RQ1 can also promote the identification of new software development practices, it is also important to identify what are the best practices that have been recommended to date for FTS (RQ2) in order to better reap its benefits. Software companies have been using the reported practices [CAR11] to develop their projects as those can enhance the usefulness of FTS.

4.1.2 Data sources

Six digital libraries were searched: *IEEEXplore, ACM Digital Library, Wiley Inter Science Journal Finder, Elsevier Science Direct, Spring Link* and *ISI Web of Knowledge, Engineering Village.* For each digital library, query strings were created according to the search tool. The SLR included published studies between 1990 and 2012. To Smite et al. [SMI10], studies in GSD began to be published in the early 1990's.

4.1.3 Search string

In literature, FTS is also referenced as around-the-clock development (see Section 2.3). Since, FTS is a recent research topic, other terms may have been used as the same meaning. Thus, these terms were included as part of the search string in order to identify as many relevant studies as possible. I have included the following terms: 24-hour development model, 24-Hour Knowledge Factory Paradigm (24HrKF), around-the-clock and shift work. The boolean search expression built with these terms is presented as follows:

(("Follow the Sun" <OR> "around-the-clock"<OR> "24-hour development" <OR> "24-Hour Knowledge Factory Paradigm" <OR> "shift work") <AND> "software")

4.1.4 Selection process

The selection of studies is one of the most critical processes in a SLR [LAN11]. It requires a great effort in the study selection to prevent inaccuracy in the findings. After an extensive search, 773 papers were found.

To select papers, I read the title followed by the abstract. Posters, panels, abstracts, presentation and summaries studies were excluded. Only studies reporting empirical evidence were selected to the SLR. At this point, I read the full paper. Repeated studies, those that did not specifically focus in FTS and the ones that did not belong to software engineering were excluded. Thus, the number of papers was reduced to 27, as shown in Table 2.

Table 2 – Selected papers.

Digital library	**Total results found**	**Not selection**	**Final selection**
IEEEXplore	106	91	15
ACM Digital Library	251	244	7
Wiley Inter Science Journal Finder	81	79	2
Elsevier Science Direct	33	32	1
Spring Link	155	154	1
ISI Web of Knowledge	54	53	1
Engineering Village	93	93	0
Total	**773**	**746**	**27**

4.1.5 Data extraction process

A data extraction form using MS Excel was created to organize obtained data. Metadata such as, author, title, year and publication source were collected with descriptive data fields such as, topic and proposed best practices.

4.1.6 Validity of the process

The main threats to the validity of the process are the study selection, inaccuracy in data extraction, incorrect classification of studies, research methods and types and potential author bias. In order to ensure that process of selection and inaccuracy in data extraction was unbiased, Kitchenham and Charters [KIT07] recommendations were followed. In relation to concepts used in the search, it was assumed that there is no incorrect definition for FTS because the research area is not consolidated yet.

At least two researchers discussed each paper of the final selection to the classification of studies and findings (see Table 2). In case of disagreement, the issue was discussed until a consensus. Therefore, there is a possibility that the extraction process may have resulted in removing some papers, which should be included.

4.2 Results

This section presents the results from the research questions defined for the SLR. First, I present the challenges reported in the literature and then the best practices recommended for FTS.

4.2.1 FTS challenges reported in the literature

To Jabangwe and Nurdiani [JAB10] FTS challenges are related to coordination, communication, and culture aspects. Therefore, to answer RQ1, I mapped the challenges in these three categories. I also calculated the frequencies of challenges in different studies (Column 3). These findings are listed in Table 3.

Table 3 - FTS development challenges.

No.	Challenge (CH)	Freq.	Reference
COORDINATION			
01	Time zone differences	9	[KRO11] [TAN11] [GUP09] [CDE09] [SET07] [TRE06] [HOL06] [YAP05] [GOR97]
02	Daily handoff cycles or handing off work-in progress (unfinished objects)	9	[HES12] [CAR10] [SOL10] [KRO11] [CDE09] [GUP09] [GUP08] [SET07] [CAR06]
03	Geographic dispersion	3	[CDE09] [SET07] [SAN12]
04	Cost estimation	3	[GUP12] [TRE06] [CAR06]
05	Loss of teamness	2	[CDE09] [ESP03]
06	Number of sites	1	[SOL10]
07	Coordination breakdown	1	[CDE09]
08	Managerial difficulties	1	[JAL04]
09	Technical platforms	1	[YAP05]
COMMUNICATION			
10	Communication difficulties (socio cultural diversity)	8	[SAN12] [HES12] [KRO11] [CAR10] [CDE09] [SET07] [TRE07] [JAL04]
11	Synchronous communication	5	[HES12] [TAN11] [GUP10] [TRE06] [ESP03]
12	Language differences	3	[CAR10] [SAN12] [SET07]
13	Loss of communication richness	2	[GUP12] [CDE09]
14	Technical difficulties	1	[JAL04]
15	Manage religious or national holidays	1	[KRO11]
CULTURE			
16	Cultural differences (increase in number of development sites, lack of synchronous communication)	7	[SAN12] [KRO11] [CAR10] [CDE09] [SET07] [HOL06] [YAP05]
17	Different technical backgrounds	3	[TRE06] [YAP05] [GOR97]

In the Coordination category, a great number of studies report time zone differences and daily handoff cycles as challenges for FTS implementation. In addition, within Coordination, seven more challenges were found in three studies. Although these findings point to lower frequencies, not identifying the challenges can lead to negative consequences for FTS projects. I found a few successful cases of FTS in literature. One of the possible reasons for this is that companies do not deal effectively with coordination challenges

Eight studies reported Communication difficulties, often related to the socio-cultural diversity of teams [SET07]. I also found five studies reporting synchronous communication as a challenge. This occurs due to the difference in time zones. Aranda et al. [ARA10] argue that the lack of face-to-face communication in GSD projects is a main obstacle to communication. In FTS, making opportunities for spontaneous interaction can result in a large amount of communication overhead introduced during task handoffs [GUP11].

Language differences and loss of communication richness is mentioned as a challenge caused by socio-culture distance [JAB10]. Technical difficulties are related to the disparity in infrastructure whereas the management of religious or national holidays poses yet another challenge as they do not coincide with those holidays in western locations.

In the Culture category, I found two challenges cited by ten studies. Cultural differences arise due to circumstances such as increased numbers of development sites, lack of synchronous communication and differing languages. Different technical backgrounds can be caused by different skills and competencies. Both are determined mainly by social, ethnic and religious aspects [JAB10].

4.2.2 FTS best practices

From the theoretical base, a set of best practices for FTS development were identified. To identify best practices for FTS, the definition given Williams [WIL09] was followed.

"*In the software engineering area, a best practice is a software development practice that, through experience and research, has proven to reliably lead to a desired result and is considered prudent and advisable to do in a variety of contexts*" [WIL09].

Table 4 shows best practices identified for FTS found in literature. Each best practice is briefly described next.

Table 4 - FTS development best practices from literature.

No.	Best practice (BP)	Reference
01	Use of agile methodologies for project management	[GUP12] [CAR10] [SMA10]] [TRE06] [YAP05]
02	Use of incremental software development approaches	[HES12] [DEN08]
03	Daily stand-up meetings	[HES12] [YA05]
04	Application of FTS for testing and development phases	[CAR10] [CDE09]
05	Application of FTS for testing phase	[CAR06]
06	Process documentation	[AVR07] [TRE06] [TAW06]
07	Daily exchange of the project status by technologies	[ESP07]
08	Conference calls outside office hours for some time zones	[BAT01]
09	Daily handoffs of 30 minutes duration with each development site	[HES12]
10	Use of screen sharing technology to exchange knowledge	[TAN11]
11	Calendar of handoff sessions should be clearly defined	[DES09]

12	Clean handoff and sticky handoff interactions	[VIS09]
13	Use of real time technologies for knowledge sharing	[TAN11] [GUP09] [TAW06] [RAM02] [GOR96]
14	Use of an FTP Server (or data repository) to exchange code and documents	[CDE09] [TAW06] [RAM02]
15	Wikis and online forums to share knowledge between FTS teams	[GUP12]
16	Backup teams	[DES09]
17	CPro concept	[GUP12] [DEN08]
18	Low task granularity	[ESP03]
19	Implementation of 'tracking system'	[DES09]
20	Task distribution by sequence or dependency	[TAW06]
21	Adopt proper technologies or tools to support communication between FTS teams	[NII11] [RAM02]
22	Time window	[TAN11] [LIN07]
23	Out-of-hours e-mails	[LIN07]
24	Informal, unplanned and ad hoc communication	[SET07]
25	Corporate technologies for team interaction	[TAN11]
26	Models of e-mails and electronic messages	[GOR96]
27	Opt out for development sites where team members could speak the same language	[LIN07]
28	Pair programming	[TAW06] [VAN05]
29	Face-to-face communication	[LIN07] [SET06]
30	At least one hour overlap between two sites	[DES09]
31	Fitting teams' working hours for a good overlap	[HOL06]
32	Teams distribution across two or three sites	[SOL10]
33	Meetings between team members for building trust	[SET07]
34	Team members with same culture	[SET07]
35	Cultural awareness training	[TRE06]
36	Similar code patterns	[TAW06]

- *BP01 - Use of agile methodologies for project management*

Agile methodologies are recommended for scenarios that have a continuous change of requirements and incremental deliveries in a short time [SMA10]. Furthermore, agile methodologies are more flexible than conventional methodologies. In literature, Scrum and XP are the most methods cited for FTS implementation.

- *BP02 - Use of incremental software development approaches*

Hess and Audy [HES12] recommend incremental software development approaches as TDD for FTS. TDD is an approach to software development, in which software units are developed in small pieces. This approach does not require initial design details as software units are incrementally developed following test-before-code stile [GUP07]. Testing on a small amount of code contributes to verify acceptance of requirements of implemented functionalities.

- *BP03 - Daily stand-up meetings*

Stand-up meetings came up from Scrum methodology. It is a daily team meeting that helps to provide a status update to the team members. The idea of stand-up meetings comes from the Scrum methodology that emphasizes on a daily team meeting that helps to provide a status update to the team members [YAP05].

- *BP04 - Application of FTS for testing and development phases*

Evidence from studies conducted on software industry shows that FTS is effective for testing as well as development phases. These phases can work well in FTS because handoffs are structured and granulate [CAR10].

Hess and Audy [HES12] also argue that FTS is feasible for the development phase. To the authors, FTS can be used to reduce the time spend in development phase.

- *BP05 - Application of FTS for testing phase*

Testing is suggested as the best software development phase to implement FTS [CAR06]. In this phase, small and low complexity tasks can be handled regularly between production sites separated by different time zones.

- *BP06 - Process Documentation*

Process documentation is used for knowledge transfer between different teams and projects [AVR07] [TAW07] [TRE06]. This practice ensures availability of technical documentation. It also can be used to maintain a history of FTS implementation, which would subsequently improve the decision making process.

- *BP07 - Daily exchange of the project status by technologies*

This practice recommends the use of technologies such as telephone calls, video conferences or e-mails for the daily exchange of the project status. Telephone calls and video conferences provide synchronous communication for real time interactions. It minimizes any possible misunderstandings [ESP07]. These technologies may be used in conjunction with others.

- *BP08 - Conference calls outside office hours for some time zones*

Companies try to maximize the overlap in office hours planning conference calls outside office hours for some time zones. To Battin et al. [BAT01], conference calls outside office hours are needed to resolve problems that required a complete information exchange between sites. Although teams could report a problem by e-mail almost instantaneously to all team members, the resolution often required detailed discussions.

- *BP09 - Daily handoff of 30 minutes duration with each development site*

Hess and Audy [HES12] recommend that handoff sessions should be of 30 minutes duration between the two sites. To these authors, 30 minutes is sufficient to transfer tasks and discuss task details.

- *BP10 - Use of screen sharing technology to exchange knowledge*

Screen sharing contributes to transfer knowledge between team members [TAN11]. Its uses make easier to understand the information that is being discussed.

- *BP11 - Calendar of handoff sessions should be clearly defined*

This practice is used to provide better communication between teams. Calendar of handoff sessions allows teams to interact daily according to the same timetable. This practice should be defined before an FTS project would start [DES09].

- *BP12 - Clean handoff and sticky handoff interactions*

Clean handoff interactions are short interactions to discuss punctual questions related to the project. On the other hand, sticky handoff interactions are more intense, but can be used effectively [VIS09].

- *BP13 - Use of real time technologies for knowledge sharing*

Many technologies are available to make knowledge sharing easier between the teams. Tang et al. [TAN11] and Gupta et al. [GUP09] recommend technologies such as, webcams and instant messaging software to improve communication between the team members distributed across multiple sites.

- *BP14 - Use of an FTP Server (or data repository) to exchange code and documents*

BP14 insists on the use of a common data repository to exchange code and documents between team members. Project files and code can be stored in this data repository. All team members should have full access to this data repository [CDE09] [TAW06] [RAM02].

- *BP15 - Wikis and online forums to share knowledge between FTS teams*

Wikis and online forums are the tools used to share knowledge among the team members [GUP12]. This practice insists on creating an internal wiki and online forums as a knowledge base in order to share problems and solutions. Both of these provide informal knowledge in a structured format.

- *BP16 - Backup teams*

Backup teams are used to give 24/7 support during holidays and weekends. Implementation of backup teams ensures that information is not lost due to a probable communication channel breakdown during the national holidays and weekends. Deshpande and Richardson [DES09] recommend that at least 10% of the teams must be available to implement this practice.

- *BP17 - CPro concept*

CPro is an agile software process that improves the CP performance. It also assigns workloads to the different members of a CP, in a way that maximizes productivity [DEN09]. A CP is a highly coordinated micro-team, which may seem like a single unit, but consists of a collection of several individuals. In such a system, each offshore site mirrors its counterpart; e.g. each site would have the same number of CPs. However, this does not imply that an equal number of developers would be present at each location as each site can have individuals belonging to more than one CPs [DEN09].

- *BP18 - Low task granularity*

FTS can be effective for software development in context to low task granularity, such as, bug correction or call center activities [ESP03].

- *BP19 - Implementation of 'tracking system'*

'Tracking system' is implemented to check teams' performance in GSD environments. This practice aims to plan and control events that can result in delays for projects [DES09].

- *BP20 - Task distribution by sequence or dependency*

In the sequencing or dependency distribution, one task is divided between two or more members who are distributed across different time zones. One member would transfer the task to another member localized in a different site. This member would take up the task and would continue from the point since the preceding team's member made the last change. This practice allows for 24 hours working development [TAW06].

- *BP21 - Adopt proper technologies or tools to support communication between FTS teams*

The communication between FTS teams can be carried out using proper communication technologies or tools [RAM02] such as telephone calls, e-mails and IM. Furthermore, many communication technologies and tools are available to support communication between distributed teams [NII11].

- *BP22 - Time window*

Time window is used by teams to minimize collaboration conflicts between sites. This practice provides opportunities for synchronous interactions without prior schedule definition [LIN07].

- *BP23 – Out-of-hours e-mails*

Time zone differences between the development sites may invariably make team members to perform part of their work at home. Out-of-hour e-mails help to reduce potential delays between sites. This practice can be implemented by providing free internet access and laptops for all teams involved in the project [LIN07].

- *BP24 - Informal, unplanned and ad hoc communication*

BP24 is important to support collaboration between the teams. It can be implemented through discussion pairs [SET07].

- *BP25 - Corporate technologies for team interaction*

BP25 recommends technologies such as video conferencing, screen sharing and other corporate resources for the teams attending meetings from their homes. This practice provides more flexible interaction windows to increase connectivity between the teams [TAN11].

- *BP26 - Models of e-mail and electronic messages*

E-mails and electronic messages reduce the communication problems [GOR06]. A unique message template could be used to assign specific meaning to a message, for example, technical and non-technical requests could be distinguished by using different message templates. These templates should describe the essential information with fields that could facilitate in recalling information typically included in the actual message.

- *BP27 - Opt out for development sites where team members could speak the same language*

Many problems occur due to language issues. Choosing offshore teams with the same language is advantageous for FTS [LIN07].

- *BP28 - Pair programming*

Pair programming is a technique for intensive collaboration, where two developers work together on a same computer doing design, code and testing [WIL02]. This BP is based on pairs of developers. Its purpose is to improve the design, reduce defects and increase the development speed [VAN05].

- *BP29 - Face-to-face communication*

In FTS context, the end-product quality may suffer due to lacking of options available for synchronous communication [SET06]. Rich communications media like face-to-face tend to be more efficient than media, such as, telephone or e-mail [LIN07].

- *BP30 - At least one hour overlap between two sites*

Management of time overlaps between sites reduces communication and coordination problems during handoff sessions. To perform handoffs at the beginning and at the end of each working day is necessary to ensure an overlap of one hour between the distributed teams, in order to provide opportunities for synchronous communication [DES09]. Moreover, effective management of overlaps helps to promote 27/4 support.

- *BP31 - Fitting teams' working hours for a good overlap*

Time management is necessary to fit the teams' working hours for a good overlap [HOL06]. However, choosing sites for a good overlap is not always possible. Time zone differences became manageable when is possible to negotiate teams working hours.

- *BP32 - Teams distributed across two or three sites*

This BP defines the number of sites for FTS, which must be at least two sites [SOL10]. More than three sites may result in coordination problems.

- *BP33 - Meetings between team members for building trust*

Meetings are used to establish or re-establish trust, increase in the number of project meetings would definitely help to increase the level of trust among the team members; whereas, reduction in it would definitely hamper the cause [SET07].

- *BP34 - Team members with the same culture*

Team members who share the same culture develop trust more quickly than those who come from different cultures [SET07]. Furthermore, team members from the same culture are more inclined to establish trust than the team members from different culture.

- *BP35 - Cultural awareness training*

BP35 aims to develop cultural awareness among team members. This practice should be implemented at the beginning to educate team members on each other culture. Cultural differences are reduced by awareness that avoids risks such as rupture of relations between distributed team members [TRE06].

- *BP36 - Similar code patterns*

Similar code patterns allow team members to understand and identify changes made in the code since the last handoff session. Furthermore, similar code patterns can avoid reworking [TAW06].

4.3 Discussion

This study investigated twenty-seven FTS studies that were published since 1990. As a result, 17 challenges and 36 best practices were identified for implementing FTS.

Related to the challenges identified, my analysis focused on their frequencies. This makes possible to see which categories have been emphasized in past research and thus to identify gaps and possibilities for future research. Coordination category appears at first with the larger number of challenges. Followed this category appears the Communication category and then Culture category. Challenges identified are focused on the main FTS characteristics. It appears an immature research area. There are many opportunities for future studies related to coordination, communication, and cultural aspects.

Related to the best practices identified, BP01 - Use of agile methodologies for project management, BP13 - Use of real time technologies for knowledge sharing, BP06 - Process documentation, BP14 - Use of a FTS server (or data repository) to exchange code and documents and BP22 - Time window are the most cited by studies. BP01 and BP13 have five studies each and the others have three studies each.

BP01 recommends agile methodologies for FTS implementation. These methodologies have high acceptance in the software industry. Additionally, they are discussed as a promise way for FTS implementation. XP and Scrum methodologies are the most cited for FTS implementation [HES12] [CAR10] [DEN09].

BP13 recommends using technology for knowledge sharing. Conference video, telephone calls and e-mails are low cost strategies utilized by companies to perform synchronous and asynchronous communication between teams [JAL09].

Process documentation is recommended by BP06. In Taweel and Brereton [TAW06], process documentation can bring some advantages such as product and quality service improvement, cost reduction and using of resources in the best way. On the other hand, process documentation practice can result in additional delays if it is not performed properly.

Use of an FTP Server (or data repository) to exchange code and documents is recommended by BP14. Three studies cite this best practice. To these studies, documents related to the project and code must be saved in order to allow it access by all teams. In FTS, a working team works in the same tasks. When a site finished its working day, another site begins its working day working on the same task [CDE09]. Thus, BP14 is important to ensure the continuity of tasks.

BP22 - Time window is used to develop handoffs. This best practice aims to provide opportunities for synchronous interactions between distributed teams. For Carmel, Espinosa, and Dubinsky [CAR10], handoff activities are difficult to coordinate due to team distribution in different time zones. Thus, BP22 can help to reduce communication barriers between FTS teams.

Other best practices identified in the SLR have two studies each. It was observed that 51% best practices report communication aspects, 40% coordination aspects and only 8% cultural aspects. The findings show a lower percentage of studies discussing cultural aspects. However, cultural aspects are not less relevant for FTS. Cultural diversity is discussed as a barrier to FTS teams and it can affect negatively on understanding level, task development and team effort [DES10].

Research findings report mainly solutions for communication and coordination issues. It occurs because communication and coordination make a base for FTS development. Moreover, communication and coordination practices can help to minimize cultural issues.

4.4 Chapter Summary

This chapter presented results from a SLR performed in FTS development. The main goal this study was to identify best practices and challenges for FTS development in the literature. As a result, 17 challenges and 36 best practices were identified for FTS. Challenges are distributed in three categories: Coordination, Communication, and Culture. The largest number of challenges were mapped in the coordination followed by communication, and then culture category. Related to the best practices, some of these are cited in more than one study. However, the majority of best practices is reported only in one single study.

Many studies report the difficulty to implement FTS in GSD projects. Moreover, since the existing literature on the area does not fully address any concrete approach to successfully implement FTS, there remains a big research gap. This study reinforces the importance of developing a software process model to support the FTS adoption in GSD environments.

The main contribution of this study for this research was the identification of the best practices that were applied to plan and develop the next stages this research. Best practices identified in this study provide the theoretical basis to define a software model for FTS.

5. CASE STUDY AT INFOSYS TECHNOLOGIES

In this chapter, I present results obtained in the Subphase 2 from the research design (see Figure 3). In Subphase 2, I conducted a case study at Infosys Technologies in Bangalore, India. This study was conducted during an internship at Infosys (see section 3.3.2). In the next sections, I describe the case study planning and results from this study. Sections are organized as follows. In Section 5.1, I describe the study settings and methods. In Section 5.2, I present how the data were collected in this study. In Section 5.3, I present the results. In Section 5.4, I present the study limitations. In Section 5.5, I present ten lessons learned from this study. Finally, in Section 5.6, I summarize this chapter and its contributions.

The results of this chapter have been published in the papers: "*A Feasibility Study of Follow-the-Sun Software Development for GSD Projects*", in the 25th International Conference on Software Engineering and Knowledge Engineering (SEKE)[5], "*Handoffs Management in Follow-the-Sun Software Projects: A Case Study"*, in the 47th Hawaii International Conference on System Sciences (HICSS)[6] and, "*Adopting Agile Methods for Follow-the-Sun Software Development*", in the 19th Americas Conference on Information Systems (AMCIS)[7].

5.1 Study Settings and Methods

I used the case study research method to conduct this study. The case study was developed at Infosys Technologies in Bangalore, India. This study was performed following recommendations defined by Yin [YIN02].

Infosys has 73 offices and 94 development centers in the United States, India, China, Australia, Japan, Middle East, and Europe. It provides business consulting, information technology, software engineering and outsourcing services. The organization is very experienced in working on distributed software projects. As a global software organization, Infosys perform global software operations between development sites that potentially give the opportunity to implement FTS development.

The case study consisted of investigating the development of a software project in the FTS mode. The development phase of the SDLC was chosen to apply FTS because

[5] The 25th International Conference on Software Engineering and Knowledge Engineering (SEKE) was held in Boston, USA from June 27 to June 29, 2013. (ISSN: 2325-9086 online)

[6] The 47th Hawaii International Conference on System Sciences (HICSS) was held in Big Island Hawaii, USA from January 6-9, 2014. Papers from this conference are available in the HICSS Digital Library at IEEE.

[7] The 9th Americas Conference on Information Systems (AMCIS) was held in Chicago, USA, from August 15-17, 2013. All papers from the conference are available in the AIS Electronic Library (AISeL).

findings from studies conducted in the software industry shows that FTS is effective for testing and development phases [CAR10]. The software project was developed by team members distributed in India, Mexico, and Australia. I describe the Infosys' software project in more details in Section 5.2.

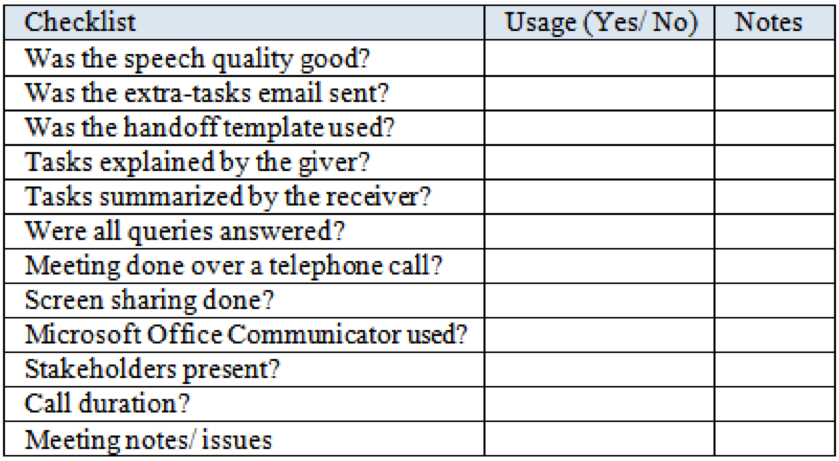
5.1.1 Data collection

In this study, documents, questionnaires and interviews were used for data collection. Data were collected from 12 participants distributed as follows. Six software developers, one project manager (PM), one scrum master, and four research scientists. Developers had different levels of working experience. Two developers had less than one year working experience, two had two years' work experience, and two others had experienced between eight and fourteen years. The other participants had more than 10 years' working experience. Table 5 presents the participants' details.

Table 5 – Design validation members' details.

Participant	**Job Title**	**Location**	**Working experience**
Participant 1	Developer (trainee)	Mexico	Less than 1 year
Participant 2	Developer (trainee)	Mexico	Less than 1 year
Participant 3	Developer	India	2 years
Participant 4	Developer	India	1 year
Participant 5	Developer	Australia	8 years
Participant 6	Developer	Australia	14 years
Participant 7	Project Manager	India	10 years
Participant 8	Scrum Master	India	More than 10 years
Participant 9	Research Scientist	India	More than 10 years
Participant 10	Research Scientist	India	More than 10 years
Participant 11	Research Scientist	India	More than 10 years
Participant 12	Research Scientist	India	More than 10 years

I collected data of the documents from the project such as, sprint backlog file, product backlog file, and project estimation file. These documents were created by the PM. I analyzed these documents to identify the teams' performance. I also created a document with 12 checklists to collect data from handoff meetings. Handoff meetings were performed at the beginning and end of each day shift. These meetings were attended by the project manager, scrum master, and developers. Infosys adopted telephone calls as a tool to support communication between development sites. Thus, data were collected by listening telephone calls among participants to perform handoff meetings. Infosys has a telephone call center system to support telephone calls between more than two participants. I present the checklist document in Figure 4.



Checklist	Usage (Yes/ No)	Notes
Was the speech quality good?		
Was the extra-tasks email sent?		
Was the handoff template used?		
Tasks explained by the giver?		
Tasks summarized by the receiver?		
Were all queries answered?		
Meeting done over a telephone call?		
Screen sharing done?		
Microsoft Office Communicator used?		
Stakeholders present?		
Call duration?		
Meeting notes/ issues		

Figure 4 – Checklist document.

Three meetings were performed with the PM. The main goal of these meetings was to identify improvements in the approach adopted by Infosys to develop the project. These meetings were performed using video conference calls. I adopted unstructured interviews with two open questions: “What is working well?”, “What should be changed in the approach adopted?”. The PM attended handoff meetings between Australia and India and between India and Mexico. His role was to discuss allocated tasks, clarify project issues, and manage the project progress.

A questionnaire to evaluate project activities was applied to developers at the end of the sprint 1 (see Appendix C). Its goal was to identify what was working well and what was not working in the FTS project activities. The evaluated activities were:

- Software engineering practices, standards and templates
- Handover template and process
- Communication flow (telephone calls, e-mail, chat)
- Task allocation
- CP experience (peer interaction, work sharing, responsibility, visibility, time zone management)
- Tools used/ not used
- Portal used, openness to change, cooperative tools (TFS and DeW)

I sent the questionnaire to the developers by e-mail. The responses given by the participants were discussing during the sprint retrospective with the PM, scrum master, and research scientists. The sprint retrospective was performed by video conference call at the end of sprint 1.

5.2 Infosys' Software Project

The first challenge faced by Infosys was how to put FTS in practice. Studies conducted in the past provided few details about the FTS experience in the software industry. Some studies report successful cases and others failures. To Sommerville [SOM11] each company has to develop its own process depending on its size, background and skills of its team, type of software being developed, customer and market requirements and company culture.

Infosys has a research center to develop new strategies for software development. For creating a software process for FTS, a group of experts from Infosys was allocated to project. Infosys's expert group had three researchers from Infolabs, one Scrum master, one project manager and one expert in deployment and development of new technologies.

A set of best practices identified in the subphase 1 of the research design (see Chapter 4) was provided to Infosys. Experts from Infosys based on own experience selected practices to create a software process for the FTS adoption. These practices were selected by Infosys' experts following guidelines for FTS teams (see Appendix A). These guidelines were created by FTS experts to develop the project.

Table 6 presents a set of best practices included in the process performed by teams during the project. Arguments given by Infosys' experts to include or not include a specific practice are also presented in the Table 6. Some practices from literature designed for around-clock environments were adapted to FTS model. The experience of Infosys' experts allowed improving practices to create a software process for FTS.

Table 6 - Best practices adopted by Infosys in the case study.

No.	Best practice (BP)	Included (Yes/No)	Infosys' experts argument
01	Use of agile methodologies for project management	**Yes**	Infosys adopted Scrum methodology to develop the project. Scrum was considered more suitable for FTS. On the other hand, XP was considered inappropriate for the FTS project. Its practices could be not developed by Infosys due to project settings and own restrictions imposed by the company.
02	Use of incremental software development approaches	**Yes**	Infosys included BP02 due to its incremental approach. In FTS, an incremental approach can help to transfer knowledge from one site to another.
03	Daily stand-up meetings	**Yes**	This practice was included to create communication opportunities between team members.

04	Application of FTS for testing and development phases	**Yes**	Infosys chose the development phase due to strategic needs in new software projects.
05	Application of FTS for testing phase	No	Infosys chose the development phase. Thus, testing phase was out of the project scope.
06	Process documentation	No	Infosys followed strictly the Scrum methodology. Thus, BP06 was not included in the process.
07	Daily exchange of the project status by technologies	**Yes**	Infosys included BP07 following the communication rule 2.1.2 defined in the Guidelines for FTS teams (Appendix A).
08	Conference calls outside office hours for some time zones	No	Due to Infosys' restrictions, laws and legislation, BP08 was not included in the process.
09	Daily handoffs of 30 minutes duration with each development site	**Yes**	Infosys included BP09 following the handoff rules 2.1.4, 2.1.5 and 2.1.6 defined in the Guidelines for FTS teams (Appendix A).
10	Use of screen sharing technology to exchange knowledge	**Yes**	Infosys included BP10 as a support resource to develop diary handoffs.
11	Calendar of handoff sessions should be clearly defined	**Yes**	Infosys included BP11 following the handoff rule 3.1.9 defined in the Guidelines for FTS teams (Appendix A).
12	Clean handoff and sticky handoff interactions	No	Infosys considered this practice irrelevant to develop the FTS project.
13	Use of real time technologies for knowledge sharing	**Yes**	Infosys used Microsoft Office Communicator as a support resource to perform handoff meetings.
14	Use of an FTP Server (or data repository) to exchange code and documents	**Yes**	Infosys adopted BP14 following the handoff rules 3.1.5 defined in the Guidelines for FTS teams (Appendix A).
15	Wikis and online forums to share knowledge between FTS teams	No	Due to Infosys' restrictions, this practice was not included in the process.
16	Backup teams	No	Due to Infosys' restrictions, this practice was not included in the process.
17	CPro concept	**Yes**	Infosys adopted BP17 as a strategy to perform task allocation.
18	Low task granularity	No	Infosys believes that FTS scenarios require high task granularity. Thus, BP18 was not included.
19	Implementation of 'tracking system'	No	Due to different experience levels, this practice was not included in the process.
20	Task distribution by sequence or dependency	**Yes**	Infosys adopted BP20 to meet the handoff rule 1.1 defined in the Guidelines for FTS teams (Appendix A).
21	Adopt proper technologies or tools to support communication between FTS teams	**Yes**	To support diary handoffs and knowledge sharing, BP21 was included in the process.
22	Time window	**Yes**	BP22 was included to allow synchronous communication between team members.
23	E-mails out-of-hours	No	Due to Infosys' restrictions, BP23 was not included in the process.
24	Informal, unplanned and ad hoc communication	No	BP24 do not satisfy FTS team rules defined in the Guidelines for FTS teams.
25	Corporate technologies for team interaction	**Yes**	Infosys included available technologies to create the software process.
26	Models of e-mails and electronic messages	No	Infosys considered BP26 irrelevant to develop the FTS project. Infosys argued that the number of e-mails exchanged between team members in a FTS project is low.

27	Opt out for development sites where team members could speak the same language	No	Due to Infosys' restrictions, BP27 was not included in the process.
28	Pair programming	No	Number of developers allocated to develop the FTS project and locations did not allow including BP28.
29	Face-to-face communication	No	BP29 was not included due to hardware restrictions.
30	At least one hour overlap between two sites	**Yes**	BP30 was included to ensure synchronous communication between two sites.
31	Fitting teams' working hours for a good overlap	**Yes**	Due to time zone restrictions, Infosys changed the working time of the team. Team members from India worked in a different working time to ensure a good overlap.
32	Teams distribution across two or three sites	**Yes**	BP32 was included to ensure time zone synchronization between different sites.
33	Meetings between team members for building trust	No	Due to Infosys' restrictions, BP33 was not included in the process.
34	Team members with same culture	No	Due to Infosys' restrictions, BP34 was not included in the process.
35	Cultural awareness training	No	Due to Infosys' restrictions, BP35 was not included in the process.
36	Similar code patterns	**Yes**	BP36 was included to support teams working on the same code file.

5.2.1 Infosys' Project Planning

Team members from three different locations and time zones were located to the project. These members were distributed in Mexico (Monterrey), India (Bangalore) and Australia (Melbourne) (shown in Figure 5). While Australia and India have some overlap working hours, Mexico joint these sites. Each site had two developers, with India the base for the project manager, scrum master and product owners.

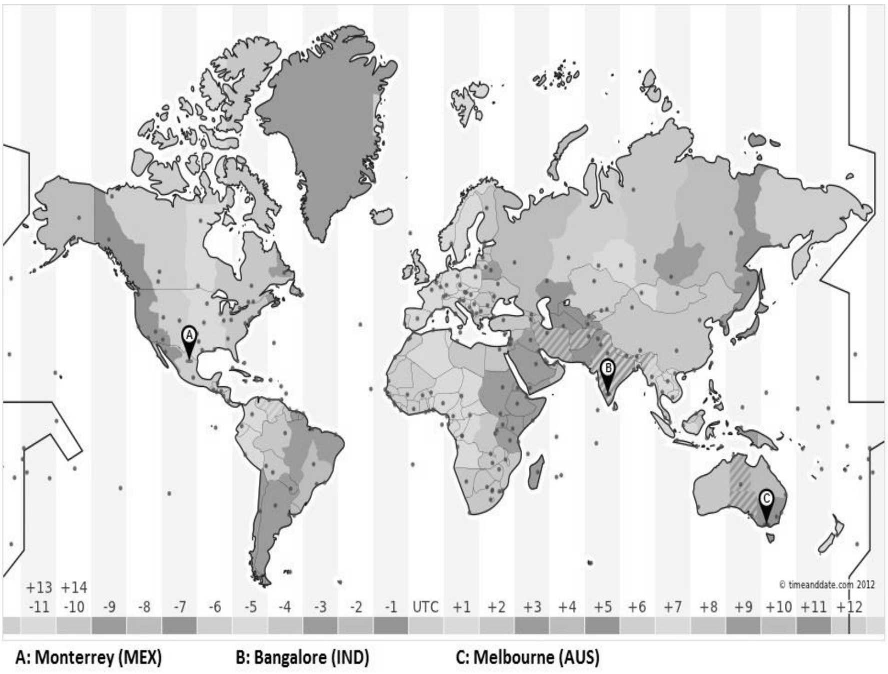


Figure 5 - Time zone differences across sites in Mexico, India and Australia.

Developers with different experience levels were allocated to develop the project. In Mexico, two trainees were allocated. In India, one project manager with approximately 10 years working experience, but without previous experience as a project manager and two developers with two years' work experience. In Australia, developers with experience between eight and fourteen years were allocated.

Following recommendations from the literature, Infosys adopted Scrum as a framework for the project execution. Four weeks duration was estimated to develop the software application. Its development was divided into two week sprints. Since did not have any standard to plan FTS projects, Infosys used standard approaches and the project manager experience to plan the project. Infosys also estimated effort hours based on typical two-location mode development. Sprint 1 started on July 24th and it finished on August 08th. Sprint 2 started on August 08th and it finished on August 21st.

CPro concept introduced by Denny et al. [DEN08] was used to perform task allocation. Based on CPro concept, Infosys created two CPs. Each CP comprised at least one team member from each site, as shows Figure 6. Tasks were allocated to CPs rather than to an individual team member. Each CP managed task allocated and its execution amongst themselves in the beginning of the project.

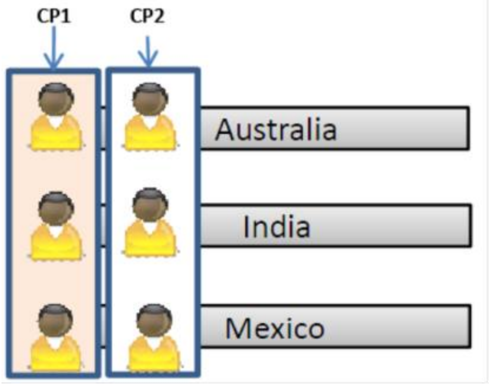
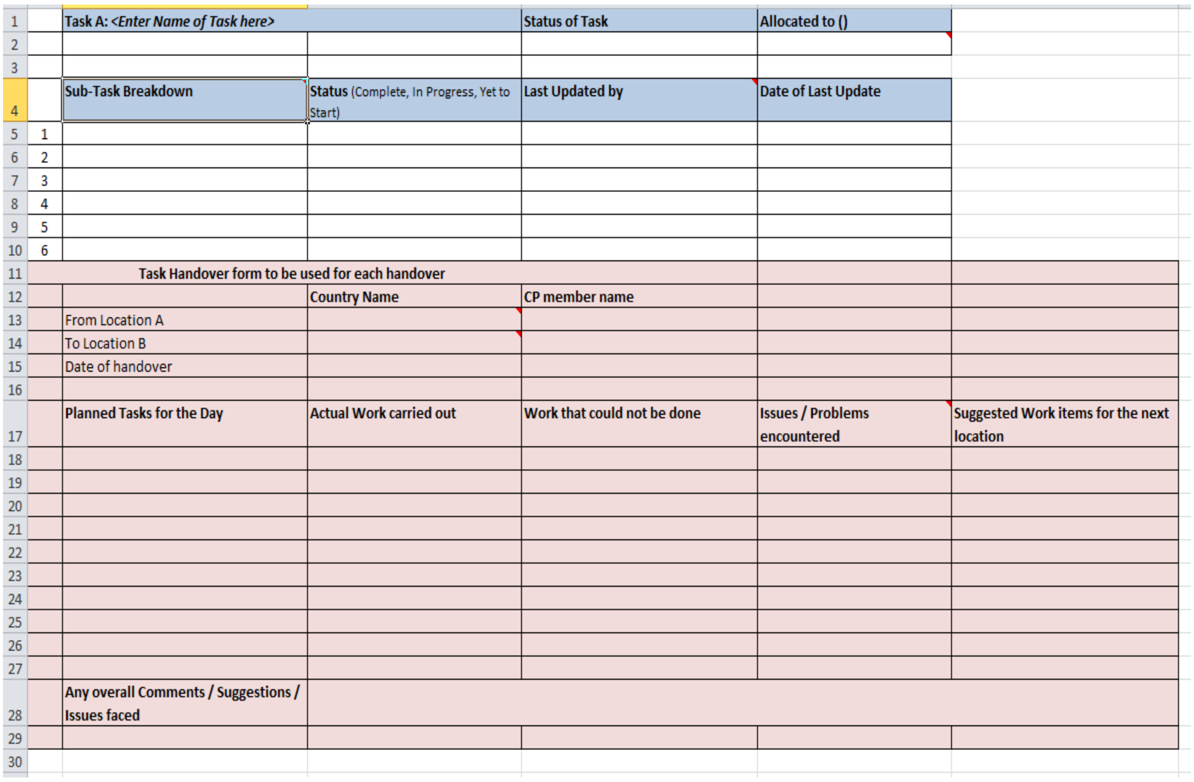


Figure 6 - Composite Persona distribution.

Project planning was created by the PM. The PM described the task's details, schedule, effort values and a product backlog file containing user stories and their mapped tasks. Project planning, product backlog file and other artifacts were saved in the project DeWportal. DeWportal is an internal data repository used by Infosys to support GSD projects.

Handoff meetings were performed over the telephone calls or communication tools at the beginning and end of each working day shift. Each handoff meeting was estimated in 30 minutes duration in the project planning.

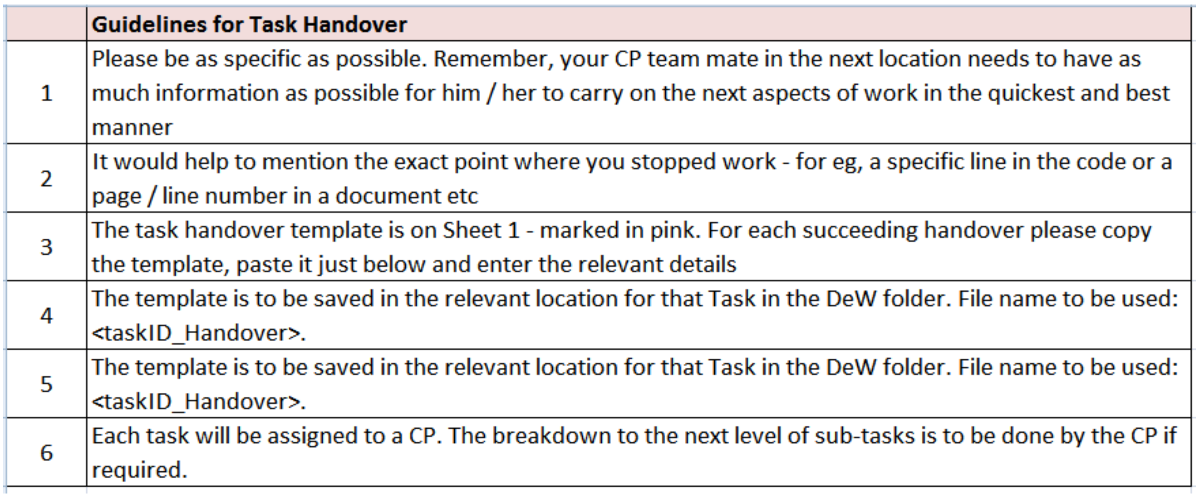
For sending tasks to another CP member, each CP member used the Excel template (Figure 7). This template was available on the TFS (Team Foundation Server) system and it was called Task Handover. CP members provide handoff information in the task handover template.



1		Task A: *<Enter Name of Task here>*		Status of Task	Allocated to ()	
2						
3						
4		Sub-Task Breakdown	Status (Complete, In Progress, Yet to Start)	Last Updated by	Date of Last Update	
5	1					
6	2					
7	3					
8	4					
9	5					
10	6					
11		Task Handover form to be used for each handover				
12			Country Name	CP member name		
13		From Location A				
14		To Location B				
15		Date of handover				
16						
17		Planned Tasks for the Day	Actual Work carried out	Work that could not be done	Issues / Problems encountered	Suggested Work items for the next location
18						
19						
20						
21						
22						
23						
24						
25						
26						
27						
28		Any overall Comments / Suggestions / Issues faced				
29						
30						

Figure 7 - Task handover template.

