Proposing a Software Process Model for Follow the Sun Development

Josiane Kroll\textsuperscript{1}, Ita Richardson\textsuperscript{2}, and Jorge L. N. Audy\textsuperscript{1}
\textsuperscript{1}Pontifical University Catholic of Rio Grande do Sul (PUCRS), Porto Alegre, Brazil
\textsuperscript{2}Lero - The Irish Software Engineering Research Centre, University of Limerick (UL), Limerick, Ireland
josi.unc@gmail.com, ita.richardson@ul.ie, audy@pucrs.br

Abstract— Many software organizations are restructuring their software development groups by extending operations to offshore software development centers. Thus, a Follow the Sun (FTS) development is a potential strategy for these organizations. FTS can help with reducing the software development life cycle duration and consequently the time-to-market. However, while the FTS concept looks promising in theory, it appears to be difficult in practice and software organizations have a pressing need for support in how to successfully implement FTS in a global software environment. In this paper, we combine the results from prior work in FTS and a design validation method conducted by experts to propose a software process model for FTS development, named FTS-SPM (Follow the Sun Software Process Model). Our paper describes how we built the FTS-SPM and draws recommendations for software organizations interested in practicing FTS.

Keywords - Follow the Sun; Global Software Engineering; Software process; Time zone management; Virtual teams.

I. INTRODUCTION

Follow the Sun (FTS) development is an alternative for global software environments when trying to manage problems related to temporal distance. Its main purpose is to reduce software development life cycle duration or time-to-market [1]. However, while the FTS concept looks promising in theory, it appears to be difficult in practice. It was observed that there is a great interest from the software industry in practicing FTS, but the lack of theoretical studies combined with software processes, models and practices make its adoption difficult [3]. In addition, many software organizations have attempted to implement FTS, but have abandoned it after some point because of this difficulty [2].

In this paper, we propose a software process model to support FTS development in global software projects. This process was called FTS-SPM (Follow the Sun Software Process Model). FTS-SPM was built based on the results from prior work [4] [5] [6] [7] and based on the design validation method conducted with experts from Lero - The Irish Software Engineering Research Centre (Ireland) and MuNDDoS research group (Brazil).

II. THEORETICAL BACKGROUND

A. Follow the Sun Development

In Global Software Engineering (GSE), team members are distributed in different places, countries or even continents. In some cases, these teams may be from the same organizations. In other cases, there is collaboration between teams from different organizations [8].

Follow the Sun (FTS) is a special case of GSE. It is applied in the context of GSE to take advantage of the temporal distance between several development sites located in different time zones. FTS is uniquely focused on speed of development [1]. It is applied to software projects when a software product needs to be developed quickly and the cost is irrelevant to the client. As team members are distributed across multiple time zones, organizations can develop software twenty-four hours continuously [3].

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 transition process from one site to another [6]. Handoffs are performed on a daily basis to present a status update and to pass on unfinished tasks (project source) from one site to another [2].

B. Related Work

Hess and Audy [9] proposed a software process for handoffs to alleviate difficulties faced by teams during the development phase of FTS projects. Findings from this study show that is possible to reduce development difficulties in FTS using the proposed process. Their process is based on Composite Persona (CP) and 24hr Design and Development concepts.

Richardson et al. [8] developed a software process called Global Teaming (GT). This process includes specific practices and sub-practices which detail specific recommendations for organizations that are implementing GSE. The main contribution of this study is a supporting mechanism for the implementation of global software projects.

Denny et al. [10] 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 a process called CPro that addresses several of the operational issues related to the 24HrKF environment.

Yap [11] also discusses agile methods, but with a different purpose. Her study describes the use of XP (Extreme Programming) to develop a globally distributed, around-the-clock software development project. In Yap’s study, a programming team was distributed across three sites and they used collective ownership of code. At the end of her study, the author concludes that XP works for a globally distributed group.
performing around-the-clock continuous development with a shared codebase.