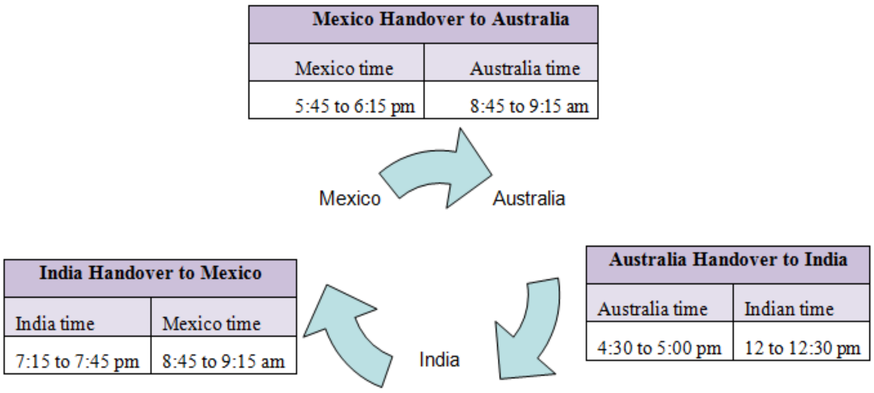
Guidelines to fill in the task handover template were developed by Infosys' experts as shown in Figure 8. These guidelines supported teams during handoff meetings and task allocation.



	Guidelines for Task Handover
1	Please be as specific as possible. Remember, your CP team mate in the next location needs to have as much information as possible for him / her to carry on the next aspects of work in the quickest and best manner
2	It would help to mention the exact point where you stopped work - for eg, a specific line in the code or a page / line number in a document etc
3	The task handover template is on Sheet 1 - marked in pink. For each succeeding handover please copy the template, paste it just below and enter the relevant details
4	The template is to be saved in the relevant location for that Task in the DeW folder. File name to be used: <taskID_Handover>.
5	The template is to be saved in the relevant location for that Task in the DeW folder. File name to be used: <taskID_Handover>.
6	Each task will be assigned to a CP. The breakdown to the next level of sub-tasks is to be done by the CP if required.

Figure 8 - Task handover guidelines.

Each working day shift had started in Australia following to India and after to Mexico. Figure 9 shows the handoff timing followed by the FTS team.



Mexico Handover to Australia	
Mexico time	Australia time
5:45 to 6:15 pm	8:45 to 9:15 am

Australia Handover to India	
Australia time	Indian time
4:30 to 5:00 pm	12 to 12:30 pm

India Handover to Mexico	
India time	Mexico time
7:15 to 7:45 pm	8:45 to 9:15 am

Figure 9 - Handoff timing.

To minimize problems related to lacking of opportunities for synchronous communication and manage the teams' office hours, developers from India and worked haft time day on the project. As FTS development was a new approach to software development at Infosys, training meetings were developed with participants to clarify FTS theory and details about the software process performed. Additionally, guidelines and documents were sent to the participants by e-mail to clarify remaining questions from training meetings. In the training meetings also were given instructions also about Scrum methodology.

5.3 Results

This section presents results from the data collection (see section 5.1.1). I start presenting results from the document analysis and handoff checklist document. Then, I present the results of the interviews conducted with the PM. As follows, I present results from questionnaire applied to developers.

5.3.1 Document analysis

In the first phase of the project (in this case, sprint 1), it was a learning phase. The team experienced processes, made mistakes and came up with recommendations to improve the process. By analyzing documents of the project, it was possible to identify the team's performance. Table 7 shows this information considering effort hours and tasks completion.

Table 7 – Teams' performance in sprint 1 and sprint 2.

	Sprint 1	Sprint 2
Estimated hours	368	464
Actual hours	432.5	350
Extra hours	64.5	0
Tasks completion %	68%	62%

As shown in Table 7, Infosys estimated 368 hours effort to develop a set of tasks in the sprint 1. At the end of this sprint, total effort expended was 432.5 hours, which was 64.5 hours more than planned. It was also found that only 68% of the planned tasks were finished. Others 32% remaining tasks were moved to sprint 2.

One of the main hurdles encountered by the team was certain delays from the internal stakeholders, which necessitated rework due to new templates introduced. This was estimated to result in approximately 50% extra work. Similarly, the setup of the project took up more time than estimated. Finally, the daily FTS handover process also took more time than estimated.

Considering the existing tasks and the carryover tasks from sprint 1, the effort estimated for sprint 2 was 464 hours. In the sprint 2, I observed that teams were more comfortable and productive having getting experience in the FTS approach from sprint 1. Several of the problems faced in the sprint 1 were minimized in the sprint 2. The effort expended in this sprint was 350 hours (see Table 7).

Not all tasks from sprint 2 were performed in FTS mode. Due to team members attending trainings and two holidays in one particular location, 62% of the planned tasks were performed using the approach. The remaining tasks of the project were completed in a subsequent phase in non-FTS mode with lesser number of the members involved (some team members were committed to other client projects and had to be released). This extra phase covered a duration of 1.5 weeks, which meant that the project was completed in 5.5 weeks as compared to the estimated target of 4 weeks. Hence, I did not consider this phase of analysis as part of this study.

5.3.2 Handoff checklist

Obtained results from checklist handoff showed low quality communication over telephone calls due to different accents and languages. In order to improve the communication quality, Infosys' experts suggested two changes in the communication protocol:

- Team members must speak slowly and clearly, in order to reduce accents and improve the communication;
- Team members must use appropriate language for the context, avoiding slang or unknown terms for the area.

With relation to extra tasks, these were allocated by e-mail by the PM. Infosys' experts identified some problems such as misunderstanding of extra tasks. Thus, two strategies were adopted to minimize this problem:

- Extra tasks allocated by the PM should be discussed in advance during handoffs;
- As an additional resource, clarifications about extra tasks should be sent by e-mail.

The handover template contributed to manage handoffs information. The team did not report problems in the using of this template. I also did not identify issues related to the use of CPro concept. Following the handover template, each CP pair discussed tasks and the next steps. To clarify code and documents with another site, team members used screen sharing.

In some handoff meetings, I observed that a CP member was not present. In these cases, the CP member (giver) sent an e-mail with handoff information to the next CP. Thus, I observed that to address the unavailability of a CP member, an additional e-mail with all handoff information could be sent to the next site.

I also observed handoffs duration between sites. In Table 8, we present data from handoffs duration in sprint 1. These data were collected automatically by the case company using a telephone call center system.

Table 8 - Timing of handoff meetings.

Date (Sprint 1)	**Location**	**Duration**
7-Aug	India – Mexico	38
7-Aug	Australia – India	49
7-Aug	Mexico – Australia	19
6-Aug	India – Mexico	34
6-Aug	Mexico – Australia	65
3-Aug	India – Mexico	44
3-Aug	Australia – India	34
3-Aug	Mexico – Australia	16
2-Aug	India – Mexico	43
2-Aug	Australia – India	44
2-Aug	Mexico – Australia	16
1-Aug	India – Mexico	37
1-Aug	Australia – India	50

1-Aug	Mexico – Australia	39
31-Jul	India – Mexico	55
31-Jul	Australia – India	69
31-Jul	Mexico – Australia	24
30-Jul	India – Mexico	53
30-Jul	Australia – India	45
27-Jul	India – Mexico	48
27-Jul	Australia – India	38
27-Jul	Mexico – Australia	18
26-Jul	India – Mexico	37
26-Jul	Australia – India	20
26-Jul	Mexico – Australia	11
Average	India – Mexico	39.78
	Australia – India	39
	Mexico – Australia	37.34
Average minutes	37.84	
Total minutes	946	
Total hours	15hrs76 min	

Handoffs among sites were performed for 38 minutes on average in the sprint 1. The longest handoff was performed by team members from Australia and India. It occurred because team members from these locations used handoffs to plan next tasks. The longest handoff took 1 hour and 10 minutes. The shortest handoff was between Mexico and Australia. It took 11 minutes.

In literature, studies recommend handoffs duration of 30 minutes (e.g. [HES12]). However, findings this study show some handoffs performed over than 30 minutes. Thus, some strategies to reduce handoffs duration were adopted by Infosys:

- Distributing the communication time between CPs;
- Creating rules for communication during handoffs. Handoff meetings were performed in 38 minutes on average in the Sprint 1. The longest handoff meeting was performed by team members from Australia and India, as shown in Table 8. Team members from Australia and India used handoff meetings to plan next tasks. Mexico and Australia performed the shortest meeting taking it 11 minutes.

All handoff meetings were performed over phone calls. IM (Instant Message) chat was used as an additional communication resource to perform handoff meetings. However, it was used in a few situations.

Results obtained from checklist handoff contributed to identify and solve problems during handoff meetings. Some changes were introduced in the FTS process as problems were being identified.

5.3.3 Interviews with the project manager

At the beginning of the project, training meetings encourage all team members to propose solutions or give ideas to improve the process. Due to the own company's restrictions and team's availability, only the PM was interviewed. The PM used his experience to propose changes according to the problems faced. The main contributions identified from the interviews with the PM are described below:

- *CP owner attribution:* at the beginning of the project, some tasks allocated to CPs were started, but were not finished. This was due to no one person being responsible for the tasks. Thus, to ensure that tasks will be started and finished by CPs, the PM assigned, by e-mail, a CP owner for each task. When a task was completed, each CP should inform the Project Manager. This also helped to manage task allocation.
- *Task allocation per working day:* following Scrum practices, the PM used the sprint backlog file to describe the tasks and its allocation to each CP. However, team members reported difficulties in understanding the sprint backlog file. Thus, the PM created a daily e-mail to inform CPs about task allocated. The task allocation per working day contributed to the definition of priorities and reduction of problems faced by teams when categorizing a task in the sprint backlog file.
- *Extra tasks:* the PM took advantage of information provided by CP owners related to tasks progress to verify the necessity of new task assignments for the working day. When necessary, the PM assigned extra tasks by e-mail.

5.3.4 Questionnaires

Findings obtained in the questionnaires are organized according to six activities (see Appendix C). These activities are Software engineering practices, standards and templates, Handover template and process, Communication flow (telephone calls, e-mail, and chat), Task allocation, CP experience (peer interaction, work sharing, responsibility, visibility, and time zone management), Tools used/ not used, and Portal used, openness to change, corporative tools (TFS and DeW).

In *Software engineering practices, standards & templates* activity, I identified that the main problem was the lacking of standards and templates. In addition, participants reported the increase of working hours due to lacking of standards and templates.

The team positively evaluated *Handover template and process* activity. However, one answer reported the difficult to share information with another site using handover template. Another answer suggests providing all tasks at the project beginning.

Communication flow (telephone call, e-mail, and chat) activity was also positively evaluated by the team. Some aspects to improve the communication flow were identified:

- Reduce time duration handoff meetings;
- Distribute taking time between members;
- Speak slow to reduce accents;
- Use of proper professional language.

In the *Task allocation,* many problems were reported by participants, such as, lateness task allocation, lack of task priorities, misunderstand of sprint backlog and task categorization. In order to minimize these problems, the PM defined a daily e-mail with task allocation. It helped to reduce misunderstandings of tasks by team members.

CP experience (peer interaction, work sharing, responsibility, visibility, time zone management) was positively evaluated. To the participants, the main benefit of using CP formation is sharing work responsibility with sites. On the other hand, participants indicated different experience levels as a disadvantage for CP using.

Tools used by team members were considered appropriate. In addition, one participant suggested some tools like Visual Studio and Microsoft Office for FTS project.

Portal used, openness to change, corporative tools (TFS and DeW) was also positively evaluated by the team. These resources contributed to do the control of versions and to maintain the punctuality of teams regarding to files uploaded. Developers also mentioned the lack of experience related to use of the portal.

Participants also reported some advantages and disadvantages using the FTS approach. Advantages are related to team punctuality, intensive communication and good time management. On the other hand, developers reported some disadvantages like lack of standards and templates, inappropriate task allocation at the project beginning and lack of project guidelines.

5.4 Limitations of this Study

This study includes some limitations. Firstly, participants with different levels of experience were allocated to the project. It includes the lacking of the team's experience in agile methodologies, methods, and practices adopted during the project. Thus, the imbalance level of experience could have influenced positively or negatively in the findings.

The software process followed by FTS teams was created by Infosys' experts based on own methodologies. It may result in a solution for the company that can be not generalized to others.

Team's availability is another limitation. In the middle of Sprint 2, team members from two locations were allocated to other client projects. For this reason, the project was completed using a non-FTS mode.

5.5 Lessons Learned in this Study

The case study reported in this chapter contributed to identify a set of lessons learned, which are described next.

- **Lesson 1 - The importance of agile methodologies for FTS:** the case study performed at Infosys showed evidences that an agile methodology, as Scrum, is feasible for FTS software development.
- **Lesson 2 - Templates and standard documents:** at the beginning of the project, teams faced problems to identify standards utilized in the project. Teams must to know templates and standard documents that will be used during the software development before the project start.
- **Lesson 3 - Coding standards:** for avoiding rework a standard to put comments in the code must be defined before project start. In the case study, FTS team spent a lot of time trying to understand the code and identifying the last changes made in the code.
- **Lesson 4 - Screen sharing to transfer knowledge***:* transferring or explaining a task using screen sharing becomes easier when teams can see the information talked about. During handoff meetings, it was observed that teams opted by using of the screen sharing to explain codes and design documents.
- **Lesson 5 - Communication resources:** it was observed that phone calls, e-mails and communicators, such as Microsoft Office Communicator, are useful to provide communication between teams, but they must be used together. It was also observed that during phone calls, some communication rules contributes to improve the communication quality, such as, speaking slowly to reduce accents, distribute time between members and summarize the tasks talked by the giver.
- **Lesson 6 - Tasks for the day:** a daily e-mail assigning tasks individually to members contribute to define task's priority and to reduce problems faced by teams to categorize a task in the sprint backlog.

- **Lesson 7 - Handover template:** an Excel spreadsheet was used to manage tasks exchanged between teams. It works very well, but it could be automated.
- **Lesson 8 - Weekend handoff:** on weekends it is difficult to manage handoff meetings. In the case study, team performed communication via e-mail. However, many problems were identified mainly in the first weekend. Receiver team faced difficulties to understand new tasks and how to continue the working. On the second weekend was better, but the tasks were discussed on Friday during handoff meetings.
- **Lesson 9 - CP owner:** some tasks were assigned to a CP owner during the sprint 1. It was observed that is a good way to ensure complete tasks. Tasks can be assigned by e-mail to CP owners per location. Each CP owner will check if the task has been completed.
- **Lesson 10 - Extra e-mail in case of day missed from work or delays:** one e-mail with detailed handoff information must be sent to the next site in case one CP receiver is late or missed his workday. With this information the next site will be able to continue the task. This extra e-mail must be sent after handoff meeting, because new information may emerge.

5.6 Chapter Summary

This chapter presented results from a case study performed at Infosys Technologies. The focus of this study was to investigate the FTS adoption in GSD projects. FTS was adopted at Infosys to develop a software project. This project aimed to develop a software application on FTS mode.

Team members from Mexico, India, and Australia were allocated to the project. Due to lack a software process for FTS, Infosys' experts created a software process based on best practices from the literature (subphase 1) and team's interactions and experience. It is important to mention that the software process presented in this chapter suits the Infosys software development context.

Nineteen best practices from Subphase 1 were included in the process. Two of them were included in the middle of sprint 1. These practices were BP10 - Use of screen sharing technology to exchange knowledge and BP36 - Similar code patterns.

At the end of sprint 2, findings from this study showed the FTS feasibility for GSD projects with some evidence that FTS can be used to compress duration. However, many untypical issues had occurred during the project. Team members attending trainings and developers without experience allocated to the project, are some examples.

Results from documents analyses, questionnaires, handoff checklists, and interviews with the PM contributed to identify ten lessons learned. It is main contribution this study for this research.

6. THE PRELIMINARY FOLLOW THE SUN SOFTWARE PROCESS MODEL

In this chapter, I present results obtained in Phase 2 - Development of the research design (see Figure 3) in which I present the preliminary FTS software process model, named FTS-SPM (Follow the Sun Software Process Model). The preliminary FTS-SPM was built based on best practices identified in Subphase 1 and lessons learned from Subphase 2 (see Chapters 4 and 5). In Section 6.1, I describe the structure of the preliminary FTS-SPM. In Section 6.2, I discuss best practices included in the sub-processes and the reasons for including each sub-practice. In this section, I also present an activity diagram for the development of each sub-process. In Section 6.3, I summarize this chapter.

6.1 Structure of the Preliminary FTS-SPM

Through an in-depth analysis of results from subphases 1 and 2, significant themes with direct correlation with best practices and lessons learned emerged. To make sense, these themes were synthesized as sub-processes in the preliminary FTS-SPM (see Table 10).

The preliminary FTS-SPM comprises SP01 - Team Setup, SP02 - Project Planning, SP03 - Communication Protocol, SP04 - Cultural Training, SP05 - Task Allocation, and SP06 - Handoff Meeting. The preliminary FTS-SPM overview is presented in Figure 10.

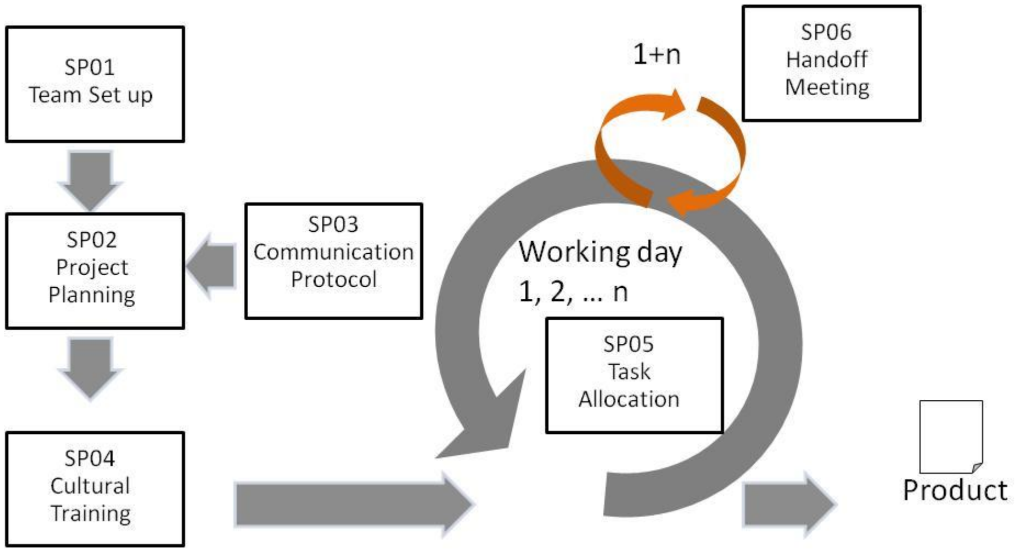


Figure 10 – Overview of the Preliminary FTS-SPM.

SP01 starts the process model. SP02 and SP03 are started following SP01. In SP02 the project planning is defined. SP03 defines communication resources and the schedule for synchronous communication between sites. SP04 is started following SP02. SP04 develops cultural training sessions in order to establish trust between team members. At the beginning of each working day, SP05 is developed. SP05 provides tasks for the day. A software project may have many working days. SP06 is started following SP05. SP06 aims to receive and to transfer tasks in progress, new tasks and project updates. At the beginning and at the end of each working day shift, SP06 is developed. One working day may have at least two working day shifts. The process finishes when at the end of a working day shift, there are no more tasks to develop.

Sub-processes are developed based on best practices. These best practices were included into sub-processes based on literature (subphase 1) and lessons learned (subphase 2) (see Table 9). Contradictory best practices to lessons learned (subphase 2) were not included in the model. Twenty-five best practices comprise the preliminary FTS-SPM, as shows Figure 11.

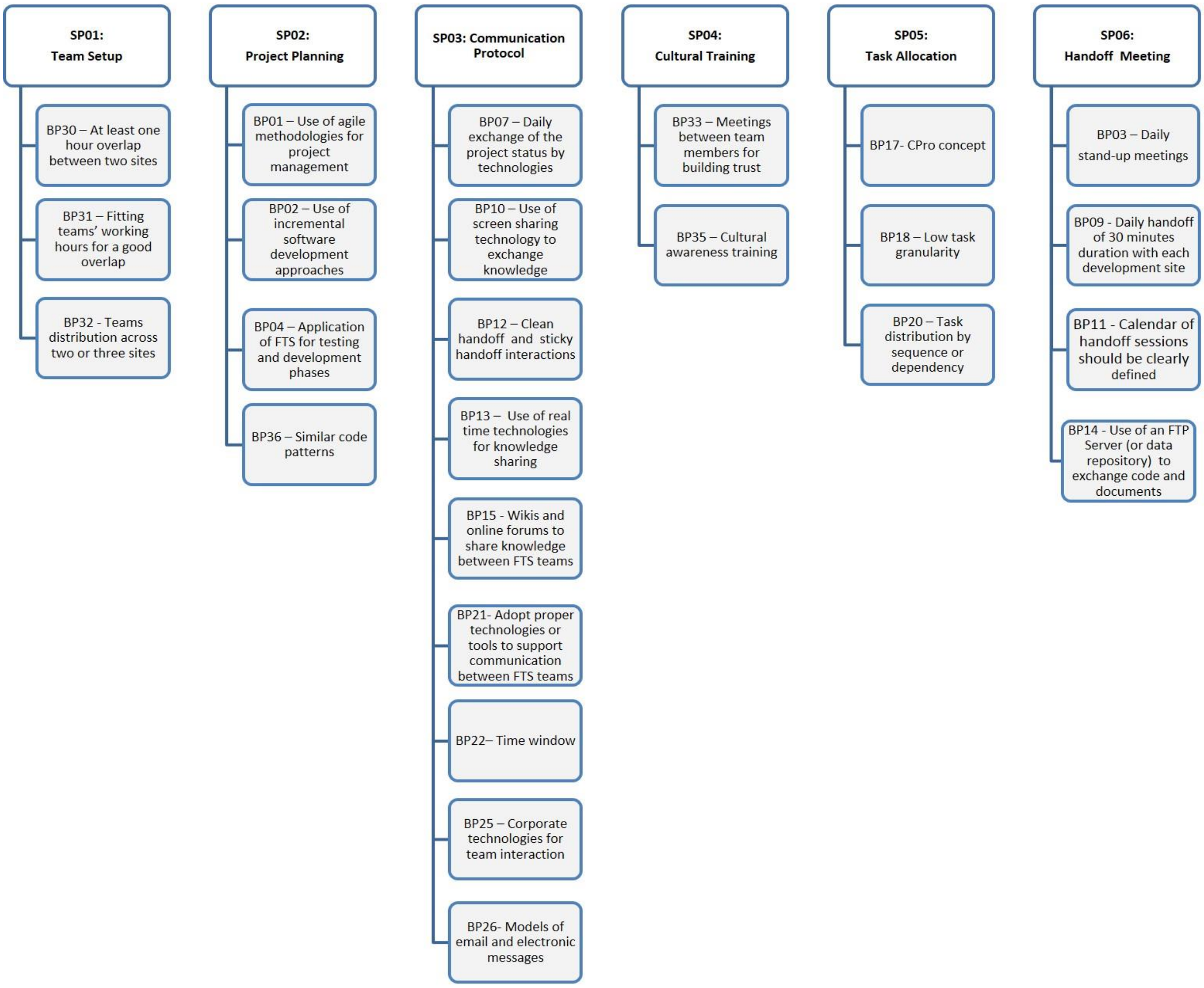

SP01: Team Setup
BP30 – At least one hour overlap between two sites
BP31 – Fitting teams' working hours for a good overlap
BP32 - Teams distribution across two or three sites
SP02: Project Planning
BP01 – Use of agile methodologies for project management
BP02 – Use of incremental software development approaches
BP04 – Application of FTS for testing and development phases
BP36 – Similar code patterns
SP03: Communication Protocol
BP07 – Daily exchange of the project status by technologies
BP10 – Use of screen sharing technology to exchange knowledge
BP12 – Clean handoff and sticky handoff interactions
BP13 – Use of real time technologies for knowledge sharing
BP15 - Wikis and online forums to share knowledge between FTS teams
BP21- Adopt proper technologies or tools to support communication between FTS teams
BP22– Time window
BP25 – Corporate technologies for team interaction
BP26- Models of email and electronic messages
SP04: Cultural Training
BP33 – Meetings between team members for building trust
BP35 – Cultural awareness training
SP05: Task Allocation
BP17- CPro concept
BP18 – Low task granularity
BP20 – Task distribution by sequence or dependency
SP06: Handoff Meeting
BP03 – Daily stand-up meetings
BP09 - Daily handoff of 30 minutes duration with each development site
BP11 - Calendar of handoff sessions should be clearly defined
BP14 - Use of an FTP Server (or data repository) to exchange code and documents

Figure 11 - Sub-processes and best practices.

6.2 The Preliminary FTS-SPM

As previously described, best practices were included into sub-processes based on literature (subphase 1) and lessons learned (subphase 2). This information is summarized in Table 9. Each sub-process and its best practices are described in details under the following sub-sections. Additionally following the sub-sections, for each sub-process, an activity diagram for its development is presented.

Table 9 – Mapped best practices into sub-processes in the preliminary FTS-SPM.

Sub-process (SP)	Best practice (BP)	Best Practice (BP) title	Included based on:
SP01: Team Setup	BP30	At least one hour overlap between two sites	[DES09] [HOL06]
	BP31	Fitting teams' working hours for a good overlap	[DES09] [HOL06]
	BP32	Teams distribution across two or three sites	[SOL10]
SP02: Project Planning	BP01	Use of agile methodologies for project management	[GUP] [CAR11] [CAR10] [YAP05] Lesson 1
	BP02	Use of incremental software development approaches	[HES12]
	BP04	Application of FTS for testing and development phases	[CAR10]
	BP36	Similar code patterns	Lesson 2
SP03: Communication Protocol	BP07	Daily exchange of the project status by technologies	Lesson 4 Lesson 5
	BP10	Use of screen sharing technology to exchange knowledge	Lesson 4 Lesson 5
	BP12	Clean handoff and stocky handoff interactions	[VIS09]
	BP13	Use of real time technologies for knowledge sharing	Lesson 4 Lesson 5
	BP15	Wikis and online forums to share knowledge between FTS teams	[GUP12]
	BP21	Adopt proper technologies or tools to support communication between FTS teams	Lesson 4 Lesson 5
	BP22	Time window	[TAN11] [LIN07]
	BP25	Corporate technologies for team interaction	Lesson 4 Lesson 5
	BP26	Models of e-mails and electronic messages	[GOR96]
SP04: Cultural Training	BP33	Meetings between team members for building trust	[SET07]
	BP35	Cultural awareness training	[TRE07]
SP05: Task Allocation	BP17	CPro concept	Lesson 6 Lesson 9
	BP18	Low task granularity	[ESP03] Lesson 6
	BP20	Task distribution by sequence or dependency	[JAL06] Lesson 6

SP06: Handoff Meeting	BP03	Daily stand-up meetings	[ESP07]
	BP09	Daily handoff of 30 minutes duration with each development site	[HES12]
	BP11	Calendar of handoff sessions should be clearly defined	[DES09]
	BP14	Use of an FTP Server (or data repository) to exchange code and documents	[CDE09] [TAW06] [RAM02]

6.2.1 Sub-process: SP01 - Team setup

Team members are geographically distributed in different time zones in FTS development [CAR11]. To Taweel and Brereton [TAW06], large software companies have a number of sites localized in different countries around the world. On the other hand, small companies tend to create virtual companies with partners in different countries. However, in both cases the potential of development time reduction depends on the number of sites involved and time zone differences between them.

The number of sites recommended to implement FTS is cited by Holmstrom et al. [HOL06] and Solingen and Valkema [SOL10]. To Holmstrom et al. [HOL06], the maximum number of sites to implement FTS depends on the working day size and its time zone differences. To Solingen and Valkema [SOL10], when the number of sites is more than two, the working becomes more difficult. In the case study performed at Infosys (see Chapter 5), team members were distributed in three different sites: Australia, India and México. Results showed team distribution in three sites suitable for FTS development. Furthermore, time zone differences between these sites were appropriate for FTS. This result confirmed outcomes found by Solingen and Valkema [SOL10]. Thus, *BP32 - Teams distributed across two or three sites* was included in *SP01- Team setup.*

Holmstrom et al. [HOL06] cite the strategy used by Intel for time zone management. Intel keeps flexible different time zones setting team office hours to obtain a good overlap between sites. The Intel's strategy is to make time zone differences manageable. Infosys used the same Intel's strategy to obtain a good overlap between sites. Thus, team members were able to perform synchronous communication between sites during the project. Studies performed by Deshpande and Richardson [DES09] and Holmstrom et al. [HOL06] confirm the results obtained in the case study. It justifies to include *BP30 - At least one hour overlap between two sites* and *BP31 - Fitting teams' working hours for a good overlap* in the SP01.

Figure 12 presents an activity diagram to develop SP01 based on best practices included.

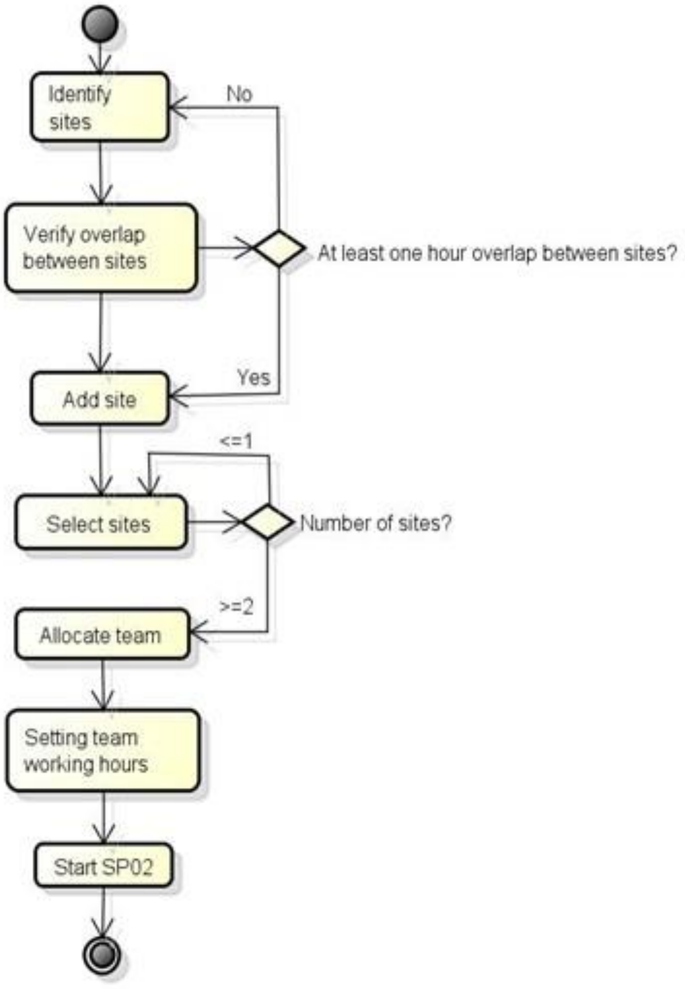

Identify sites
No
Verify overlap between sites
At least one hour overlap between sites?
Yes
Add site
<=1
Select sites
Number of sites?
>=2
Allocate team
Setting team working hours
Start SP02

Figure 12 - SP01 Activity diagram.

SP01 is developed in six main activities. These activities are developed by the PM. Each activity is described as follows.

- **Identify sites**

This activity aims to identify available sites to develop a FTS project. Information about each site should be collected in order to make future decisions. It is important to verify if there are staff, cost or scope restrictions in each site. These restrictions and others related to the project goals should be considered to define priorities in order to select appropriate sites.

- **Verify overlap between sites**

This activity verifies overlaps differences between sites in order to identify the best configuration for a FTS project. Its development is based on *BP30 - At least one hour overlap between two sites*. As recommended by BP30, sites should be separated by at least one hour difference. Its implementation allows synchronous communication between team members during handoff sessions. Furthermore, overlap management at the beginning of the project may reduce communication and coordination problems in FTS projects [DES09].

- **Add site**

  This activity aims to build a database of available sites for the project.

- **Select sites**

  Once available sites for the project were identified, the next step is to select some of them. Thus, this activity aims to select appropriate sites to develop the project. Its development is based on *BP32 - Teams distribution across two or three sites.* BP32 recommends selecting two or three sites to develop a FTS project. To select appropriate sites, time zone differences, overlap time between sites, staff allocation with the best profile and technical, business and behavioral skills should be considered [DES09].

- **Allocate team**

  This activity aims to allocate human resources to the project. Team allocation has to consider staff profile and previous experience in technologies adopted in the project [DES09].

- **Setting team working hours**

  This activity aims setting team's working hours to obtain a good overlap between sites. Its development is based on *BP31 - Fitting teams' working hours for a good overlap.*

  BP31 recommends making time zone manageable setting team working hours. Each site has its own working hours according to governmental laws or internal politics. Changes in team's working hours should allow establishing time windows for synchronous communication between sites. This best practice also contributes to performing daily handoffs.

- **Start SP02**

  This activity indicates the end of SP01 and the beginning of *SP02 - Project planning.* Deliveries from SP01 are used to develop SP02.

6.2.2 Sub-process: SP02 - Project Planning

In literature, agile methodologies such as Scrum and XP are the most recommended to develop FTS. To Gupta et al. [GUP12], Carmel, Espinosa, and Dubinsky [CAR10] and Yap [YAP05], agile methodologies contribute to increase speed in software development. FTS focus to reduce software development cycle duration or reduce time-to-market [CAR11]. Thus, agile methodologies address the FTS main goal. The *Lesson 1 - The importance of agile methodologies for FTS* identified in the subphase 2 contribute with results found in the subphase 1. These results justify to include *BP01 - Use of agile methodologies for project management* in SP02.

BP02 - Use of incremental software development approaches came from agile methodologies. To Hess and Audy [HES12], incremental approaches like TDD (Test Driven Development) contribute in performing daily handoff meetings. Thus, BP02 is also included in SP02.

BP04 - Application of FTS for testing and development phases is recommended by Carmel, Espinosa, and Dubinsky [CAR10]. To these authors, testing and development phases are recommended to FTS because handoffs are structured and granulate. Thus, the Carmel, Espinosa and Dubinsky [CAR10] study justify including BP04 in SP02.

BP36 - Similar code patterns was also included in SP02. *Lesson 2 - Templates and standard documents* from subphase 2 show the importance of code patterns for software development and to improve team's performance. BP36 also helps teams to identify last changes made in the code. Thus, BP36 was included in SP02 based on Lesson 2.

SP02 is developed based on six activities as shown in Figure 13. SP02 is developed by the PM. These activities are based on four best practices included in SP02 as presented in Figure 11.

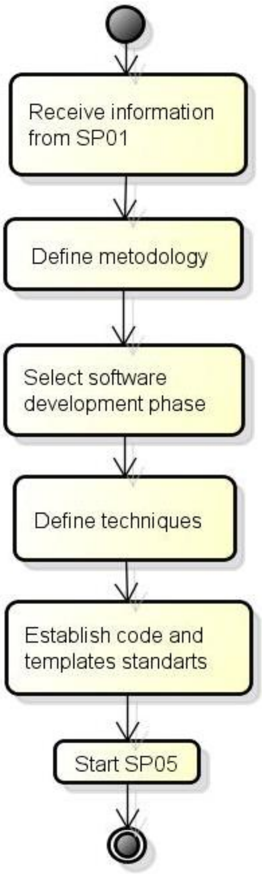

Receive information from SP01
Define metodology
Select software development phase
Define techniques
Establish code and templates standarts
Start SP05

Figure 13 - SP02 Activity diagram.

1. **Receive information from SP01**

SP01 provides information to develop SP02. The project manager uses this information to create the project planning.

2. **Define methodology**

This activity aims to define a methodology to be followed in the project. Its development is based on *BP01 - Use of agile methodologies for project management*. BP01 recommends agile methodologies such as Scrum and XP for FTS development. However, some aspects such as number of members and experience level should be considered before choosing a methodology for the project [GUP12].

3. **Select software development phase**

This activity aims to select a phase of SDLC to apply FTS. FTS strategy can be applied in all phases of the SDLC, but in some phases its implementation becomes more difficult [CAM04]. Thus, this activity implements *BP04 - Application of FTS for testing and development phases.* BP04 recommends testing and development phases for FTS development.

4. **Define techniques**

This activity aims to define techniques to support FTS. Its development is based on *BP02 - Use of incremental software development approaches.* This best practice recommends techniques for code development based on short development iterations [GUP07].

5. **Establish code and template standards**

The PM should define code and template standards before the project start. These standards will be used by team members to develop tasks. Thus, this activity is developed based on *BP36 - Similar code patterns*

For the BP36 implementation is necessary to define patterns and how comments will be inserted into the code by the team. All team members must be able to understand the last changes made in the code from the last handoff meeting.

Use of code and template standards contributes to document information produced by the team over the project [TAW06]. Templates must be completed according to previous recommendations. Teams also must avoid putting incomplete or irrelevant information.

6. **Start SP05**

This activity indicates the ending of SP02 and the beginning of *SP05 - Task Allocation.*

6.2.3 Sub-process: SP03 - Communication Protocol

SP03 - Communication protocol includes nine best practices, as shows Figure 11. Five of them, recommend the use of technologies for FTS*: BP07 - Daily exchange of the project status by technologies, BP10 - Use of screen sharing technology to exchange knowledge, BP13 - Use of real time technologies for knowledge sharing* and *BP21 - Adopt proper technologies or tools to support communication between FTS teams* and *BP25 - Corporate technologies for team interaction*. These best practices were included in SP03 based on *Lesson 4 - Screen sharing to transfer knowledge* and *Lesson 5 - Communication resources*. These lessons discuss the use of technologies to increase collaboration and reduce communication difficulties between team members in FTS projects.

SP03 includes *BP12 - Clean handoff and sticky handoff interactions* for handoff management. BP12 is recommended by Visser and Solingen [VIS09]. Its goal is to establish short interactions between FTS teams in order to make handoff meetings more productive and shorter.

In FTS projects, team members may have different experience levels and skills. The imbalance experience level may have a negative effect on the project. Thus, *BP15 - Wikis and online forums to share knowledge between FTS teams* was included in SP03. Its main purpose is to identify challenges faced by FTS teams and share solutions. Wiki and on-line forums aim to provide opportunities for discussing a particular issue from the project, where each team member can give your contribution. BP15 is recommended by Gupta et al. [GUP12]. To these authors, the main benefit of BP15 implementation is to provide information to teams from different locations.

BP22 - Time window was also included in SP03. This best practice provides synchronous communication between team members from different sites, without previous schedule. Moreover, it allows establishing daily communication between team members. It is recommended for FTS by Tang et al. [TAN11] and Lings et al. [LIN07].

E-mail is a common communication resource for FTS teams. However, if e-mails are not properly managed, some information may be lost. Some e-mails or electronic messages have excess or lacking of information. In both cases, excess or lacking of information may have negative consequences during the project development. In addition, the e-mail or electronic message flow increases in FTS development and its management become more difficult. Gorton, Hawryszkiewycz, and Fung [GOR96] recommend *BP26 - Models of e-mail and electronic messages* to manage information between team members. Thus, BP26 was included in SP03 based on literature recommendations.

Activities to develop SP03 are presented in Figure 14. SP03 activities are developed by the PM. Each activity are described next.

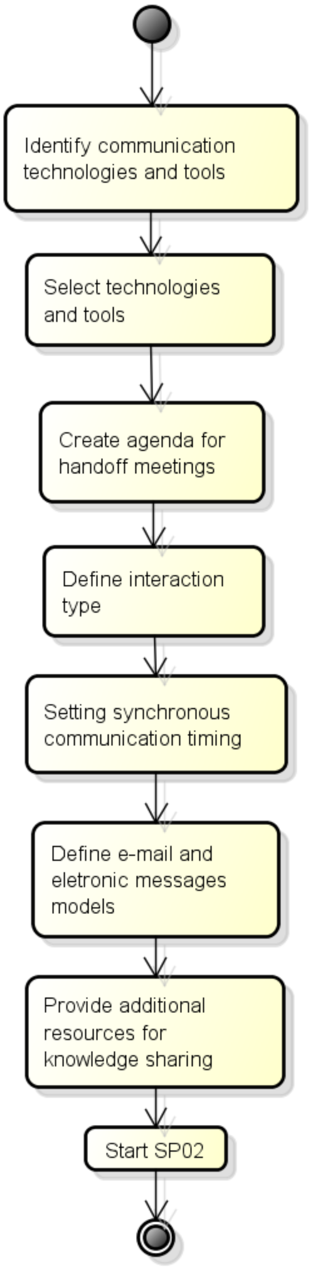

Identify communication technologies and tools
Select technologies and tools
Create agenda for handoff meetings
Define interaction type
Setting synchronous communication timing
Define e-mail and eletronic messages models
Provide additional resources for knowledge sharing
Start SP02

Figure 14 - SP03 Activity diagram.

1. Identify communication technologies and tools

This activity aims to identify available technologies and tools to support FTS development. The project manager can suggest technologies or tools already used in other projects.

2. Select technologies and tools

There are many available technologies and tools to perform communication in distributed projects. However, hardware compatibility between sites and reliability of these technologies and tools should be evaluated in first. Technologies and tools usually used by companies to perform communication between distributed teams are telephone calls, video conference, instant message software and others [JAL09]. However, many technologies and tools may be not appropriate to FTS projects. Thus, this activity aims to select appropriate technologies and tools for FTS development. This activity is developed based on *BP21 - Adopt proper technologies or tools to support communication between FTS teams* and *BP25 - Corporate technologies for team interaction.*

3. Create agenda for handoff meetings

To ensure that all team members will attend meetings, an agenda for handoff meetings should be created. Thus, this activity aims to define an agenda to perform handoff meetings during the project. Its development is based on *BP11 - Calendar of handoff sessions should be clearly defined.*

BP11 define an agenda for handoff meetings considering team members office time differences. To set handoff timing, time restrictions should be considered in order to ensure that all team members are available to attend the meetings.

4. Define interaction type

This activity aims to define the interaction type that will be adopted by teams during the project. Its development is based on *BP12 - Clean handoff and sticky handoff interactions.*

Clean handoff interactions are short and it can be used to discuss punctual topics. Sticky handoff interactions are intense and it is recommended to discuss issues related to project planning. Both can be applied to FTS contexts and it can be used together [VIS09].

5. Setting synchronous communication timing

This activity aims to define a schedule for synchronous communication between team members.

6. Define e-mail and electronic message models

Information exchanged by e-mail or electronic messages may be intensified in FTS projects. It occurs mainly due to the lacking of opportunities for synchronous communication. Thus, this activity aims to define models for e-mail and electronic messages. Its development is based on *BP26 - Models of e-mail and electronic messages*.

BP26 recommends creating models for each type of message. Each type of model should have a purpose and fields to describe typical information pertaining to each e-mail or message.

7. **Provide additional resources for knowledge sharing**

This activity aims to provide resources for knowledge sharing among FTS teams. Its development is based on *BP10 - Use of screen sharing technology to exchange knowledge, BP13 - Use of real time technologies for knowledge sharing* and *BP15 - Wikis and online forums to share knowledge between FTS teams*.

BP10 recommends the use of screen sharing technologies as an additional resource to exchange knowledge between teams. Additionally, BP13 recommends the using of real time technologies to support teams in FTS projects. In addition, knowledge sharing among FTS teams can be performed by a wiki or online forums. These additional resources provide benefits such as an internal knowledge database that can be used by other FTS teams in other FTS projects.

8. **Start SP02**

This activity indicates the end of SP03 and the beginning of the *SP02 – Communication protocol.*

6.2.4 Sub-process: SP04 - Cultural Training

FTS teams face many challenges related to cultural diversity. Some examples are language, national and political differences, individual perceptions and motivation, and work ethics [DES10]. Holmstrom et al. [HOL06] argue that when constraints such as temporal, geographical and socio-cultural distance are identified and while they increase in the scope of organizational operation, these constraints result in challenges for FTS projects. For example, usual problems like supporting collaboration are compounded by language and diversity [CAR04]. Yap [YAP05] argue that cultural differences often created misunderstandings and lead to frustration and conflicts between teams.

Setamanit, Wakeland, and Raffo [SET07] and Treinen and Miller-Frost [TRE06] studies discuss cultural aspects involving FTS teams. Setamanit, Wakeland, and Raffo [SET07] recommend *BP33 - Meetings between team members for building trust* and Treinen and Miller-Frost [TRE06] recommends *BP35 - Cultural awareness training* for FTS. These best practices aim to establish or re-establish trust among team members and reduce problems related to cultural differences. Thus, these best practices were included in SP04 based on literature recommendations.

Activities to develop SP04 are presented in Figure 15. SP04 is developed by the PM. Each activity is described next.

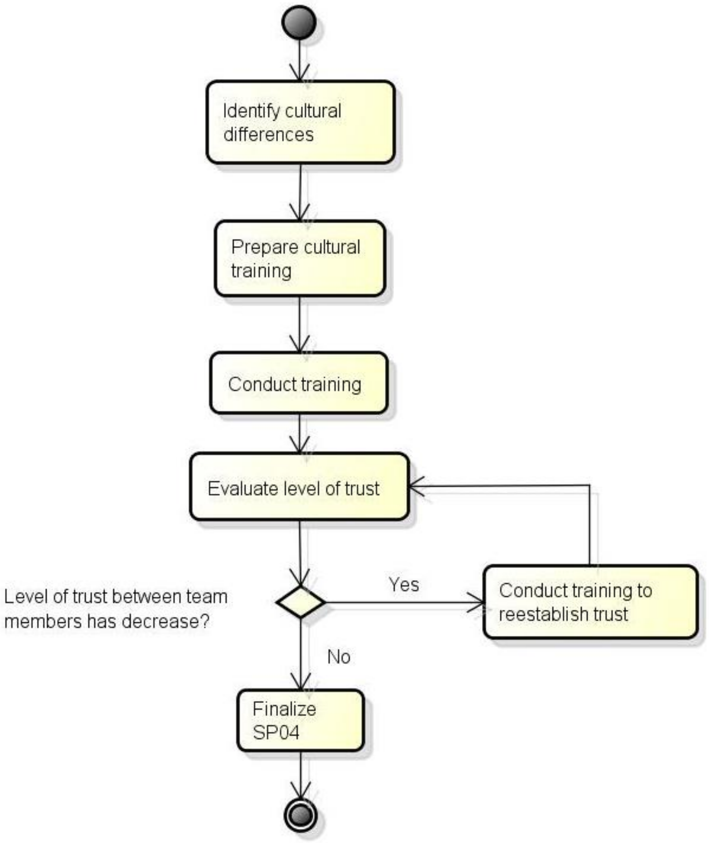

Identify cultural differences
Prepare cultural training
Conduct training
Evaluate level of trust
Level of trust between team members has decrease?
Yes
Conduct training to reestablish trust
No
Finalize SP04

Figure 15 - SP04 Activity diagram.

1. Identify cultural differences

Team members may have different cultures in FTS projects. It may affect the level of trust between team members. Thus, this activity aims to identify cultural differences between team members in order to minimize it.

Cultural differences are mainly related to languages, organizational operations, traditions (national traditions, values and norms of behavior), laws and legislation differences [HOL06].

2. Prepare training

This activity aims to plan cultural trainings to be conducted during the project. Its development is based on *BP35 - Cultural awareness training.*

Each software project has different characteristics such as team setup. Thus, the project manager should create a plan according to the information collected in the "*Identify cultural differences"* activity*.*

3. **Conduct training**

This activity implements *BP35 - Cultural awareness training.* BP35 recommends developing meetings before the project start with all team members. Issues related to work style and schedules differences should be discussed in order to eliminate potential issues.

4. **Evaluate level of trust**

The level of trust among team members may decrease during the project [SET07]. Therefore, this activity aims to evaluate the level of trust of the team during the project. Each organization can adopt the most suitable evaluation tool to identify the level of trust. Jalali and Zlatkovic [JAL09] describe causes of lacking or losing trust.

If the level of trust has decreased, new training meetings should be performed in order to re-establish it. This activity is developed based on *BP33 - Meetings between team members for building trust.* The output of this activity indicates if the *"Conduct training to re-establish trust"* activity should be performed.

5. **Conduct training to re-establish trust**

This activity is developed if the output from "*Evaluate level of trust"* activity indicates the level of trust has been decreased during the project. Its main goal is to recover the level of trust between team members. Its development is based on *BP33 - Meetings between team members for building trust.*

6. **Finalize SP04**

This activity indicates the end of the SP04. SP04 may be developed many times during the project.

6.2.5 Sub-process: SP05 - Task Allocation

Team members are distributed across different sites and time zones in FTS projects. When a working team finishes its own regular working hours, another team located in another location and time zone starts its workday. Unfinished tasks are handed from one team to another by the end of each working day [CDE09]. At the beginning and ending of each working day, handoff meetings are performed to discuss future tasks and tasks in progress.

Clearly, to maximize the reduction in completion time over 24 hours development, task allocation in FTS projects needs to be done carefully [JAL06]. Each development site may have different process steps depending on how tasks are allocated and specific activities are performed [SET07].

Hence, based on literature recommendations and lessons learned in the case study, three best practices were included in SP05. These best practices are *BP17 - CPro concept, BP18 - Low task granularity,* and *BP20 - Task distribution by sequence or dependency.*

BP17 was included in SP01 based on *Lesson 6 - Tasks for the day* and *Lesson 9 - CP owner.* These lessons learned show advantages of BP17 to implement FTS. The main advantage is highly structured work.

BP18 was included in SP05 based on the Espinosa and Carmel [ESP03] study and on *Lesson 6 - Tasks for the day*. Espinosa and Carmel [ESP03] argue that tasks of low levels of granularity may enhance recognition accuracy. In addition, Lesson 6 emphasizes task allocation for the day as a solution to define priorities and reduce problems faced by teams to categorize tasks. BP18 recommends dividing tasks into small parts contributing to allocate tasks for the day.

BP20 also was included in SP05 based on literature recommendations and Lesson 6. To Jalote and Jain [JAL06] since tasks may have some relationship among them, they cannot be executed independently. Lesson 6 reports problems faced by team members to define task's priority. Therefore, BP20 contributes to define sequence and dependency among tasks in FTS development.

Activities to develop SP05 are presented in Figure 16. SP05 activities are developed by the PM. Each activity is described next.

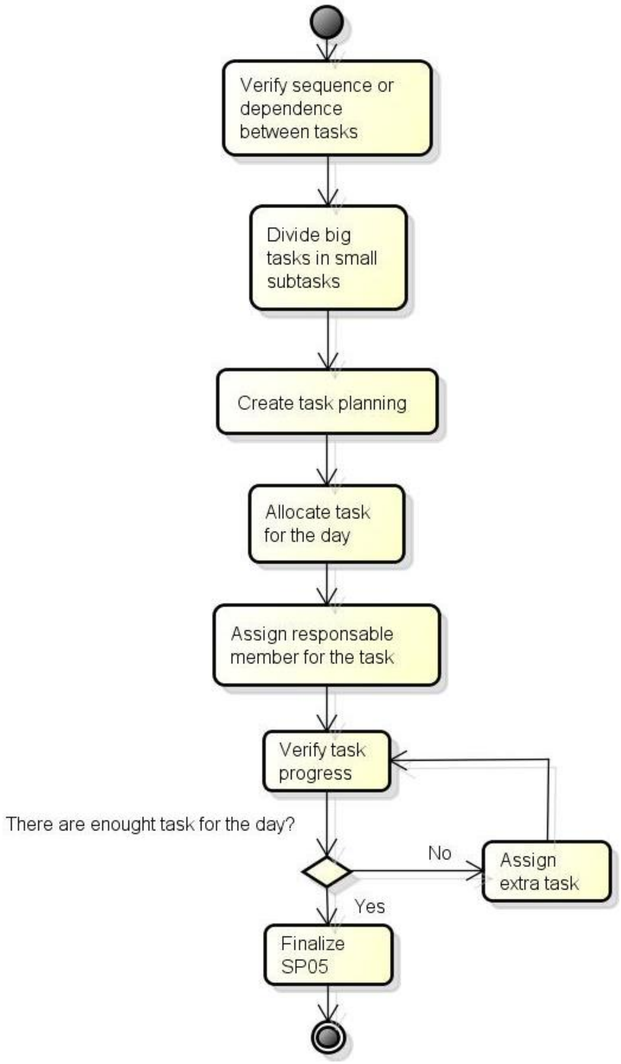

Verify sequence or dependence between tasks
Divide big tasks in small subtasks
Create task planning
Allocate task for the day
Assign responsable member for the task
Verify task progress
There are enought task for the day?
No
Assign extra task
Yes
Finalize SP05

Figure 16 - SP05 Activity diagram.

1. Verify sequence or dependency between tasks

Some tasks have relationships one to another and they cannot be separately developed [TAW06]. Therefore, this activity aims to verify sequence and dependency relationships between tasks. Its development is based on *BP20 - Task distribution by sequence or dependency.*

To develop this activity, sequence and dependency relationships between tasks must be identified. All tasks sequence and dependency details should be described in the project planning.

2. Divide big tasks in small subtasks

Some tasks take many hours or days to be developed by team members. Therefore, this activity aims to divide big tasks, which will take many hours or days of effort to be developed, in small parts. Its purpose is to make easier to manage tasks. This activity is implemented based on *BP18 - Low task granularity.*

3. Create task planning

This activity aims to develop the task allocation plan. This planning should describe tasks details such as time effort, sequence or dependency between tasks, team members assigned to develop each task.

This activity implements *BP17 - CPro concept*. BP17 define CPs (Composite Persona) for each site. At least one team member from each site forms each CP. CPs formation is carried out by the project manager. Its goal is to set team members in CPs.

4. Allocate tasks for the day

The project manager carries out this activity. The project manager must send a daily e-mail allocating tasks to each CP and not to individual members. Each CP must receive a daily e-mail with tasks for the day. Its development is based on *Lesson 6 - Task for the day*. This activity is based on *BP17 - CPro concept*.

5. Assign a responsible member for the task

This activity aims to ensure that tasks assigned to CPs will be finished before another task start. Its development is based on *Lesson 9 - CP owner.*

The project manager assigns one team member of each CP as responsible for a specific task. This CP member must report to the project manager when his/her task have finished.

6. Verify task progress

Tasks allocated in daily e-mails could be not enough for a 24-hour working day. Therefore, this activity aims to verify the task progress. If it is observed that tasks allocated for the day will be finished with less than estimated time, the "*Assign extra task*" activity is performed.

7. Assign extra task

This activity is carried out by the project manager in order to ensure sufficient tasks for the day. An e-mail is sent allocating new tasks to each CP.

8. Finalize SP05

This activity indicates the end of SP05. SP05 may be developed many times during the project. This sub-process finish when there are no more tasks to be allocated.

6.2.6 Sub-process: SP06 - Handoff Meeting

SP06 is developed at the beginning and end of each working day shift. Its main goal is to transfer new or unfinished tasks to other team members localized in a different site [CAR11]. Its development is based on four best practices as shows Figure 11.

BP03 - Daily stand-up meetings and *BP11 - Calendar of handoff sessions should be clearly defined* aim to support handoff activities. BP03 was included in SP06 to emphasize daily communication between teams [ESP07] and BP11 to provide daily interactions between teams [DES09]. BP03 and BP11 were included in SP06 based on literature recommendations.

BP09 - Daily handoff of 30 minutes duration with each development site recommends handoffs meetings performed at maximum 30 minutes. This best practice was included in SP06 considering results obtained by Hess and Audy [HES12] work. To these authors, 30 minutes duration is enough to transfer and to discuss tasks.

SP06 also includes *BP14 - Use of an FTP Server (or data repository) to exchange code and documents*. This best practice recommends technologies for sharing documents and code. BP14 was included in SP06 based on studies performed by Carmel, Dubinsky, and Espinosa [CDE09], Taweel and Brereton [TAW06] and Ramesh and Dennis [RAM02]. Figure 17 shows the activity diagram to develop SP06. SP06 activites are developed by team members alocated to the project.

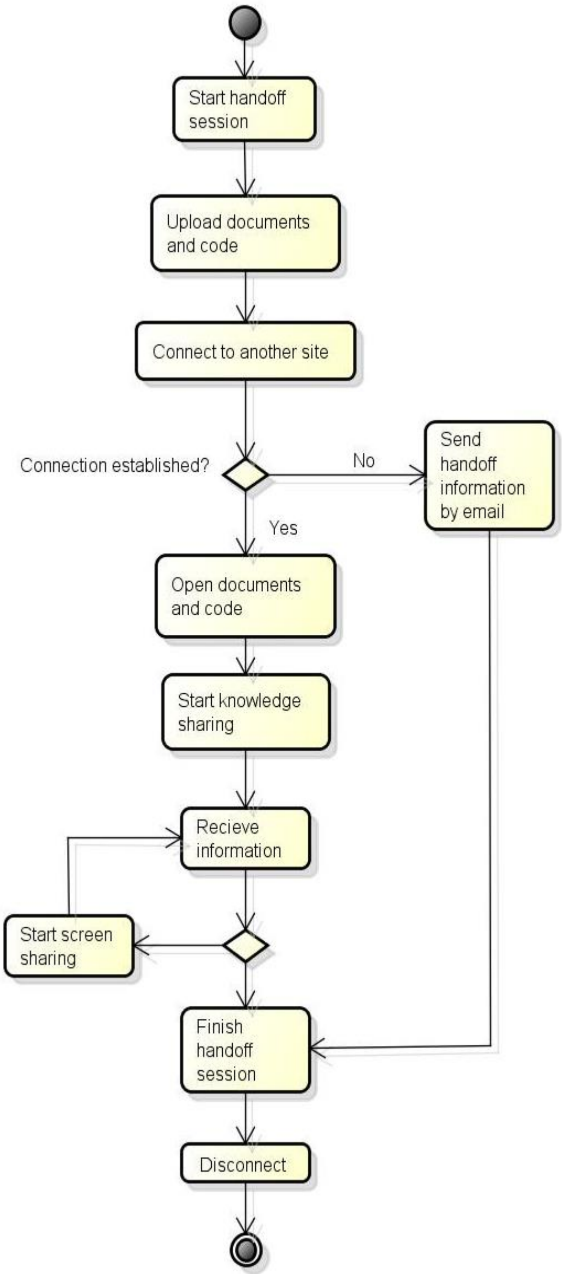


Figure 17 - SP06 Activity diagram.

1. **Start handoff session**

   This activity indicates the beginning of a handoff meeting between two sites.

2. **Upload documents and code**

A FTS team produces documents and code over the working day shift. These documents and code are transferred to the next team at the end of each working day shift. Thus, this activity aims to transfer documents and code from a site to another. This activity is developed by team members allocated to the project. Its development implements *BP14 - Use of an FTP Server (or data repository) to exchange code and documents.*

3. Connect to another site

This activity is carried out by team members from two sites that are performing a handoff meeting. Its purpose is to allow synchronous communication between sites. To develop this activity are used communication technologies and tool defined in the *SP03 - Communication protocol.*

4. Send handoff information by e-mail

Different sites have different hardware infrastructure. Due to these differences, connection problems may occur. Thus, this activity is developed if the connection is lost. This activity is based on *Lesson 10 - Extra e-mail in case of day missed from work or delays.*

5. Open documents and code

This activity aims to verify if documents and code were uploaded by team members from the previous site. Documents and code that will be used during handoff meetings should be kept open during handoff meetings.

6. Start knowledge sharing

At the end of a working day shift, unfinished or on progress tasks are transferred to another site. Information about these tasks is transferred during handoff meetings in order to team members to continue the working day. This activity is developed based on *BP09 - Daily handoff of 30 minutes duration with each development site.*

7. Receive information

This activity aims to receive handoff information. This information is used to give the continuity to the working day.

8. Start screen sharing

Screen sharing aims to make easier to exchange information between sites. During handoff meetings, team members may use screen sharing as an additional resource to improve the communication.

This activity is optional. Its development occurs when has a request from another site. This activity is developed based on *Lesson 4 - Screen sharing to transfer knowledge.*

9. Finish handoff session

This activity indicates the end of SP06. SP06 is developed at the beginning and end of each working shift. Thus, SP06 is developed many times during the working day. The process finish, when there are no more tasks to handover.

6.3 Chapter Summary

This chapter presented the preliminary FTS-SPM (Follow the Sun Software Process Model) that was built based on results from Phase 1 - Exploratory of the research design (see Figure 3). The preliminary model comprises twenty-five best practices mapped into six sub-processes.

Sub-processes emerged from a further analysis of the significant themes with direct correlation with best practices and lessons learned. Sub-processes in the preliminary model are developed based on its best practices. For each sub-process included in the model, an activity diagram for its development was presented. Activity diagrams were defined based on best practices.

Best practices were included into sub-processes based in the literature (Subphase 1) and lessons learned (Subphase 2). They support the main FTS characteristics and contribute to increase the probabilities of successfully implement FTS projects.

Next chapter presents the preliminary FTS-SPM evaluation following the research methodology adopted in this thesis. Its results will contribute to refine the preliminary model and to identify the usefulness and relevance of each best practice and sub-process defined in the model.

7. THE PRELIMINARY FTS-SPM EVALUATION AND EVOLUTION

In this chapter, I present results obtained in Phase 3 – Evaluation and Evolution of the research design (see Figure 3). The evaluation and evolution of the preliminary FTS-SPM was conducted in two stages. The first stage to address the preliminary FTS-SPM evaluation was made through the design validation. The main goal in performing the design validation was to improve the preliminary software process model design by collecting input from research experts. The design validation was carried out using the experts from Lero - The Irish Software Engineering Research Centre. The results from the design validation have been published in the paper "*Proposing a Software Process Model for Follow the Sun Development*", in the 26th International Conference on Software Engineering and Knowledge Engineering (SEKE 2014)[8].

The second stage to address the preliminary FTS-SPM evaluation was made through an expert panel. Specific questions were asked to a group of GSD experts to uncover the usefulness and relevance of each best practice and sub-process mentioned in the preliminary FTS-SPM. The evaluation proved helpful in highlighting the strengths and weaknesses in the FTS-SPM and providing further directions for improving the model. The preliminary results from the expert panel have been published in the paper "*FTS-SPM: A Software Process Model for Follow the Sun Development – Preliminary Results*", in the PARIS workshop[9] collocated with ICGSE 2014.

In the next sections, I describe how the design validation and the expert panel approach were conducted in the evaluation process. Section 7.1 presents the FTS-SPM evaluation made through the design validation. Section 7.2 presents the FTS-SPM evaluation approach made through an expert panel. Section 7.3 summarizes this chapter.

7.1 The Design Validation

The design validation was performed during the scholarship at Lero (see Section 3.3). Its main goal was to improve the preliminary software process model design (see Figure 10) by collecting input from research experts.

[8] The 26th International Conference on Software Engineering and Knowledge Engineering (SEKE 2014) was held in Vancouver, Canada from July 1 to July 3, 2014 (ISSN: 2325-9086 online).

[9] The PARIS: Methods and Tools for Project/Architecture/Risk Management in Globally Distributed Software Development Projects is a workshop collocated with ICGSE. The PARIS workshop was held during the 9th International Conference on Global Software Engineering (ICGSE) in Shanghai, China from August 18th to 21st, 2014. All papers presented at the conference venue were indexed by IEEE Computer Society (CSDL) digital libraries.

Data collection was performed in two stages. In the first stage, I had the opportunity to discuss the preliminary FTS-SPM with researchers and visiting researchers at the Annual NUIG-UL (National University of Ireland, Galway/University of Limerick) Research Day held in Galway (Ireland). There, I collected data by notetaking from feedback provided by research experts.

Based on data collected at NUIG-UL Research Day, I made some changes in the preliminary model as shown in Figure 18. The sequence flows between SP02, SP03, SP04, and SP05 were modified and an initial and final state was added to the process model. I also changed SP06's name and included arrows to show how the information moves through sub-processes. SP06 was called *Handoff Sessions*. These changes aimed to provide a better understanding in which sequence flow each sub-process is developed.

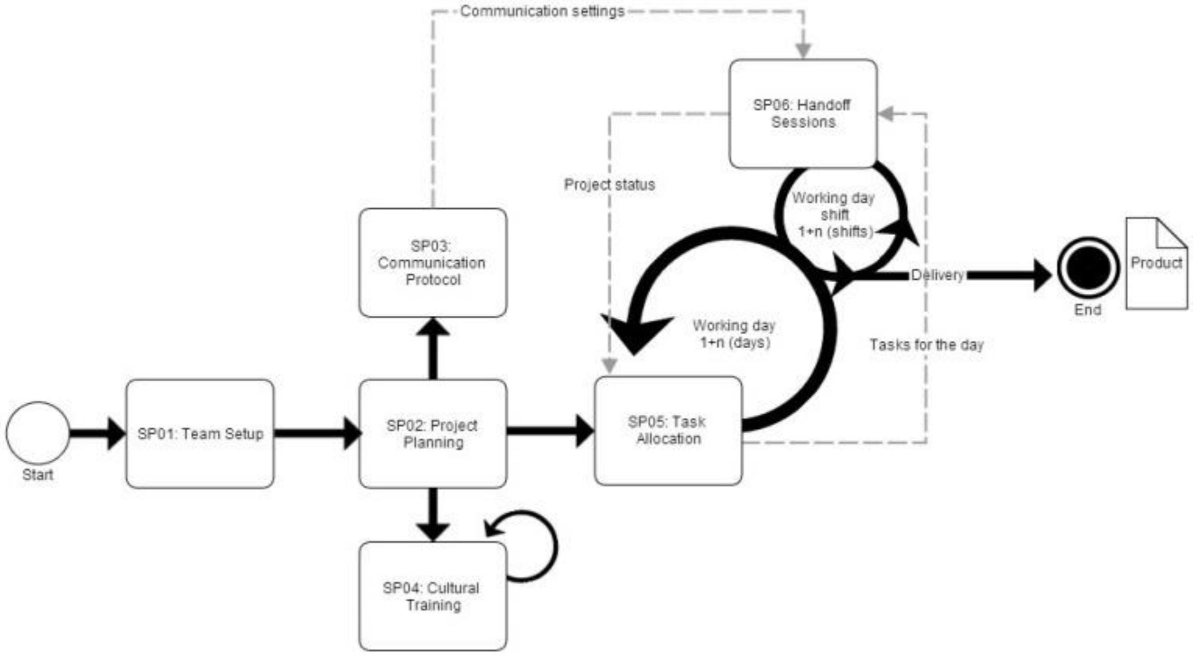
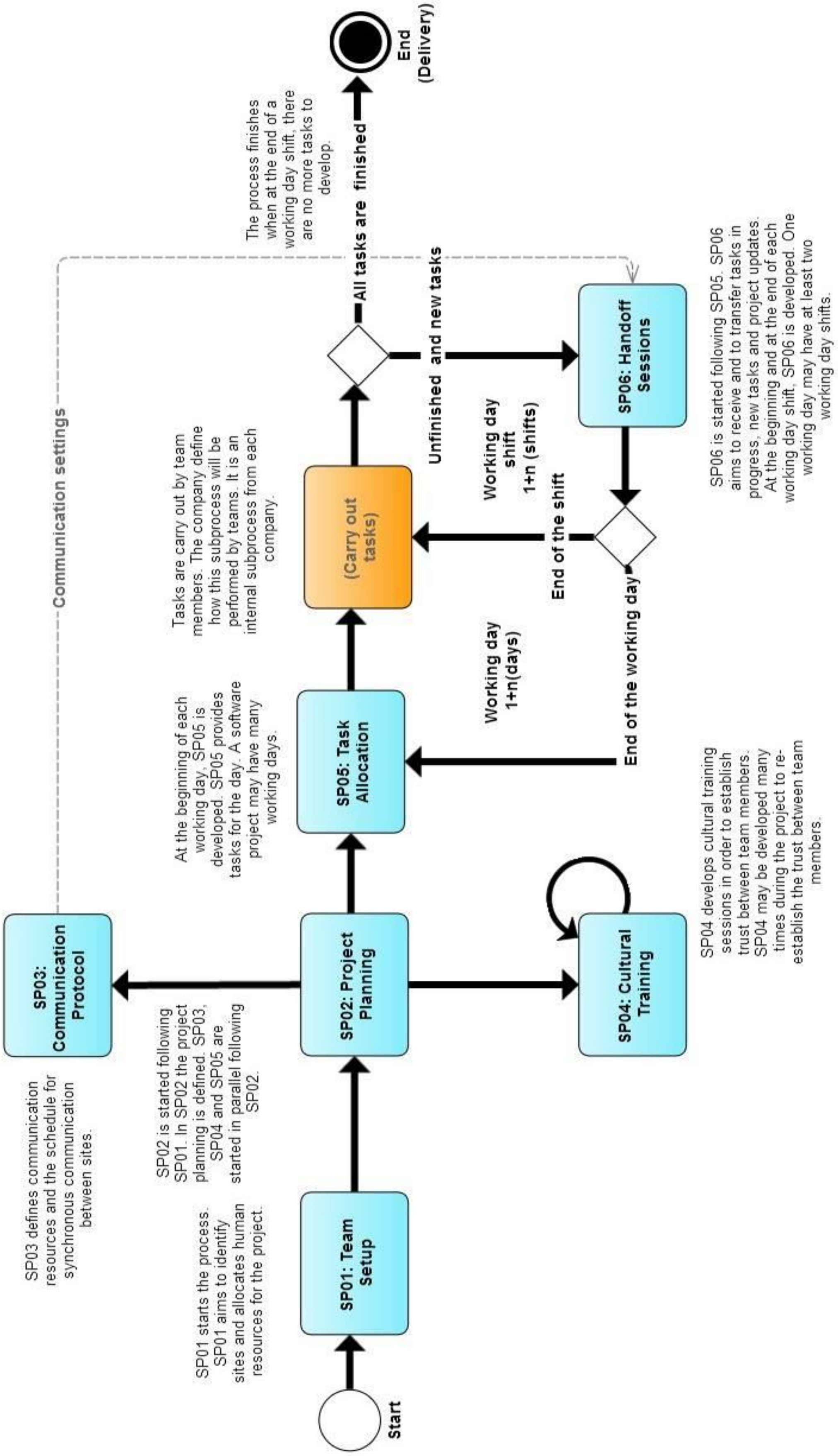


Figure 18 – The Preliminary FTS-SPM[i].

In the second stage, I presented the preliminary FTS-SPM[i] at Lero workshop. Workshops are conducted at Lero to present research in progress and its results. Eleven research experts from Lero participated of the design validation in this second stage. Five experts were research fellow and the other six were doctoral researchers. Data collection was performed by notetaking from feedback and further discussion with research experts.

During the Lero workshop some questions about the sequence flow between SP05 and SP06 emerged. The sequence flow between sub-processes was considered inadequate by the majority of experts from Lero. Following recommendations from Lero researchers, changes were made to the preliminary FTS-SPM[i] design. I included a square called ‘Carry out tasks’. This square represents an internal sub-process of the organization. Each organization defines how it should be executed. I also included two diamonds for the decision point between SP05 and SP06. In the first diamond, the process can finish if all tasks are finished or can start SP06, if there are unfinished or new tasks to transfer to another site. In the second diamond, a new working day shift starts if the end of the shift is or SP05 starts if is the end of the working day. These changes were made to support FTS characteristics. The preliminary FTS-SPM[ii] is shown in Figure 19.

As shown in Figure 19, the model stars in SP01: Team Setup. SP02: Project planning is started following SP01. SP03, SP04 and SP05 are started in parallel following SP02. SP03 defines communication settings to SP06. SP04 develops cultural training sessions. SP04 may be developed many times during the project to re-establish the trust between team members (loop arrow). At the beginning of each working day, SP05 is undertaken, as it provides tasks for the day. SP06 is started following SP05. SP06 aims to receive and to transfer tasks in progress, new tasks and project updates. At the beginning and at the end of each working day shift, SP06 is undertaken. One working day may have at least two working day shifts. The process finishes when at the end of a working day shift, there are no more tasks to develop.



SP01 starts the process. SP01 aims to identify sites and allocates human resources for the project.

SP02 is started following SP01. In SP02 the project planning is defined. SP03, SP04 and SP05 are started in parallel following SP02.

SP03 defines communication resources and the schedule for synchronous communication between sites.

SP04 develops cultural training sessions in order to establish trust between team members. SP04 may be developed many times during the project to re-establish the trust between team members.

At the beginning of each working day, SP05 is developed. SP05 provides tasks for the day. A software project may have many working days.

Tasks are carry out by team members. The company define how this subprocess will be performed by teams. It is an internal subprocess from each company.

SP06 is started following SP05. SP06 aims to receive and to transfer tasks in progress, new tasks and project updates. At the beginning and at the end of each working day shift, SP06 is developed. One working day may have at least two working day shifts.

The process finishes when at the end of a working day shift, there are no more tasks to develop.

Figure 19 - The preliminary FTS-SPM[ii].

7.1.1 The Design Validation Contributions

The design validation conducted for the FTS-SPM evaluation offered two main contributions to this research. The first contribution is related to improvements made to the sequence flow between sub-processes in the preliminary process model design. By conducting the design validation was possible to collect data from experts to rearrange the sequence flow between sub-processes making it into a more acceptable position in the model. Consequently, it promoted a better understanding about the process model design.

The second contribution is related to improvements made in the preliminary process model design to support FTS characteristics. Designing a software process model to support FTS characteristics requires to pay attention in a lot of details of the project management. With data collected from the design validation was possible to identify an internal sub-process in the preliminary process model. Furthermore, it was possible to better define the sequence flow between SP05 and SP06.

The preliminary FTS-SPM^{ii} design presented in this section (see Figure 19) was used to conduct the second stage of the FTS-SPM evaluation made through an expert panel. The next section describes it in details.

7.2 The Expert Panel Evaluation

In this study the evaluation phase was kept interpretive to anticipate expert opinion on the preliminary FTS-SPM. The participants were asked to provide evidence or reasons such as, experience in practice to support the basis for best practices and sub-processes that are mentioned in the preliminary FTS-SPM. The responses from the interviews are mapped with some recommendations for FTS development.

Table 10 presents the information about participants who were interviewed in order to evaluate the preliminary FTS-SPM. Participants were selected based on academic or industry experience in GSD. Two questions were written based on each best practice, sub-process, and process model overview. The questionnaire included 64 questions. For answering the questions 51 to 64, which are related to process model overview evaluation, the preliminary FTS-SPM^{ii} picture (see figure 19) was provided to the participants. Appendix D gives the questionnaire and the summary of the questions asked to each expert.

Table 10 – Expert panel participants' information.

Expert	**Job title**	**Type of experience** (Academic or Industry or Both)	**Expert location**	**Experience in GSD**
Exp1	Senior Member of Technical Staff	Both	USA	9 years
Exp2	Postdoctoral Researcher	Academic	Italy	+10 years
Exp3	Professor/ Researcher	Academic	New Zealand	+4 years
Exp4	Managing Director	Both	Germany	15 years
Exp5	Software Engineer	Both	USA	17 years
Exp6	Project Manager	Industry	Romania	7 years
Exp7	Professor/ Researcher	Both	New Zealand	+20 years
Exp8	Project Manager	Both	Brazil	7 years
Exp9	Project Manager / Researcher	Both	India	10 years
Exp10	Head Marketing	Industry	India	10 years
Exp11	Professor	Academic	USA	+20 years
Exp12	Researcher	Both	Ireland	4 years
Exp13	CTO (Chief Technology Officer), Professor	Both	Netherlands	20 years
Exp14	Project Manager	Both	Brazil	7 years
Exp15	Researcher	Both	Ireland	5 years
Exp16	Professor	Academic	Italy	13 years
Exp17	Senior Researcher	Academic	Finland	14 years
Exp18	Professor/ Researcher	Academic	Spain	10 years
Exp19	IT Senior Manager	Both	Brazil	13 years
Exp20	Project Manager	Both	Poland	3 years

As per the answers given by the experts, all best practices, sub-processes, and the process model overview were analyzed and classified as VALID, PARTIALLY VALID, CONTEXT SPECIFIC or INCONCLUSIVE in the evaluation process. The classification schema is based on Deshpande [DES13].

- VALID: A particular best practice or sub-process is classified as VALID and is considered to have compliance with the model if the majority of experts are in full agreement with the best practice or sub-process.
- PARTIALLY VALID: A particular best practice or sub-process is classified as PARTIALLY VALID if the majority of experts are fully in agreement and partially in dis/agreement with the best practice or sub-process.
- CONTEXT SPECIFIC: A particular best practice or sub-process is classified as CONTEXT SPECIFIC if the majority of experts recommend this best practice or sub-process is only applicable in a particular context.

- INCONCLUSIVE: A particular best practice or sub-process is classified as INCONCLUSIVE if the majority of experts are all in disagreement with the best practice or sub-process.

7.2.1 Expert Panel Results

This section presents the results of the preliminary FTS-SPM evaluation made through an expert panel. Table 11 summarizes the evaluation results. Following the evaluation process and analysis of each best practice, sub-process, and process model overview, some amendments were identified to the preliminary FTS-SPM in order to make it more acceptable to GSD projects. The results are described in details under the following sub-sections.

Table 11 - Evaluation process summary for the preliminary FTS-SPM.

Sub-process (SP)	Best practice / Sub-process	Best Practice (BP) title	Evaluation by the Experts
SP01: Team Setup	BP30	At least one hour overlap between two sites	PARTIALLY VALID
	BP31	Fitting teams' working hours for a good overlap	VALID
	BP32	Teams distribution across two or three sites	VALID
SP02: Project Planning	BP01	Use of agile methodologies for project management	VALID
	BP02	Use of incremental software development approaches	VALID
	BP04	Application of FTS for testing and development phases	VALID
	BP36	Similar code patterns	VALID
SP03: Communication Protocol	BP07	Daily exchange of the project status by technologies	VALID
	BP10	Use of screen sharing technology to exchange knowledge	VALID
	BP12	Clean handoff and stocky handoff interactions	CONTEXT SPECIFIC
	BP13	Use of real time technologies for knowledge sharing	VALID
	BP15	Wikis and online forums to share knowledge between FTS teams	VALID
	BP21	Adopt proper technologies or tools to support communication between FTS teams	PARTIALLY VALID
	BP22	Time window	VALID
	BP25	Corporate technologies for team interaction	VALID
	BP26	Models of e-mails and electronic messages	VALID
SP04: Cultural Training	BP33	Meetings between team members for building trust	VALID

	BP35	Cultural awareness training	VALID
SP05: Task Allocation	BP17	CPro concept	VALID
	BP18	Low task granularity	PARTIALLY VALID
	BP20	Task distribution by sequence or dependency	VALID
SP06: Handoff Sessions	BP03	Daily stand-up meetings	VALID
	BP09	Daily handoff of 30 minutes duration with each development site	VALID
	BP11	Calendar of handoff sessions should be clearly defined	VALID
	BP14	Use of an FTP Server (or data repository) to exchange code and documents	VALID
Process model overview	SP01	Sequence flow between SP01 to SP02	VALID
	SP02	Sequence flow between SP02 to SP03, SP04, and SP05	INCONCLUSIVE
	SP03	Sequence flow between SP02 to SP03	INCONCLUSIVE
	SP04	SP04 sequence flow (loop)	INCONCLUSIVE
	SP05	SP05 sequence flow	INCONCLUSIVE
	SP06	Sequence flow between SP05 to SP06	VALID
	Whole model	Sequence floe between sub-processes as a whole	PARTIALLY VALID

7.2.1.1 SP01: Team Setup

SP01: Team Setup starts the preliminary FTS-SPM, as shown in Figure 19. It aims to identify available sites and allocates human resources for the project. Its development is based on three best practices. These best practices are: BP30: At least one hour overlap between two sites, BP31: Fitting teams' working hours for a good overlap, and BP32: Teams distribution across two or three sites. Next, I describe the evaluation results for these best practices.

- **BP30: At least one hour overlap between two sites**

Time overlap between sites in FTS development is important because allows to establish synchronous communication during handoff sessions. Also, it is important to define the minimum time overlap between sites that is needed to transfer the working tasks from one site to another. In order to evaluate the usefulness and relevance of the BP30 to the FTS-SPM, I asked to the experts how time overlap between sites help in better communication and coordination in GSD projects and how at least one hour of overlap between sites is useful for performing synchronous communication between sites.

In response to how time overlap between sites help in better communication and coordination in GSD projects, all experts agree that time overlap between sites is important for synchronous communication. The main benefit reported by the experts is the opportunity for real time communication that allows quickly identifying and solving problems. Such information is described by the experts as follows.

> “Overlap is important because we can have same time that goes to synchronous communication. It is necessary to transfer work to site to site. It is very important because you are able to connect, to message, to create a live meeting connection... So, the overlap is very important, but the important thing is to know when the other site is available.” **(Expert 1)**

> “Overlap is the most important because it allows you to select between synchronous and asynchronous communication. So, the more options you have, that’s a good thing.” **(Expert 2)**

> “You can have synchronous communication, that it is very important. Of course, you have other like e-mail, but it is not efficient like synchronous communication. Real time communication is very important.” **(Expert 3)**

In response to how at least one hour of overlap between sites is useful for performing synchronous communication between sites, different responses were given by the experts. Expert 1 is in full agreement that is needed at least one hour overlap between sites to perform synchronous communication.

> “You need at least one hour overlap between two sites. That is my answer. Less than one hour will be very bad.” **(Expert 1)**

Expert 2 explained that depends on the team skills. If a team has experience of working together, this team will be faster than other teams to perform handoffs. An experienced team can have a quick chat because they have a long history of working together. It allows establishing a good communication and reducing the time to perform daily handoffs between sites.

> “My answer is it depends. In the mostly on the history of two team settings when they start, one hour will not be enough. If they have a long history of working together, they can be composed of the short overlap in time. They will be able to communicate. If they have long experience, you can have a quick chat. When a team has a long history of collaboration a distance, one hour could be enough for a small transition.” **(Expert 2)**

Half hour is mentioned by the Expert 3. However, based on his experience one hour should be the maximum overlap time between the two sites.

> “The teams that I have investigated basically distributed between India and US don’t have any overlap during working hours. What they do is stretching the working hours, so, creating flexible working hours to get some overlap. So, the minimum overlap that they need is a half hour for fundamental daily scrum meetings. When is small time, they are able actually to get feedback, have an immediate response. One hour is probably the maximum

> time that they need. But from my empirical evidence, they need at least a half hour from understanding stages of the project." **(Expert 3)**

On the basis of the inputs from the experts, Table 12 gives the results about the BP30: At least one hour overlap between two sites.

Table 12 - Evaluation results of BP30: At least one hour overlap between two sites.

	Compliance to the model	**Result**
Expert 1	In full Agreement	PARTIALLY VALID
Expert 2	Context Specific	
Expert 3	In partial Agreement	

Amendments to the model

With the inputs given by the experts, BP30 is classified as PARTIALLY VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- It is important to have overlapping hours between two distributed teams in order to perform synchronous communication.
- Keeping team members with previous experience in working together for running new projects helps to increase the team's performance.
- Half hour may be enough for teams understanding stages of the project.

- **BP31: Fitting teams' working hours for a good overlap**

In GSD projects the time overlap differences between sites may not allow synchronous communication. This may result in problems to perform daily handoffs. Thus, BP31has been indicated to make time overlap differences manageable. In order to evaluate the usefulness and relevance of this best practice to the FTS-SPM, I asked to the experts how useful is to make time zone differences manageable in GSD projects and how fitting teams' working hours helps to manage time zone differences.

In response to how useful is to make time zone differences manageable in GSD projects, all experts consider critical to manage some time zones differences in GSD projects. That is because communications is constantly missed in global projects. How critical it is, depends on the project settings. The project manager has to make sure that have time windows, scheduled teams for team members work together. In some projects, there is no overlapping time and it becomes harder to manage teams globally distributed.

All experts are in full agreement that fitting team's working hours is a solution to make time zones manageable. However, it may result in problems if the teams' working hours are constantly changed. All experts are also in agreement that the fitting teams' working hours for a good overlap should be done for the minimum time. Such information is explained by one expert based on his experience.

> “I have faced it in some of the projects where I have worked, especially in Asia. So, have some critical phases of the project. For example, writing before deploy where you have to change our working time as well Asia team changed their working time. So, few days we changed our working time (the company) and few days they changed their working time. Thus, we worked together in a critical stage of the project. Actually, it is a bit hard to the team do things outside of their working hours. So, if is required to keep a long time doing this, it can be a bad thing for the team. So, it should be done when it must be done. If there is no agreement that is a critical stage of the project, my advice is not to do that. But if is a critical stage of the project go ready, it must be done, but for the minimum time.” **(Expert 8)**

On the basis of the inputs from the experts, Table 13 gives the results about the BP31: Fitting teams’ working hours for a good overlap.

Table 13 - Evaluation results of BP31: Fitting teams’ working hours for a good overlap.

	Compliance to the model	**Result**
Expert 2	In full Agreement	VALID
Expert 5	In full Agreement	
Expert 8	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP31 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Team’s working hours should be changed for the minimum time and only to support critical phases of the project.
- Team members must be in agreement on changing their working time hours.

- **BP32: Teams distribution across two or three sites**

Team members may are distributed in several sites in GSD projects. That is the main characteristic of GSD projects. In FTS projects, the number of sites may impact in communication and coordination aspects. BP32 is related to teams’ settings to a FTS project. This BP defines teams’ distribution across two or three sites. To evaluate the usefulness and relevance this best practice to the FTS-SPM, I asked to the experts how the number of sites can affect communication and coordination in GSD projects and how teams’ distribution across two or three sites helps in better coordination and communication in FTS projects.

In response to the first question, the experts stated that when the number of sites increases, it affects communication and coordination in GSD projects. That is because in same projects may have different cultures, holidays, skills, etc. It increases the number of problems in GSD projects. Expert 1 gave an example where different cultures affect communication and coordination on GSD projects.

> “Like in India have many religious holidays and it in some occasions take a week. It is called Indian festival. What happens is (...) if it takes too much longer it results in delays, delays of work.” **(Expert 1)**

Additionally, time zone differences between sites are a factor that increases the complexity of coordination. Such information is described by the Expert 11.

> “If the number of sites goes up, the complexity of coordination goes up. The one of the explanations is differences in time zones. Typically, when you have more sites involved, there are more time zone areas and so, there is a coordination complexity.” **(Expert 11)**

In response to the second question, two experts agreed that team distribution in two or three sites helps in better coordination and communication in FTS projects. Expert 11 explains that FTS needs at least two sites. In addition, Expert 7 describes a situation where projects with extreme time zone overlap differences have negatively impacted on coordination and communication. For this expert, two or three can better collaborate and help in reducing collaboration challenges between sites.

On the other hand, Expert 1 recommends the use of tools to support communication and coordination between sites. For this expert, a lot of problems can be solved by the use of appropriate technologies. Team members can work from home and communicate with people from different time zones. Expert 1 suggests screen sharing and live meetings as tools to support communication. This expert also explains the main problem in having several locations in FTS projects. To this expert, culture and working skills are the bigger problem than communication.

> “The communication is not the problem. What is difficult is the culture, the language, the assumptions, the style of the each person, what kind of skills they have. There are 2 parts of communication: first is to converge the information, the other part is the other person to understand what you said. The convergence of the information is the difficult part.” **(Expert 1)**

On the basis of the inputs from the experts, Table 14 gives the results about the BP32: Teams distribution across two or three sites.

Table 14 - Evaluation results of BP32: Teams distribution across two or three sites.

	Compliance with the model	Result
Expert 1	In partial agreement	VALID
Expert 7	In full Agreement	
Expert 11	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP32 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Creating teams with good expertise in order to have teams self-managed;
- Adopting tools to support communication and coordination like screen sharing and live meeting;
- New sites have to attend training sessions to become easier to work together on FTS projects.

7.2.1.2 SP02: Project Planning

SP02 aims to define the project planning. It is performed by the project manager at the beginning of the project and it can be reviewed along on the project. SP02 is developed based on four best-practices: BP01: Use of agile methodologies for project management, BP02: Use of incremental software development approaches, BP04: Application of FTS for testing and development phases, and BP36: Similar code patterns. Next, I describe the evaluation results for these best practices.

- **BP01: Use of agile methodologies for project management**

Software companies can adopt different working processes and different development methods to develop GSD projects. Most of the studies suggest adapting to the agile methodologies to implement FTS. To Gupta et al. [GUP09] and Carmel, Espinosa, and Dubinsky [CAR10], agile methodologies contribute to increase the speed of software development. Some studies also cite Scrum and XP methods for FTS, but there is no consensus on suitability of one on another one for FTS. In order to evaluate the usefulness and relevance of the BP01 to the FTS-SPM, I asked two questions to the experts. In the first question, I asked to the experts some examples of agile methodologies for GSD projects. In the second question, I asked how agile methodologies are appropriate for GSD projects.

In response to the first question, experts mentioned TDD (Test Driven Development), Scrum, and XP. Expert 2 stated that TDD is very effective for GSD projects. Expert 12 describes that is possible to mix agile methodologies for GSD projects. This expert reported the use of XP with Scrum as an example.

> "The project manager provides a high level requirement definition. We usually worked by 3 or 4 weeks in it. In this case we implemented agile for GSD. I also could say that we used a mix of Scrum and XP in the project. Scrum, because we had daily meetings with the development team, project manager and engineers." **(Expert 12)**

This question was also asked to an expert without experience with agile methodologies to GSD projects. However, this expert realizes the benefits of agile methodologies for GSD projects.