Studies performed by Carmel, Espinosa, and Dubinsky [12] and Carmel, Dubinsky, and Espinosa [13] discuss mainly FTS definition, characteristics, and challenges. The first study provides a conceptual foundation and a formal definition of FTS. The second study, presents the details of the FTS concept and the outcomes of a first quasi-experiment designed to measure the speed of software work on FTS mode.

III. PRIOR WORK

We conducted some prior work in FTS before carrying out the study presented in this paper [4] [5] [6] [7]. We present an overview of this prior work, focusing on the aspects relevant to the software process model. We have categorized our studies into 2 phases: Phase 1 - Exploratory studies and Phase 2 - Case studies.

A. Phase 1 – Exploratory studies

Kroll et al. [4] conducted a mapping study of the literature in GSE to identify best practices for FTS development. They limited their study to identify practices conducted in GSE and at the same time recommended for FTS. Although not described in the literature, the evidence demonstrates that FTS is carried out by software engineers, but only in part. Nine best practices and key aspects for FTS implementation were presented in this paper.

Kroll et al. [5] extends the study published by Kroll et al. [4]. This study provides new information about FTS best practices and challenges. They substantially extend the empirical evaluation of FTS which was conducted in that previous study. As a result, they identified 36 best practices and 17 challenges for FTS implementation.

B. Phase 2 – Case studies

Kroll et al. [6] conducted a case study at Infosys Technologies in Bangalore, India. This study examined the feasibility and outcomes of FTS. Infosys’ experts considered some best practices reported in the literature [4] [5] to design a software process for FTS. This study presents details of software practices, presents solutions performed to overcome the challenges when developing a software application in FTS mode and highlights eight lessons learned. The authors conclude this study by showing that FTS works for GSE projects with some evidence that FTS can be used to compress duration.

Kroll et al. [7] report how handoffs management should be performed in FTS development. They present an experience report describing handoffs development and management in a FTS software project. The results describe the participants’ perception about software engineering activities performed, challenges faced and solutions performed to minimize these challenges. They also highlight management elements for handoffs.

IV. RESEARCH METHODOLOGY

We proposed the initial software process model for FTS, called FTS-SPM (Follow the Sun Software Process Model) version 1, based on the results from prior work (see Figure 1).

Subsequently, we built a second and third version of the FTS-SPM, with a view to improving the initial software process model design. This was done by collecting input from research experts, and this design validation was carried out using expert input from those in the Lero and MuNDDos research groups.

We had the opportunity to discuss the initial proposal 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, we collected data by notetaking from feedback provided by research experts. Based on these data, we propose the second version of our FTS-SPM.

Following the validation design planning, we presented the second version of the FTS-SPM at Lero workshops. We collected data during the workshops also by notetaking from feedback and further discussion with research experts. These data were also discussed in parallel with experts from the MuNDDos research group. Based on all data collected during the validation design, we proposed the third and final version of the FTS-SPM.

V. SOFTWARE PROCESS MODEL FOR FOLLOW THE SUN

A. The Design Validation

The FTS-SPM first version is presented in Figure 1. Through in-depth analysis of results from prior work (Section 3), significant themes directly correlating with best practices and lessons learned emerged. To make sense, these themes were synthesized as sub processes (SP) in the proposed FTS-SPM.

The proposed FTS-SPM version 1 comprised six sub processes: SP01: Team Setup, SP02: Project Planning, SP03: Communication Protocol, SP04: Cultural Training, SP05: Task Allocation, and SP06: Handoff Meeting. The sequence flow (arrows) between sub processes shows in which sequence each sub process is developed.

Figure 2, which is an evolution of the FTS-SPM version 1, shows modified sequence flows between sub processes. In addition, an initial and a final state were added to the FTS-SPM. We also changed SP06’s name and included arrows to show how the information moves through sub processes. SP06 was called Handoff Sessions on the FTS-SPM second version.