In response to the second question, the importance of agile methodologies for GSD projects is described by all experts. To them, agile methodologies help to solve problems faster, keep the focus on requirement requests from the client and reduce the time spend working on documentation.

Agile methodologies are also described as the most suitable for FTS projects by two experts. To these experts, agile methodologies focus on faster software development. Thus, agile methodologies help in achieving the FTS main goal. On the other hand, one expert reported that it is difficult to manage FTS projects having agile methodologies. For this expert, agile methodologies are not the solution for FTS.

> "I don't think that agile (methodologies) is the solution to follow the sun development. Because a follow the sun project is a complete different project setup that agile. Follow the sun does always like around the clock, moving from one time zone to another, have different deliveries parts or software development. The segment of the project at the time is changed. So, do you have an agile methodology in this particular setup I find difficult as proved by my experience. It's been always difficult to have an agile methodology in follow the sun." **(Expert 9)**

On the basis of the inputs from the experts, Table 15 gives the results about the BP01: Use of agile methodologies for project management.

Table 15 - Evaluation results of BP01: Use of agile methodologies for project management.

	Compliance to the model	**Result**
Expert 2	In full agreement	VALID
Expert 9	In disagreement	
Expert 12	In full agreement	

Amendments to the model

With the inputs given by the experts, BP01 is classified as VALID. The input given by the experts about BP01 did not allowed to identify recommendations for the preliminary FTS-SPM.

- **BP02: Use of incremental software development approaches**

Test Driven Development (TDD) is an approach for incremental software development in which software units are developed in small pieces. This approach does not require initial design details as software units are incrementally developed following test-before-code stile [GUP07]. Incremental software development approaches like TDD are recommended for FTS development. In order to evaluate the usefulness and relevance of the BP02 to the FTS-SPM, I asked to the experts how TDD technique is helpful for development across multiple sites and how incremental software development approaches are useful for FTS development.

In response to the use of TDD for development across multiple sites, all experts stated that TDD is a good practice for GSD projects. TDD is important for GSD because helps in identifying requirements, writing test cases, and for continuous integration. Such information is described by Expert 14 based on his experience.

> "For TDD is useful to get all the issues up front. So, the idea is to start developing the task, the task fails, and then creating the classes, interface database doesn't matter. Then tasks classes, then continue task fails again. So they are improving the coding by the time they are developing the tasks. Then, when they complete the tasks, the coding is done. So, this is helpful for continuous integration. So, when you play the fields to deploy the code, I say no production, production environment, doesn't matter, the code runs the built, the tasks and deploy the code." **(Expert 14)**

However, TDD technique is only helpful for GSD if it is effectively managed. Otherwise, TDD does not offer benefits for software development. Such information is described by Expert 19.

> "Depends on how you perform TDD. It should have a great deal of coordination across the different teams and a great deal of methodology on how to deliver, what is found across the multiple sites. I think it can be helpful and it can delivery productivity and effectiveness. Just if you have a good deal of management thought out your whole process. Otherwise, I don't see any benefit from TDD." **(Expert 19)**

I have obtained similar responses regarding to the use of incremental software development approaches for FTS development. The experts stated that incremental software development approaches are useful for FTS development. That is because these approaches support short cycles' implementation like daily or weekly, delivery cycles. If a team is working on a common code, as happen in FTS development, an incremental approach will help to structure the work. Expert 14 gave an example the using of incremental software development approaches to fix defects. Based on this example, the expert explains how incremental software development approaches can be useful for FTS.

> "I'll take the defect example, because we do not use FTS for coding. So, we stabilize the code, four weeks before production, we want to fix the defects. TDD means first we need to make the test, and then fix the defects. I believe that helps because if someone writes a code and just handover to another one like a developer and this guy concludes the code and doesn't write any unit test. Then, we do check-in, doesn't have any commit that the code is going to work. We didn't develop the unit test first. For FTS, it is useful because if I start fixing the defects in the unit testing my pair in another country will have to follow the same standard. If I start the unit test, they will have to continue developing the unit testing because the unit testing will go to fail. And it is going to fail; he can do (wrong) the build. When the unit testing fails, we send an e-mail that doesn't have any package deployed, let say development. (We) Don't need to be productive developers, for example. So, only one unit testing fails, the code doesn't to production or no production. So, that means that is very useful." **(Expert 14)**

How useful incremental software development approaches are for FTS development depends on how it is managed. Based on the FTS aspects and incremental software development approaches, one can be a complement of another. However, it is required a good management and a software process to deliver a product with some level of quality. If a team cannot deliver quality, the FTS and incremental approaches do not offer any benefit. Such information is described by Expert 19.

> "If there is no closer management, I don't see any benefit. If there is no solid process for effective management for TDD or FTS, you have to be much disciplined in terms of documentation, requirements, how you track requirements, how you manage the technical aspects of the implementation, and how you translate that in documents that have been built though out of the process." **(Expert 19)**

The definition of a process for effective management helps to increase the product quality. Incremental software development approaches can help in enhancing quality in such FTS dynamic. If team members are delivering part of the work to someone else who does not have a clue in what has been done to that point, the definition of a process can be useful.

On the basis of the inputs from the experts, Table 16 gives the results about the BP02: Use of incremental software development approaches.

Table 16 - Evaluation results of BP02: Use of incremental software development approaches.

	Compliance to the model	**Result**
Expert 4	In full Agreement	VALID
Expert 14	In full Agreement	
Expert 19	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP02 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- This BP can be followed by team members working on the same code base. It allows splitting the code base in different locations.

- It is needed to define a software process for effective FTS management.

- **BP04: Application of FTS for testing and development phases**

In literature, FTS is mainly recommended for testing and development phases of the SDLC. Studies conducted in the software industry show that testing and development can work well in FTS. In order to evaluate the usefulness and relevance of the BP04 to the FTS-SPM, I asked to the experts how adopt FTS for different software development phases impact on handoffs development and how testing and development phases are appropriate for developing software in FTS mode.

In response to the first question the experts stated that development and testing phases are the most suitable for FTS. Handoffs can work well in testing because this phase does not require process automation. Thus, handoffs are easier to perform during the testing phase. Such information is described by Expert 10.

> "You can do it easier in testing. A lot of coding can happen if I got a piece of code that I have been built, and I could have to validate, could do the unit testing. Actually be ready with the code or I built some code and you can run a unit test. This can happen, but that the easier to shift is testing. Follow the sun for testing is extremely useful." **(Expert 10)**

In the development phase it is also possible to perform handoffs to transfer the work from one site to another. However, the complexity these handoffs depends on the level of coding sophistication. Development phase can be difficult if developers do not understand the work due to the task's complexity. Some factors that affect the team's performance are the working experience and the process adopted to perform handoffs in different stages of the project. An expert describes his experience doing FTS for testing and development phases.

> "I have experience doing it (FTS) in my company. It works for testing and coding. For example, I'm working on a module that test goes out very soon, then I split the piece of code and get another class for instance and starts implementing. Then, someone does the testing around. So, developers in US starts developing class number 1, the guy in India will be working in the number 2 for the same design. Then implementing teams and checking teams. This can work. But anything early, you can forget." **(Expert 10)**

FTS is not recommended for the early stages of the project as mentioned by Expert 10. On the other hand, Expert 19 recommends FTS for planning and other activities of the project like writing requirements. This expert suggests experienced teams to work on FTS projects. A good coordination and an effective communication between teams allow adopting FTS in other phases of the SDLC. Such information is described by Expert 19.

> "I think it depends on how experimented is your team in different phases of the project. If you have, for instance, a very skillful business analyst, together with analyst requirements and they talk with each other very often, they know what they are talking about, they are pretty much aware about the business objectives on the specific project that they are working on. Again, if there is a very solid coordination and a very effective communication between these teams. I believe, for instance, that you can have FTS for planning and other activities of the project like writing requirements. I can see that happening today." **(Expert 19)**

In response to the second question, the experts stated that testing and development phases are appropriate for developing software in FTS mode. An expert mentioned that is important to know that testing and development phases are different from the testing phase. Tasks are separated in testing and development. Such information is described by the Expert 10.

> "If you have to do testing and development then you need to separate the tasks and say: you write test cases for this functionality, I will write test cases for this functionality. But, it really independent and it is not the follow the sun. Follow the sun is something that you doing the same task in two locations and continuing it. Follow the sun is running all thought the clock." **(Expert 10)**

In FTS, team members work on the same software development phases. However, executing testing and development phases in parallel may not work for FTS. That is because team members have different skills.

On the basis of the inputs from the experts, Table 17 gives the results about the BP04: Application of FTS for testing and development phases.

Table 17 - Evaluation results of BP04: Application of FTS for testing and development phases.

	Compliance to the model	**Result**
Expert 1	In full Agreement	VALID
Expert 10	In full Agreement	
Expert 19	In full Agreement	

<u>Amendments to the model</u>

With the inputs given by the experts, BP04 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Daily handoffs must be supported by specific documents. These documents should describe what has been done to a certain point of time and the goal for the next stage or for the one that is receiving the work.
- Skillful team members should be allocated to the project in order to promote solid coordination between sites;

- High level of communication and feedback should be performed. Team members must be open to receive those feedbacks on how things are progressing (positive or negative) and what are the project's objectives.
- Handoffs' information should be very clear for all parts (team members), the one that is handing off and the other that is receiving the work.
- Since team members have different skills, it is recommended to implement FTS for only one phase of the SDLC per time.

- **BP36: Similar code patterns**

BP03 aims to define similar code patterns to avoid rework. Since, team members are distributed in different locations and working on the same tasks, similar code patterns allow team members to understand and identify changes made in the code since the last handoff session. In order to evaluate the usefulness and relevance of the BP36 to the FTS-SPM, I asked to the experts some examples where similar code patterns are helpful for GSD projects and how similar code patterns are helpful for team members working on the same tasks.

In response to the first question, two experts stated that similar code patterns across different locations allows people understanding the team's expectations, roles, and responsibilities. It helps organizations in general have a common goal and achieve the customer's objective. On the other hand, one expert does not see the use of similar code patterns as an advantage for GSD projects. This expert suggests short interactions to improve the communication between team members. Such information is explained by the Expert 16.

> "I think that is something relate to a process that can be in the middle, for short interaction. How they require to communicate and you cannot, or you are not allowed to communicate. If you have short interactions, I think it is one of the most important." **(Expert 16)**

The use of similar code patterns as seen as an advantage for team members working on the same tasks. Two experts mentioned the need of similar code patterns to understand the tasks developed by team members distributed in different sites. These experts stated that code patterns help team members to develop tasks following the same string line, it especially during the maintenance stage of the project. Also, similar code patterns are important to share a common knowledge about the tasks. Team members may have different skills and knowledge. Thus, similar code patterns are helpful to better understand the terminology and technical terms used during the project.

An expert also suggests similar code patterns to improve the tasks' quality, but to this expert it does not help team members working on the same tasks. Well defined software architecture can be more useful for team members working on the same piece of code or task. A good version control system can avoid rework if tools adopted in the project are based on asynchronous coding. Such information is described by the Expert 16.

> "If you have a good architecture, it is useful because you have a group to assign a piece of code to people. Different pieces of code assigned to different people. I think good code patterns allow you to have quality. There is a need of people working at the same time and on the same coding task. If you need something like that, related to distance to draw something. I never had used that, but there some task that use the same algorithms. I use Google docs, for example. If you want to do coding, but not in real time, it is a good version control system. But particularly in this case, if you use a centralized version control system. You can't avoid a kind of rework if you use tools that are based on asynchronous coding. If you want to avoid a kind of rework, you need to use editors for a concurrent editing." **(Expert 16)**

On the basis of the inputs from the experts, Table 18 gives the results about the BP36: Similar code patterns.

Table 18 - Evaluation results of BP36: Similar code patterns.

	Compliance to the model	**Result**
Expert 3	In full Agreement	VALID
Expert 8	In full Agreement	
Expert 16	In partial Agreement	

Amendments to the model

With the inputs given by the experts, BP36 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Team members must work following the same practices to build the code;
- Code practices must be well defined before the project starts;
- Team members must have a common knowledge about code patterns performed during the project;
- It is needed to define a version control system, editors, and tools for asynchronous communication in order to avoid rework.

7.2.1.3 SP03: Communication Protocol

SP03: Communication Protocol aims to define communication resources and the schedule for synchronous and asynchronous communication between team members. SP03 includes nine best practices: BP07: Daily exchange of the project status by technologies, BP10: Use of screen sharing technology to exchange knowledge, BP12: Clean handoff and sticky handoff interactions, BP13: Use of real time technologies for knowledge sharing, BP15: Wikis and online forums to share knowledge between FTS teams, BP21: Adopt proper technologies or tools to support communication between FTS teams, BP22: Time window, BP25: Corporate technologies for team interaction, and BP26: Models of e-mails and electronic messages. Next, I describe the evaluation results for these best practices.

- **BP07: Daily exchange of the project status by technologies**

BP07 recommends the use of technologies such as telephone calls, video conferences or e-mails for the daily exchange of the project status. In order to evaluate the usefulness and relevance of the BP07 to the FTS-SPM, I asked to the experts some examples of technologies for synchronous communication in GSD projects and how these technologies are helpful to perform daily handoffs in FTS projects.

In response to these questions the experts stated that the telephone is the most popular technology. Other technologies are also adopted in GSD projects are video and conference calls, teleconference, Skype, and live meeting. Experts stated that technologies based on synchronous communication are important to perform daily handoffs between sites. One expert explains the importance of these technologies and when it should be applied. Asynchronous technologies like e-mail are also useful to perform daily handoffs. However, team members prefer using synchronous communication.

> "When team members are using Skype or video conference, it is almost talking face-to-face. They are able to discuss the problems without using asynchronous communication like e-mail. Asynchronous tools help to interact directly and make the concepts clear. E-mails are used at the most time in the software companies because the time zone overlaps. When there is no time overlap, people prefer using e-mail, but when there is time overlap, especially in follow the sun, people most of the time prefer using synchronous communication." **(Expert 9)**

To the experts, technologies based on synchronous communication allow reducing the time spent to perform daily handoffs. Team members can have a quick chat to discuss critical information focusing on the most relevant aspects of a task.

On the basis of the inputs from the experts, the results about the BP07: Daily exchange of the project status by technologies are given in Table 19.

Table 19 - Evaluation results of BP07: Daily exchange of the project status by technologies.

	Compliance to the model	**Result**
Expert 4	In full Agreement	VALID
Expert 9	In full Agreement	
Expert 18	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP07 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Synchronous communication during daily handoffs can be supported by telephone, video and conference calls, and teleconference technologies.
- The use of e-mail can be adopted to support asynchronous communication when there is no overlapping time between sites.
- If the objective is to make even more clear concepts discussed by teams during handoff, then this should be done by asynchronous communication.

- **BP10: Use of screen sharing technology to exchange knowledge**

BP10 recommends the use of screen sharing technologies to exchange knowledge. Screen sharing contributes to transfer knowledge between team members. It uses aim to easier understand the information that is being discussed during handoff meetings or any meeting performed between two or more participants. In order to evaluate the usefulness and relevance of the BP10 to the FTS-SPM, I asked to the experts some examples of the information discussed by teams during handoffs and how the use of screen sharing technology is useful to exchange knowledge between team members distributed across different sites.

Experts gave different examples of information discussed by teams during handoffs. Two examples are related to the use of tools to support daily handoffs. These tools are online tools to manage defects and screen sharing and e-mail to discuss code details. In addition, Expert 14 describes a particular strategy adopted to transfer knowledge between developers and make sure that the information is correctly understood by the receiver. This strategy is based on the use of screen sharing followed by an e-mail. Such information is described by the Expert 14.

> “We have a simple standard in my project to establish communication. First, the developer, since he has IM to share the screen, this developer shares the screen and shows the code. Then, he explains what he did and what is missing. First, he explains what he is doing, what is missing that means what has to be done to complete the task. Then, another guy has to acknowledge about the information. So, the other guy has to explain back again. “Now it is your turn, what this code does, what is done and what is missing to do to complete this code”. So, this guy explains, and the other that was working has to say “Ok, you understood”. So, the handover is done. First, sharing the screen and go through the code. Then, after that, both send e-mails saying “Do you agree that you will continue task A, B, C, and these are the missing steps that should be complete task” and you has to send this e-mail again if you need to handover. If the Indian developer doesn’t complete the code, he has to handover again. It is very simple. Screen sharing and e-mail.” **(Expert 14)**

Experts stated that screen sharing technology is the main practice adopted to exchange knowledge between team members. This practice is useful in the context of technical discussions. Since teams are distributed in different locations, explaining a code by telephone becomes difficult. Developers are visual. Thus, the use of screen sharing is seen as essential to transfer knowledge between team members. Such information is described by the Expert 14.

> "Today I can't see how does not use screen sharing. They (team members) need to see the lines of code or at least the other guy follows in your local box the code lines. They need to synchronize the line of code for example. "Now I'm reading line 1, 2, 3 e 4 and class A", for example. This way, it is hard to keep synchronized. I don't see sharing knowledge without screen sharing." **(Expert 14)**

On the basis of the inputs from the experts, the results about the BP10: Use of screen sharing technology to exchange knowledge are given in Table 20.

Table 20 - Evaluation results of BP10: Use of screen sharing technology to exchange knowledge.

	Compliance to the model	**Result**
Expert 5	In full Agreement	VALID
Expert 12	In full Agreement	
Expert 14	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP10 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- During daily handoffs team members who are receiving the tasks' updates should summarize the information to the giver team members at the end of each daily handoff session.
- An e-mail must be sent at the end of each daily handoff session with the agreements made between team members.
- Team members who are giving the tasks' updates should describe what the team has been done, what the team is doing and the next steps that should be done by the next team to complete the tasks.

- **BP12: Clean handoff and sticky handoff interactions**

BP12 aims to define clean and sticky handoff interactions. Clean handoff interactions are short interactions to discuss punctual questions related to the project and sticky handoff interactions are more intense. In order to evaluate the usefulness and relevance of the BP012 to the FTS-SPM, I asked to the experts how daily meetings between team members distributed across different sites should be performed and how short and intense interactions help in better communication between team members during handoffs.

Experts gave some examples to explain how daily meetings should be performed. One expert describes the use of live meetings with screen sharing between participants. This kind of daily meeting is adopted to discuss problems faced by the team to perform a particular task. Other expert explains how daily meetings are performed when there are a few overlap hours between sites. They use Skype or telephone calls to discuss tasks for the day and the next tasks that should be done for the next team. The use of Scrum methodologies which require daily meetings is reported by another expert. This expert describes the use of video conference when team members are geographically distributed.

Regarding to handoff meetings planning, two experts mentioned that short and intense interactions between teams help in better communication. However, the time spent during interactions between teams should be flexible in some cases. The communication does no should be blocked if there are more topics to be discussed. It depends on the project context or the particular work that has been carried out. Such information is described by the Expert 9:

> "That can be short, but depend on the particular work that has been carried out or it can be intense if there are media working is going on and have a lot of communication and then can be that. Like discuss "I did this task, I did this part of coding" just talk quickly, briefly and handover the task to each other. Most depends on which stage the project has been carried out. So, communication, can be short or intense depend on the situation of the project." **(Expert 9)**

Additionally, one expert recommends defining an agenda to make team members available. This agenda should keep interactions between team members short and intense, that because people get stressed in attending meetings.

On the basis of the inputs from the experts, the results about the BP12: Clean handoff and sticky handoff interactions are given in Table 21.

Table 21 - Evaluation results of BP12: Clean handoff and sticky handoff interactions.

	Compliance to the model	**Result**
Expert 1	Particular context	CONTEXT SPECIFIC
Expert 9	Particular context	
Expert 17	In full agreement	

<u>Amendments to the model</u>

With the inputs given by the experts, BP12 is classified as CONTEXT SPECIFIC. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- An agenda for team interactions should be defined.
- Teams' interactions can be flexible, but it is not recommended to be longer than one hour.

- **BP13: Use of real time technologies for knowledge sharing**

BP13 aims to make knowledge sharing easier between teams defining real time technologies. In order to evaluate the usefulness and relevance of the BP013 to the FTS-SPM, I asked to the experts some examples of technologies for real time communication and how real time technologies help in better communication between team members in FTS projects.

Many technologies for real time communication were cited by the experts such as telephone, video conference and instant message (IM). Some software applications for real time were also cited. These software applications are Skype, Microsoft communicator and live meeting from Microsoft. One expert mentioned that the use of the telephone is decreasing in some organizations. Instead of the use of telephone, organizations are adopting instant communicators and chats.

Real time technologies are considered important for knowledge sharing between teams. However, a good infrastructure to support real time technologies is needed. Without a good infrastructure the effective communication can be hindered. One expert gives an example to explain some advantages of real time technologies.

> "One big advantage of the get of real time is especially with chat. I can send a request if I don't understand this (something) and he doesn't get disturbed in what he's doing. He looks when he wants. It is not really real time, it is near real time. So, he gets then he looks and thinks "someone needs some help". He will respond to you. That makes much easier because you always talk with each other. It doesn't look an e-mail that becomes very formal, flogs your e-mail box. It just happens badly and doesn't distract you. When an e-mail comes to you, the reaction is… you are trained to take a look at the e-mail. If a chat comes, we have a condition don't look immediately. So we can have a chat when a request comes. When an e-mail request comes, then you say it is strictly urgent. That is really distractive in terms of a breaking a talk process." **(Expert 10)**

As described by the Expert 10, real time technologies allow performing informal communication. Thus, team members can self-organize their time to answer a request avoiding distractions. However, it doesn't mean that a request is less important than an e-mail. E-mail in general sounds something urgent.

In addition, one expert describes the use of real time technologies to solve the problems faster than by the use of other technologies. Real time technologies are also recommended to improve the trust between team members and to discuss the problems.

On the basis of the inputs from the experts, the results about the BP13: Use of real time technologies for knowledge sharing are given in Table 22.

Table 22 - Evaluation results of BP13: Use of real time technologies for knowledge sharing.

	Compliance to the model	**Result**
Expert 3	In full Agreement	VALID
Expert 10	In full Agreement	
Expert 15	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP13 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Tools such as telephone, video conference, Skype calls, Microsoft communicator, and live meeting from Microsoft can be used to support real time communication.
- An adequate infrastructure should be provided to support real time technologies.

- **BP15: Wikis and online forums to share knowledge between FTS teams**

BP15 consist of on creating an internal wiki and online forums as a knowledge base in order to share problems and solutions. Wikis and online forums are the tools used to share knowledge among the team members. Both of these resources provide informal knowledge in a structured format. In order to evaluate the usefulness and relevance of the BP015 to the FTS-SPM, I asked to the experts some example of technologies used to sharing knowledge in GSD projects and how wikis or online forums are helpful to share knowledge in FTS projects.

In response to the first question, experts reported the use of editing tools for sharing knowledge. One expert describes the use of Wikipedia for training people in different cultures. Some organizations also adopt technologies as CVS (Concurrent Versions System) repositories and tools for performing synchronous and asynchronous communication. However, these tools are not used only for a particular project, but for creating a database of information. In this database teams can discuss not only the technical aspects of the project.

Two experts stated that the use of wikis or online forums is helpful for sharing and transferring knowledge between team members. One expert describes as a benefit the possibility to analyze, fix, and improve contributions given by team members. Such information is described by the Expert 16.

> “The writing documentation is a very old invention to communicate or transfer knowledge. What is new in the wiki, is that people is not just the owner, but read this provide comments or if I analyze the answer, I can make changes if I find something to fix.” **(Expert 16)**

Another expert describes the using of wikis, online forums and challenges faced by the teams to share knowledge. The main challenge is making teams start to use wikis and online forums. Teams do not understand the benefits of wikis and online forums for the project at the beginning. Expert 6 provides this information.

> “We do have a platform. We start with this platform recently. It is the most important to share knowledge. It is difficult to use these new tools. It is difficult to include people in happy hours. Not everyone from the team see the use of these tools in the beginning. A lot of people think it is not necessary to wait for the technical aspects. Takes a while for people to get used to these tools. In the beginning is very slow to get an answer. The responsibility for answer something is not clear. Like to send an e-mail to a group of people and wait for an answer. It will take a day. It depends. I mean, this platform is structured as you are a part of a certain group, area of expertise. Then have a configuration that somebody post as you get the notification in your e-mail. So, it is in a real time. In fact, I know the answer, but I don’t have to respond to that. So, I’m not gonna to respond until the end of the day and then at the end of the day I will handle it from home. So I look for the notification, it is a short answer, I go there to answer. These don’t take a lot of time from me. For these online forums, it is socialized platform. There are things that are technical and others not. Sometimes the people do a joke. It’s put together the community. So, it is not just a technical template.” **(Expert 6)**

On the other hand, Expert 12 does not see significant benefits with the adoption of wikis and online forums for knowledge sharing. Expert 12 stated that there are some delays in getting the right answer on wikis and online forums. That is because some messages posted by participants are useless. However, an answer can be posted in a few minutes or hours. For this expert how the wikis and online forums will help for sharing knowledge depends on cultural aspects of organizations.

On the basis of the inputs from the experts, the results about the BP15: Wikis and online forums for sharing knowledge between FTS teams are given in Table 23.

Table 23 - Evaluation results of BP15: Wikis and online forums for sharing knowledge between FTS teams.

	Compliance to the model	**Result**
Expert 6	In full Agreement	VALID
Expert 12	Particular Context	
Expert 16	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP15 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Web pages, CVS (Concurrent Versions System) repositories and editing tools can be used to support knowledge sharing in GSD projects.
- Training programs must focus on giving training on better use of wikis and online forums for knowledge sharing.

- **BP21: Adopt proper technologies or tools to support communication between FTS teams**

BP21 recommends the adoption of proper technologies or tools to support communication between FTS teams. Communication between FTS teams can be carried out using communication technologies or tools such as telephone calls, e-mails and IM. In order to evaluate the usefulness and relevance of the BP021 to the FTS-SPM, I asked to the experts some examples of technologies or tools to support communication in GSD projects and how adopt proper technologies or tools to support communication helps in FTS software development.

Telephone, Skype, Google hangout, Microsoft communicator, conference calls, e-mail, and chat were mentioned as examples by the experts. Experts also mentioned the using of internal repositories to share documents and to support communication.

These technologies and tools should support project and team characteristics. The project manager can choose the best fit and the most appropriate technology or tool for a project. There are many generic technologies or tools that can be applied for all projects as one expert describes.

> "Some teams can use a generic tool and its work very well. Other teams, maybe needs a particular tool to work. I don't see a generic answer to that because depends on the situation, depends on the team, depends on the tool. It is always about distance. If this helps to reduce the perception of distance, it is fine." **(Expert 13)**

On the other hand, one expert describes the adoption of proper technologies or tools as helpful to support communication in FTS development. This expert makes a comparison between GSD projects and FTS development to explain how helpful are the adoption of proper technologies or tools to support communication in FTS development.

> "The same way helps with GSD projects. Like with code repositories we can code available, with the sharing point we can have documents live to anyone that makes parts of the project. With live meetings it is easier to understand and provide clarifications, the communication as well. However, the communicator is only efficient if you are working with some overlap otherwise the people from the other side is offline. Any e-mail can be used for a tool for offline line or short communications." **(Expert 8)**

On the basis of the inputs from the experts, the results about the BP21: Adopt proper technologies or tools to support communication between FTS teams are given in Table 24.

Table 24 - Evaluation results of BP21: Adopt proper technologies or tools to support communication between FTS teams.

	Compliance to the model	**Result**
Expert 2	In partial Agreement	PARTIALLY VALID
Expert 8	In full Agreement	
Expert 13	In partial Agreement	

Amendments to the model

With the inputs given by the experts, BP21 is classified as PARTIALLY VALID. The following recommendations identified to the preliminary FTS-SPM as a result of the evaluation process.

- Technologies or tools such as telephone, Skype, Google hangout, Microsoft communicator, conference calls, e-mail and chat can be used to support communication on FTS projects.
- The project manager should choose technologies or tools to support communication according to project's characteristics.
- Team members have to learn how to communicate over the time;
- Technologies and tools based on synchronous communication require some time overlap between sites.

- **BP22: Time window**

BP22 aims to define an opening of limited duration during which something can be accomplished meetings or short interactions between team members. Time window is used by the teams to minimize collaboration conflicts between sites. In order to evaluate the usefulness and relevance of the BP022 to the FTS-SPM, I asked to the experts some examples where synchronous interaction between distributed sites is required and how time window helps in better collaboration and communication between sites.

Three different examples were given by the experts relate to synchronous interaction between distributed sites. One expert described the using of synchronous interaction for brief communication to clarify something or a particular requirement of the project. Another expert described the need of synchronous interaction to discuss task's details. And the last expert described the need of synchronous interaction for project planning. However, as mentioned by the experts there are some projects adopting asynchronous communication for planning. However, in some situations it is difficult to solve problems without synchronous interaction between teams.

Experts stated that time window maximize the opportunities for synchronous communication. It allows team members to interact with other team members during the day. Time window can be used to ask questions and to solve problems as described by one expert:

> "It is very important to have a synchronous time window for communication. Supposing you can have a video conference or teleconference, then you can know people better when you can join with them and not just sending e-mails. So, solve difficult problems, making the planning together. It is very important." **(Expert 17)**

However, the time window can not be applied to all cultures as described by another expert. Organizations should consider team's culture in order to make it work.

> "You don't have to consider it applied to all cultures as a benefit. You have to consider cultural aspects. Maybe you don't have the main benefit with that. Maybe they don't make themselves available for a simple chat. You don't have to institute it in the company." **(Expert 5)**

On the basis of the inputs from the experts, the results about the BP22: Time window are given in Table 25**.**

Table 25 - Evaluation results of BP22: Time window.

	Compliance to the model	**Result**
Expert 5	Particular context	PARTIALLY VALID
Expert 11	In full Agreement	
Expert 17	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP22 is classified as PARTIALLY VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Time window should be applied according to the teams' and organizations' culture.
- Team members must make themselves available to increase collaboration and communication.

- **BP25: Corporate technologies for team interaction**

BP25 recommends technologies such as video conferencing, screen sharing and other corporative resources for the teams attending meetings from home. In order to evaluate the usefulness and relevance of the BP025 to the FTS-SPM, I asked to the experts some examples of corporate technologies and how these technologies are useful for team's interaction in GSD projects.

In response to the first question, experts mentioned tools for version control, repositories, forms and online forums. Some software applications were also mentioned by experts like Foundation Sever, Link from Microsoft, Clarity, Changepoint and Microsoft communicator. Foundation server is a platform that helps teams communicate, save coding and provide a website for the teams. Link from Microsoft is used conferencing teams and project meetings. Clarity is used for project management. Changepoint is used for portfolio management. Microsoft communicator is used for video conference and instant message.

To the experts, corporate technologies provide many benefits for team's interaction. One expert mentioned that is needed to collaborate though corporate technologies. That because corporate technologies provide more benefits than other technologies for simple interaction. For this expert, it is possible to follow a structured process for exchange information.

Expert 19 stated that team members can be more productive by the using of corporate technologies because they can avoid some software incompatibilities. Corporate technologies also allow sharing artifacts between team members using a same technology. Such information is described by the Expert 19:

> "So, I think if you can use corporate technologies is better. I can have more benefits from corporate technologies. People can be more productive, more effective and you can avoid incompatibilities, avoid talk something that others do not understand what is about. I think it provides the same language in the company. If I schedule a meeting for using a link, for instance, for the communication purpose, I think everyone understands, everyone knows how to deal with the technologies, everyone is productive using it. I totally agree if possible using corporate technologies as much as possible, because everyone can benefit from it." **(Expert 19)**

On the basis of the inputs from the experts, the results about the BP25: Corporate technologies for team interaction are given in Table 26.

Table 26 - Evaluation results of BP25: Corporate technologies for team interaction.

	Compliance to the model	**Result**
Expert 7	In full Agreement	VALID
Expert 15	In full Agreement	
Expert 19	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP25 is classified as VALID. The following recommendation was identified to the preliminary FTS-SPM as a result of the evaluation process.

- Such technologies like tools for version control, repositories, forms, online forums and software applications like Foundation Sever, Link from Microsoft, Clarity, Changepoint, and Microsoft communicator can be adopted as corporate technologies in organizations.

- **BP26: Models of e-mails and electronic messages**

BP26 recommends the use of models of e-mails and electronic messages. A unique message template could be used to assign specific meaning to a message, for example, technical and non-technical requests could be distinguished by using different message templates. In order to evaluate the usefulness and relevance of the BP026 to the FTS-SPM, I asked to the experts some examples of resources used to perform asynchronous communication between sites and how models of e-mails and electronic messages help in better communication between team members.

All the experts mentioned the use of e-mail for asynchronous communication. To them, e-mail is the most used technology by organizations today.

Models of e-mails and electronic messages help in better communication between team members because helps to structure the information and describe the essential information that is required. Such information is described by the Expert 18:

> “Templates or something similar helps people to structure the information and focus on the more important part. I think is very convenient and a good idea having templates or script to complete the information. Another advantage is that all e-mail has the same structure. A person interested in a specific part of a message can look straight there besides takes time looking all the e-mail.” **(Expert 18)**

On the basis of the inputs from the experts, the results about the BP26: Models of e-mails and electronic messages are given in Table 27.

Table 27 - Evaluation results of BP26: Models of e-mails and electronic messages.

	Compliance to the model	**Result**
Expert 4	In full Agreement	VALID
Expert 11	In full Agreement	
Expert 18	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP26 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Similar templates help people to better structure the information.

- It is suggested to define the priority information required to complete the form (e-mail or electronic message).
- Each e-mail or electronic message should address only one topic (information).

7.2.1.4 SP04: Cultural Training

SP04 aims to develop trust between team members of different cultures involved in a same project. Cultural differences involve characteristics of a particular group of people or locals. Since team members come from different countries, the issue of managing culture becomes more challenging. Each culture has its own set of unwritten rules and etiquettes [GUS07].

SP04 is developed based on two best practices: BP33: Meetings between team members for building trust and BP35: Cultural awareness training. Next, I describe the evaluation results for these best practices.

- **BP33: Meetings between team members for building trust**

BP33 recommends conducting meetings between team members to establish or re-establish trust. These meetings should address cultural differences between team members involved in the project. In order to evaluate the usefulness and relevance of the BP033 to the FTS-SPM, I asked to the experts some examples of cultural differences between sites in GSD projects and how meetings between teams helps to build or increase the level of trust.

Experts mentioned three different examples of cultural differences between team members from different sites in GSD projects. Two examples are related to different rules to transfer the information. In some cultures, team members can just talk and solve the problems. In others, it is needed a detailed e-mail to explain the problem. Expert 14 describes this information.

> “Taking about my experience as a developer, when you handover code to an Indian developer, they expect after the handover session gets an e-mail with the steps. If you do not do that, there is a great change does have something wrong or part of the code that is not correct. After the handover you always have to send an e-mail with all steps. I don’t know why, but I believe that is part of their culture to follow structured steps. You have to write the steps in sequence. If you just send a task probably you will have defects. This approach works very well for Indians developers. In Austin, for example, when you do that they don’t like steps, they like to understand first. You need to explain the code, once the guy understood the code, it is ok. You do not need to send an e-mail. They do not like. Even you send an e-mail, they are already doing, because they understood the code. We just send an e-mail as a follow up just to remember “Hey, this is what we discussed and

> just have a formal e-mail about our meeting". For Indians developers, if you don't do that, he will ask at the end, 'Can you send me the steps by e-mail?'." **(Expert 14)**

Some cultural differences between sites are also related to spoken and written language, differences approached to perform communication, and national holidays. Teams from different countries have different holidays. Managing holidays can be difficult in FTS development. Such information is described by the Expert 9.

> "For example, I can have teams in the Middle West and Arabic countries. Arabic countries have weekend's holiday on Fridays and Saturday. It is different from other places. So, what teams can do in these cases, if doing follow the sun? They have to decide to make plans to meet during holidays. It is an example of cultural diversity." **(Expert 9)**

In response to the second question, experts recommend meetings face-to-face or the use of conference or video calls to develop trust and cultural awareness. As teams are distributed in different locations, they can use video conference tools. Regular team meetings are recommended for the most projects. Meetings at the beginning of the project are important for building trust between team members and reduce problems relate to cultural differences.

Some organizations do not perform meetings between teams for building or increasing the level of trust. That is because companies have their own culture as described by one expert.

> "We don't use it that much. We have some special tools and rooms to do video conference and see another team. But we do not use it that much. We do not use it, not because we don't want. I believe that is a culture from the company. We are running a lot of projects at the same time, than to put all the team together in a single room sometimes is difficult. There are a lot of conference meetings happening at the same time and it can cause conflicts because there are many time zones. But, I believe that is helpful because, for example, there is a turn collocation when you have a big project, we try to put the all leaders in a site, doesn't matter in which site. They will have face-to-face meetings." **(Expert 14)**

On the basis of the inputs from the experts, the results about the BP33: Meetings between team members for building trust are given in Table 28**.**

Table 28 - Evaluation results of BP33: Meetings between team members for building trust.

	Compliance to the model	**Result**
Expert 1	In full Agreement	VALID
Expert 9	In full Agreement	
Expert 14	In full Agreement	

<u>Amendments to the model</u>

With the inputs given by the experts, BP33 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- There is a need to conduct proper cultural training programs to help team members to learn how to better communicate with team members from different cultural backgrounds.
- Video conference and teleconferences are recommended as tools to perform meetings between team members from different cultures.
- It is recommended a meeting between team members at the beginning of the project to introduce team members and cultural differences between them.
- It is recommended a meeting between team members at the end of the project to discuss what the team members have learned during a particular project.
- The number of meetings between team members to build trust should be planned according to the type of the project.
- At the beginning of the project, it is suggested for big projects to plan a face-to-face meeting between all leaders in a site. In this meeting they will define a common agreement to be followed by all sites.
- **BP35: Cultural awareness training**

BP35 aims to develop cultural awareness among team members working in a FTS project. This practice is performed to educate team members on each other culture. In order to evaluate the usefulness and relevance of the BP035 to the FTS-SPM, I asked to the experts explain how cultural diversity that exists between team members impact in GSD projects and how cultural awareness training is useful in GSD projects.

In response to the impact of cultural diversity between team members for GSD projects, experts stated that are types of cultural diversities. In some cases, cultural aspects are not critical. Companies have their own professional culture that helps to minimize cultural differences. However, team members have to learn how to better work with those differences.

Cultural diversity between team members can be positive in some cases. Team members can learn how to work with differences over the time. On the other hand, cultural diversity can have a huge impact on global projects if team members do not understand those differences. For example, communication between teams can be bad if there are misunderstandings. That will result in team's frustration and on delays along of the project.

Experts have a mixed of opinions about the benefits of cultural awareness training for GSD projects. Two experts are in full agreement and one expert is in partial agreement that cultural awareness training is useful for GSD projects. For those that are in full agreement, cultural awareness training can help in better collaboration and deal with teams' expectations. Emotional and social characteristics of team members are also mentioned by experts as a challenge to establish trust between teams. One expert describes how cultural awareness trainings are performed in his company.

> "We do trainings. It is a part of the regular company. We do online training where we have to read, there are online movies, a bit interaction, there are some situations where there are interactions between countries. It takes a half hour. It doesn't happen at the beginning of each project. It is a stage. In the middle you get practice. It is important at the beginning when you start working on global projects. Cultural awareness, we don't have at the beginning of each project. When the project starts you have to meet our colleagues and then discuss the scope of work. We can go to other countries to meet the team. We can delegate which countries will attend remote or physically. So it depends on the project. People also can use web cam and video conference." **(Expert 6)**

The same expert described that cultural awareness training provides motivation between team members. It results in a strong team and reduces the cultural diversity between team members.

On the other hand, one expert reported that cultural awareness training by itself it's not enough. It is important to make people feel as a part of the team. To this expert, fair communication between team members is the most important.

On the basis of the inputs from the experts, the results about the BP35: Cultural awareness training are given in Table 29**.**

Table 29 - Evaluation results of BP35: Cultural awareness training.

	Compliance to the model	**Result**
Expert 6	In full Agreement	VALID
Expert 10	In partial Agreement	
Expert 16	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP35 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Team members have to learn how to work with a particular team;
- The project manager as a leader has to manage cultural differences between team members;
- Communication during the project must be fair and motivational;
- Individual meetings should be performed with each team member that is not presenting a good performance during the project;

- Online movies, books and documents can be used to support cultural awareness training;
- It is suggested for youngest and oldest teams to give them the opportunity of working together in collaborative tasks. It helps to develop new skills and increasing the trust between team members;
- Sort program trainings of the 3 weeks or 4 weeks should allow team members to travel to other sites to meet the team and discuss issues of the project. In these program trainings, team members will have the opportunity to know one to each other and socialize. It is also helpful for exchange knowledge.

7.2.1.5 SP05: Task Allocation

In FTS development, incomplete tasks from one site are handed over to the subsequent site in order to continue the working day. These tasks handover involves task allocation. The idea is to convey specific details and responsibilities of the remaining to be developed tasks to the corresponding team members who are in line to take over the unfinished work. The goal is to dispense work responsibilities to the proceeding team. Thus, SP05: Task allocation includes three best practices: BP17: CPro concept, BP18: Low task granularity, and BP20: Task distribution by sequence or dependency. Next, I describe the evaluation results for these best practices.

- **BP17: CPro concept**

BP17 is based on a cooperative working model called Composite Persona (CP). BP17 consist of task allocation to CPs and not for individual team members or sites. CP members work on the vertically decomposed subcomponent in series. In order to evaluate the usefulness and relevance of the BP017 to the FTS-SPM, I asked to the experts some examples of approaches for task allocation in GSD projects and how task allocation performed on the vertical way helps to improve the team’s productivity.

The experts stated that task allocation in GSD projects can be performed in the different ways. Some projects adopt agile methodologies for project management. Thus, task allocation is defined in the backlog file and the priority is defined by the team. In agile methodologies, like Scrum, each team defines rules for the task and task allocation. In other projects, task allocation is supported by tools to assign tasks to team members. These tools help to manage deadlines, number of effort hours, number of tasks and deliverables. Task allocation is also performed by the interaction between project manager and developers’ leaders. They interact in order to define a certain level of granularity where tasks can be uncovered and can be properly assigned to the individuals. However, sometimes it is difficult to manage tasks, as described by one expert.

> “Sometimes what we can see in some of globally distributed projects that we run, it's that sometimes project managers and sometimes project leaders don’t go too far on unleashing the activities. They stop at a much higher level, what is difficult to define what should be done and sometimes for that particular level, it is difficult to estimate how long will take to complete the task, to whom should be assigned and eventually we lost track about the project.” **(Expert 19)**

In response to how task allocation performed on the vertical way helps to improve the team’s productivity, two experts stated that the most important aspect is the knowledge sharing between locations and team members. These experts recommend performing trainings to define how teams should work and communicate.

Another expert reported that is more common to allocate tasks in the horizontal way. However, there are some projects that are possible to work in a vertical way. There are some projects where the business puts a lot of pressure and the timeline is very short. In these cases, the project manager does teams’ assessment to identify where the most senior team members are.

On the basis of the inputs from the experts, the results about the BP17: CPro concept are given in Table 30.

Table 30 - Evaluation results of BP17: CPro concept.

	Compliance to the model	**Result**
Expert 3	In full Agreement	VALID
Expert 12	In full Agreement	
Expert 19	Particular context	

<u>Amendments to the model</u>

With the inputs given by the experts, BP17 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Tools can be used to support task allocation;
- It is recommended to define a certain level of task granularity to properly assign tasks to team members and define the estimated time to develop a task;
- A work breakdown structure (WBS) should be created to help project managers to start planning the work. It allows the project managers elicit what should be done and assign tasks in the right way;
- Time estimates of tasks can be used in the initial stages of the project to better allocate tasks.

- **BP18: Low task granularity**

BP18 recommends tasks broken down into small numbers of few parts. Tasks of low levels of granularity may enhance recognition accuracy [ESP03]. Additionally, low dependency tasks are supposedly easier to manage than high dependency tasks. In order to evaluate the usefulness and relevance of the BP018 to the FTS-SPM, I asked to the experts how granular should be tasks allocated to distributed teams and how low task granularity helps to enhance recognition accuracy of a task.

To the experts, the level of granularity where a task can be uncovered helps in terms of quality, productivity, and architectural control. The level of granularity can be defined according to the methodology adopted to develop a project. If the project adopts agile methodologies, tasks will break down in many parts. If the project adopts waterfall methodology, tasks will break into a few parts. Usually, waterfall projects adopt tools to support task management. In waterfall projects, the project manager has a high level planning and high level tasks. In agile projects, the project manager has a very detailed tasks and low level planning.

The response about how low task granularity helps to enhance recognition accuracy of a task was mixed. The first expert stated that low task granularity is very useful at the beginning, but very complicate and costly to put all little tasks together. Sometimes little tasks do not fit perfectly. The second expert stated that for breaking down tasks in very low level, it is needed well defined requirements. If requirements are not very well written or not well understood, it is difficult to break down tasks in low level. The low task granularity decreases the chances to have defects in the code. On the other hand, if tasks break down in high level to code the whole business requirements, the changes to have defected increases. The third expert stated that depends on how tasks are divided. If tasks are divided without to follow a logic structure, it may result in problems because a lot of communication e collaboration between team members is needed. This expert suggests dividing tasks in small parts, but respecting the relationship and the dependency between the modules.

On the basis of the inputs from the experts, the results about the BP18: Low task granularity are given in Table 31.

Table 31 - Evaluation results of BP18: Low task granularity.

	Compliance to the model	**Result**
Expert 5	In partial Agreement	PARTIALLY VALID
Expert 14	In partial Agreement	
Expert 18	Particular context	

Amendments to the model

With the inputs given by the experts, BP18 is classified as PARTIALLY VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Tasks should be broken down in accordance with the methodology adopted in the project;
- Dependencies between tasks and modules should be identified before starting the project. It helps to manage task allocation.

- **BP20: Task distribution by sequence or dependency**

BP20 recommends performing task distribution by sequence or dependency. In order to evaluate the usefulness and relevance of the BP020 to the FTS-SPM, I asked to the experts some examples where tasks cannot be divided between two or more members who are distributed across different time zones and how task distribution by sequence or dependency helps on 24 hours working development.

In response to the first question, experts stated that successful projects are very well planned. Dividing a task between two or more members who are distributed across different time zones can also be complex and difficult. In these situations, organizations adopt around-the-clock or FTS development. Such information is described by one expert:

> “For instance, when tasks can be not divided and there are two people in different locations, then we do around-the-clock and follow the sun. Actually, more follow the sun in major number. Somebody starts and then another continues. There are situation where we work with Scrum, so they can choose the tasks, discuss issues, negotiate. And there are other situations where have a big task, then we do follow the sun. On starts, another continues and if the task is not finished yet, send back to the first one.” **(Expert 6)**

In response to the second question two experts stated that task distribution by sequence or dependency is useful for 24 hours working development. The third expert on the basis of their experience stated that there is a problem when the distributed teams are architected. The first two experts suggested performing task distribution more by sequence than dependency. The tasks distribution by sequence is important because the project manager starts doing the planning, after that tasks breakdown and definition, and then tasks are assigned. Team members need to understand pieces of functionalities and how these tasks are related. One expert describes how task distribution by sequence or dependency is appropriate for FTS development.

> “I think more the sequence can happen. When we talk about follow the sun, sequence because depends on the team are working. Tasks should be divide by sequence rather dependency. I say more sequence than dependency.” **(Expert 9)**

To the Expert 9, software companies adopt more task distribution by sequence. However, task distribution by dependency is more indicated for FTS development.

On the basis of the inputs from the experts, the results about the BP20: Task distribution by sequence or dependency are given in Table 32.

Table 32 - Evaluation results of BP20: Task distribution by sequence or dependency.

	Compliance to the model	**Result**
Expert 6	In full Agreement	VALID
Expert 9	In full Agreement	
Expert 16	In partial Agreement	

Amendments to the model

With the inputs given by the experts, BP20 is classified as VALID. The following recommendation was identified to the preliminary FTS-SPM as a result of the evaluation process.

- It is recommended to perform task distribution more by sequence than by dependency.

7.2.1.6 SP06: Handoff sessions

In FTS projects, work in progress is transferred to the subsequent team localized in different site and time zone. When one production site finishes its working day, another production site start the day working on the same tasks. Team members depend on the handoff information to continue the work. The goal of a handoff session is to precisely communicate the details of the subtasks accomplished during a particular work period so that it can be efficiently continued after handoff. Thus, SP06 is one of the main sub-process in FTS development.

SP06 is developed based on four best practices: BP03: Daily stand-up meetings, BP09: Daily handoff of 30 minutes duration with each development site, BP11: Calendar of handoff sessions should be clearly defined, and BP14: Use of an FTP Server (or data repository) to exchange code and documents. I describe the evaluation results for these best practices next.

- **BP03: Daily stand-up meetings**

BP03 is based on stand-up meetings from Scrum methodology. SP03 aims to provide the project status update to team members. In order to evaluate the usefulness and relevance of the BP03 to the FTS-SPM, I asked to the experts how daily stand-up meetings is helpful to transfer project status updates from one site to another and how appropriate are stand-up meetings to perform daily handoffs.

In response to the first question the experts mentioned that daily stand-up meetings are helpful to transfer project status updates from one site to another. However, daily stand-up meetings should be focused and provide motivation for the participants. Such information is described by the Expert 10.

> "It (stand-up meetings) is really wonderful because as you meet early you can be very, very focus to have a look as the same you can have a celebration. You can say what is the good news and what are the bad news. Always is important to be fair. The high management brings motivation. So, when the project managers are doing badly, the guys should be warned. It (Daily meetings) should be not only for updating, but also for criticism." **(Expert 10)**

Daily stand-up meetings should have solid communication happing between two sites or between teams involving in delivering something that has been done collaboratively. A good documentation helps to support daily stand-up meetings. Team members should feel comfortable to use the documentation and to answer any question that other team members like to have. The main goal of FTS development is to deliver faster. Thus, daily stand-up meetings should be supported by a good documentation to help teams to be more productive. Expert 19 described this information.

.

> "If you are going to stop your work because you are not able to progress further to develop the things that you need to deliver in the other day. It will impact in productive that are based on the main objective of FTS that is to be productive. It is to deliver faster. If I'm not productive with the documentation that I have, will not be a stand-up meeting that will solve all the problems. I'm not talking about a specific methodology like stand-up meeting from Scrum. I'm talking about a general handoff meeting." **(Expert 19)**

Stand-up meetings are considered appropriate to perform daily handoffs by all the experts. One expert describes how stand-up meetings should be performed in FTS projects.

> "In a stand-up meeting is the best place to do a handoff. A stand-up meeting will be where six guys will be there, where six guys will take tasks. If you have six handoffs to be taken, the project 1 have these issues, project 2 has task 2, tasks 3. Which of these handoffs can say who is going to be responsible, who is going to take the ownership, when I will get the state of the handoff? I do get start the handoff maybe later you came back to me and I say ok, because the handoff has to be the two in two people. As I say "I finish it", he says "I finish it". When all the tasks are done, you have to go to the project manager and tell you that all tasks are done. He doesn't need to look into those. He just needs to say how the transfer has been or not done at any point. But you have to make sure the handover is done. Actually, it is the main problem of follow the sun." **(Expert 10)**

Other expert also stated that stand-up meetings do not replace a very good documentation. To this expert, a good documentation is needed to make productive teams and to support daily handoffs.

On the basis of the inputs from the experts, the results about the BP03: Daily stand-up meetings are given in Table 33.

Table 33 - Evaluation results of BP03: Daily stand-up meetings.

	Compliance to the model	**Result**
Expert 10	In full Agreement	VALID
Expert 17	In full Agreement	
Expert 19	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP03 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- There is a need of documentation to support teams during daily stand-up meetings.
- Daily stand-up meetings should be focused and provide motivation for the participants.

- **BP09: Daily handoff of 30 minutes duration with each development site**

BP09 recommends daily handoff meetings of 30 minutes duration with each development site. In order to evaluate the usefulness and relevance of the BP09 to the FTS-SPM, I asked to the experts what information is carried out during synchronous communication in GSD projects and how appropriate are daily handoffs of 30 minutes duration with each development site in FTS projects.

To the experts during synchronous communication in GSD projects are discussed mainly information related to handoffs, problems that teams may have, what have been done since the last meeting, technical aspects, conflict management and negotiation, delivery deadlines and request for changes in certain functionalities. Usually it is a kind of communication to address critical issues from the project.

In response to the question related to daily handoffs of 30 minutes duration, the experts stated that 30 minutes with each development site is sufficient to perform daily handoffs. During 30 minutes team members can transfer the essential information that another site needs to continue the work.

Daily handoffs can help teams to discuss activities and make sure that the next team will have a good understanding of the previous work. However, team members have to use the same tools, communication structure, and technologies. It is also recommended to define a template to structure the information discussed during handoffs.

On the basis of the inputs from the experts, the results about the BP09: Daily handoff of 30 minutes duration with each development site are given in Table 34.

Table 34 - Evaluation results of BP09: Daily handoff of 30 minutes duration with each development site.

	Compliance to the model	Result
Expert 8	In full Agreement	VALID
Expert 12	In full Agreement	
Expert 18	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP09 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- It is suggested to define a template to structure information discussed during daily handoffs;
- Team members should perform daily handoffs using the same tools, communication structure and technologies. It helps to avoid incompatibilities between technologies;
- In order to keep daily handoff meeting of 30 minutes duration with each development site, team members should discuss only the essential information that the other site needs continue the work.

- **BP11: Calendar of handoff sessions should be clearly defined**

BP11 aims to define a calendar to handoff sessions. Its implementation makes team members available to interact according to the same timetable. In order to evaluate the usefulness and relevance of the BP011 to the FTS-SPM, I asked to the experts how does the resource calendar help for better coordination and communication between team members working on GSD projects and how helpful is establishing a calendar for handoff sessions in FTS projects.

In response to the first question, all experts stated that resource calendar is useful to define an agenda for meetings and make team members available. Resource calendar is useful for the project planning meetings, stand-up meetings and sprint retrospectives. Resource calendar is also helpful to define a specific data for teams provide status of the work that has been executed. Resource calendar can also be used to plan handoff meetings and deliverables. Such information is explained by one expert.

> "If handoffs should occur at the end of the day, people are aware that at that particular time, they will have to deliver something. It is a goal. It reinforces accountability, ownership. You should have, because at the end of the day is what the project managers do. They schedule a specific meeting. So, everyone can provide their status about something, they owner and they should deliver a certain period of time." **(Expert 19)**

On the other hand, implementing resource calendar does not have to be used to schedule individual meetings. The experts recommend for individual meetings to check the availability of the participants. For this reason, the team members' calendar should be public for all teams.

In response to the second question, two experts stated that establishing a calendar for handoff sessions is helpful for FTS development. The main benefits are receiving commit data on time and organizing weekly and daily meetings. On the other hand, another expert suggests define an informal calendar for handoff sessions. This expert stated that communication between team members must be informal. Such information is described by the Expert 19:

> "We can use the communicator to transfer the work. It could be half hour or 5 minutes. This doesn't have to be in the calendar. 'I'm an adult, you are an adult'. We have to be aware of our work. If he is not available, I'll set a meeting for tomorrow. The other site that doesn't have a handoff session to receive the explanation of the task' status can pick another task in the backlog file. Of course, there is a delay, because in the other round, there is no handoff to receive the task. It is not perfect as we wanted." **(Expert 19)**

The explanation given by the expert to make informal handoff sessions is the possibility to start news tasks. Since, a software project has a set of tasks, team members can pick new tasks. If there are no more new tasks, then team members should schedule a handoff session. Also, it is recommended to send notes by e-mail at the end of the day. Team members should be able to understand in the right way the information received by e-mail. Structured information in e-mail can help to reduce misunderstandings. Therefore project managers need to be aware of the project status. Such information is described by the Expert 19.

> "If I have this situation that there are no more tasks to pick or a situation like that, we have to commit at the beginning one to each other to make available for the handoff session. Another situation is when I have to talk with you at the end of the day and you are not there. That is a bridge of trust of the commits. We can define it in the calendar. At the end of the day, half hour of my business day I'll be available. It is a part of our commits. But if we define that I can send an e-mail with the information is ok. It is our agreement. I send notes by e-mail by the end of the day. Imagine if you are in England and I'm in Romania. When my business day finished, you are at work. When your business day finishes, I'm not at the work because I'm sleeping. So, what you do then? You send out an e-mail. This works. Sometimes the people don't understand the information in the e-mail. This happens. It is the real. Tomorrow, I'll be again in the office and I'll tell you that I couldn't understand you, the task that you gave to me. So, we will talk and explain the task to understand you and you understand me. When we start to work together, I have to learn how to send the

> messages that you can get right. Do you need a more detail's message or do you need straight crisp's messages. You need structured messages or certain of the stories. This is a kind of thing that we learn working one with others." **(Expert 19)**

On the basis of the inputs from the experts, the results about the BP11: Calendar of handoff sessions should be clearly defined are given in Table 35.

Table 35 - Evaluation results of BP11: Calendar of handoff sessions should be clearly defined.

	Compliance to the model	**Result**
Expert 4	In full Agreement	VALID
Expert 6	In full Agreement	
Expert 19	In partial Agreement	

<u>Amendments to the model</u>

With the inputs given by the experts, BP11 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- The team members' agenda must be public for the whole team. It helps to schedule meetings;
- Handoff sessions can be also supported by the use of e-mails in case of time zone constraints between teams;
- Team members should exchange an e-mail at the end of each handoff session with the information discussed and all agreements made by the team;
- E-mails exchanged by team members should contain structured information in order to avoid misunderstandings;
- Team members have to learn how to write and understand e-mails in a proper way;
- When a team cannot attend a handoff meeting, a new task should be taken. If there are no more new tasks, a handoff meeting should be scheduled with the previous team members.

- **BP14: Use of an FTP Server (or data repository) to exchange code and documents**

BP14 recommends the use of an FTP Server (or data repository) to exchange code and documents. This practice aims to facilitate access to the project data. In order to evaluate the usefulness and relevance of the BP14 to the FTS-SPM, I asked to the experts some examples of technologies for exchange software code and documents in GSD projects and how it helps to exchange software code and documents in GSD projects.

All experts stated that the most popular technology for exchange software code and documents are SVN (Apache Subversion) repositories. To the experts, these technologies are useful for GSD projects.

In response to the use of an FTP server (or data repository) to exchange code and documents, experts mentioned that the major problem is coordination against communication. The definition of an FTP server (or data repository) instead of multiple resource information is important because motive all the teams to use only a repository. Some benefits are better process communication, experience and maturity. Some experienced teams can help for better collaboration.

On the basis of the inputs from the experts, the results about the BP14: Use of an FTP Server (or data repository) to exchange code and documents are given in Table 36.

Table 36 - Evaluation results of BP14: Use of an FTP Server (or data repository) to exchange code and documents.

	Compliance to the model	**Result**
Expert 7	In full Agreement	VALID
Expert 11	In full Agreement	
Expert 20	In full Agreement	

Amendments to the model

With the inputs given by the experts, BP14 is classified as VALID. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Such technologies as SVN repositories can be used for exchange software code and documents in GSD projects;
- It is recommended to use only a data repository instead of multiple resource information. It helps in better information and data coordination.

7.2.1.7 Process model overview: SP01 - Team setup

SP01 starts the FTS-SPM. SP01 aims to identify sites and allocates human resources for the project. In order to evaluate the usefulness and relevance of the SP01 to the FTS-SPM, I asked two questions to the experts. First, I asked how important is to identify available sites and human resources before starting the project planning and then, how appropriate is the sequence flow between SP01 and SP02: Project planning shown in Figure 19.

Experts stated that is important to identify available sites and human resources before starting the project planning. That is because software development is made by people and depends on teams' availability in the sites. Identifying available sites before starting to plan the project allows arranging teams set up and minimize risks. However, FTS does not work because its designs, but because it have people in more locations to develop software faster. Such information is described by the Expert 3.

> "FTS don't happen because its designs. FTS the most of the time happens because we have people in more locations to work faster. It is never an idea or a situation. Of course it makes sense, to do team's forming before planning. But the question is "is it feasible?" FTS is not the goal is the way to work. It's not just the project manager gets assigned the number of people and decide to do FTS. There is another way around. We have this business proposition, reducing the time-to-market. So far, we cannot do it with local team then you decide to do it with FTS settings. I don't think so, it is a kind of project manager's decision, because it doesn't work that way. If you are a project manager, you want the people close to you. I don't believe that it works that way." (**Expert 3)**

The sequence flow between SP01 and SP02 is considered appropriate. To the experts is better first to have a team. The project planning is made based on the available team members. One expert also suggests adding an arrow going back from SP02 to SP01. If an inappropriate site was selected, it is possible to go back to SP01 and decide for another site.

> "There is no chance to have an arrow going back (SP02 to SP01). Maybe the project planning has wrong team selected. I suggest an interaction here. Maybe the project manager wants feedback from the company and says "Look the team set up in the project planning or in my mind doesn't are matching". This looks a meta-step, should be teams setup inside of project planning. To me it is not very clear to separate. Seems something that should be more interactive." **(Expert 2)**

On the basis of the inputs from the experts, the results about the SP01: Team setup are given in Table 37.

Table 37 - Evaluation results of SP01: Team setup.

	Compliance to the model	**Result**
Expert 2	In partial Agreement	VALID
Expert 3	In full Agreement	
Expert 16	In full Agreement	

<u>Amendments to the model</u>

With the inputs given by the experts, SP01 is classified as VALID. The following recommendation was identified to the preliminary FTS-SPM as a result of the evaluation process.

- The SP02: Project planning should interact with SP01: Team setup and vice-versa. An arrow going back from SP02 to SP01 allows the project manager to exclude or include new sites in the project.

7.2.1.8 Process model overview: SP02 - Project planning

SP02 is started following SP01 as shown in Figure 19. The project planning is defined in SP02. In order to evaluate the usefulness and relevance of the SP02 to the FTS-SPM, I asked to the experts some examples of the information described in the project planning and how appropriate are the sequence flow between SP02 to SP03: Communication protocol, SP04: Cultural training and SP05: Task allocation.

Experts reported that in the project planning are described the most important aspects of the project. The PM describes the project goals, how much resource the project will have, the project budget, deadlines and available resources. The project planning is also made based on the company's culture. There is a lot of information described in the project planning, but the most companies spend significant time trying to understand the project timeline. The project manager spends o lot of time trying to understand exactly what is needed and when. There are projects that can take more than four years.

In response to the sequence flow defined in the FTS-SPM between SP02 to SP03, SP04 and SP05, there was a mixed response. The first expert on the basis of his experience stated that SP01, SP03, and SP04 are integrated. SP01 and SP02 should be developed at the same time.

> "To me there are two parts. One is based on my research. It works differently and better. In which is much better setup teams here ("project planning")…what are the locations, where the people are. It works better. The way that you describe here, shown in the picture (sequence flow between SP02, SP03, SP04 and SP05) doesn't make sense to me. To me it is very interactive, then you have to define here (SP02) you have the sequence back (SP02 to SP03 or SP04). To me communication protocol, cultural training and teams setup is something integrated." **(Expert 5)**

The second expert also on the basis of his experience stated that SP03 and SP04 are part of the project planning. He stated that how the team will communicate and how the tasks will be handover to another site are discussed earlier.

> "The communication protocol, cultural training is part of the project planning. Usually when you do it in the project planning, you say how the team will communicate. I say you have three parts that you need. One is this in my daily tasks. I need a plan that says what is the time, what are the milestones, how are my process. I need a project that says this is the milestones, this is my timeline, this is what I've designed, coding completed, tested and fixed and whatever I did. That has come earlier." **(Expert 10)**

The third expert stated that the sequence flow between SP02 to SP03, SP04 and SP05 is appropriate. However, SP04 do not need to be developed in some projects. That is because team members may have previous experience in working in that culture. However, if the team does not have experience in that culture, SP04 should be developed. This expert recommends identifying if team members have previous experience in that culture before developing SP04.

> “SP04 is performed always? What happens if the participants already have experience or knowledge? Sometimes, they will not need cultural training. My point ii…SP04 is not always necessary at the beginning because sometimes the participants may have experience in those cultures. So, they need it at the beginning of the project. Participants involved in other projects with the same countries. You don’t need to repeat the same training and spend the time doing it. It will be necessary to identify if people have knowledge of those cultures before planning a cultural training. If they do not have experience in that culture, so you can provide them. If they have previous experience, they step should be not performed.” **(Expert 15)**

On the basis of the inputs from the experts, the results about the SP02: Project planning are given in Table 38.

Table 38 - Evaluation results of SP02: Project planning.

	Compliance to the model	**Result**
Expert 5	In disagreement	INCONCLUSIVE
Expert 10	In disagreement	
Expert 15	In partial agreement	

Amendments to the model

With the inputs given by the experts, SP02 is classified as INCONCLUSIVE. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- It is suggested to integrate SP01: Team setup and SP02: Project planning. These sub-processes should communicate in order to better define teams’ settings and the project planning;
- It is suggested to develop SP03 and SP04 into SP02. These sub-processes can be integrated;
- It is recommended to identify if team members have previous working experience in cultures involved in the project before developing SP04.

7.2.1.9 Process model overview: SP03 - Communication protocol

SP03 is started in parallel following SP02 as shown in Figure 19. SP03 defines communication resources and the schedule for synchronous communication between sites. In order to evaluate the usefulness and relevance of the SP03 to the FTS-SPM, I asked to the experts how project settings impact on the communication protocol planning and how appropriate is the sequence flow between SP02 to SP03.

Two experts stated that has a high correlation between the project settings and the communication protocol planning. As more distributed a team is, more communication is needed. The communication protocol depends on the project settings. Thus, the communication protocol should be reviewed all the time in order to help team members better communicate.

In response to the sequence flow between SP02 to SP03, I found a mixed response. The first expert stated that SP02 is made once during the project. On the other hand, SP03 can be improved as the SP02 is built. This expert suggested two arrows between SP02 to SP03. One arrow goes from SP02 to SP03 and another arrow goes SP03 to SP02. The second expert suggests SP04 before SP01 and SP02.

> "I suggest cultural training at the beginning. They know how the team is setup up, they know which communication mechanisms and in which situation will be used. They will know if an action can make someone angry one to each other and if is helpful to send an e-mail forward. Thus, cultural training must be very early. It helps for team building." **(Expert 4)**

The third expert stated that the sequence flow between SP02 to SP03 is appropriate. However, SP03 should be built in parallel with SP02. SP03 will be finished when SP02 is also finished. The PM does not have to wait to finalize the SP02 to start SP03.

On the basis of the inputs from the experts, the results about the SP03: Communication protocol are given in Table 39.

Table 39 - Evaluation results of SP03: Communication protocol.

	Compliance to the model	**Result**
Expert 3	In disagreement	INCONCLUSIVE
Expert 4	In disagreement	
Expert 19	In full agreement	

<u>Amendments to the model</u>

With the inputs given by the experts, SP03 is classified as INCONCLUSIVE. The following recommendations identified added to the preliminary FTS-SPM as a result of the evaluation process.

- The communication protocol should be defined and reviewed according to the project settings and the team members;
- SP02: Project planning and SP03: Communication protocol should be done in parallel;

- It is suggested to perform a cultural training with available team members before start the project. It helps to build teams.

7.2.1.10 Process model overview: SP04 - Cultural training

SP04 is started in parallel following SP02 and SP03 as shown in Figure 19. SP04 develops cultural training sessions in order to establish trust between team members. SP04 may be developed many times during the project to re-establish the trust between team members. In order to evaluate the usefulness and relevance of the SP04 to the FTS-SPM, I asked to the experts how cultural issues can affect of the team's performance and how cultural training sessions performed along of the project are helpful for better communication and collaboration between team members. This second question aims to evaluate the sequence flow to this sub-process.

All experts stated that there are several cultural differences. The first expert described the difficult to arrange meetings between teams from different cultures. This expert gave an example of Indian and German teams. German teams follow a strict time for meetings whereas Indian teams do not follow a strict time. The second expert also mentioned German teams as an example. To this expert, delays are considered a lack of respect for German teams. The same expert also described the use of e-mail in different cultures. In some cultures to write an e-mail with a lot of information can get people frustrated and misunderstands can happen between them. Another example is given by the third expert. This expert described a situation where Brazilians, Indians and Japanese faced difficulties to communicate due to the cultural differences. Such information is described by the Expert 14.

> "We have a project with a Japan team and we have between Brazil, India and Japan. In Japan, they have a special thing that during the calls like we were discussing technical requirements. They ask permission to step away from the call to discus in Japanese language and then come back to the call. "Ok, we agree with these requirements, go on". When, we discussed the second requirement, the guys go way the call and discuss internally. That causes some noise in the team because "Why they are doing that, if we are working as a team? Why the step way to talk?" So, that time again, we have some budget to move the leaders to Japan, that time I was a developer. The leader that project went with other leaders to Japan. That time, we learn that they do that because one of the challenges for Japanese developers, at least in this project was to understand the accent of Brazilian, US and Indian team. Sometimes during the meeting, only one or two guys were able to understand the whole conversation. So, they decide to move on and explain one to others, because if a Japanese team member has a question, he was not able to ask because he didn't understand the whole conversation." **(Expert 14)**

In response to how cultural training sessions performed along of the project are helpful for better communication and collaboration between team members, I found two different answers. Two experts stated that performing cultural training sessions along of the project do not help for better communication and collaboration between team members. They recommend adopting other strategies at the beginning of the project to train people in different cultures. Such information is explained by one expert.

> “People do not do trainings. I think in the middle is not helpful. My preference is to have this kind of travels, videos at the beginning. After we know these are our team members, this is the locations, this is the scope of the work, what is what we need to delivery and when. Exactly after this event, the client does team building, cultural awareness, building trust, learning about each other, that is the moment. If you do it in the middle of the project, you may have a wide bomb in your team, in your project. And then, it’s gonna explode and you have to fix things. If you start early building the relationship between team members, it will work better.” **(Expert 6)**

Another expert on the basis of his experience stated that performing cultural training sessions along on the project can be helpful and positive for all teams. However, in his company they do not do cultural trainings. The company offers corporate trainings and courses for individual development. Team members can choose to attend or not the corporate trainings and courses. It is not mandatory for team members.

> “I think it is extremely helpful if a team can do that. I think is extremely positive for everyone. In my company, we don’t do that. What we do have is a corporate training that we talk about cultural differences. So there are corporate communications that come out from corporate communication teams and there are specific trainings available for everyone. We have web base training where you can have all sorts of trainings related to cultural differences. We can also attend a course that can talk specifically about India, US, Chine… It is more like individual development work rather than something that is proactively being established by the project.” **(Expert 19)**

On the basis of the inputs from the experts, the results about the SP04: Cultural training are given in Table 40.

Table 40 - Evaluation results of SP04: Cultural training.

	Compliance to the model	**Result**
Expert 6	In disagreement	INCONCLUSIVE
Expert 19	In full agreement	
Expert 20	In disagreement	

Amendments to the model

With the inputs given by the experts, SP04 is classified as INCONCLUSIVE. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- It is recommended to have a budget from the project for team leaders to meet and to discuss how cultural issues can be alleviated for those particular team members;
- It is suggested the local team members to meet and spread their knowledge about culture for all team members before the project start;
- Team members can be transferred to other sites for 2 or 3 weeks to learn how to better collaborate;
- It is suggested to transfer team members for 2 or 3 weeks for cultural immersion in other sites.

7.2.1.11 Process model overview: SP05 - Task allocation

In the preliminary FTS-SPM, SP05 is developed at the beginning of each business working day as shown in Figure 19. SP05 provides tasks for the day. In order to evaluate the usefulness and relevance of the SP05 to the model, I asked to the experts how the task allocation is managed in GSD projects and how appropriate is to execute daily task allocation at the beginning of each working day (24-hour) in FTS projects.

In response to the first question, the experts mentioned two strategies to manage task allocation in GSD projects. The first strategy is based on agile methodologies. The use of agile methodologies allows having three rounds for task allocation. The first and second rounds are used for task allocation and the third round is used as an additional activity to synchronize tasks between sites. The second strategy is based on teams’ experience and availability.

In response to the second question, I found a mixed response. The first expert stated that in his company, they follow Scrum methodology and the teams make the decision for each task will be picked. They did not have a daily task allocation at the beginning of each working. In further discussion about the daily task allocation, this expert stated that it is inappropriate for FTS projects. The second expert stated that the project manager takes care of the task allocation management. The task allocation is supported by tools and methodologies. On the other hand, if the project has good team’s leaders as good team members, they are able to manage tasks doing daily task allocation. However, it can be very costly for the project manager. Thus, daily task allocation for FTS depends on the project. The third expert stated that tasks should be allocating take account how handoffs are performed and which countries are involved.

On the basis of the inputs from the experts, the results about the SP05: Task allocation are given in Table 41.

Table 41 - Evaluation results of SP05: Task allocation.

	Compliance to the model	**Result**
Expert 7	In disagreement	INCONCLUSIVE
Expert 15	Particular context	
Expert 20	In disagreement	

<u>Amendments to the model</u>

With the inputs given by the experts, SP05 is classified as INCONCLUSIVE. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- Different strategies can be used to manage task allocation in GSD projects like task allocation based on agile methodologies, teams' experience and availability.
- Task allocation should be performed according to handoffs definitions and team setup.

7.2.1.12 Process model overview: SP06 - Handoff sessions

SP06 is started following SP05 as shown in Figure 19. SP06 aims to receive and to transfer tasks in progress, new tasks and project updates. At the beginning and at the end of each working day shift, SP06 is developed. The process finishes when at the end of a working day shift, there are no more tasks to develop. In order to evaluate the usefulness and relevance of the SP06 to the FTS-SPM, I asked to the experts to describe the relation between task allocation and handoffs and how appropriate is the sequence flow between SP05 to SP06.

In response to the first question, the experts recommend to break down tasks into small parts. That is because it helps tasks and handoffs management. Task allocation also should consider some factors such as available locations, available resources (part time or full time), and it is needed to make sure that the skills that team members have will match with the skills required to perform those tasks. One expert recommends to the project manager to discuss the tasks with the whole team at least at the beginning of the project to clarify some questions that the team members may like to have.

In response to the second question, the experts stated that the sequence flow between SP05: Task allocation to SP06: Handoff sessions is appropriate. The experts mentioned that as defined the sequence flow between these two sub-processes, it is possible to perform the daily task allocation. The problem of performing the task allocation is related to how small tasks can be divided. If tasks are small, a team can carry on during a day and it makes easier for the next team to continue the work. However, if tasks are difficult to split, those can be not done during one day and it will be more difficult to transfer the work to another person. The experts recommend defining a good communication protocol to solve this problem. The communication protocol should be established according to how handoffs are planned. Such information is described by one expert:

> “You have to make sure when you establish communication protocol, you establish the handoffs. So, my point is, SP06 is the handoff execution or definition? Usually when you establish the communication protocol, you also establish how handoffs will be performed. You define tools, communication frequency, that every day the team has a handoff of 30 minutes, etc… it is in the communication protocol. Usually the communication protocol or planning is done as part of the project planning. It is kind a project planning has several scope plan, communication plan, several items in the project planning.” **(Expert 8)**

On the basis of the inputs from the experts, the results about the SP06: Handoff sessions are given in Table 42.

Table 42 - Evaluation results of SP06: Handoff sessions.

	Compliance to the model	**Result**
Expert 8	In full agreement	VALID
Expert 15	In full agreement	
Expert 17	In full agreement	

<u>Amendments to the model</u>

With the inputs given by the experts, SP06 is classified as VALID. The following recommendation was identified to the preliminary FTS-SPM as a result of the evaluation process.

- Time window should be applied according to the teams’ and organizations’ culture.

7.2.1.13 Process model overview: Whole model

In order to confirm the results from previous sections, I asked two questions to the experts to evaluate the usefulness and relevance of the preliminary FTS-SPM as a whole. First, I asked to the experts how appropriate are the sub-processes included in the FTS software process model as a whole and then, how appropriate is the sequence flow between sub-processes as a whole.

In response to the first question, the experts suggested some changes related to sub-processes included in the preliminary model. The first expert suggested to perform SP04:

Cultural training before SP02: Project planning. To this expert, it will help to build the team. SP04 should be part of SP02.

> "I think "Cultural training" should be done after project planning. It could make part of the team set up. It could make sense to do "Cultural training" before you do project planning. Then you know if the team can work together before you do the planning. To plan is something that the project manager does. Planning is something that team does. In my it means to take a couple of days. We don't do that, because there people have culture. We can change that. What we can do is training to you, understand me and you understand me. We can avoid conflict and creating a better way to understanding one to another." **(Expert 13)**

The second expert stated that the sub-processes included in the preliminary model are appropriate to run a FTS project. Additionally, this expert mentioned the importance of identifying different levels of culture to train people about the process. For this expert, there are two different levels of culture. There are national culture and professional culture country, as described by the expert:

> "I think you are gonna to run your FTS model, and then really talk about the team's culture that exists in this process. So, you have to think about culture in different levels. So, not only the national culture, but could also be professional culture country in terms of the whole software development, testing or wherever. The professional country level, the professional has to give trained in this different mode or something in software development. You have to think how they have to be trained about the process." **(Expert 7)**

The third expert suggested including in the preliminary FTS-SPM an integration phase and an additional quality assurance phase. To this expert when all tasks are finished, it is needed to integrate them into a larger project.

> "I think what you are trying to do here is to define how a task starts and goes through the sites and what point exactly becomes rude. If a task is finished, it means to be integrated into a larger project and I don't see that here. So, a task goes around and around and updated and it is finished. Technically, to put together on the rest of the software, I don't really see that. What happens in the software, software puts the pieces that are probably integrated. I don't see integrating here. There is no integration at all. I'm not sure it is correct. It is not just how it looks. Other things happen and you have to think about that. I suggest putting an integration phase, an additional quality assurance." **(Expert 11)**

In response to the second question related to the sequence flow between sub-processes, I found a mix of responses. Expert 13 stated that the sequence flow between sub-processes as a whole is appropriate for FTS development. Additionally, this expert mentioned that the most companies do not have a process. Thus, these companies fail mainly in the culture because they do not understand how important is to perform cultural awareness trainings. This expert suggests keeping the teams that have been working together for new projects. These teams have learned how to work together. This practice is called by the expert as “table teams”.

> “You design a perfect process. The most companies do not a process like this. When they fail is the culture. Of course cultural awareness training is important to understand, but have much more important stuff. These other sub-processes are more important. But still, the software process makes sense. I have learned in the past years that a team that have worked together and they have learned how to work together; we should never to separate this team at the end. You should keep this team together. In my company, we created table teams. These teams have worked together years. The project manager goes to the team and it is not we make a team for the project. It is much more effective because we are not setting teams all the time, training and teach them all the time. Overall, these other steps seem well. It is how it works in practice.” **(Expert 13)**

The second expert stated that there are different levels of activities. SP02 and SP03 are on the same level, but SP05: and SP06 are on another level. For example, SP03 cannot be performed without SP02.

> “You have SP03 and SP04, which is very nice. Then you have many levels of activities. For example, the communication protocol is something not isolated, very well established by the team, and the order to do. So, there is no real process there, just continue something. The practices you can use for a while. So, I think this is good, but I not comfortable with different levels of activities, SP02, SP03. Maybe consider different SP06 and SP05.” **(Expert 11)**

The third expert stated that there is a recursive link (sequence flow) from SP04 to SP02. Further, this expert explains that is important to identify national and professional culture to train people about the FTS-SPM. This expert also suggested changing the words in the sequence flows between SP05 and SP06, “Global working day” besides “Working day” and “Local working day” besides “Working day shift”.

On the basis of the inputs from the experts, the results about the whole model are given in Table 43.

Table 43 - Evaluation results of the whole model.

	Compliance to the model	Result
Expert 7	In full agreement	INCONCLUSIVE
Expert 11	In partial agreement	
Expert 13	In partial agreement	

<u>Amendments to the model</u>

With the inputs given by the experts, the preliminary process model and the sequence flow between sub-processes are classified as INCONCLUSIVE. The following recommendations were identified to the preliminary FTS-SPM as a result of the evaluation process.

- It is suggested to perform SP04: Cultural training before SP02: Project planning in order to select the most appropriate teams;
- When all tasks are finished is needed to integrate those tasks. An integration phase should be added to the FTS-SPM;
- It is recommended to add to the FTS-SPM a quality assurance phase;
- It is important for running the FTS-SPM to train people about sub-processes and best practices;
- The project manager should identify cultural levels differences between team members in order to provide the appropriate cultural training;
- Teams with experience in working together can be more effective for new projects. It is recommended to create “table teams” in order to keep experienced teams in working together in the same project;
- It is recommended to use the words “Global working day” besides “Working day” and “Local working day” besides “Working day shift”. These words are in the sequence flows between SP05 and SP06 in the FTS-SPM.

7.2.2 The Expert Panel Contributions

The preliminary FTS-SPM evaluation made through an expert panel aimed to gather the view of experts about the applicability of best practices included into sub-processes for FTS development and to gain an understanding of how best practices included into sub-processes can support FTS projects (see Section 3.2). As a result, the adoption of an expert panel offered two main contributions for this research. The first contribution is related to the applicability of best practices. It is interesting to observe that not all twenty-five best practices are perceived as high value practices for FTS projects by the experts. I identified 21 best practices (84%) as Valid, 3 best practices (12%) as Partially valid, and 1 best practice as Context specific (4%). This result shows that there are perceived benefits more in some best practices than others. Such contribution will help to refine the preliminary FTS-SPM.

The second contribution is related to the support given by best practices and sub-processes to FTS projects. I observed that some best practices mentioned in the preliminary model can provide more benefits for FTS development if some amendments are made. Some experts have reported their own experience adopting a particular best practice in GSD projects. Thus, it was possible to collect some recommendations to improve best practices and sub-processes to support FTS characteristics. It can promote a better acceptance of the model in the software industry. Additionally, such contribution may provide useful information to both practitioners and researchers.

7.3 Chapter Summary

In this chapter, I presented results from the Phase 3 – Evaluation and Evolution (see Figure 3). In this research phase, I conducted the preliminary FTS-SPM evaluation made through the design validation and an expert panel. The design validation method was conducted with research experts from Lero. The design validation showed: 1) How best practices and lessons learned contributed to define sub-processes; 2) How sub-processes are related and in which sequence flow the sub-processes should be carried out; 3) How specific sub-process status, SP05: Task allocation promotes SP06: Handoff sessions to determinate the sequence flow to SP05 or to determine the beginning or the end of the day (final state).

The expert panel method was conducted with twenty experts. Experts with an average of 10 years of experience in GSD participated in the expert panel. I collected more than 10 hours of data recording interviews. Based on these data, I analyzed and classified all best practices and sub-processes defined in the preliminary FTS-SPM. Twenty-one best practices (84%) as Valid, 3 best practices (12%) as Partially valid, and 1 best practice as Context specific (4%). Related to the sub-processes, the sequence flows between four sub-processes were classified as Inconclusive and two as Valid. The sequence flow between sub-processes as a whole was classified as Partially valid.

The opinion of experts about best practices and the sequence flow between sub-processes took into consideration mainly its benefits for increasing the software development productivity and quality. Some experts have reported their own experience adopting a particular best practice in GSD projects. Thus, it was possible to collect some recommendations to improve best practices and sub-processes to support FTS characteristics.

8. FTS-SPM: THE FOLLOW THE SUN SOFTWARE PROCESS MODEL

This chapter presents the FTS-SPM (Follow the Sun Software Process Model). The model is the main contribution this thesis. Section 8.1 describes the process model construction. Section 8.2 describes the structure of the FTS-SPM. Section 8.3 describes best practices included into sub-processes and demonstrates its significance for FTS development. Section 8.4 describes the final considerations about the model. Section 8.5 summarizes this chapter.

8.1 Process Model Construction

The FTS-SPM was built based on results from the research phases defined in the research design (see Figure 3). The research design comprised three research phases: Phase 1 – Exploratory, Phase 2 – Development and Phase 3 - Evaluation and Evolution.

In Phase 1, I collected data for the preparation of a preliminary software process model for FTS. In Phase 2, I built the preliminary software process model for FTS called FTS-SPM (Follow the Sun Software Process Model). The preliminary process model was evaluated in Phase 3. The evaluation process was made through the design validation and expert panel research methods. The adoption these methods resulted in a set of contributions which were used to propose the FTS-SPM, as shown in Figure 20.

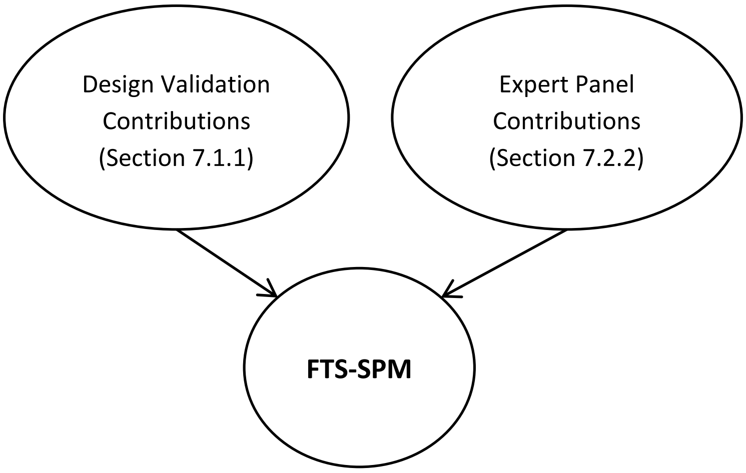


Figure 20 – Process model construction overview.

From the design validation, I collected data from the research experts regard to the preliminary FTS-SPM design (see Section 7.1). As a result, I made some changes in the process model design following recommendations given by the experts. These changes aimed to improve the sequence flow between sub-processes and support FTS characteristics. Changes made in the preliminary process model design denote the contributions from the design validation for this study. These contributions are described in details in Section 7.1.1.

From the expert panel, I collected data from experts in GSD regards to the usefulness and relevance of each best practice and sub-process included in the preliminary FTS-SPM. I analyzed and classified all best practices, sub-processes, and the process model overview as Valid, Partially Valid, Context Specific or Inconclusive in the evaluation process. To evaluate the process model overview, the preliminary FTS-SPM[ii] picture (see figure 19) defined in the Design Validation was provided to the participants. The adoption of the expert panel resulted in key contributions which helped to enhance the model. I describe these contributions in details in Section 7.2.2.

The proposed FTS-SPM includes best practices measured as Valid in the evaluation process and its ancillary recommendations. These best-practices were considered to have compliance with the model. Regarding to the sub-processes and the process model overview, only the sequence flow between SP01 to SP02 and SP05 to SP06 were measured as Valid in the evaluation process (see Table 12). Thus, I have kept the sub-processes definition and the process model overview as previously defined in the Design Validation. However, I have included in the model recommendations given by the experts regards to the sequence flow between SP01 to SP02 and SP05 to SP06. Based on these recommendations, an arrow going back from SP02 to SP01 was inserted. This arrow shows that SP02 can interact with SP01 to exclude or include new sites in the project. I present the proposed FTS-SPM in Figure 21.

8.2 Structure of the FTS-SPM

The FTS-SPM comprises six sub-processes as shown in Figure 21: *SP01 - Team Setup, SP02 - Project Planning, SP03 - Communication Protocol, SP04 - Cultural Training, SP05 - Task Allocation,* and *SP06 - Handoff Sessions.*

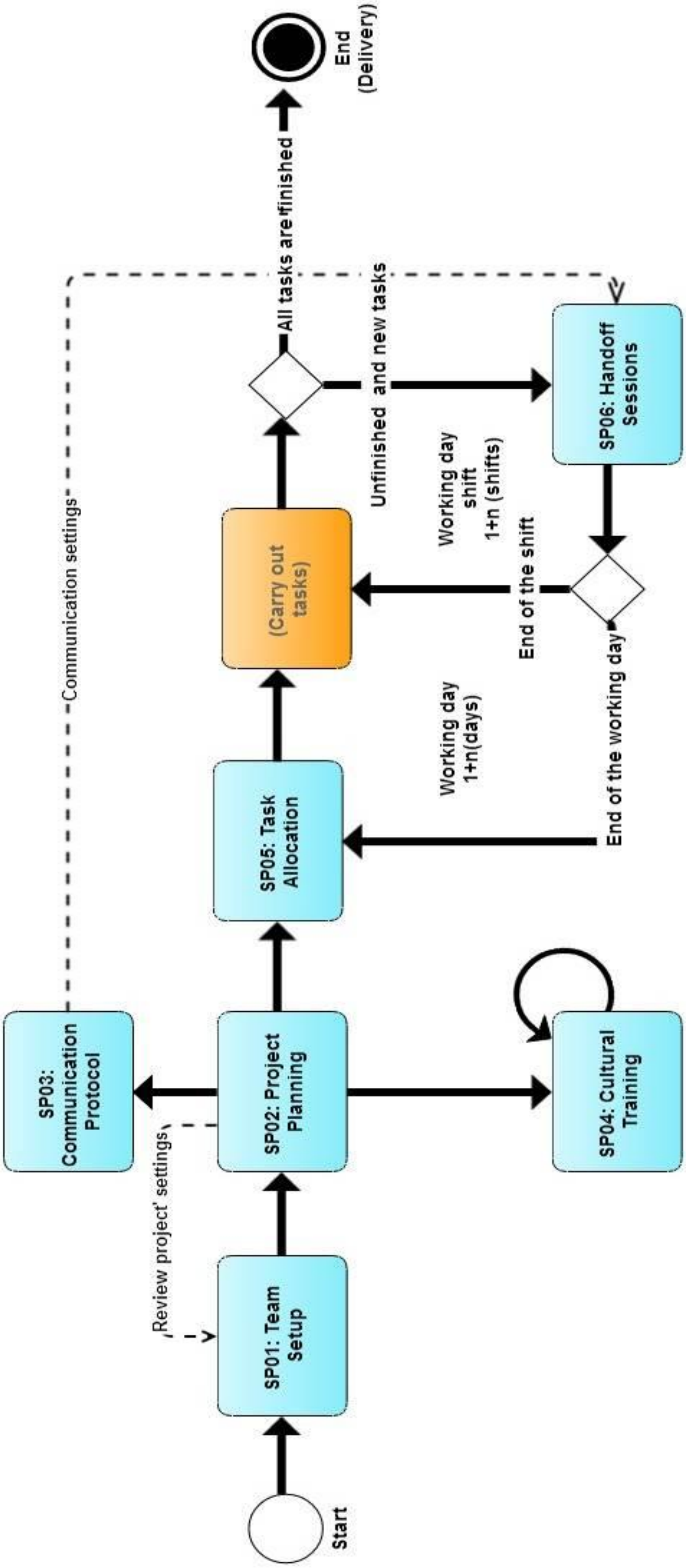


Figure 21 - FTS-SPM: The Follow the Sun Software Process Model.