We presented the second version of the FTS-SPM at the Lero workshops and questions about the sequence flow between
SP05 and SP06 emerged. However, the sequence flow between sub processes was considered inadequate by some experts from Lero. Following recommendations from Lero researchers, particularly those belonging to the Process Quality Research Group, we introduced new changes to the FTS-SPM version 2. These changes were made between SP05 and SP06 to support FTS characteristics. All changes in the process model were also supported by experts from the MuNDDos research group.

B. The FTS-SPM Overview

The final version of the FTS-SPM (shown in Figure 3), comprises six sub processes: SP01: Team Setup, SP02: Project Planning, SP03: Communication Protocol, SP04: Cultural Training, SP05: Task Allocation and SP06: Handoff Sessions.

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 which 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.

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 described during 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. We show this sub process in our FTS-SPM to represent 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 is end of the shift 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.

VI. DISCUSSION

The proposed FTS-SPM comprises six sub processes. All sub processes were presented the second version of the FTS-SPM at the Lero workshops. However, questions about the sequence flow between SP05 and SP06 emerged. However, the sequence flow is also implemented in global software projects.
It makes sense, because FTS is a subset of GSE [1]. However, some characteristics of the FTS approach make its sub processes more difficult to implement. For example, in the FTS approach, team members work on the same tasks and daily handoffs are a strong characteristic of the FTS projects [1]. Thus, at the beginning and the end of the day there is a handoff to transfer unfinished or new tasks. In a traditional global project handoffs also happen, but with less frequency. In the absence of a structured handoff procedure, defects may occur due to lack of understanding of the current state of the task [10].

As shown in Figure 3, sub processes are related to each other. Thus, activities from a particular sub process can support one or more sub processes (e.g. activities from SP03, Communication protocol, supports the activities from SP06 – Handoff sessions). This is an important characteristic of software process models. Besides this, software process models help to better understand the software development process and present the possibility of discussing process improvements [8].

Through the analysis of the FTS-SPM, we highlight some recommendations for software organizations interested in practicing FTS. These recommendations aim to improve successful implementation of FTS: a) Project planning (SP02) should be performed after defining Team Setup (SP01). In some cases, to allow global teams to work in FTS mode, team members need to be re-allocated within different time zones. This makes time zones manageable; b) Communication protocol (SP03) is defined according to the team characteristics. Thus, cultural and temporal diversity in FTS projects requires more attention before selecting collaborative technologies for the project; c) An imbalance in team members’ experience can be solved developing Cultural Training (SP04). This sub process is also used to define standards, rules and strategies to improve software product development.

Finally, we highlight the relevance of a software process model to bring theory closer to practice. Organizations can benefit from the FTS-SPM to establish their own process for FTS.

VII. IMPLICATIONS AND CONCLUDING REMARKS

Our findings propose a software process model for FTS implementation, the FTS-SPM (Follow the Sun Software Process Model). Specifically, our qualitative evidence, gathered through prior work in FTS and design validation method, shows: 1) How best practices and lessons learned contribute to define sub processes; 2) How sub processes are related and in which sequence flow the process should be carried out; 3) How specific sub process status, SP05: Task allocation promotes SP06: Handoff sessions to determine the sequence flow to SP05 or to determine the beginning or the end of the day (final state).

Our work has practical implications for organizations that want to adopt FTS. In order to make FTS work effectively, an organization should pay attention not only technical issues such as adopting rich collaborative technologies, but also to better allocate team members in different continents and time zones.

It is useful for improving coordination across development sites.

While different experience levels can make the transfer of knowledge between team members difficult, managers should consider it as an opportunity for spontaneous learning across team members. Moreover, an effective team can better collaborate if team members keep working together on other projects. Finally, we highlight the need for more work to identify how each sub process should be developed and maintained over the project. Other studies are being developed to fill this gap.

ACKNOWLEDGMENT

The first author is supported by the PDTI program, financed by Dell Computers of Brazil Ltd. (Law 8.248/91) and the second author is partially supported by the Science Foundation Ireland grant 10/CE/I1855 to Lero (www.lero.ie).

REFERENCES