The FTS-SPM has an initial and final state. The initial state causes the process to start with Team Setup (SP01). The final state is the end of the process when all tasks were finished, at this point there is a software delivery. SP01, Team Setup, starts the process. It aims to identify available sites and allocates human resources for the project. Information about each site should be collected in order to make future decisions. It is important to verify if there are staff, cost or scope restrictions in each site. These restrictions and others related to project goals should be considered to define priorities in order to select appropriate sites.

SP02, where project planning is defined, is started following SP01. SP01 provides information to develop the project plans, and these are developed by the project manager. SP02 interact with SP01 and vice-versa to include or exclude new sites in the project.

SP03, SP04 and SP05 are started in parallel following SP02. SP03 defines communication resources and the schedule for synchronous communication between sites. The project manager can suggest technologies or tools already used in other projects. SP04 develops cultural training sessions in order to establish trust between team members. SP04 may be developed many times during the project to re-establish the trust between team members (loop arrow).

At the beginning of each working day, SP05 is undertaken, as it provides tasks for the day. A software project may have many working days. Within SP05, the sequence and dependency relationships between tasks must be identified. All details about tasks sequence and dependency should be identified and described in the project planning.

SP06 is started following SP05. SP06 aims to receive and to transfer tasks in progress, new tasks and project updates. At the beginning and at the end of each working day shift, SP06 is undertaken. One working day may have at least two working day shifts. The process finishes when at the end of a working day shift, there are no more tasks to develop.

'*Carry out tasks'* is an internal sub-process of the organization. Each organization defines how it should be executed. This sub-process in the FTS-SPM represents how it is related to other sub-processes.

In the first diamond, the process can finish if all tasks are finished or can start SP06, if there are unfinished or new tasks to transfer to another site. In the second diamond, a new working day shift starts if the end of the shift is or else, if is the end of the working day, SP05 starts.

Arrows in the FTS-SPM show the sequence flows between sub-processes. An additional arrow is included between SP03 to SP06 indicating the relationship between those sub-processes. The communication settings defined in SP03 are used in SP06.

Sub-processes in the FTS-SPM are developed based on best practices. The model includes twenty-one best practices, as shown in Table 44. Sub-processes in the model are referred as SP01, SP02 up to SP06 while the best practices are identified as BP01, BP02 up to BP21. I renamed best practices described in the previous sections in order to establish a numeral sequence in the FTS-SPM (e.g. BP31 to BP01, BP32 to BP02, BP01 to BP03). Table 44 shows the renamed best practices.

Table 44 – FTS-SPM sub-processes and best practices.

Sub-process (SP)	Sub-process (SP) title	Best practice (BP)	Renamed for the model	Best Practice (BP) title
SP01	Team Setup	BP31	BP01	Fitting teams' working hours for a good overlap
		BP32	BP02	Teams distribution across two or three sites
SP02	Project Planning	BP01	BP03	Use of agile methodologies for project management
		BP02	BP04	Use of incremental software development approaches
		BP04	BP05	Application of FTS for testing and development phases
		BP36	BP06	Similar code patterns
SP03	Communication Protocol	BP07	BP07	Daily exchange of the project status by technologies
		BP10	BP08	Use of screen sharing technology to exchange knowledge
		BP11	BP09	Calendar of handoff sessions should be clearly defined
		BP13	BP10	Use of real time technologies for knowledge sharing
		BP15	BP11	Wikis and online forums to share knowledge between FTS teams
		BP22	BP12	Time window
		BP25	BP13	Corporate technologies for team interaction
		BP26	BP14	Models of e-mails and electronic messages
SP04	Cultural Training	BP33	BP15	Meetings between team members for building trust
		BP35	BP16	Cultural awareness training
SP05	Task Allocation	BP17	BP17	CPro concept
		BP18	BP18	Task distribution by sequence or dependency
SP06	Handoff Sessions	BP20	BP19	Daily stand-up meetings
		BP03	BP20	Daily handoff of 30 minutes duration with each development site
		BP09	BP21	Use of an FTP Server (or data repository) to exchange code and documents

On the basis of these sub-processes and best-practices, the model is structured in three levels. Level 1 in the model denotes the 6 key sub-processes (SP01 to SP06) while level 2 corresponds to the best practices (BP01 to BP21) under each sub-process. Additionally, some of the best practices in the model include ancillary recommendations that form the level 3 within the model. These recommendations are described later in the Section 8.3 under each sub-section which gives detail information about the best practices in the model.

Additionally, in the FTS-SPM each best practice under a particular sub-process includes its significance for FTS development. Therefore, the significance is described for each best practice suggested in the FTS-SPM.

The significance and the specific recommendations for each best practice presented under each sub-process are based on findings from the expert evaluation as discussed in the previous chapter.

8.3 FTS-SPM: The Follow the Sun Software Process Model

The following sub-sections describe each sub-process and each best practice suggested in the model. Furthermore, the description of each best practice demonstrates its significance for FTS development and recommendations that can be followed to implement each best practice. The basis for the ‘Significance’ and ‘Recommendations’ within each best practice is cited while presenting a particular best-practice.

8.3.1 Sub-process: SP01 - Team setup

A team can be defined as a group of people working together to achieve a common goal [DES13]. In GSD projects, teams are distributed in different locations. In FTS development, team members are also distributed in different time zones. The temporal distance that exists between team members is important for FTS development because allows to create a software development cycle of the 24 hours. In FTS development, team members work in their specific hours in the day, but in the same tasks. Thus, at the beginning and end of each working day there is a handoff.

SP01: Team setup aims to identify sites and allocates human resources for the project. The team setup should allow to create a software development cycle between sites. The SP01 - Team Setup includes 2 best practices: BP01: Fitting teams’ working hours for a good overlap and BP02: Teams distribution across two or three sites. The significance and recommendations of each best practice are presented in the next sub-sections.

8.3.1.1 BP01: Fitting teams' working hours for a good overlap

Overlapping hours between sites is important because allows synchronous communication. Team members separated by temporal distance can benefit of overlapping hours between sites to establish communication to another site. Synchronous communication facilitates daily handoffs between two sites and increases the collaboration between team members.

Team members in different development sites follow different working hours. For example, some teams start to work at 8am until 18pm and others at 10am until 20pm. Laws and legislations of each development site and the companies' culture define the teams' working hours.

In some cases, time zone differences do not allow overlapping hours between sites. In FTS development, overlapping hours between development sites is needed mainly to support daily handoffs. Thus, BP01 aims to fit teams' working hours to define a good overlap between two or more sites. Next, I present the significance of this best practice for FTS development as well as the recommendations given by the experts to implement BP01.

Significance

- Allows creating time windows between development sites that is important to establish synchronous communication between two or more sites.
- Enables to manage time zones, establishing different teams' working hours.
- Allows creating overlapping hours between sites that is helpful to increase the collaboration between sites.
- Keeps teams outside of their working hours and reduces the teams' overtime hours.

Recommendations

- Team's working hours should be changed for the minimum time and only to support critical phases of the project.
- Team members must be in agreement on changing their working time hours.

8.3.1.2 BP02: Teams distribution across two or three sites

Increasing the number of sites in FTS projects may result in coordination problems. When more than one site is added to the project, this increases the difficulties to coordinate aspects that involve team management, cultural and geographical differences. The increase of the number of sites also adds difficulties to communication. These difficulties occur due to the increasing of the number of teams allocated to the project and consequently loss of communication richness.

Teams distributed in different development sites have different skills, culture, working infrastructure and overlapping hours between sites. These differences associated with FTS' characteristics may result in failures and over budget projects. Thus, BP02: Teams distribution across two or three sites aims to reduce communication and coordination problems that may occur due to the increasing of the number of sites. BP02 recommends two or three sites to implement FTS projects. FTS development needs at least two sites [CAR11]. Thus, the implementation of this best practice also aims to attend FTS characteristics. The significance this best practice for FTS development and the recommendations given by experts to implement BP02 are presented as follows.

Significance

- Helps in reducing the communication and coordination problems in the project.
- Reduces the coordination complexity of different time zones involved in the same project.
- Helps in better management of cultural differences between teams.

Recommendations

- It is recommended to select team members with good expertise in order to have teams self-managed and productive.
- Screen sharing and live meeting technologies can be adopted to support communication and coordination between FTS teams.
- It is suggested team members from new development sites attend training sessions to better work together in FTS projects.

8.3.2 Sub-process: SP02 - Project planning

Project planning is a formal, approved document used to guide both project execution and control [PMB04]. In the most companies, the project manager is responsible for creating this document. A lot of information related to the project management is described in the project planning such as project goals, budget, information about teams, available tools, systems and technologies, schedule baselines, and others. The project planning is developed as part of any software project. In FTS development, the project planning also addresses particular characteristics related to FTS development (e.g. handoffs' planning).

The project planning in the FTS-SPM is performed following SP01. SP01 identifies available sites and human resources before starting the project planning. However, SP02 interact with SP01 and vice-versa to define the project planning.

The SP02: Project planning for the FTS-SPM is defined as part of the project management. The SP02 - Project planning in the FTS-SPM includes 4 best practices: BP03: Use of agile methodologies for project management, BP04: Use of incremental software development approaches, BP05: Application of FTS for testing and development phases, and BP06: Similar code patterns. The description, significance, and recommendations of each best practice are presented as follows.

8.3.2.1 BP03: Use of agile methodologies for project management

Software projects adopt agile methodologies as part of the project management. Agile methodologies in general share the same philosophy, as well as many of the same characteristics and practices [SMA10]. There are many agile methodologies proposed for project management, such as Scrum, XP, Kanban, and others.

Agile methodologies for FTS development offers many benefits such as help to faster solve problems, keep the focus on requirements requests from the client and reduce the time spend working on documentation. Thus, the implementation of BP03 establishes the use of agile methodologies for project management in FTS development. The significance this best practice for FTS development is presented as follows.

Significance

- Allows having short development intervals, continues releases and integration that is feasible for FTS.
- Keeps the focus on requirements requests.
- Accommodate new requirements from the client.
- Helps to increase the software development speed because has less focus on documentation.

Recommendations

No recommendations are given by the experts for the BP03.

8.3.2.2 BP04: Use of incremental software development approaches

BP04 recommends the use of incremental software development approaches for FTS, in which are defined techniques for code development based on short development iterations. In FTS development, incremental software development approaches contribute to structure the work. Additionally, the use of incremental software development approaches allows team members to take advantage of what was being learned during the development of early (last handoff) to continue the work. The uses of incremental software development approaches allow defining short implementation cycles like daily or weekly, delivery cycles. However, this best practice requires good coordination across differences teams.

The significance and recommendations for BP04 are presented below.

Significance

- Helps team members to better understand the requirements.
- Increases the software quality and at the same time improve the productivity and effectiveness of the team.
- Allows making improvements in the code over the time.
- Helps in continuous integration of tasks.
- Helps to accommodate changing requirements from the client.
- Allows establishing standard practices between sites to code and fixing defects.

Recommendations

- This BP can be followed by team members working on the same code base. It allows splitting the code base in different locations.
- It is necessary to define strategies for effective FTS management.

8.3.2.3 BP05: Application of FTS for testing and development phases

Testing and development phases are the most suitable for FTS development [CAR10]. In these phases it is easier to split tasks in small tasks. Furthermore, other development phases like requirements and design require a lot of dialogue that becomes difficult when team members are separated by temporal distances. Thus, BP05 recommends the application of FTS for testing and development phases.

The significance this best practice for FTS development and recommendations for its implementation are presented as follows.

Significance

- Reduces the difficulties to perform daily handoffs.
- Allows splitting the tasks between development sites.

Recommendations

- Daily handoffs must be supported by specific documents. These documents should describe what has been done to a certain point of the time and the goal for the next stages or for the one that is receiving the work.
- Skillful team members should be allocated to the project in order to promote solid coordination between sites.
- High level of communication and feedback should be performed. Team members must be open to receive those feedbacks on how things are progressing (positive or negative) and what are the project's objectives.
- Handoffs' information should be very clear for all parts (team members), the one that is handing off and the other that is receiving the work.
- Since team members have different skills, it is recommended to implement FTS for only one SDLC phase per time.

8.3.2.4 BP06: Similar code patterns

Team members across different locations have different skills and knowledge. Thus, the work is performed the different ways in different locations. Since FTS development requires team members working on the same tasks, the adoption of similar code patterns can facilitate the work and support the maintenance of the project.

BP06: Similar code patterns in SP02 aims to establish code patterns to be followed by teams to perform the work. This best practice is mainly important for FTS development because allows team members to understand and identify changes made in the code since the last handoff session. Thus, team members have a common goal during the project. BP06 also helps to avoid rework and increase the teams' productivity.

The significance and recommendations for BP06 are presented below.

Significance

- Allows team members to understand the teams' expectations, roles and responsibilities.
- Helps to understand and identify changes made in the code since the last handoff session.
- Defines a common goal to all team members.
- Helps to achieve the customer's objective.
- Helps to avoid rework and increase the teams' productivity.
- Establishes practices and standards to be followed by teams.

- Allows defining a terminology to be followed by team members.
- Support the maintenance of the project.

Recommendations

- Team members must work following the same practices to build the code or standards in the project's documents.
- Code practices must be well defined before the project starts.
- Team members must have a common knowledge about code patterns performed during the project.
- It is needed to define a version control system, editors, and tools for asynchronous communication in order to avoid rework.

8.3.3 Sub-process: SP03 - Communication protocol

Communication is performed to exchange information between two or more people [SHA12]. In FTS development, temporal distance that exists between team members reduces the opportunities for synchronous communication [CAR11]. In these cases where the overlapping hours do not allow synchronous communication, asynchronous communication can be adopted. The SP03: Communication protocol aims to define how the communication will be performed between team members on a FTS project. It includes communication resources, schedule for synchronous and asynchronous communication between teams, tools and technologies.

SP08 includes 8 best practices: BP07: Daily exchange of the project status by technologies, BP08: Use of screen sharing technology to exchange knowledge, BP09: Calendar of handoff sessions should be clearly defined, BP10: Use of real time technologies for knowledge sharing, BP11: Wikis and online forums to share knowledge between FTS teams, BP12: Time window, BP13: Corporate technologies for team interaction, and BP14: Models of e-mails and electronic messages. The description, significance, and recommendations of each best practice are presented as follows.

8.3.3.1 BP07: Daily exchange of the project status by technologies

BP07 define the use of technologies such as telephone, video conference or e-mails for the daily exchange of the project status. These technologies aim to support communication that is performed during daily handoff cycles. Furthermore, these technologies aim to increase the collaboration between team members. Companies can adopt technologies for synchronous or asynchronous communication. The significance and recommendations for BP07 are presented below.

Significance

- Allows establishing synchronous communication between development sites.
- Helps to synchronize tasks, project update, and address features.
- It is helpful to perform short meetings and handoff sessions.

Recommendations

- Synchronous communication during daily handoffs can be supported by telephone, video and conference calls, and teleconference technologies.
- The use of e-mail can be adopted to support asynchronous communication when there is no overlapping time between sites.
- If the objective is to make concepts discussed during handoff even more clearly to the team, then this should be done by asynchronous communication.

8.3.3.2 BP08: Use of screen sharing technology to exchange knowledge

Screen sharing technology is adopted to exchange knowledge between team members. It contributes to transfer knowledge between team members during handoffs. Team members can make using this best practice for technical discussions where the specific details of the code or documents are needed to be discussed. The BP08 adoption in the FTS-SPM aims to improve the communication between teams as well as increase the handoffs ‘efficiency.

The significance this best practice for FTS development and the recommendations for its implementation are presented as follows.

Significance

- Allows establishing synchronous communication between development sites.
- Supports handoffs and the communication between teams geographically distributed.
- Helps to explain pieces of code, test information, and documents.
- Helps to exchange knowledge between team members.
- Provides a visual interface between two or more team members.
- Helps to synchronize and make clear the information which is discussed by two or more team members.

Recommendations

- Team members who are receiving the tasks’ updates during handoffs should summarize the information to the giver team members at the end of each daily handoff session.
- An e-mail must be sent at the end of each daily handoff session with the agreements made between team members.
- Team members who are giving the tasks’ updates should describe what the team has been done, what the team is doing and the next steps that should be done by the next team to complete the tasks.

8.3.3.3 BP09: Calendar of handoff sessions should be clearly defined

Handoff sessions are performed to transfer the project ‘status, unfinished, and new tasks to the site that will start the working day. In the most cases, companies adopt synchronous communication to perform handoffs. The definition of a calendar of handoff sessions helps teams to make themselves available to attend handoff sessions. In this context, BP09 aims to establish a schedule for handoff sessions.

The significance and recommendations for BP09 are presented next.

Significance

- Allows establishing synchronous communication between development sites.
- Makes team members available to attend handoff sessions.
- Defines a schedule for teams to present the status of the work.
- Reinforces the teams' accountability with the deadlines.

Recommendations

- The team members’ agenda must be public to all the team members. It helps to schedule the project meetings.
- Handoff sessions can be also supported by the use of e-mails in case of time zone constraints between sites.
- Team members should exchange an e-mail at the end of each handoff session with the information discussed and all agreements made by the team.
- E-mails exchanged by team members should contain structured information in order to avoid misunderstandings.
- Team members have to learn how to write and understand e-mails in a proper way. It can avoid misunderstandings during the project.

- When a team cannot attend a handoff session, a new task should be taken. If there are no more new tasks, a handoff session should be scheduled with the previous team members.

8.3.3.4 BP10: Use of real time technologies for knowledge sharing

BP10 defines the real time technologies adoption to support knowledge sharing between teams. The use of these technologies improves the communication and collaboration in a software project. There are many situations where the use of real time technologies can be adopted. For example, a small problem can be faster solved with a quick chat between two or more team members.

BP10 aims to support communication during handoff sessions. Team members can use the interface provided by real time technologies to clarify misunderstandings during the communication.

The BP10 significance and recommendations given by the experts are presented as follows.

Significance

- Allows real time communication (e.g. chat).
- Helps team members to self-organize their time to answer a request for communication from another team member.
- Avoid distractions during the working day hours.
- Helps to faster solve problems.

Recommendations

- Tools such as telephone, video conference, Skype calls, Microsoft communicator and live meeting from Microsoft can be used to support real time communication in FTS projects.
- A proper infrastructure should be provided to support real time technologies.

8.3.3.5 BP11: Wikis and online forums to share knowledge between FTS teams

Wikis and online forums provide an informal knowledge in a structured format. Software companies can take benefits of wikis and online forums to create a knowledge base for the project. Thus, BP11 aims to create an internal wiki and online forums as a knowledge database to share problems and solutions between teams.

The significance and recommendations for BP11 are presented below.

Significance

- Provides an information repository for the project.
- Provides a platform for trainings. (e.g. Cultural and technical trainings).
- Support defect reporting or maintenance.
- Help team members to discuss and solve problems related to the project or even related to cultural differences.
- Provides a social platform, where all team members can collaborate.

Recommendations

- Web pages, CVS repositories and editing tools can be used to support knowledge sharing in FTS projects.
- Training programs must focus on giving training on better use of wikis and online forums for knowledge sharing.

8.3.3.6 BP12: Time window

BP12 is adopted for synchronous communication in FTS projects. This best practice provides a short space of time for short interactions between team members. BP12 is defined according to teams' and organizations' culture. This best practice can also be applied to develop SP06: Handoff sessions.

The significance this best practice for FTS development and the recommendations for its implementation are presented as follows.

Significance

- Allows synchronous communication between sites.
- Allows defining a common timetable for teams' interaction.
- Establishes a quickly relation between teams in order to ask and solve questions.

Recommendations

- Time window should be applied according to the teams' and organizations' culture.
- Team members must make themselves available to increase collaboration and communication.

8.3.3.7 BP13: Corporate technologies for team interaction

Temporal distances between sites that exist in FTS projects create barriers for team interaction. In some software projects, there are no overlapping hours during the teams' working hours. Thus, in some cases team members perform their work from home. BP13 aims to define corporate technologies and resources for the teams attending meetings from home. The adoption of this best practice avoids some software incompatibilities that may occur. Furthermore, the use of same technologies allows teams to share software project artifacts. The significance and recommendations for this best practices are presented as follows.

Significance

- Reduces problems of incompatibility between technologies of different development sites.
- Helps to increase the teams' productivity.

Recommendations

- Such technologies like tools for version control, repositories, forms, online forums and software applications like Team Foundation Server, Link from Microsoft, Clarity, Changepoint and Microsoft communicator can be used as corporate technologies to support team interaction.

8.3.3.8 BP14: Models of e-mails and electronic messages

BP26 recommends the use of models of e-mails and electronic messages to support the communication between team members. Its adoption aims to avoid the lack of information or misunderstandings in e-mails or electronic messages.

Models of e-mails and electronic messages should describe the all information needed or requested by another user. This BP is used to support asynchronous communication.

The significance and recommendations for BP14 are presented below.

Significance

- Helps team members to understand which information should be described in e-mails or electronic messages.
- Provides the necessary information requested by another site or team member.
- Provides the structured information.
- Reduces problems of incomplete information carried out in e-mails or electronic messages.

Recommendations

- Similar templates help people to better structure the information.

- It is suggested to define the priority information required to complete the form (e-mail or electronic message).
- Each e-mail or electronic message should address only one topic (information).

8.3.4 Sub-process: SP04 - Cultural training

FTS projects have team members from different cultures. Since, FTS development explores time zone differences between sites, team members comes from different cultures. The cultural diversity make difficult coordination and communication in FTS projects. Thus, SP04 aims to train people in cultural aspects and provide knowledge of different national culture and religious values of the team. SP04 aims also to build trust between team members.

SP04 includes 2 best practices: BP15: Meetings between team members for building trust and BP16: Cultural awareness training. The significance and recommendations of each best practice are presented next.

8.3.4.1 BP15: Meetings between team members for building trust

BP15 is adopted to establish or re-establish trust between team members. Meetings at the beginning of the project are important for building trust between team members and reduce problems relate to cultural differences. The level of trust may decrease in certain stages of the project. Thus, meetings between team members help to re-establish trust. These meetings should address cultural differences between team members involved in the project.

The significance and recommendations for this best practices are presented as follows.

Significance

- Provides a better understanding about a specific culture or behavior.
- Helps to solve problems related to the cultural differences.
- Increases the level of trust between team members.

Recommendations

- It is suggested to plan cultural training programs to help team members to learn how to better communicate with team members from different cultural backgrounds.

- Video conference and teleconferences are recommended as tools to perform meetings between team members from different cultures.
- It is recommended meetings between team members at the beginning of the project to introduce team members and cultural differences between them.
- It is recommended a meeting between team members at the end of the project to discuss what the team members have learned during that particular project.
- The number of meetings between team members to build trust should be planned according to the type of the project.
- It is suggested for large projects to plan a face-to-face meeting between all leaders in a site before the project starts. At this meeting they will define a common agreement to be followed by all sites.

8.3.4.2 BP16: Cultural awareness training

BP16 is associated to BP15. BP16 implementation also addresses cultural differences that exist in FTS projects. Its main goal is to educate team members on other cultures.

FTS teams are required to develop a cultural awareness to better collaborate and deal with teams' expectations. Emotional and social characteristics of FTS team members can interfere on the teams' performance. The significance this best practice for FTS development and the recommendations given by experts to implement BP16 are presented as follows.

Significance

- Helps teams to learn about other cultures involved in the project.
- Makes team members to respect cultural differences.
- Increases the team's performance.
- Motivate teams to reduce the cultural diversity between them.

Recommendations

- Team members have to learn how to work with a particular team.
- The project manager as a leader has to manage cultural differences between team members.
- Communication during the project must be fair and motivational.
- Individual meetings should be performed with each team member that is not presenting a good performance during the project.

- Online movies, books and documents can be used to support cultural awareness training.
- It is suggested for youngest and oldest teams to give them the opportunity of working together in collaborative tasks. It helps to develop new skills and increasing the trust between team members.
- Sort program trainings of the 3 weeks or 4 weeks should allow team members to travel to other sites to meet the team and discuss issues of the project. In these program trainings, team members will have the opportunity to know one to each other and socialize. It is also helpful for exchange knowledge.

8.3.5 Sub-process: SP05 - Task allocation

A software project has a set of activities or tasks. A task is the smallest unit of work with a well-defined functionality and external interface with other tasks. For example, a task could be developing a software module, writing a technical document, testing a piece of code or any other effort in the process of software development [JAL04].

Tasks in FTS projects are difficult to be managed. In FTS development, tasks are allocated to the team and not to individual team members. Team members work on the same tasks until each task to be completed. Unfinished tasks are transferred to the next site during handoffs. Tasks are developed sequentially, and the team member who takes over the development continues to work on the task regardless of whether there are code dependencies between tasks. Thus, SP05 in the FTS-SPM aims to implement the task allocation. SP05 is performed at the beginning of each working day. The project manager allocates tasks for each working cycle of the 24 hours.

The SP05 – Task allocation includes 2 best practices: BP17: CPro concept and BP18: Task distribution by sequence or dependency. The significance and recommendations of each best practice are presented next.

8.3.5.1 BP17: CPro concept

BP17 is based on a cooperative working model called Composite Persona (CP) [DEN09]. This BP consists of task allocation to CPs and not for individual team members or sites. Tasks are allocated in the vertical way. Its goal is increasing the teams' performance.

The significance and recommendations for this best practices are presented as follows.

Significance

- Allows transferring knowledge between different sites and team members.
- Helps to set up teams based on different skills.

Recommendations

- Tools can be used to support task allocation.
- It is recommended to define a certain level of task granularity to properly assign tasks to team members and define the estimated time to develop a task.
- A work breakdown structure (WBS) should be created to help project managers to start planning the work. It allows project managers elicit what should be done and assign tasks in the right way.
- Time estimates of tasks can be used in the initial stages of the project to better allocate tasks.

8.3.5.3 BP18: Task distribution by sequence or dependency

BP18 recommends performing task distribution by sequence or dependency. In the sequencing or dependency distribution, one task is divided between two or more members who are distributed across different time zones. Thus, BP18 is useful to structure the work. The significance and recommendations for BP18 are presented below.

Significance

- Reduces problems related to communication between the sites.
- Help teams to understand the software functionalities.
- Help to define tasks' priorities at different levels.

Recommendations

- It is recommended to perform task distribution more by sequence than by dependency.

8.3.6 Sub-process: SP06 - Handoff sessions

The FTS main characteristic is the daily handoffs. When team members from a site finishes their regular working hours day, other team members located in an another time zone take the work tasks in order to continue the working day. This process of transition of a task to another site is called handoff.

Daily handoffs cycles create dependencies between development sites. The team that will start a working day shift depends on the project status update and project source from the last production site to start its working day shift [CAR11]. FTS teams work on the same task until its completion.

The SP06 is implemented in the FTS-SPM in order to execute the unfinished task and project's update transition from one site to another. SP06 includes 3 best practices: BP19: Daily stand-up meetings, BP20: Daily handoff of 30 minutes duration with each development site, and BP21: Use of an FTP Server (or data repository) to exchange code and documents. The significance and recommendations of each best practice are presented as follows.

8.3.6.1 BP19: Daily stand-up meetings

BP19 came up from Scrum methodology. Its adoption promotes daily interactions between teams that can be easier adapted to perform handoff sessions. The BP19 main benefit is related to the teams' productivity. Furthermore, BP19 helps to manage tasks and brings the team's motivation to meet the deadlines. The significance and recommendations for this best practice are presented as follows.

Significance

- Motivate teams to increase the productivity.
- Establishes a sequence of steps to transfer the information between team members.
- Provides the project' status update to the project manager and team members.

Recommendations

- It is recommended to document daily stand-up meetings to support team members.
- Daily stand-up meetings should be focused and provide motivation for the participants.

8.3.6.2 BP20: Daily handoff of 30 minutes duration with each development site

BP20 aims to establish daily handoffs of 30 minutes duration with each development site. Since, handoffs occur at the beginning and end of each working day shift, one hour is required to perform handoffs. For example, eight hours working day shift have one hour of handoff meeting. Thirty minutes at beginning of the day and 30 minutes at the end of the day. Thus, a working day has seven productive working hours.

The significance and recommendations given by the experts for BP20 are presented next.

Significance

- Allows transferring the essential information that another site needs to continue the working day.
- Allows discussing activities and making sure that the next team will have a good understanding of the previous work.

Recommendations

- It is suggested to define a template to structure information discussed by teams during daily handoffs.
- Team members should perform daily handoffs using the same tools, communication structure, and technologies. It helps to avoid incompatibilities between technologies.
- In order to keep daily handoff meeting of 30 minutes duration with each development site, team members should discuss only the essential information that is needed to the other site continue the working day.

8.3.6.3 BP21: Use of an FTP Server (or data repository) to exchange code and documents

BP21 provides an interface to exchange code and documents between sites. At the end of each working day, unfinished objects are transferred from one site to another. The site that will start the working day needs these unfinished objects to start the working day. The adoption of BP21 allows to exchange big pieces of code or documents that may be not supported by e-mail or other technologies. Additionally, BP21 provides a common repository to save all artifacts of the project. The significance and recommendations for BP21 are presented below.

Significance

- Allows exchanging big pieces of code and documents between sites.
- Provide a common repository to save all project artifacts.

Recommendations

- Such technologies as SVN repositories can be used for exchange software code and documents in GSD projects.
- It is recommended to use only a data repository instead of multiple resource information. It helps in better coordination.

8.4 FTS-SPM Final Considerations

In Software Engineering the definition of new models, processes, and practices helps to build new theories for the area. Furthermore, models, processes, and practices carry and transfer knowledge obtained from past researches. In the FTS research, since 1999 studies have been reported the lacking of processes and practices for FTS implementation. Thus, the proposed process model may become a reference for futures studies.

The FTS-SPM aims to support the FTS adoption in GSD projects. Companies interested in implementing FTS development can adopt the model..

From the software industry' perspective, the FTS-SPM can be combined with other software practices and sub-processes. All six sub-processes that comprise the FTS-SPM are essential for the FTS implementation. However, software companies can add new sub-processes and best practices in the model.

Resulting of the FTS–SPM evaluation, recommendations were added to the best practices. These recommendations allow to make improvements in best practices as well as sub-processes. The benefits provided by the evaluation approach enhance the probability of success in achieving FTS projects.

The FTS-SPM proposal highlights the need for further research in this area, particularly to define new practices for FTS development. Furthermore, research is needed to investigate the type of configurations that will benefit from FTS.

The FTS-SPM may benefit smaller organizations by the use of best practices and sub-process. Alternatively, it can be more beneficial to larger organizations with adopting the offshore insourcing model.

8.5 Chapter Summary

This chapter presented the FTS-SPM (Follow the Sun Software Process Model). It is the main contribution this thesis.

Twenty-one best practices mapped into six sub-processes comprise the FTS-SPM. Sub-processes are developed based on best practices. On the basis of these sub-processes and best practices, the model is structured in three levels. Level 1 in the model denotes the 6 key sub-processes while level 2 corresponds to the best-practices under each sub-process. Additionally, some of the best practices in the model include ancillary recommendations that form the level 3 within the model. These recommendations aim to support the best practices (BP) implementation.

The proposed process model aims to support FTS implementation on GSD projects. As any process model, new sub-processes and best practices can be integrated in the model. Moreover, recommendations included in each best practice can provide new best practices for the model.

This chapter also presented the final considerations about the FTS-SPM. Next chapter will present the research conclusions, contributions, limitations, and future work.

9. CONCLUSION

This chapter summarizes the research presented in this thesis and presents suggestions and directions for future work. The chapter begins with a review of the research objectives while addressing the research question in Section 9.1. Section 9.2 summarizes the contributions of this thesis. Section 9.3 describes the research limitations of the study. The chapter ends with a number of suggestions for future work in Section 9.4.

9.1 Review of the Research Objectives

The work of developing software has become more complex over the last years. With the GSD emergency, new challenges in software engineering field have been identified. FTS development has been applied in GSD projects to explore temporal differences between sites. However, many companies have abandoned it after some point because of the difficult to put it into practice. The absence of empirical evidence led practitioners to stop adopting FTS, and consequently, stopping the theory to evolve as expected.

FTS is rarely practiced partly due to organizations misunderstanding how and when to apply this configuration. Perhaps a prime reason for the lack of take up is that FTS team members are spread across different time zones which complicates the project management. Despite the added complexity, the speed advantages offered by FTS can be realized if FTS is properly implemented.

The primary research goal this thesis was to develop a software process model for the adoption of Follow the Sun (FTS) development in GSD projects. I meet the primary objective through the development of the FTS-SPM. This is achieved through extensive review of literature, analysis of a case study, internships and scholarships did to renowned research groups. Subsequently, on account of this, the FTS-SPM was evaluated by experts to make it more acceptable to GSD. This model contributes to the GSD research area, providing a better understanding about FTS software development and the definition of a software process model to support FTS implementation.

In order to achieve the primary research goal, four secondary research objectives were defined as described in Section 1.1.

- To further theoretical knowledge about GSD, FTS, around-the-clock, and agile software development.
- To analyse best practices for software development in the GSD contexts.
- To identify best practices for FTS development in GSD environments.

- To identify best practices for FTS and around-the-clock development in the literature.

The literature review presented in Chapter 2 has provided the theoretical knowledge about GSD, FTS, around-the-clock and agile software development. The concepts presented in Chapter 2 contributed to guide further investigation in the FTS research area. Furthermore, these results contributed to define the steps followed in this research.

The second and fourth secondary research objectives were achieved with a SLR in FTS development. This study identified 36 best practices for FTS development. A prior mapping study of the literature in GSD was conducted to identify best practices for FTS development. This study identified practices conducted in GSD and at the same time recommended for FTS development.

The case study presented in Chapter 5 allowed to carry out a software project applying FTS to develop internal software application. Ten lessons learned were identified from this study. This study showed a significant contribution to this research. Thus, the third secondary research objective this thesis was achieved.

This thesis addressed the research question: *How can software be developed using Follow the Sun development in Global Software Development projects?”.* This question was answered by developing the FTS-SPM. The research methodology followed in this thesis (see Chapter 3) contributed to achieve the research objectives and to answer the research question.

9.2 Contributions of this Thesis

The objectives of this thesis are focused on FTS development in the GSD context. The SLR study identified the state-of-art of FTS research and best practices for its implementation. My contribution with the SLR showed gaps in the current research, challenges for FTS implementation, and the importance of a software process model definition for FTS. Furthermore, it was possible to characterize FTS.

A case study conducted at Infosys Technologies contributed to identify lessons learned and to give me a better understanding about FTS practice. From the case study, the results revealed the feasibility of FTS for GSD projects. Furthermore, this study has shown good results using Scrum practices, but there is a need to gain more experience and understanding on when it works well, and how making it work better.

The adoption of an expert panel to evaluate the model identified the usefulness and relevance of each sub-process and best practice suggested in the FTS-SPM. It contributed to enhance the model and to build new theories for software engineering. Additionally, the design validation contributed to improve the process model design to support FTS characteristics.

Throughout the thesis, I identified best practices and created six key sub-processes for FTS implementation. These sub-processes and best practices comprise the main contribution this thesis that is a software process model for FTS development. This model is called FTS-SPM (Follow the Sun Software Process Model).

This thesis contributes to reinforce the importance of defining models, processes, and best practices for software development. The definition of models, processes, and best practices for software development is relevant for Software Engineering as well as for the software industry.

From the Software Engineering perspective, this thesis contributed to identify best practices and sub-processes to support FTS characteristics and to define new concepts and theories around GSD and FTS. From the software industry perspective, this thesis contributes to reduce the gap between FTS theory and practice, provide new information about FTS characteristics and its benefits, and provide a software process model for FTS adoption in GSD projects.

Do not limited to these contributions, concepts and theories discussed in this thesis are relevant for future research in Software Engineering. In the Software Engineering field, FTS was acknowledged more than a decade ago. However, the gap between theory and practice make its implementations in the software industry difficult.

9.3 Research Limitations

The main threats to the validity of this research are related to the methodological process. Selection and application of the research methods are conducted by the researcher, who obtains an understanding about recommendations from the literature and define strategies to conduct the research itself. To ensure the rigor of the methodological process, I adopted guides and recommendations from different authors. In all planning and execution method, another researcher reviewed the process. I defined a formal protocol to guide the execution of each research method aiming to reduce these limitations. I also submitted the research proposal to the doctoral symposium of the 7th IEEE International Conference on Global Software Engineering (ICGSE). It gave me the opportunity to discuss the research objectives, methods, and results at early stages of the research.

I performed only one case study in this research. However, the company case is the third-largest India-based IT services company with global software development centers distributed around the world. The company case is a global leader in consulting, technology and outsourcing solutions.

I adopted the SLR method to get a better understanding of GSD and FTS research area. I also adopted the SLR method to identify best practices for FTS implementation. As any SLR, threats to the validity of the process such as, study selection, inaccuracy in data extraction, incorrect classification of studies, research methods and types and potential author bias must be considered. To reduce author's bias, at least two researchers reviewed the interpretation given to the collected data. In case of disagreement between reviewers and the main researcher (the author), disagreements were reviewed until consensus is achieved.

I also adopted an expert panel as evaluation approach. During the interviews, some participants may restrain or be concerned about confidentiality. There is also a possibility that the results can be influenced by personal reasons creating an inaccurate view or making the output biased. To reduce inaccuracy in data extraction from experts, I created a summary of questions by experts (Appendix D), in which each question was answered three times by different participants.

The researcher introduces another threat. In order to minimize this threat, the research proposal and its preliminary results were presented in workshops and submitted to renowned conferences. The proposal was presented at the Annual NUIG-UL Research Day (as a poster) and Lero workshops. At the Annual NUIG-UL (National University of Ireland, Galway / University of Limerick) Research Day held in Galway (Ireland). At the Lero workshops, I had opportunities to discuss the proposal with other researchers from Lero and visiting researchers. Initial results were submitted to the ICGSE (International Conference on Global Software Engineering), SEKE (International Conference on Software Engineering & Knowledge Engineering), HICSS (The Hawaii International Conference on System Sciences), ICEIS (International Conference on Enterprise Information Systems), AMCIS (Americas Conference on Information Systems), and WDDS (Workshop de Desenvolvimento Distribuído de Software).

Finally, the proposed FTS-SPM is based on theory and empirically informed. Data obtained from theory can give us equivocate interpretations and conceive possible incorrect data for this research. Many studies lack information about the data extraction process. In relation the empirical research, all research methods have their own peculiarities that can interfere directly or indirectly with obtained results.

It is important to highlight the knowledge obtained from research visits, internships and conferences. This knowledge contributed to build and improve the research design, getting a better understanding about obtained results and gaps in the research, and propose a software process model for FTS. Additionally, this knowledge contributed to support decisions made and decisions about the steps of the research.

9.4 Future Work

This thesis presents six contributions to the literature, which have been summarized in Section 9.2. Based on these research results, I suggest a number of directions for future work. Since this thesis addresses the definition of a software process model for FTS adoption in GSD projects, there are several topics that have emerged from this research work which need to be examined further.

The definition of a software process model is important to create new theories in the software engineering research area. However, more research is needed to increase its applicability in the software industry. The partnership with other research groups will help to fill this gap between theory and practice. Currently, there is a research project running with Siemens Company (USA) and researchers from PUCRS (Brazil), UFRJ (Federal University of Rio de Janeiro - Brazil), Lero (Ireland), Aalto University (Finland), and SSERG (The software system engineering research group - Finland). This research project aims to develop a framework for optimization of GSD, which involves developing survivability models for GSD. FTS has being investigated in this research project because its characteristic poses extra challenges and difficulties in terms of rigorous process, and higher risk due to high dependencies among sites. Furthermore, FTS requires significant process discipline to achieve the desired speed-up.

Improvements in sub-processes and best practices can be made with future research. Recommendations added to best practices (Chapter 8) will help to increase its efficiency and applicability for FTS development. Thus, new best practices could be proposed and added to the model. Since, the FTS-SPM presents key elements for FTS implementation, new elements could also be identified in future research.

The FTS-SPM evaluation highlighted the usefulness and relevance of each best practice and sub-process suggested in the model. Resulting of the evaluation approach, amendments were made to the model. It contributed to improve the model. The next step would be to apply the FTS-SPM to develop a GSD project in a global software company. The definition of a new case study to evaluate the FTS-SPM does not make part of this research scope. However, it is observed it importance to propose improvements in the model and increase its applicability in the software industry.

Not all best practices identified in Chapter 4 were added to the model. I included best practices in sub-processes based on literature and lessons learned, as describe Chapter 6. Thus, the evaluation approach was applied to 25 best practices from 36 best practices. As part of the future research it will be important to evaluate these best practices to identify its significance for FTS development.

During the evaluation process, the experts have raised their concern about technologies and tools to support the FTS-SPM adoption. The future plan is to develop tools for the management and support of sub-processes and best practices suggested in the FTS-SPM. These tools can also demonstrate the relation of various best practices and how they are beneficial to one another.

REFERENCES

[ABR02] Abrahamsson, P.; Salo, O.; Ronkainen, J.; Warsta, J. “Agile software development methods - Review and analysis”. VTT Publications 478, VTT Electronics, 2002.

[ABR04] Abran A.; Moore J. W.; Bourque, P.; Dupuis, R. “Guide to the Software Engineering Body of Knowledge (SWEBOK®)”. IEEE Computer Society, 2004.

[ALT06] Altintas, N. I.; Cetin, S.; Dogru, A. “Industrializing Software Development: The ‘Factory Automation’ Way”. In TEAA-2006, The 2nd Trends in Enterprise Application Architecture Conference, LNCS, Berlin - Germany, 2006.

[ARA10] Aranda, G. N.; Vizcaíno, A.; Piattini, M “Analyzing and Evaluating the Main Factors that Challenge Global Software Development”, The Open Software Engineering Journal, vol. 4, pp. 14-25, 2010.

[AUD07] Audy, J. L. N.; Prikladnicki, R. “Desenvolvimento Distribuído de Software: Desenvolvimento de Software com Equipes Distribuídas”. Série Campus - SBC, Editora Campus-Elsevier, 2007. *(in Portuguese)*

[AVR07] Avram, G. “Knowledge Work Practices in Global Software Development”. Electronic Journal of Knowledge Management Volume 5 Issue 4, 2007, pp. 347 - 356.

[AVR14] Avritzer, A.; Beecham, S.; Kroll, J.; Menasché, D. S.; Noll, J.; Paasivaara, M. “Survivability Models for Global Software Engineering”, 9th International Conference on Global Software Engineering (ICGSE), Shanghai, China, 2014.

[BAS09] Bass, M; Herbsleb, J. D.; Lescher, C. “A Coordination Risk Analysis Method for Multi-Site Projects: Experience Report”, ICGSE '09: Proceedings of the IEEE International Conference on Global Software Engineering (ICGSE'09), 2009.

[BAT01] Battin, R.; Crocker, R.; Kreidler, J.; Subramanian, K. “Leveraging resources in global software development”. IEEE *Software*, 18(2), 2001, pp. 70-77.

[BEC04] K. Beck. “Programação extrema explicada: acolha as mudanças”. Porto Alegre: Bookman, 2004. *(in Portuguese)*

[BEE03] Beecham, S.; Hall, T. “Expert panel questionnaire: validating a requirements process improvement model”, 2003. Retrieved from: http://uhra.herts.ac.uk/bitstream/handle/2299/384/101470.pdf;jsessionid=7A1AF0656E4654EECEADC2B0C43A3292?sequence. Visited June 2013.

[BEE05] Beecham, S.; Hall, T.; Britton, C.; Cottee, M.; Rainer, A. “Using an expert panel to validate a requirements process improvement model”. Journal of Systems and Software, Volume 76, Issue 3, 2005, pp. 251-275.

[BEE13] Beecham, S.; O'Leary, P.; Baker, S. Richardson, I.; Noll, J. “Who are we doing Global Software Development research for?” In: 8th IEEE International Conference on Global Software Engineering (ICGSE '13), Bari, Italy; 2013.

[BEL06] Bellur, U. “An academic perspective on globalization in the software industry’, Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC’06), 17-21 September, Chicago, Illinois, pp.53–54, 2006.

[BIO05] Biolchini, J.; Gomes, P.; Cruz, A.; Travassos, G. “Systematic review in software engineering”, Rio de Janeiro, Brazil: Systems Engineering and Computer Science Department, UFRJ, 2005.

[BRU06] Bruegge, B.; Dutoit, A.; Wolf, T. “Sysiphus: Enabling informal collaboration in global software development”. In the proceedings of the International Conference on Global Software Engineering (ICGSE), 2006.

[CAM04] Cameron, A. “A Novel Approach to Distributed Concurrent Software Development using a Follow-the- Sun Technique”. Unpublished EDS working paper, 2004.

[CAR05] E. Carmel; P. Tjia. “Offshoring Information Technology: Sourcing and Outsourcing to a Global Workforce”. UK: Cambridge, 2005.

[CAR06] Carmel, E. “Building your Information Systems from the Other Side of the World: How Infosys manages time differences”. Management Information Systems Quarterly - Executive, 2006.

[CAR09] Carmel, E.; Dubinsky, Y.; Johnston, J. “The Impacts of Global IS Sourcing on Engineering, Technology and Innovation Management”. Proceedings of the Third Global Sourcing Workshop, 2009, pp.1-9.

[CAR10] Carmel, E.; Espinosa, A.; Dubinsky, Y. “Follow the Sun Workflow in Global Software Development”. Journal of Management Information Systems Vol. 27, No. 1, 2010, pp. 17 - 38.

[CAR11] Carmel, E.; Espinosa, A. “I'm Working While They're Sleeping: Time Zone Separation Challenges and Solutions”. Kindle Edition, 2011, 188p.

[CAR99] E. Carmel. “Global Software Teams: collaborating across borders and time zones”. Published by Prentice Hall-PTR, 1999.

[CAS06] Casey, V.; Richardson, I. “Project management within virtual software teams”, In Proceedings of the IEEE International Conference on Global Software Engineering (ICGSE ’06), Florianopolis, Brazil, October 2006, pp. 33-42.

[CAS08] Casey, V.; Deshpande, S.; Richardson, I. “Outsourcing Software Development. The Remote Project Manager’s Perspective”. The 2nd Global Sourcing Workshop - Services, Knowledge and Innovation, Val D’Isere, France, 2008.

[CAS09] Casey, V.; Richardson, I. “Implementation of Global Software Development: a structured approach”. Software Process 14, 5 (September 2009), 2009, 247-262.

[CAT07] Cataldo, M., Bass, M., Herbsleb, J., and Bass, L. “On Coordination Mechanism in Global Software Development,” In Proceedings of the International Conference on Global Software Engineering (ICGSE’07), Munich, Germany, 2007.

[CAT10] Cataldo, M.; Nambiar, S. “The Impact of Geographic Distribution and the Nature of Technical Coupling on the Quality of Global Software Development Projects”. Forthcoming in Journal of Software Maintenance and Evolution: Research and Practice, 2010.

[CDE09] Carmel, E.; Dubinsky, Y.; Espinosa, J. A. “Follow the Sun Software Development: New Perspectives, Conceptual Foundation, and Exploratory Field Study”. In: Proceedings of the 42nd Hawaii International Conference on System Sciences, 2009.

[COH03] Cohen, D.; Lindvall, M.; Costa, P. “Agile Software Development A DACS State-of-the-Art Report”. Fraunhofer Center for Experimental Software Engineering Maryland and The University of Maryland, 2003.

[COL10] Colazo, J.; Fang, Y. L. “Following the Sun: Temporal Dispersion and Performance in Open Source Software Project Teams”, Journal of the Association for Information Systems, In Press., 2010.

[COL14] Colazo, J., “Structural Changes Associated with the Temporal Dispersion of Teams: Evidence from Open Source Software Projects”, System Sciences (HICSS), 2014 47th Hawaii International Conference on, pp.300 - 309, 6-9 Jan. 2014.

[CZE11] Czekster, R.; Fernandes, P.; Prikladnicki, R.; Sales, A.; Santos, A.; Webber, T. “Follow-The-Sun Methodology in a Stochastic Modeling Perspective,” in 6th IEEE International Conference on Global Software Engineering (ICGSE): Methods and Tools for Project/Architecture/Risk Management in Globally Distributed Software Development Projects (PARIS), Helsinki, Finland, August 2011, pp. 54-59.

[DAM06] Damian, D.; Moitra, D. “Global Software Development: How Far Have We Come?”, IEEE Software, v.23 n.5, pp.17-19, 2006.

[DEI11] Deiters, C.; Herrmann, C.; Hildebrandt, R.; Knauss, E.; Kuhrmann, M.; Rausch, A., Rumpe, B.; Schneider, K. “GloSE-Lab: Teaching Global Software Engineering”. Global Software Engineering (ICGSD), 6th IEEE International Conference on, 2011, pp.156-160.

[DEN08] Denny, N.; Mani, S.; Nadella, R. S.; Swaminathan, M.; Samdal, J. “Hybrid Offshoring: Composite Personae and Evolving Collaboration Technologies”. IRMJ 21(1), 2008, pp. 89-104.

[DEN09] Denny, N.; Crk, I.; Seshu, R.; Gupta, A. “Agile Software Processes for the 24-Hour Knowledge Factory Environment”. Journal of Information Technology Research, Vol. 1, Issue 1, 2009, pp. 57-71.

[DES09] Deshpande, S.; Richardson, I. “Management at the Outsourcing Destination - Global Software Development in India”. In: Proceedings of the 2009 Fourth IEEE International Conference on Global Software Engineering (ICGSE '09). IEEE Computer Society, Washington, DC, USA, 2009, pp. 217-225.

[DES10] Deshpande, S.; Richardson, I.; Casey, V.; Beehcam, S. “Culture in Global Software development - a Weakness or Strength?”. In: 5th IEEE International Conference on Global Software Engineering (ICGSD), Princeton, USA, 2010.

[DES13] Deshpande, S. “A Coordination Model for Global Software Development Teams”. PhD thesis, Lero (The Irish Software Engineering Research Centre), University of Limerick (UL), Republic of Ireland, 2013. Available at http://www.lero.ie/phdthesis.

[DIL02] Di Lucca, G.A.; Di Penta, M.; Gradara, S. “An approach to classify software maintenance requests”. Proceedings. International Conference on Software Maintenance, 2002, pp.93-102.

[DIN14] Dingsoyr, T.; Smite, D. “Managing Knowledge in Global Software Development Projects”, IT Professional, vol.16, no.1, pp.22-29, 2014.

[DUB08] Dubinsky, Y.; Carmel, E. “Follow the Sun in an Agile Environment: A First Quasi-Experiment. Development”. (n.d.), 2008.

[ENC07] Espinosa, J. A.; Nan, N.; Carmel, E. “Do Gradations of Time Zone Separation Make a Difference in Performance? A First Laboratory Study”. International Conference on Global Software Engineering (ICGSE 2007), 2007, pp.12-22.

[END03] Endres, A.; Rombach, D. A. “Handbook of Software and Systems Engineering: Empirical Observations, Law and Theories”. Addison Wesley, 2003.

[ESP03] Espinosa, J. A.; Carmel, E. “The impact of time separation on coordination in global software teams: a conceptual foundation”. Software Process: Improvement and Practice, 8(4), 2003, pp.249-266.

[ESP07] Espinosa, J. A.; Nan, N.; Carmel, E. “Do Gradations of Time Zone Separation Make a Difference in Performance? A First Laboratory Study”. International Conference on Global Software Engineering, Munich, Germany, August, 2007, pp. 27-30.

[EAS08] Easterbrook, S.; Singer, J.; Storey, M. A.; Damian, D. “Selecting empirical methods for software engineering research. Guide to advanced empirical software engineering”, Chapter 11 in Shull et al., pp.285–311. Available at: http://www.springerlink.com/index/n815725515063p2m.pdf.

[FAD00] Fadel, G.; Lindemann, U.; Anderl, R. “Multi-National Around the Clock Collaborative Senior Design Project”, Invited paper, Honorable mention at the ASME Curriculum Innovations Award, 2000.

[GM96] Gorton, I.; Motwani, S. “Issues in co-operative software engineering using globally distributed teams”. Information and Software Technology, Volume 38, Issue 10, 1996, pp. 647-655.

[GOR96] Gorton, I.; Hawryszkiewycz, I.; Fung, L. “Enabling software shift work with groupware: a case study”. System Sciences, Proceedings of the Twenty-Ninth Hawaii International Conference on, vol.3, 1996, pp. 72-81.

[GOR97] Gorton, I.; Hawryszkiewycz, I.; Ragoonaden, K.”Collaborative tools and processes to support software engineering shift work”. BT Technology Journal 15, 1997, pp. 189-198.

[GUP07] Gupta, A.; Jalote, P. “An Experimental Evaluation of the Effectiveness and Efficiency of the Test Driven Development”. Empirical Software Engineering and Measurement (ESEM), 2007, pp.285-294.

[GUP08] Gupta, A.; Bondade, R.; Denny, N. “Software Development Using the 24-Hour Knowledge Factory Paradigm”, April, 2008. Available at: http://ssrn.com/abstract=1130062.

[GUP09] Gupta, A.; Mattarelli E.; Seshasai, S.; Broschak, J. “Use of collaborative technologies and knowledge sharing in co-located and distributed teams: Towards the 24-h knowledge factory”. The Journal of Strategic Information Systems, Volume 18, 2009, pp.147-161.

[GUP11] Gupta, A.; Crk, I.; Bondade, R. “Leveraging temporal and spatial separations with the 24-hour knowledge factory paradigm,” Information Systems Frontiers 13, 3 (July 2011), 397-405, 2011.

[GUP12] Gupta, A.; Hu, L.; Hedberg, T.; Prendergast, C.; Crk, I. “Creating the 24-Hour Knowledge Factory”. February 13, 2012. Available at: http://ssrn.com/abstract=2004791.

[GUS07] Gupta, A.; Seshasai, S. “24-hour knowledge factory: Using Internet technology to leverage spatial and temporal separations”, ACM Transactions on Internet Technology (TOIT), v.7 n.3, p.14, 2007.

[HEL11] Helen, M.; Nahar, N. “Key barriers of globally distributed software products development”. Proceedings of Technology Management in the Energy Smart World (PICMET), 2011, pp. 1-13.

[HER00] Herbsleb, J. D.; Mockus,A.; Finholt, T. A.; Grinter, R. E. “Distance, dependencies, and delay in a global collaboration”, in Proceedings of the ACM conference on Computer supported cooperative work (CSCW '00), New York, NY, USA, pp. 319-328, 2000.

[HER01] Herbsleb, J. D.; Mockus, A.; Finholt, T. A.; Grinter, R. E. “An Empirical Study of Global Software Development: Distance and Speed”. Proceedings of the 23rd International Conference on Software Engineering, 2001, pp. 81-91.

[HER07] Herbsleb, J. D. “Global Software Engineering: The Future of Socio-technical Coordination”. II IEEE International Conference on Global Software Engineering (ICGSD), 2007.

[HES12] Hess, E. R.; Audy, J. L. N. “FTSProc: A Process to Alleviate the Challenges of Projects that Use the Follow the Sun Strategy”. IEEE Seventh International Conference on Global Software Engineering (ICGSE), 2012, pp. 56-64.

[HOL06] Holmstrom, H.; O’Conchuir, E.; Agerfalk, P. J.; Fitzgerald, B. “Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance”. Proceedings of the IEEE international conference on Global Software Engineering (ICGSD '06). IEEE Computer Society, Washington, DC, USA, 2006, pp.3-11.

[HUM87] W. S. Humphrey, “Managing the Software Process”. Reading, MA: Addison-Wesley, 1987.

[JAB10] Jabangwe, R.; Nurdiani. I. “Global Software Development Challenges and Mitigation Strategies: A Systematic Review and Survey Results”. Master´s program in Software Engineering, Blekinge Institute of Technology, OM/School of Computing, 2010.

[JAL04] Jalote, P.; Jain, G. “Assigning tasks in a 24-hour software development model”. 11th Asia-Pacific Software Engineering Conference, 2004, pp. 309- 315.

[JAL09] Jalali, S.; Zlatkovic, B. “Success Factors in Building and Maintaining Trust among Globally Distributed Team Members”. Master Thesis, School of Engineering Blekinge Institute of Technology, 2009.

[JAL10] Jalali, S.; Wohlin, C. “Agile Practices in Global Software Engineering: A Systematic Map”. International Conference on Global Software Engineering (ICGSE), Princetown, USA, 2010, pp. 45-54.

[JAN12] Jangir, S. K.; Gupta, N.; Agrawal, S. “Evolution and Melioration of Software Management Processes”. International Journal of Software Engineering & Applications (IJSEA), Vol.3, No.1, 2012.

[JAV02] Javelin Technologies, “Best Practice Definition”. Oakville, Ontario, Canada, 2002.

[JIM09] Jimenez, M.; Piattini, M.; Vizcaino, A. “Challenges and improvements in distributed software development: A systematic review”. Advances in Software Engineering, 2009.

[KIR01] Kircher, M.; Jain, P.; Corsaro, A.; Levine, D. “Distributed eXtreme Programming”. In Proceedings of XP Conference, Cagliari, Villasimius, Sardinia, Italy, 2001.

[KIT07] Kitchenham, B.; Charters, S. Guidelines for performing systematic literature reviews in software engineering (version 2.3). Technical report, Keele University and University of Durham, 2007.

[KOR14] Korkala, M.; Maurer, F. “Waste identification as the means for improving communication in globally distributed agile software development”, Journal of Systems and Software, 2014.

[KRO11] Kroll, J.; Audy, J. L. N.; Prikladnicki, R. “Mapping the Evolution of Research on Global Software Engineering: A Systematic Literature Review”. In: 13th International Conference on Enterprise Information Systems (ICEIS), Beijing, Chine, 2011.

[KRO12] Kroll, J.; Audy, J. L. N. “Mapping Global Software Development Practices for Follow the Sun Process”. IEEE Seventh International Conference on Global Software Engineering, 2012, pp. 164-168.

[LAM09] Lamersdorf, A; Munch, J., “TAMRI: A Tool for Supporting Task Distribution in Global Software Development Projects,” Fourth IEEE International Conference on Global Software Engineering (ICGSE), pp.322-327, 2009.

[LAN08] Lane, M.; Ågerfalk P. “Suitability of Particular Software Development Roles to Global Software Development”. 3rd IEEE International Conference on Global Software Engineering, 2008.

[LAN11] Lane, S.; Richardson, I. “Process models for service-based applications : A systematic literature review”. Information and Software Technology, 2011, pp. 424-439.

[LAY06] Layman, L.; Williams, L.; Damian, D.; Bures, H. “Essential communication practices for extreme programming in a global software development team”. Information and Software Technology, 2006, pp. 781-794.

[LIN07] Lings, B.; Lundell B.; Ågerfalk P. J.; Fitzgerald B. “A reference model for successful Distributed Development of Software Systems”. IEEE International Conference on Global Software Development (ICGSE), 2007, pp. 130-139.

[LUC02] Lucca, G. A.; Di Penta, M. Di ; Gradara, S., An Approach to Classify Software Maintenance Requests. Conference On Software Maintenance, 2002.

[MAT09] Mattarelli, E.; Gupta, A., “Offshore-onsite subgroup dynamics in globally distributed teams”. Information Technology & People, Vol. 22 Issue 3, 2009, pp. 242-269.

[MCL10] McLoughlin, F., “The Rosetta Stone Methodology - A Benefits Driven Approach to Software Process Improvement”. PhD thesis, Lero (The Irish Software Engineering Research Centre), University of Limerick (UL), Republic of Ireland, 2010. Available at http://www.lero.ie/phdthesis.

[MIS11] J. Mishra; A. Mohanty. “Software Engineering”. Pearson Education, New Delhi, India: Dorling Kindersley, 2011.

[NII11] Niinimaki, T. “Face-to-Face, E-mail and Instant Messaging in Distributed Agile Software Development Project”. Sixth IEEE International Conference on Global Software Engineering Workshop (ICGSDW), 2011, pp.78-84.

[NUE11] Nuevo, E. D.; Piattini, M.; Pino, F. J. “Scrum-based Methodology for Distributed Software Development”. Proceedings of the 2011 IEEE Sixth International Conference on Global Software Engineering (ICGSE '11). IEEE Computer Society, Washington DC, USA, 2011, pp. 66-74.

[OCO06] O’Conchúir, E.; Holmström, H.; Ågerfalk, P. J.; Fitzgerald, B. “Global Software Development: Never Mind the Problems - Are There Really Any Benefits?”. Proc. 29th Information Systems Research Seminar in Scandinavia, 2006.

[OCO09] O´Conchúir, E.; Holmstrom O., H.; Ågerfalk, P.; Fitzgerald, B. “Benefits of Global Software Development: Exploring the Unexplored”. Software Process: Improvement and Practice. Vol. 14, Issue 4, 2009, pp. 199-241.

[PAA08] Paasivaara, M.; Durasiewicz, S.; Lassenius, C. “Distributed Agile Development: Using Scrum in a Large Project”, IEEE International Conference on Global Software Engineering (ICGSD), 2008, pp.87-95.

[PAS10] Prikladnicki, R.; Audy, J. L. N.; Shull, F. “Patterns in Effective Distributed Software Development”. IEEE Software, v. 27, 2010, pp. 12-15.

[PFL04] S. L. Pfleeger. “Engenharia de Software: teoria e prática”.*(in portuguese)* 2ed., São Paulo - Prentice Hall, 2004.

[PMB04] PMBOK. “A Guide to the Project Management Body of Knowledge (PMBOK® Guide)”. 3ed, Project Management Institute Publications, USA, 2004.

[POR07] Porter, A.; Yilmaz, C.; Memon, M.; Schmidt, C.; Natarajan, B. “Skoll: A Process and Infrastructure for Distributed Continuous Quality Assurance”. IEEE Transactions on Software Engineering, 2007, pp. 510-525.

[PRE10] R. S. Pressman. “Software Engineering: a practitioner’s approach”. New York: McGraw Hill, 7th Edition, 2010, 928p.

[PRI08] Prikladnicki, R.; Damian, D.; Audy, J. “Patterns of Evolution in the Practice of Distributed Software Development in Wholly Owned Subsidiaries: A Preliminary Capability Model”. IEEE International Conference on Global Software Engineering (ICGSD), 2008, pp.99-108.

[PRI10] Prikladnicki, R.; Audy, J. L. N. “Process Models in the Practice of Distributed Software Development: A Systematic Review of the Literature”. Information and Software Technology, v. 1, 2010, pp.1-1.

[PRI13] Prikladnicki, R.; Carmel, E. “Is time-zone proximity an advantage for software development? the case of the Brazilian IT industry”. Proceedings of the 2013 International Conference on Software Engineering (ICSE '13). IEEE Press, Piscataway, NJ, USA, 2013, pp.973-981.

[RAM02] Ramesh, V.; Dennis, A. “The object oriented team: Lessons for virtual teams from global software development”. In: HICSS ’02: Proceedings of the 35th Annual Hawaii International Conference on System Sciences, volume 1. IEEE Computer Society, 2002.

[RCS14] Colomo-Palacios, R; Casado-Lumbreras, C.; Soto-Acosta, P.; Peñalvo, F. J. G.; Tovar, E. “Project managers in global software development teams: a study of the effects on productivity and performance”. Software Quality Journal 22, 2014, pp. 3-19.

[RIC10] Richardson, I.; Casey, V.; Burton, J.; McCaffrey, F. “Global Software Engineering: A Software Process Approach”. Collaborative Software Engineering, Editors I. Mistrík, J. Grundy, A. Hoek, J. Whitehead, Publisher Springer, London,2010, pp. 35-56.

[ROB04] M. Robinson; R. Kalakota. “Offshore outsourcing: Business models, ROI and best practices”. Mivar Press, Inc.; 2nd edition, 2004, 336p.

[ROB08] Robarts, J. M. “Practical Considerations for Distributed Agile Projects”, Conference AGILE '08. 2008, pp.327-332.

[RUA14] Ruano-Mayoral, M.; Casado-Lumbreras, C.; Garbarino-Alberti, H.; Misra, S. “Methodological framework for the allocation of work packages in global software development”, Journal Software: Evolution and Process, 26: pp. 476–487, 2014.

[RUN12] Runeson, P.; Host, M; Rainer, A.; Regnell, B. “Case Study Research in Software Engineering: Guidelines and Examples”, Wiley Publishing, 2012.

[SAI05] Saiedian, H.; Weide, B. W. “The New Context for Software Engineering Education and Training”. Journal of Systems and Software 74, (15 January), guest editors' introduction to special issue of journal, 2005, pp.109-111.

[SAM06] R. H. Sampieri. “Metodologia de Pesquisa”. McGraw-Hill, 583 p. 3ª Ed., 2006. (*In portuguese*).

[SAM91] Sampieri, R. H.; Collado, C. F.; Lucio, P. B. “Metodologia de la Investigación”. México: McGraw Hill, 1991, 200p.

[SAN07] R. Sangwan; M. Bass, N. Mullick, D. Paulish; J. Kazmeier. “Global Software Development Handbook”. Auerbach Publications, 2007.

[SAN12] Santos, A.R.; Sales, A.; Fernandes, P. “Setting Up a Stochastic Model for Teams Working in a Follow-the-Sun Environment,’ IEEE Seventh International Conference on Global Software Engineering (ICGSE), pp. 179, 2012.

[SDJ07] Sjoberg, D. I. K.; Dyba, T.; Jorgensen, M. “The Future of Empirical Methods in Software Engineering Research”, in 2007 Future of Software Engineering (FOSE '07). IEEE Computer Society, Washington, DC, USA, pp. 358-378, 2007.

[SET07] Setamanit, S.; Wakeland, W.; Raffo, D. “Improving Global Software Development Project Performance Using Simulation”. Management of Engineering and Technology, Portland International Center, pp. 2458-2466, 2007.

[SHA12] Shah, Y. H.; Raza, M.; UlHaq, S. “Communication Issues in GSD”. IJAST, vol. 40, 2012.

[SHE01] Shepperd, M.; Cartwright, M. “Predicting with Sparse Data”. IEEE Transactions on Software Engineering, vol. 27, no. 11, 2001, pp.987-998.

[SJO03] Sjøberg, D. I. K.; Anda, B.; Arisholm, E.; Dybå, T.; Jørgensen, M.; Karahasanovic, A.; Vokác, M &ldquo. “Challenges and Recommendations when Increasing the Realism of Controlled Software Engineering Experiments, Empirical Methods and Studies in Software Engineering”. Lecture Notes in Computer Science Volume 2765, 2003, pp 24-38.

[SLO05] Slocum, N. “Participatory methods toolkit: A practitioner's manual. [Participatory manual]”, 2005. Retrieved from: http://archive.unu.edu/hq/library/Collection/PDF_files/CRIS/PMT.pdf

[SMA10] D. Šmite; N. B. Moe; P. J. Ågerfalk. “Implementing Agile methods in Global Software Projects”. 1st Edition, 2010, XXXVI, 341 p.37.

[SMI10] Smite, D.; Wohlin, C.; Gorschek, T.; Feldt, R. “Empirical evidence in global software engineering: a systematic review”. Empirical Softw. Eng., 2010, pp. 91-118.

[SOL10] Solingen, V. R.; Valkema, M. “The Impact of Number of Sites in a Follow the Sun Setting on the Actual and Perceived Working Speed and Accuracy: A Controlled Experiment”. Global Software Engineering (ICGSE), 5th IEEE International Conference, 2010, pp. 165-174.

[SOM11] I. Sommerville, I., Software Engineering, Ninth edition, Addison-Wesley, 2011, 792p.

[SOO08] Sooraj, P.; Mohapatra, P. K. J. “Modeling the 24-h software development process”. Strategic Outsourcing: An International Journal, 2008, pp. 122-141.

[SUT07] Sutherland, J.; Viktorov, A.; Blount, J.; Puntikov, N. “Distributed Scrum: Agile Project Management with Outsourced Development Teams”. 40th Annual Hawaii International Conference on System Sciences (HICSS), 2007, 274p.

[TAN11] Tang, J. C.; Zhao, C.; Cao, X.; Inkpen, K. “Your time zone or mine?: a study of globally time zone-shifted collaboration”. Proceedings of the ACM 2011 conference on Computer supported cooperative work (CSCW '11). ACM, New York, NY, USA, 2011, pp. 235-244.

[TAW06] Taweel, A; Brereton, P. “Modelling software development across time zones”. Inf. Softw. Technol. 48, 2006, pp. 1-11.

[TRA02] Travassos, G. H.; Gurov, D.; Amaral, G. “Introdução a Engenharia de Software Experimental”. *(in portuguese)* Technical report, PESC 590/02, COPE/UFRJ, 2002, 66 p.

[TRE06] Treinen, J. J.; Miller-Frost, S. L. “Following the Sun: Case Studies in Global Software Development”. IBM Systems Journal, Volume 45, Number 4, 2006.

[TUF10] Tufekci O.; Cetin S.; Arifoglu A. “Proposing a Federated Approach to Global Software Development.” Fourth International Conference on Digital Society. 2010, pp.150-157.

[VAN05] Vanhanen, J.; Lassenius, C. “Effects of pair programming at the development team level: an experiment”, International Symposium on Empirical Software Engineering, 2005, 10p.

[VIS09] Visser, C.; Solingen R. V. “Selecting Locations for Follow the Sun Software Development: Towards a Routing Model”. Fourth IEEE International Conference on Global Software Engineering, 2009.

[WAK08] Wakefield, R. L.; Leidner, D. E.; Garrison, G. “Research Note - A Model of Conflict, Leadership, and Performance in Virtual Teams”. Information Systems Research, 2008, pp. 434-455.

[WIL02] L. Williams; R. Kessler. “Pair Programming Illuminated”. Addison-Wesley, Boston, 2002.

[WIL09] L. Williams. “A (partial) Introduction to Software Engineering Practices and Methods”. 2008-2009 (Fifth) Edition.

[WOH02] Wohlin, C.; Runeson, P.; Höst, M.; Ohlsson, M.; Regnell, B.; Wesslén, A. “Experimentation in software engineering: an introduction”. Boston: Kluwer Academic Publishers, 2002, 204p.

[YAP05] Yap, M. “Follow the sun: distributed extreme programming development”, Agile Conference Proceedings, 2005, pp.218- 224.

[YIN02] Yin, R. K. “Case Study Research, Design and Methods”, 3rd ed. Newbury Park, Sage Publications, 2002.

APPENDIX A - GUIDELINES FOR FTS TEAMS

Three rules must be followed by FTS teams:

1. <u>Rules for the course weekly meeting for FTS</u>
   - Meeting with the entire team can be held when the project manager with the sub-team that is his time zone for this iteration and the other sub-teams is over the phone or video conference.
2. <u>Rules to follow between course meetings</u>

   3.1 <u>Communication</u>
   - A team member cannot talk with teammates from another site after his time office.
   - A team member can communicate by phone or using a communication tool with teammates from the other team. This communication is for all purposes like for example talking about a specific customer story, specific bug, issues with respect the personal role. This type of communication don't have a time limit.
   - Time windows for interaction are available one hour (*maximum*) per day.
   - The first time window for interaction is available in the first 30 minutes (max.) of a working day.
   - The second time window for interaction is available on the last 30 minutes (max.) of a working day.
   - Overlap diary one hour (max.) for exchange tasks and communication.
   - Tester and others have to synch meetings especially at the end of an iteration.
   - The iteration presentation is performed together over phone or communication tool.

   3.2 <u>Development procedure</u>

   a. One sub-team works in their specific hours in the day and another sub-team work works in different hours, but in the same tasks.

   b. The integrators in both sub-teams are responsible to generate reports about who committed changes and when so to keep the rule of work as much as possible.
3. <u>Rules to follow FTS requirements</u>

   <u>4.1 Handoffs</u>
   - One sub-team at the end of a working day, must send the task (still not complete) to the next sub-team, localized in a different local, to continue the task;
   - Sub-teams depend on the handoff to continue the task;
   - At any point in time, only a site has the product;

- There is one deliverable at the end of the each iteration.
- There is a common digital product repository (such as a software configuration management system), which allows all sites to “commit” the code/objects at the end of the workday.
- After every task - there is a commit.
- There should be at least one hour overlapping session between two teams in different time zone.
- Discussion about what is done by the previous team and what needs to carry on by the next team should occur.
- A clear agenda for these sessions should be defined.
- The handoff from one site to the next can occasionally be empty in the case of holidays or emergencies.

APPENDIX B - SOFTWARE APPLICATION

The application is called *M-Buddy program*. M-Buddy program enables seamless connectivity to the *mothers to be* (MTB) employee through a Buddy, an employee who volunteers to help MTB, with valuable information about the projects/work and providing the latest updates of Infosys policies, processes, project status, last news, such as, trainings, certifications and alike. The application be used by the MTB (mother to be) in order to nominate a “buddy” (an Infosys employee) to keep her informed she is on Maternity Leave. The primary objectives of this program are:

1. To keep women going on maternity leave to stay connected to her work place.
2. To minimize her anxiety and make it relevant for her to join back.

Any MTB employee can voluntarily opt for the M-Buddy program when she applies for a maternity leave (ML) or she could register for this program through the auto-generated e-mail having the link to nominate a Buddy. During registration, she should provide the details like employee id & e-mail id of her identified Buddy.

Typically the team will include:

- MTB - Mother-to-be employee
- Buddy - employee who volunteers for M-Buddy program
- MTB’s Manager - Most recent project manager of MTB captured by IS-Systems
- MTB’s BP-HR - Unit BP-HR of MTB captured by IS-Systems
- D&I Team - Infosys team, who would provide relevant updates to MTB fortnightly.

APPENDIX C - SPRINT RETROSPECTIVE

SPRINT ACTIVITY - Software engineering practices, standards & templates

ID Member / Location	What went well?	What did not go well? How could it be made better?
1 - India	N/A	Coding Standards should have been defined before the coding started.
2 - India	It was ok. There were some delays in getting confirmation on Design templates from IS team.	N/A
3 - Australia	IS Coding standards clearly defined in the documents	Templates and standards documents received from IS after coding had commenced. It would have limited the amount of rework if they had been available from the start
4 - Australia	Everyone wants the project to succeed and is working very hard.	Not having the correct document templates and coding standards made everything very difficult and created a large amount of rework. This should not be an issue in the second *sprint*. Many times there was a failure to follow design plans when coding and if changes were made in one place (e.g. requirements document) they were not always made everywhere else (design document) which made it difficult to know what the point of truth was.
5 - Australia	That's good we got templates early in the project	Got standards very late because of that we need to do some rework and that increase our review and fixing time
6 - Mexico	All the guys have good programming skills and help me very much in coding.	N/A
7- Mexico	We all had similar knowledge about procedures and practices that made the job easier.	I think standards must be reviewed by parts, reviewing a whole project may be very difficult.

SPRINT ACTIVITY - Handover template and process

ID Member / Location	What went well?	What did not go well? How could it be made better?
1 - India	Process went off very well.	Handover meeting is taking lot of time. We should either go for handover sheet or Mail. As the content remains the same.
2- India	It is nice way to communicate and handing over the task to next team	N/A
3 - Australia	Handover template made it easy and efficient to document the tasks that were done and the tasks that were not finished that day	N/A
4 - Australia	The template was good. Having twice daily meetings helps explain further information to receiving team members	It's hard to tell how much information is getting across in the meeting. It's also hard to see if the information provided in the template is being taken on board and used to make central decisions or whether if the handover recipient ignores it, it is lost forever.
5 - Australia	Handover is going very well	NA

6 - Mexico	Template helped to have all tasks organized and clear between members.	N/A
7- Mexico	Format helped to have all tasks and advance organized between members.	Only the first days, I could not understand how did it work, but now I am fine.

SPRINT ACTIVITY - Communication flow (call, e-mail, chat)

ID Member / Location	What went well?	What did not go well? How could it be made better?
1 - India	All went well.	But writing daily handover e-mail was painful as he content in handover excel and mail remains same
2- India	We have tried our best to communicate the message to the next team.	The meeting time span could be reduced to better utilize the time.
3 - Australia	Task handovers were well communicated - by call, or e-mail, or both	We could all be more pro-active with notifying the rest of the team by e-mail whenever a decision/change/issue happens that has an impact on the team
4 - Australia	Swati's 24 hour e-mails were very helpful- well done!	It took some time to develop a good communication practice. I think this will be much better in *sprint* 2. I would like to see an high level view of what's happening in the project to help understand the work context.
5 - Australia	First couple of days we face some issue but after that everything is fine	NA
6 - Mexico	The different options of communication (phone, e-mail, communicator, etc) helped to get to understand each other.	Accents and language are still not very understandable for me. I have to work in that.
7- Mexico	The different options of communication (phone, e-mail, communicator, etc) helped to get to understand each other.	Accents and language may be sometimes a wall when making a call, but I think we are going well with that.

SPRINT ACTIVITY - Task allocation

ID Member / Location	What went well?	What did not go well? How could it be made better?
1 - India	Swati distributed the task very well later on. It helped us to know what need to be done in next 24 Hours	Tasks were not allocated properly in the initial stage.
2- India	task is divided in two CPs	N/A
3 - Australia	The daily e-mail that Swati sends with the tasks for the day was extremely useful to see what needed to be done for that day	Initially it was unclear what the highest priority tasks were. This has largely been resolved by the 'tasks for the day' e-mail
4 - Australia		
5 - Australia	next 24 hour task allocation I like	in starting of the project it was a feeling that we are going on a different path. And felt blind about what to do next after handover from Mexican team
6 - Mexico	The variety in the kind of tasks make the work easier	I feel *Sprint* Backlog don't have enough tasks to describe what we really do.

7- Mexico	I think we all had a very good variety of tasks so that help us to get know better the application.	About *Sprint* Back Log allocations of the tasks, I am still not very should how to categorize a task in the *sprint* back log.

SPRINT ACTIVITY - CP experience (peer interaction, work sharing, responsibility, visibility, time zone management)

ID Member / Location	What went well?	What did not go well? How could it be made better?
1 - India	Peer interaction - Good Time zone management - Good	Need to work on - Work sharing and taking responsibility
2- India	good way of managing team at different location having different time zone	There were some problem in communication initially but as we moved forward with the development, everything seems to be in sync now.
3 - Australia	Everyone was consistently attending and participating in the handover meetings	It was not always easy to share ownership of a document or code. This was due to varying levels of experience, different line of thinking and understanding
4 - Australia	A lot of work is being done and everyone takes responsibility for their work and the project success.	It was not always clear why changes were being made to code by other people working on it. Code was being unnecessarily re-worked when new content still remained to be created. I'm not sure how to resolve this - possibly a rule that if you edit someone else's work you must put comments explaining what changes were made and why you made them.
5 - Australia	Onshore CP interaction went well	About offshore CP interaction my feeling is like I don't have any CP as except last 2 days the work which I was doing I could not feel any share from any offshore ppl. And I felt like that task is not in sharing it's just for me. Not only for code but I felt same in documentation too. That if I missed something only i have to finish it. No one is going to do that from offshore.
6 - Mexico	It's very exciting working in a team from different countries, this is the first time I do something like this and I think it's very good, personal and professionally	
7- Mexico	It was my first chance to be in a project where I had to share information and ideas with two other different countries, and for me, it was an experience that added me professional and personal skills in many different ways.	There are some activities that can only be carried by one person, like having the credentials of an e-mail account. So sometimes we have to work together fast before the other member leave the office, because if the task is not done, we must wait until the next day.

SPRINT ACTIVITY - Tools used/ not used

ID Member / Location	What went well?	What did not go well? How could it be made better?
1 - India	NA	NA
2- India	tfs is the nice way to manage the source control and versioning	N/A

3 - Australia	Common set of tools used by everyone on the team	
4 - Australia	Visual studio has been a good management tool.	
5 - Australia	NA	NA
6 - Mexico	All the tools are very useful and easy to use	
7- Mexico	For this application, I think k we had used the corrected tools as sql, visual studio, Microsoft office, etc. Those are just the ones that our project need.	At the beginning, there were some tools that I did not know very well, but now I have learned to use them.

SPRINT ACTIVITY - Portal used, openness to change (TFS, DeW)

ID Member / Location	What went well?	What did not go well? How could it be made better?
1 - India	We need to follow strict timeliness	We need to make it a practice i.e. To test the code before check in. It really wasted a lot of time.
2- India	tfs is the nice way to manage the source control and versioning	N/A
3 - Australia	Every project document was version controlled using TFS - making it easy to ensure the latest versions were always available to everyone on the team	
4 - Australia	NA	NA
5 - Australia	TFS is a nice source safe application	haven't look Dew yet
6 - Mexico	I think it's very good that all guys just modify "one file" in the server and everybody can see the changes	Just at the beginning wasn't so easy to get used to, but now I am
7- Mexico	Visual Studio helped a lot because it keep us from exchanging files via e-mail.	The same, at the beginning I had a lot of problems using this type of sharing files but now I have understood the whole process.

SPRINT ACTIVITY - Any other feedback on how the *sprint* was carried out?

ID Member / Location	What went well?	What did not go well? How could it be made better?
1 - India	We need to follow strict timeliness	We need to make it a practice i.e. To test the code before check in. It really wasted a lot of time.
2- India	Time utilization at different location is nicely done to speed up the work	Work can be done with more speed if we had confirmation regarding templates and standards
3 - Australia	The processes and communication were adapted and improved throughout the 2 weeks of the *sprint*	

4 - Australia	I think we made lots of positive changes based on our learning during the *sprint*.	It was really difficult not having the IS standards and templates available from the start. I think it would have reduced our workload a lot to start with if we knew what we were aiming for.
5 - Australia	nice communication	Feels like we won't be able to finish whatever we want to because of various reason like, we receive standards very late. Too much communication gap between different location team about design, lack of design guidance and lack of task allocation in starting.
6 - Mexico		
7- Mexico		

APPENDIX D - EXPERT PANEL QUESTIONNAIRE

Interview Planning (ICGSE 2013)

Questionnaire for face-to-face interviews

- **Experts number: 20**
- **Number of the questions per expert: 10 or 8**
- **Interview time duration: 30 minutes (estimated)**
- **Each question will be answered 3 times by different experts**
- **Total of the answers at the end: 192**
- **Participant experience required: academic or industry experience in GSD.**
- **For answering questions 51 to 64, the FTS software process picture will be provided to the participant.**
- **Experts will be contacted in advance by e-mail in order to arrange meetings during the ICGSE conference.**

PART 1 - *Participant Information*

1. Name:
____

2. E-mail address :
____

3. Job Title:
____

4. Type of experience in Global Software Development (GSD):
( ) Industry experience ( ) Academic experience ( ) Both (industry and academic experience)

Industry Experience *(if applicable)*

5. Company name:
____

6. Company location (city and country):
____

7. Number of Global Software Development (GSD) projects managed (if applicable):
____

8. How long have you been involved in GSD projects? (years and months) *
____

9. Do you have experience on Follow the Sun (FTS) projects? *
( ) Yes ( ) No

10. Do you have experience on around-the-clock projects? *
( ) Yes ( ) No

11. Do you have work experience on agile methodologies applied to GSD projects? *
( ) Yes ( ) No

12. In which agile methodologies are you have experience? (If applicable)
( ) XP (Extreme Programming)
( ) DSDM (Dynamic Systems Development Method)
( ) FDD (Feature-Driven Development)
( ) Scrum
( ) Kanban
( ) Crystal
( ) Other: ____

13. How long have you been researching or working with agile methodologies? (years and months) (if applicable)
____

Academic experience *(if applicable)*

14. How long have you been researching GSD? (years and months) *

____

15. How long have you been researching Follow the Sun (FTS)? (years and months) *

____

How long have you been researching around-the-clock? (years and months) *

____

16. Have you performed research in agile methodologies? *

( ) Yes ( ) No

17. In which agile methodologies are you have experience? (if applicable)

( ) XP (Extreme Programming)
( ) DSDM (Dynamic Systems Development Method)
( ) FDD (Feature-Driven Development)
( ) Scrum
( ) Kanban
( ) Crystal
( )
Other:____

18. How long have you been researching or working with agile methodologies? (years and months) (if applicable)

____

PART 2: *Questionnaire to evaluate the FTS-SPM*

SP01: TEAM SETUP

(BP30 - At least one hour overlap between two sites)

1. Can you describe how time **OVERLAP BETWEEN SITES** help in better communication and coordination in GSD projects?

2. Can you explain how **AT LEAST ONE HOUR OF OVERLAP BETWEEN SITES** is useful for performing synchronous between sites? (handoffs)

____

(BP31 - Fitting teams' working hours for a good overlap)

3. Can you explain how useful is to make time zone differences manageable in GSD projects?

4. Can you explain how **FITTING TEAMS' WORKING HOURS FOR A GOOD OVERLAP** makes time zone differences manageable?

____

(BP32 - Teams distribution across two or three sites)

5. Can you explain how the **NUMBER OF SITES** can affect communication and coordination in GSD projects?

6. Can you explain how **TEAMS DISTRIBUTION ACROSS TWO OR THREE SITES** help in better coordination and communication in FTS projects?

____

SP02: PROJECT PLANNING

(BP01 - Use of agile methods for project management)

7. Can you give an example of **AGILE METHODOLOGIES** for GSD projects?

8. Can you describe how **AGILE METHODOLOGIES are** appropriate for GSD projects?

____

(BP02 - Use of incremental software development approaches)

9. Can you describe how TDD (Test Driven Development) technique is helpful for development across multiple sites consistent?

10. Can you explain how **INCREMENTAL SOFTWARE DEVELOPMENT APPROACHES** (i.e. Test Driven Development) are useful for FTS development?

____

(BP04 - Application of FTS for testing and development phases)

11. Can you explain how adopt FTS for different **SOFTWARE DEVELOPMENT PHASES** impact on handoffs development? (Requirements, Design, Coding, Testing)

12. Can you describe how **TESTING AND DEVELOPMENT PHASES** are appropriate for developing software in FTS mode?

____

(BP36 - Similar code patterns)

13. Can you give an example where **SIMILAR CODE PATTERNS** are helpful for GSD projects?

14. Can explain how **SIMILAR CODE PATTERNS** is helpful for team members working on the same tasks?

SP03: COMMUNICATION PROTOCOL

(BP07 - Daily exchange of the project status by technologies)

15. Can you give some examples of **TECHNOLOGIES** for synchronous communication in GSD projects?

16. Can you explain how these technologies are helpful to perform **DAILY HANDOFFS** (daily exchange of the project status) in FTS projects?

____

(BP10 - Use of screen sharing technology to exchange knowledge)

17. Can you give an example of the information discussed by teams during handoffs?

18. Can you explain how **SCREEN SHARING TECHNOLOGY** is useful to exchange knowledge between team members distributed across different sites?

(BP12 - Clean handoff and sticky hands-off interactions)

19. Can you describe how daily meetings between team members distributed across different site should be performed?

20. Can you explain how short and intense interactions helps in better communication between team members during handoffs?

(BP13 - Use of real time technologies for knowledge sharing)

21. Can you give examples of technologies for real time communication?

22. Can you explain how **REAL TIME TECHNOLOGIES** help in better communication between team members in FTS projects?

(BP15 - Wikis and online forums to share knowledge between FTS teams)

23. Can you give an example of technologies used to **SHARING KNOWLEDGE** in GSD projects?

24. Can you explain how **WIKIS OR ONLINE FORUMS** are helpful to share knowledge in FTS projects?

(BP21- Adopt proper technologies or tools to support communication between FTS teams)

25. Can you give examples of **TECHNOLOGIES OR TOOLS TO SUPPORT COMMUNICATION** in GSD projects?

26. Can you explain how **ADOPT PROPER TECHNOLOGIES OR TOOLS TO SUPPORT COMMUNICATION** helps in FTS software development?

(BP22 - Time window)

27. Can you give an example where synchronous interaction between distributed sites is required?

28. Can you explain how **TIME WINDOW** helps in better collaboration and communication between sites?

(BP25 - Corporate technologies for team interaction)

29. Can you give and example of **CORPORATE TECHNOLOGY**?

30. Can you explain how **CORPORATE TECHNOLOGIES** are useful for team’s interaction in GSD projects?

(BP26 - Models of e-mail and electronic messages)

31. Can you give an example of resources used to perform asynchronous communication between sites?

32. Can you explain how **MODELS OF EMAIL AND ELECTRONIC MESSAGES** helps in better communication between team members?

SP04: CULTURAL TRAINING

(BP33 - Meetings between team members for building trust)

33. Can you give an example of cultural differences between sites in GSD projects?

34. Can you explain how **MEETINGS BETWEEN TEAMS** helps to build or increase the level of trust?

(BP35 - Cultural awareness training)

35. Can you explain how cultural diversity that exists between team members impact in GSD projects?

36. Can you explain how **CULTURAL AWARENESS TRAINING** is useful in GSD projects?

SP05: TASK ALLOCATION

(BP17 - CPro concept)

37. Can you give an example of **APPROACH FOR TASK ALLOCATION** in GSD projects?

38. Can you explain how task allocation performed on the vertical way helps to improve the team's productivity?

(BP18 - Low task granularity)

39. Can you explain how granular should be tasks allocated to distributed teams?

40. Can you explain how **LOW TASK GRANULARITY** (tasks broken down into few parts) helps to enhance recognition accuracy of a task?

(BP20 - Task distribution by sequence or dependency)

41. Can you give an example where tasks cannot be divided between two or more members who are distributed across different time zones?

42. Can you explain how **TASK DISTRIBUTION BY SEQUENCE OR DEPENDENCY** helps on 24 hours working development?

SP06: HANDOFF SESSIONS

(BP03 - Daily stand-up meetings)

43. Can you explain how **DAILY STAND-UP MEETINGS** is helpful to transfer project status updates from one site to another?

44. How appropriate are **STAND-UP MEETINGS** to perform daily handoffs?

(BP09 - Daily handful of 30 minutes duration with each development site)

45. Can you explain what information is carried out during synchronous communication in GSD projects?

46. Can you explain how appropriate are **DAILY HANDOFFS OF 30 MINUTES DURATION WITH EACH DEVELOPMENT SITE** in FTS projects?

____

(BP11 - Calendar of handoff sessions should be clearly defined)

47. How does the **RESOURCE CALENDAR** help for better coordination and communication between team members working in GSD projects?

48. How helpful is establishing a **CALENDAR FOR HANDOFF SESSIONS** in FTS projects?

____

(BP14 - Use of an FTP Server (or data repository) to exchange code and documents)

49. Can you give an example of **TECHNOLOGIES FOR EXCHANGE SOFTWARE CODE AND DOCUMENTS** in GSD projects?

50. Can you explain how the **USING OF AN FTP SERVER (OR DATA REPOSITORY)** helps to exchange software code and documents in GSD projects?

____

FTS SOFTWARE PROCESS OVERVIEW

(SP01: Team setup)

51. How important is to identify available sites and human resources before starting the project planning?

52. How appropriate is the sequence flow between **SP01: TEAM SETUP** and SP02: Project planning?

____

(SP02: Project planning)

53. Can you give some examples of the information described in the project planning?

54. How appropriate is the sequence flow between **SP02: PROJECT PLANNING TO SP03: COMMUNICATION PROTOCOL, SP04: CULTURAL TRAINING AND SP05: TASK ALLOCATION?**

____

(SP03: Communication protocol)

55. Can you explain how project settings impact on the communication protocol planning?

56. How appropriate is the sequence flow between **SP02: PROJECT PLANNING TO SP03: COMMUNICATION PROTOCOL**?

____

(SP04: Cultural training)

57. Can you describe how cultural issues can affect of the team’s performance?

58. Can you explain how **CULTURAL TRAINING SESSIONS** performed along of the project are helpful for better communication and collaboration between team members?

____

(SP05: Task allocation)

59. Can you explain how task allocation is managed in GSD projects?

60. Can you explain how appropriate is to execute **DAILY TASK ALLOCATION** at the beginning of each working day (24 hours) in FTS projects?

(SP06: Handoff session)

61. Can you describe the relation between task allocation and handoffs?

62. How appropriate is the sequence flow between **SP05: TASK ALLOCATION TO SP06: HANDOFF SESSIONS**?

(Process and sequence flow)

63. How appropriate are the sub-processes included in the FTS software process as a whole?

64. How appropriate is the sequence flow between sub-processes as a whole?

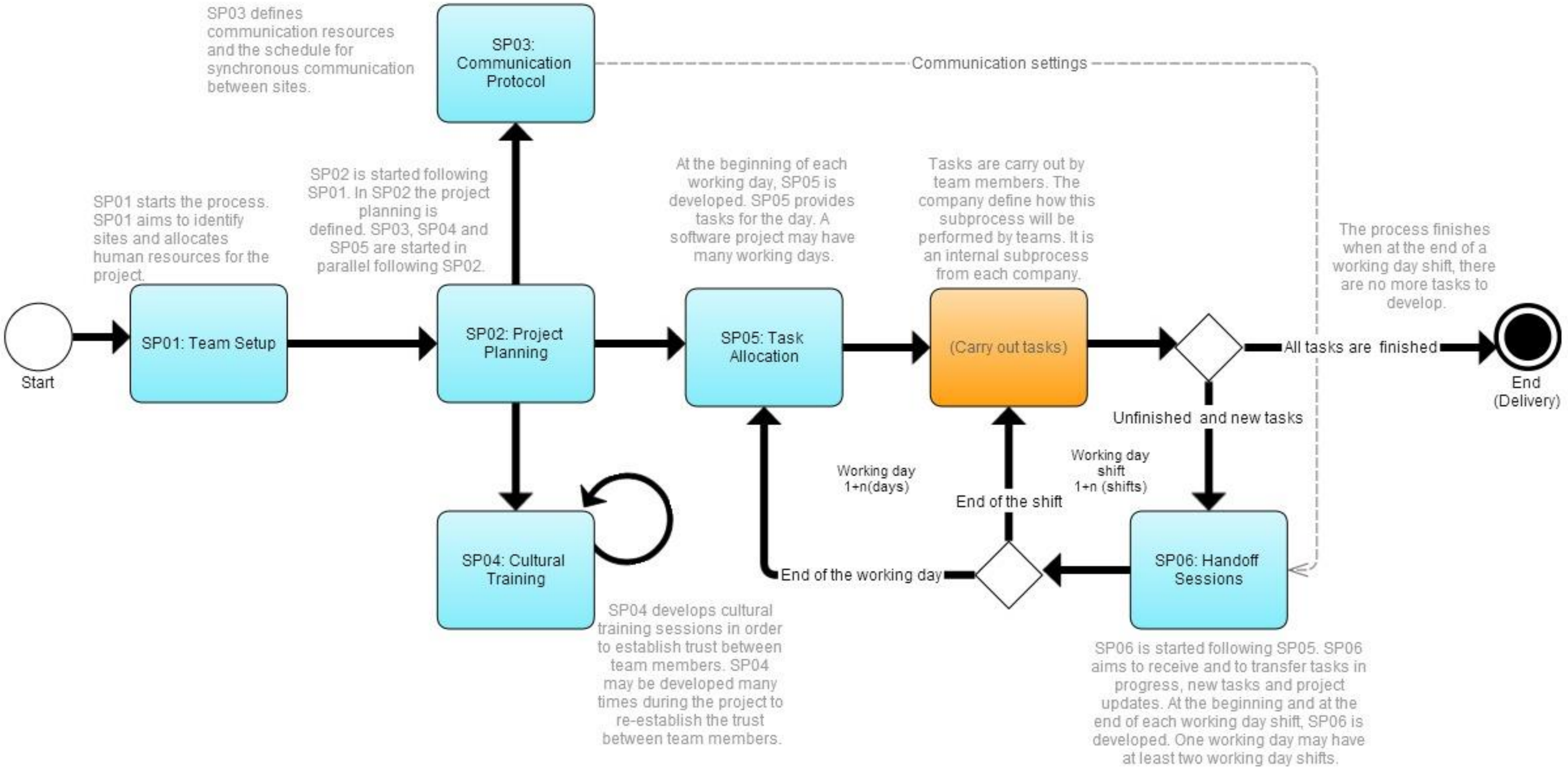
Summary of Questions by Experts

Question	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	Exp 6	Exp 7	Exp 8	Exp 9	Exp 10	Exp 11	Exp 12	Exp 13	Exp 14	Exp 15	Exp 16	Exp 17	Exp 18	Exp 19	Exp 20	Nº. experts
1	x	x	x																		3
2	x	x	x																		3
3		x			x			x													3
4		x			x			x													3
5	x						x				X										3
6	x						x				X										3
7		x							x			x									3
8		x							x			x									3
9				x										x					x		3
10				x										x					x		3
11	x									x					x						3
12	x									x					x						3
13			x					x								x					3
14			x					x								x					3
15				x					x									x			3
16				x					x									x			3
17					x							x		x							3
18					x							x		x							3
19	x								x								x				3
20	x								x								x				3
21			x							x					x						3
22			x							x					x						3
23						x						x				x					3
24						x						x				x					3
25		x						x					x								3

26		x						x					x								**3**
27					x						x						x				**3**
28					x						x						x				**3**
29							x								x				x		**3**
30							x								x				x		**3**
31				x							x							x			**3**
32				x							x							x			**3**
33	x								x					x							**3**
34	x								x					x							**3**
35						x				x						x					**3**
36						x				x						x					**3**
37			x									x	x								**3**
38			x									x	x								**3**
39					x									x				x			**3**
40					x									x				x			**3**
41						x			x							x					**3**
42						x			x							x					**3**
43			x							x							x				**3**
44			x							x							x				**3**
45								x				x						x			**3**
46								x				x						x			**3**
47				x		x													x		**3**
48				x		x													x		**3**
49							x				x									x	**3**
50							x				x									x	**3**
51		x											x			x					**3**
52		x											x			x					**3**
53					x					x					x						**3**
54					x					x					x						**3**

55				**x**									**x**						**x**		**3**
56				**x**									**x**						**x**		**3**
57						**x**								**x**						**x**	**3**
58						**x**								**x**						**x**	**3**
59							**x**								**x**					**x**	**3**
60							**x**								**x**					**x**	**3**
61								**x**									**x**			**x**	**3**
62								**x**									**x**			**x**	**3**
63							**x**				**x**		**x**								**3**
64							**x**				**x**		**x**								**3**
65																	**x**	**x**	**x**		**3**
66																	**x**	**x**	**x**		**3**
Total	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**10**	**8**	**198**

FTS software process overview


SP03 defines communication resources and the schedule for synchronous communication between sites.
SP03: Communication Protocol
Communication settings
SP01 starts the process. SP01 aims to identify sites and allocates human resources for the project.
SP02 is started following SP01. In SP02 the project planning is defined. SP03, SP04 and SP05 are started in parallel following SP02.
At the beginning of each working day, SP05 is developed. SP05 provides tasks for the day. A software project may have many working days.
Tasks are carry out by team members. The company define how this subprocess will be performed by teams. It is an internal subprocess from each company.
The process finishes when at the end of a working day shift, there are no more tasks to develop.
Start
SP01: Team Setup
SP02: Project Planning
SP05: Task Allocation
(Carry out tasks)
All tasks are finished
End (Delivery)
Unfinished and new tasks
Working day 1+n(days)
Working day shift 1+n (shifts)
End of the shift
SP04: Cultural Training
End of the working day
SP06: Handoff Sessions
SP04 develops cultural training sessions in order to establish trust between team members. SP04 may be developed many times during the project to re-establish the trust between team members.
SP06 is started following SP05. SP06 aims to receive and to transfer tasks in progress, new tasks and project updates. At the beginning and at the end of each working day shift, SP06 is developed. One working day may have at least two working day shifts.

Preliminary FTS-SPM overview

APPENDIX E – SYSTEMATIC LITERATURE REVIEW ON GSD

Appendix E summarizes results from an SLR performed on GSD. This SLR was performed in 2010 when I have started to study GSD. This study contributed to better understanding the research area and what have been studied over the years. This SLR also aimed to identify studies reporting FTS development in internal offshoring environments. I observed from the results this study a little research on FTS development and new opportunities for research in this field.

Results this study were published in the paper: "*Mapping the Evolution of Research on Global Software Engineering: A Systematic Literature Review*" published in the 13th International Conference on Enterprise Information Systems (ICEIS)[10].

- **The Evolution of Research on Global Software Development: A Systematic Literature Review**

Studies on GSD have increased in recent years. However, it was not clear in the literature how research in this area has evolved in terms of topics being investigated. For this reason, I conducted a study to identify software engineering areas explored by studies in the GSD research area. This study was important to better understand the GSD research area, to identify relevant studies, and to indicate possible gaps in this recent research area.

A Systematic Literature Review (SLR) was adopted as a research method to conduct this study. I conducted this study from September to November of 2010 in accordance with the procedures recommended by Kitchenham and Charters [KIT07]. Next, I present the research protocol, obtained results, and the main contributions from this study.

Research Questions

I defined two research questions (RQ) for this study as follows:

RQ1: *Which software engineering areas have studies in GSD? What are the discussed topics in each area identified?*

RQ2: *And even, are there studies reporting the FTS development strategy in internal offshoring environments?*

The first research question aimed to provide a general view of the whole research field in GSD and its topics discussed in each software engineering area. The second question is focused on FTS topic. This question aimed to identify studies discussing FTS applied to internal offshoring environments. This research question was included in this study already aiming to investigate the main topic this thesis.

Search string and Data Sources

To answer the RQ1, I defined four keywords to compose the first search string:
(global software development <OR> global software engineering <OR> distributed software development <OR> distributed software engineering)

[10] The13th International Conference on Enterprise Information Systems (ICEIS) was held in Beijing, China from June 8 -11st, 2011. All papers presented at the conference venue were included in the SciTePress Digital Library.

I adopted the same search string to answer the RQ2. Additionally, I included a second search string to confirm my results.

(Follow the Sun <OR> around-the-clock <OR> round-the-clock <OR> 24-hour development <AND> software engineering)

The second string adopted the word Follow the Sun and similar terms. With the second search string, I obtained a greater number of results. However, the most studies have not been discussing FTS. Anyway, I analyzed these studies observing contributions in the area.

I searched studies through five digital libraries:

- IEEEXplore (http://ieeexplore.ieee.org/)
- ACM Digital Library (http://www.sciencedirect.com/)
- Wiley InterScience (http://www.interscience.wiley.com/)
- Elsevier Science Direct (http://www.sciencedirect.com/)
- Scopus (http://www.scopus.com/home.url/)

In each digital library, I utilized a search string in accordance with each search engine available. I considered the period from 1990 to 2011 for searching, because studies in this area were published from 1990. To Prikladnicki, Audy, and Shull [PAS10], GSD phenomenon started in early 1990, but has it become a powerful competitive strategy over the past 10 years.

I adopted as criteria to select studies consisting in reading title followed by abstract and in some case the full paper. I deleted repeated studies and studies not belong to the software engineering area. I present the number of publications found in the Table 45.

Table 45 – Paperlibraryn each digital libraries.

Digital Library	**First search string**	**Second search string**
IEEExplore	469	10
Wiley InterScience	146	49
Elsevier Science Direct	258	31
ACM Digital Library	108	34
Scopus	595	40
Total	**1576**	**164**

Table 46 presents the number of selected and excluded studies after further analysis in this study.

Table 46 – Selected papers.

Digital Library	**Studies**			
	First search string		Second search string	
	Excluded	**Selected**	**Excluded**	**Selected**
IEEExplore	66	403	2	8
Wiley InterScience	122	24	49	-
Elsevier Science Direct	1	257	28	3
ACM Digital Library	29	79	32	2
Total	**218**	**763**	**108**	**13**

I deleted Scopus digital library from this study, because more than 30% of studies from that library were repeated with other libraries selected for this study.

Data analysis

To answer RQ1, I used Knowledge Areas (KAs) from SWEBOK (Software Engineering Body of Knowledge) and PMBOK (Project Management Body of Knowledge) to classify the studies. SWEBOK is a guide that categorizes the software engineering domain defining ten KAs in software engineering [ABR04]. PMBOK describes a collection of processes and nine KAs describing best practices for the project management discipline [PMB04]. However, during the classification process I observed the lack of areas from SWEBOK and PMBOK to classify studies. Thus, I defined three new KAs:

- **General Area:** studies in GSD without specific KA in SWEBOK and PMBOK guides.
- **Position Paper:** studies reporting an author's point of view about any aspect in GSD area.
- **Systematic Review:** studies using systematic literature review (SLR) method. Studies have mapped in this area may also be mapped in other KAs from SWEBOK or PMBOK.

Obtained Results

To answer RQ1, first I identified the number of the studies in each knowledge area (KA) of the SWEBOK and PMBOK. Studies not mapped in KA from SWEBOK or PMBOK were mapped in new KAs defined in this study. Table 47 presents the number of studies found in each KA from SWEBOK, PMBOK and New areas.

Table 47 – Studies mapped in the KAs**.**

	Knowledge Area (KA)	**Studies Number**
SWEBOK	Software Engineering Processes	87
	Methods and Tools of Software Engineering	69
	Software Project	51
	Software Engineering Management	46
	Software Requirements	33
	Software Construction	32
	Software Quality	15
	Software Maintenance	13
	Software Testing	11
	Software Configuration Management	4
PMBOK	Project Communications Management	51
	Project Human Resource Management	40
	Project Quality Management	11
	Project Risk Management	11
	Project Integration Management	9
	Project Scope Management	4
	Project Cost Management	3
	Project Time Management	0
	Project Procurement Management	0
New areas	General Area	99
	Systematic Review	8
	Position Paper	5

As shows Table 47, there are studies distributed in all areas from SWEBOK. In the PMBOK, studies are distributed in seven KAs and in the new areas, the majority of studies were found in the General Area.

As follow, I present topics discussed in each KA area. First, I present topics from SWEBOK KAs following by PMBOK KAs and New KAs. In Table 48, I present topics found in the SWEBOK KAs.

Table 48 - SWEBOK KAs topics.

SWEBOK KAs	Topics	Area%	Total%
Software Engineering Processes	Management and improvement of processes	26.4	3.8
	Process and capacity models	18.4	2.7
	Teams and organization’s management	13.8	2.0
	Tools	5.7	0.8
	Software development environments; Concepts of the GSD	4.6	0.7
	Methods; Agile methods; Process support systems; Knowledge management	3.4	0.5
	Collaborative Development; Metrics; Emerging topics and challenges	2.3	0.3
	Patterns; Frameworks; Software Quality; Components; UML diagrams; Software life cycles; Risks; User profiles; Performance analysis; Applications; Allocation of tasks	1.1	0.2
Methods and Tools of the Software Engineering	Process support tools	18.8	2.2
	Communication tools	11.6	1.3
	Research on existing GSD tools	10.1	1.2
	Modeling of processes; Personnel management tools	7.2	0.8
	Project management tools	5.8	0.7
	Tools for requirements; Tools for real-time simulation	4.3	0.5
	Methods and tools for the Web; Tools for simulating projects; Time optimization tools; Tools for assessment of case tools documentation; Tools for replacement and acquisition of the code	2.9	0.3
	Artificial intelligence; Tools for the development team’s skills; Construction of middleware; Quality tools; New methods unifying methodologies; Wiki; Route calculation tools; Testing tools; Offshore development tools	1.4	0.2
Software Project	Software architecture and structure	33.3	2.8
	Development tools	19.6	1.7
	Development methodologies	9.8	0.8
	Construction of models; UML models and prototypes; Environment management processes; Design patterns; Characteristics of the structure of GSD; Frameworks	5.9	0.5
	Inner source software development; Taxonomies to identify GSD dimensions; Wiki for GSD; Development via mobile	2.0	0.2
Software Engineering Management	Models; Process improvement	13.0	1.0
	Teams and organization's management; Challenges	10.9	0.8
	Tools and techniques; Development strategies	8.7	0.7
	Risk management and reduction; Approaches to projects with similar environments	6.5	0.5
	Business models	4.3	0.3
	Project Scope Management; Agile methods; Estimation of effort; Metrics; Ontology; Middleware; Support agents; Frameworks; Valuation in virtual environments; Security	2.2	0.2

Software Requirements	Approaches for elicitation and trading requirements; Use of social networks, semantic data and wiki	18.2	1.0
	Tools for systems requirements management	12.1	0.7
	Simulators for training and learning requirements elicitation	9.1	0.5
	Process support system; Culture and distance aspects; Challenges in the requirements negotiation; Requirements extraction techniques	6.1	0.3
	Notations for requirements modeling; Validation of requirements; Communication aspects in process of requirements extracting; Text-based communication for elicitation of requirements; UML models of requirements; Frameworks; Models; Volatile requirements	3.0	0.2
Software Construction	Project theories and models	28.1	1.5
	Open source development; Implementation tools for virtual environments; Development Approaches	12.5	0.7
	Languages	9.4	0.5
	Methodologies; Managing artifacts; Tools; Using components; Patterns; Business models	6.3	0.3
	Task allocation; Agile methods; Platforms for object oriented languages; CORBA based distributed components; Development testing; Social networks	3.1	0.2
Software Quality	Project quality management	73.0	1.8
	Quality assessments; Management and quality process; Influence of distance and geographical dispersion in software quality; Analysis of validation experiences	13.3	0.3
	Open source software quality; Frameworks; New approaches to software quality; Models; Performance indicators	6.7	0.2
Software Maintenance	Sources of error in products and processes; Management mechanisms	30.8	0.7
	Cost estimation	23.1	0.5
	Collaborative work; Models	7.7	0.2
Software Testing	Processes for defect detection and simulation	27.3	0,5
	Experiences across teams; Effectiveness test models	18.2	0.3
	Performance assessment methodologies of applications; Allocation of developers and testers to perform tests; Training tutorials; Internet-based approaches to testing; Verification and validation; Tools	9.1	0.2
Software Configuration Management	Managing documentation versions; Tools coordination	50.0	0.3
	Security	25.0	0.2

I found a larger percentage of studies in the Software Engineering Processes KA. This KA has 14.4% of the studies found in the SWEBOK. The most discussed topic among the studies in Software Engineering Processes KA was the Management and improvement processes. Twenty-one studies in this topic were found, which corresponded to 24.6% of the studies in the KA. This topic has the second largest percentage (3.8%) of all the found studies in GSD. It makes sense, when one notices that this topic is applied in several software engineering KAs. The Process and capacity models topic is largely discussed in studies on Software Engineering Processes, corresponding to 18.4% of the KA. Studies in this topic focus on the definition, implementation, change and improvement processes. The Teams and organizations management topic has 13.8% of the found studies in the KA and 2.0% of the found studies in the SLR. Other identified topics in the Software Engineering Processes KA have smaller than 0.6% percentages.

In the Methods and Tools of Software Engineering KA, 69 studies were found, mapped into 24 topics. This KA has 11.4% of found studies in the SLR. The most discussed topic in the Methods and Tools of Software Engineering KA is Process support tools. This topic is among the most discussed studies in GSD. Related to this topic, 13 studies were identified, which corresponded to 2.2% of found studies in the SLR. The Communication tools are also greatly discussed in GSD. Studies on this topic are relevant to the GSD research field, because the communication is much more present and relevant in GSD than in traditional development. In this KA, the Research on existing GSD tools topic has 7 studies, corresponding to 10.1% of the studies in the KA and 1.2% in the SLR. With the obtained results, it can be verified that the topics that discuss the basic characteristics of GSD have the most studies. However, the total identified studies for these topics are 6 studies and corresponded to 10.1%.

I found 51 studies in the Software Project KA. It was the third KA with the largest percentage of studies in GSD (8.5%). The Software architecture and structure topic had the highest number of studies in the Software Project KA. This result was expected, because the Software architecture and structure topic addresses basic aspects to the development of software projects. Developing tools is the second most discusses topic in Software Project KA. This topic also is presented among the most discussed topics in the Software Requirements and Software Construction KA. This way, it can be verified that there is a great need of building tools in these areas. In these KAs, the lack of tools for GSD can be pointed out [SOL10] [JAB10]. Among other topics in Software Project, results show the least explored topics with 5.9% and 2.0% of the studies.

In the Software Engineering Management KA, 46 studies and 19 research topics were found. This KA has 7.6% of total studies identified in this SLR. I have observed that the Software Engineering Management KA is covered mainly in the Models and Processes improvement topics. However, this KA is explored through its many topics. In the Models topic, mainly coordination and team management models are addressed. In the Processes improvement topic, the found studies present evaluation process, task allocation processes and practices for vendors and software processes. These topics, although prevalent in the Software Engineering Management KA, correspond only to 1% of the found studies in the SLR. This result shows that these topics are hardly discussed in GSD.

In the Software Requirements KA, the most discussed in the found studies during the SRL were Approaches to the elicitation and trading requirements and Use of social networks, semantics data and wiki. This result shows that studies are carried out to explore specific topics in this KA, related to the requirements elicitation. The topic Tools for systems requirements management is also greatly discussed in Software Requirements KA. This topic appears in 12.1% of the studies within the KA. However, regarding the percentages of the two most discussed topics in this KA, I observed that they together represent less than 2% of all studies in GSD. It makes sense if one notices the low number of found studies for this KA.

The Software Construction KA corresponded to 5.3% of the found studies in GSD. The Project theories and models topics were the most discussed among the identified studies in the Software Construction KA. However, it matches only 1.5% of all the found studies in the SLR. It makes sense, considering that the Software Construction KA is one of the most critical KAs in GSD [JAB10]. Out of the 17 identified topics in Software Construction KA, 12 topics had 1 or 2 studies, corresponding to 3.1% and 6.3% respectively. Other topics had 3 or 4 studies corresponding to 9.4% and 12.5% respectively.

In the Software Quality KA 15 studies and 10 research topics were found. This KA has 2.4% of found studies in the SLR. The topic with the largest percentage of studies in the Software Quality KA is Projects quality management. This topic is also recognized as a specific KA in the PMBOK, what explains the high percentage of found studies on this topic. The Quality assessment, Management and quality processes, Influence of distance and geographical dispersion in software quality and Analysis of validation experiences topics obtained 13.3% of the found studies in the KA. This percentage corresponds to 2 studies by topic. Other topics obtained 6.7% of studies, with one study each in the KA.

The Software Maintenance KA topics are appointed as challenges of GSD [JAB10]. To Cataldo et al. [CAT07], in software maintenance, several activities are involved and must be coordinated to avoid errors and decrease flaws. This way, studies about these topics are required. The Cost estimation, Collaborative work and Models topics are also discussed in the Software Maintenance KA, but represent only 0.9% of the GSD research area.

The Software Testing KA is the second least exploited area by studies in GSD. It concentrates 1.8% of the total of studies in GSD. In the Software Testing KA three topics were identified as the most discussed: Processes for defect detection and simulation, Experience across teams and Effectiveness test models. The Processes for defect detection and simulation topic had three found studies, which corresponded to 27.3% according to Table 48. The large percentage of studies on this topic occurs due to the need of making the GSD more efficient. The Software Testing KA is motivated by the quest for highly reliable, efficient and free of defect systems [CAT10]. The Experiences across teams topic has 18.2% studies in the Software Testing KA. The presence of these studies on this topic is motivated by challenges in teams' management. It makes sense, when one observes the main features of the GSD: the dispersed teams and multi-site production [JIM09] [LAN08]. Only one study was found related to other topics in Software Testing KA, which corresponded to 9.1%.

In the Software Configuration Management KA, 4 studies and 3 topics were found. This area corresponds to 0.6% of the found studies in GSD. This KA has the least number of studies in GSD. The percentages found for each topic from this KA are high. However, the percentages found in the GSD research field are low. It shows that this KA is not greatly explored. With the results obtained in the Software Configuration Management KA, it is possible to observe that research on this KA is not prioritized in GSD. However, the complexity of the software systems reflects of this configuration [BRU06]. Therefore, new practices and tools are required to support this type of development.

By the analysis of the data of this section, I observe that in most SWEBOK KAs, there were not many studies concentrated on a single research topic. There is a wide distribution of the studies among the topics. In addition, there are many topics with just a single study.

One hundred and twenty nine studies related to project management were identified. Topics these studies were mapped in the PMBOK KAs. In Table 49, I present the topics discussed in each KA area.

Table 49 - PMBOK KAs topics.

PMBOK KAs	Topics	Area%	Total%
Project Communications Management	Communication support tools	17.6	1.5
	Communication flaws, barriers, case studies and experiments	15.7	1.3
	Communication management and coordination of knowledge; Communication patterns between project groups	7.8	0.7
	Communications requirements; Communication models; Agile methods	5.9	0.5
	Coordination mechanisms of communication; Structures of delay; Effects of distance in communication; Social networks and wiki	3.9	0.3
	Studies of communication language and linguistic differences; Use of instant messages in GSD; Predictive modeling processes using notations to improve communication among teams; Transcripts of communication among teams; Ontology for communication	2.0	0.2
	Teams management and development organizations	37.5	2.5
	Virtual teams	20.0	1.3

Project Human Resource Management	Tools; Ability of teams and individuals	12.5	0.8
	Impact of the culture on the design team	10.0	0.7
	Models	7.5	0.5
	Adaptations of roles	5.0	0.3
	Allocation of tasks and teams; Simulation approaches Communication practices	2.5	0.2
Project Quality Management	Defect detection and quality evaluation	27.3	0.5
	Quality frameworks; Quality approaches	18.2	0.3
	Quality in offshore projects; Distance and software quality; Analysis of validation experiences; Quality and geographic dispersion	9.1	0.2
Project Risk Management	Analysis and coordination methods	45.5	0.8
	Frameworks	36.4	0.7
	Risk reduction; Risk assessment types; Identification of problems related to the emergence of risks; Risk management models; Agile methods; Risk factors in outsourced offshoring	18.2	0.3
	Web Risks; Tools	9.1	0.2
Project Integration Management	Coordination models; Approaches;	22.2	0.3
	Coordination processes; Software life cycles; Knowledge management; Project phases; Offshore development tools	11.1	0.2
Project Scope Management	Development routes studies; Group project management; Models; Collaborative software	25.0	0.2
Project Cost Management	Cost analysis of collaboration	100.0	0.4
Project Time Management	None	0	0
Project Procurement Management	None	0	0

In the Project Communications Management KA, I found 51 studies corresponding to 8.4% of the GSD research field. The Project Communications Management KA is not discussed directly by SWEBOK. On the other hand, in GSD, communications management is considered one of the most critical areas [JAB10]. In the PMBOK, the communications management focuses on the project itself, but the communication is also being discussed in the context of software development. The Communication support tools topic had 9 identified studies, corresponding to 17.6% of the studies in KA. The Communication flaws, barriers, case studies and experiments topic have also a high percentage of studies, about 15.7%.

In the Project Human Resource Management KA, 40 studies were mapped into 10 topics. This KA had 6.6% of found studies in GSD. Project Human Resource Management KA is not directly present in the KAs from the SWEBOK. However, in GSD, human resources management is considered the greatest challenge [JAB10]. In this KA most studies are present in the Teams management and development organizations topic. These studies discuss team coordination, employee's turnover, teams effectiveness, selecting groups, among other topics. The Virtual teams topic had 20% of found studies found in the Project Human Resource Management KA. This result is due to the distribution of several production teams in a GSD environment. In GSD, teams range from sub remote teams to production teams, with different functional roles [LAN08].

In the Project Quality Management KA, 11 studies were found mapped into 7 research topics. This KA had 1.4% of found studies in the SLR, where the most studies discuss defect detection and quality evaluation. The Defect detection and quality evaluation topics had 27.3% of found studies in the KA. This topic is not directly present in the Software Quality KA from SWEBOK. However, it is in the Project Quality Management topic of the

Software Quality KA. The Quality approaches and Quality frameworks topics were also discussed in the Project Quality Management KA. Each topic corresponds to 18.2% of found studies in KA. However, these topics have only 0.3% found studies in the GSD research field. Other identified topics in the Project Quality Management KA had only 1 study each, which corresponds to 9.1%.

In the Project Risk Management KA, 11 studies were found and mapped into 10 topics. This KA had 1.8% of found studies in the SLR. The Analysis and coordination methods topic is present in 45.5% of the studies in the Project Risk Management KA. In this KA, the same amount of found studies was in the Project quality management KA. However, the percentage of discussed subjects was larger in one KA then in the other, but with the same amount studies. Although, it is observed that the smaller the percentage of studies in a KA, the more distributed are the studies. The high value obtained in only one of the topics of the Project Risk Management KA showed that this topic is extremely important within that KA. In GSD, the analysis and coordination of risks are key factors for the project's success [BAS09].

In the Project Management Integration KA, 9 studies were identified, corresponding to 1.4% of found studies in the SLR, so as 7 research topics. Coordination models and Approaches had 22.2% of studies each in the KA. However, the high percentage of found studies in these topics does not indicate a significant number of studies. The topics with 22.2% of studies corresponded to 2 studies while the topics with 11.1% corresponded to 1.

In the Project Scope Management KA, 4 studies were identified, and each one has 1 found topic. The Project Scope Management KA is barely explored by research in GSE. Four studies were found which represented 0.2% of the studies in GSD. With the low percentage of found studies in this KA, it was not possible to verify a predominant topic, among the 4 identified ones.

In the Project Cost Management KA, 3 studies and only 1 topic was found. The identified studies in the Project Cost Management KA discuss the successful practices used in product development projects to reduce costs. This topic corresponds to 0.4% of the found studies in the SLR.

In the Project Procurement Management and Project Cost Management KAs, no studies were identified. These KAs are not explored in GSD research. However, comparing all the results obtained in the PMBOK KAs, I observed that there is a lack of studies in the most areas. Areas that had a small amount of studies did not have a well-defined predominant topic.

In the Project Communications Management and Project Human Resource Management KAs, the predominant topic in each area reflects the challenges of GSD. In addition, issues were identified, such as communication barriers and flaws and the lack of studies on practical applications and techniques to manage dispersed teams and organizations [JIM09] [JAB10]. The found results are caused by a research field that is still under development.

In the new KAs defined in this study, I found the largest percentage of studies in the General area. In Table 50, I present the topics discussed in each KA area.

Table 50 - New KAs topics.

New Kas	Topics	Area%	Total%
General Area	Education	31.3	5.1
	Revisions of workshops	18.2	3.0
	Challenges related to spatial and temporal aspects, cultural partners	17.2	2.8
	Lessons learned and practices	12.1	2.0
	New research areas	9.1	1,5

	Evolution of GSD	4.0	0.7
	Agile methods; Strategies for GSD	3.0	0.5
	Analysis of the technical and non-technical challenges for GSD; Difficulties for the research in the area	2.0	0.3
	Research groups; Concepts; GSD benefits	1.0	0.2
Systematic Review	Challenges and solutions in GSD; Critical barriers in GSD	25.0	0.3
	Process models; Tools; Development patterns and practices; Communication aspects; Success factors; Agile methods	12.5	0.2
Position Paper	Study prospects in GSD	60.0	0.5
	Considerations about culture, location, coordination in GSD	40.0	0.3

The General Area concentrates the largest percentage of studies (16.4%). Ninety nine studies were identified, distributed in thirteen topics. Education was the most discussed topic in the General Area. This topic represented 31.3% of the KA and 5.1% of found studies in the SLR. This topic represents the largest percentage of studies in the SLR. To Bellur [BEL06], with the globalization process, it is necessary to introduce GSD to the educational context. In addition, there is a higher percentage of studies on the Challenges related to spatial and temporal aspects, cultural partners topic (17.2%), followed by the Lessons learned and practices (12.1%). The Presentation of research groups and Concepts, GSD Benefits topics were barely explored, corresponding to 1% of found studies. The results obtained from the General Area show that many studies focus on a specific software engineering KA. This is expected from a research field as GSD, which is still in phase of growth and improvement.

In the Systematic review KA, 8 studies were mapped into 8 topics. The identified topics in the Systematic Review KA are also mapped in the SWEBOK KAs.

Any topic was mapped into the PMBOK KAs, because systematic reviews that discuss aspects of projects were not identified. Studies in this area discuss Challenges and solutions in GSD and Critical barriers in GSD. These topics corresponded to 25% of the KA. The other topics had only one identified study, corresponding to 12.5% each.

In the Position Paper KA, 5 studies and 2 topics were identified. The most discussed topic in the Position Paper KA is Study prospects in GSD. This topic had 60% of the area studies. Studies in this topic discuss mainly on new directions pursued by research in GSD. In the Considerations about culture, location and coordination in GSD topic, studies discussed the characteristics of the GSD environment. With the results obtained in this area, it was not possible to verify a study tendency.

To answer the RQ2, I adopted KAs from SWEBOK, PMBOK, and New KAs defined in this study. As result, I identified studies discussing FTS in four areas from SWEBOK and one area from General studies. I did not identify studies about FTS in KA from PMBOK guide. I show these data in Table 51.

Table 51 – Topics from FTS studies.

KAs	Topics	Total
Software Engineering Process	Environments to implement FTS; Methodologies and Models; Definition and process improvement; Task allocation	6
Software Project	Production site	1
Software Construction	Extreme Programming (XP)	1
Software Quality	Quality process	1

I identified four studies in the General area. These studies discuss the benefits and reflections about FTS practice. I found nine studies with research topics related to SWEBOK areas.

The largest number of studies conducted in FTS was mapped in the SWEBOK KAs. These KAs aims to define process and models for management, measurement, implementation and evaluation in the software engineering. From our results, I observed FTS as an immature research topic. To Carmel, Dubinsky, and Espinosa [CDE09], FTS have many development aspects that need to be identified and improved.

I found only one study in the Software project, Software Construction and Software Quality KAs. These studies discuss production site, extreme programming (XP) and quality process, respectively.

I did not find studies discussing internal offshoring environments. The lower number of studies found for FTS shows an immature research area with research opportunities in all KAs.

Main contributions from the SLR

GSD is considered a tendency for organizations due to the globalization of businesses. However, in the literature, there are still few studies that explore this type of software development.

The studies that discuss GSD are focused mainly on the following KAs: Software Engineering Processes (14%), Methods and Tools of Software Engineering (11%), Project Communications Management (8%) and Project Human Resource Management (6%). The first two KAs provide subsidies so that GSD can be implemented in organizations, while the last two handle characteristics that are more present in GSD, which are communication and distributed teams. The high percentage of studies in these KAs was predictable, when one notices that GSD depends mainly on these KAs to be applied.

The Project Procurement Management and Project Cost Management KAs, both from PMBOK, did not have identified studies. However, other KAs, such as Project Procurement Management and Time Management had less than 10 studies each. With this result, it was observed that these KAs did not have as much influence on GSD as others. However, studies that discuss topics related to these KAs are relevant to GSD improvement.

The Software Engineering Processes, Methods and Tools of Software Engineering and Design software KAs from SWEBOK correspond together to 27.1% of found studies in GSD. These areas obtained a higher percentage of studies, in comparison to KAs of the PMBOK. This result showed that most challenges of GSD are concentrated in these KAs.

Regarding the most discussed research topics in GSD, 11 topics were found with more than 10 studies in all KAs. These topics are:

- Education (4,1% - 31 studies);
- Management of improvement processes (3,0% - 23 studies);
- Revisions of workshops (2,4% - 18 studies);
- Challenges related to spatial and temporal aspects, socio cultural (2,2% - 17 studies);
- Architecture and structure of software (2.2% - 17 studies);
- Process templates and capacity (2.1% - 16 studies);
- Tools support processes (1.7% - 13 studies);
- Practices and lessons learned (1.6% - 12 studies);
- Management teams and organizations (1.6% - 12 studies);
- Quality management projects (1.4% - 11 studies);
- Tools for developing (1.3% - 10 studies).

Topics with the greatest percentage of studies are presents in the General Area KA. In this KA 4 topics are discussed. This result shows that research field is developing and there are many research topics that can be exploited.

Observing the research questions formulated for this study, can be inferred that there are studies for most software engineering KAs, but also, that these KAs are not completely exploited. There are several research topics with one or more studies, and same those topics that were prevalent among the results found, have study opportunities.

Regarding to FTS research, I conclude this study observing that a few studies in FTS area have been published so far. Increasing the research in this area might contribute to increase the FTS adoption by companies and to develop new theories for FTS.

APPENDIX F –PUBLICATIONS

2014

Kroll, J., Richardson, I., Audy, J. L. N., Fernandez, J., **"Handoffs Management in Follow the Sun Software Projects: A Case Study",** In 47th Hawaii International Conference on System Sciences (HICSS), Hawai'i Island, USA, 2014.

Kroll, J.; Richardson, I., Audy, J. L. N. **"Proposing a Software Process Model for Follow the Sun Development"**. In: International Conference on Software Engineering (SEKE), 2014, Vancouver, Canada.

Kroll, J., Richardson, I., Audy, J. L. N. **"FTS-SPM: A Software Process Model for Follow the Sun Development - Preliminary Results",** In PARIS: Methods and Tools for Project/Architecture/Risk Management in Globally Distributed Software Development Projects, Proceedings of International Conference on Global Software Engineering (ICGSE), Shanghai, China, 2014.

Kroll, J., Estácio, B. J. S., Audy, J. L. N., Prikladnicki, R., "**An Initial Framework for Researching Follow-the-Sun Software**", Proceedings of International Conference on Global Software Engineering (ICGSE), Shanghai, China, 2014.

2013

Kroll, J. ; Prikladnicki, R. ; Audy, J. L. N. ; Carmel, E. ; Fernandez, J. **"A Feasibility Study of Follow the Sun Software Development for GSD Projects"**. In: International Conference on Software Engineering (SEKE), 2013, Boston, USA.

Kroll, J., Richardson, I. **"A Propose Process for Follow the Sun Software Development",** in NUIG/UL PhD Research Day, Galway, Ireland, 2013. (Poster).

Kroll, J., Audy, J. L. N. "**Adopting Agile Methods for Follow the Sun Software Development**." In: 19th Americas Conference on Information Systems (AMCIS), 2013, Chicago.

Kroll, J., Audy, J. L. N. "**Desenvolvimento de Handoffs em Projetos de Software Follow the Sun: Um Relato de Experiência."** In: Workshop de Desenvolvimento Distribuído de Software (WDDS), 2013, Brasilia. CBSoft 2013 - VII WDDS 2013, 2013.

Kroll, J., Hashmi, S. I, Richardson, I., Audy, J. L. N.,**"A Systematic Literature Review of Best Practices and Challenges in Follow the Sun Software Development",** In PARIS: Methods and Tools for Project/Architecture/Risk Management in Globally Distributed Software Development Projects, Proceedings of International Conference on Global Software Engineering (ICGSE), Bari, Italy, 2013.

2012

Kroll, J., Audy, J. L. N., "**Mapping Global Software development Practices for Follow the Sun Process**", Proceedings of International Conference on Global Software Engineering (ICGSE), Porto Alegre, Brazil, 2012.

Kroll, J., Santos, A. R, Prikladnicki, R., Hess, E. R, Glazer, R., Sales, A., Audy, J. L. N., Fernandes, P. **"Follow the Sun Software Development: A Controlled Experiment to Evaluate the Benefits of Adaptive and Prescriptive Approaches",** Proceedings of the

24th International Conference on Software Engineering & Knowledge (SEKE 2012), 551-556.

Kroll, J., Audy, J.L.N., **“Follow the Sun Strategy: A Process for Global Software Development”**, Proceedings of International Conference on Global Software Engineering (ICGSE), Porto Alegre, Brazil, 2012.

2011

Kroll, J., Audy, J. L. N., Prikladnicki, R., “**Mapping the Evolution of Research in Global Software Development**”, Proceedings of International Conference on Enterprise Information Systems (ICEIS), Beijing, China, 2011.

Kroll, J., Hess, E. R., Audy, J. L. N., Prikladnicki, R., “**Researching into Follow the Sun Software Development: Challenges and Opportunities**”, Proceedings of International Conference on Global Software Engineering (ICGSE), Helsinki, Finland, 2011.

Kroll, J., Audy, J. L. N., Prikladnicki, R., “**Desmitificando o Desenvolvimento de Software Follow the Sun: Caracterização e Lições Aprendidas**”, Proceedings of Workshop de Desenvolvimento Distribuído de Software (WDDS), São Paulo, Brasil, 2011.