FTSProc: a Process to Alleviate the Challenges of Projects that Use the Follow-the-Sun Strategy

Estevão Ricardo Hess
Computer Science School
PUCRS
Porto Alegre, Brazil
estevao.hess@acad.pucrs.br

Jorge Luis Nicolas Audy
Computer Science School
PUCRS
Porto Alegre, Brazil
audy@pucrs.br

***Abstract* — Searching for competitive advantages as low coast and productivity gains, organizations choose to distribute their software development to other countries with more affordable production costs. Increasingly, projects are being developed in geographically distributed environments, featuring the distributed software development. However, the challenges inherent in this software development environment are significant. Among these challenges is the time zone difference, which can also be tackled as an advantage, through the use of the follow-the-sun development. However, the follow-the-sun strategy presents some challenges, mainly alongside the handoffs. Therefore, this experimental research focuses to present a development process to alleviate the challenges found in project that uses this strategy, focusing in the development phase from the SDLC. Yet, it performs an experiment to evaluate the created process' efficiency. In this experimental process it was found evidences the created process actually alleviate the challenges found in the follow-the-sun strategy.**

Keywords - Global Software Development; Development Process; Follow-the-Sun.

I. INTRODUCTION

Global software development (GSD) is becoming a trend for companies that aim to keep competitive in the software development industry. GSD is also referenced as Global Software Engineering (GSE) or Distributed Software Development (DSD), and can be defined as software development with teams spread among different geographical locations [1]. It is characterized when one or more individuals involved in the project are physically distant from another [2]. In GSD one of the main characteristics is the time zone differences between development centers [3].

According to several studies [4, 5, 6] the time zone difference is difficult to manage. However, this difference can also be used as an advantage and not only as a disadvantage [7, 4, 8, 9, 10, 6]. In this sense, emerge the Follow-the-Sun (FTS) concept of.

The FTS approach utilizes distributed team members spread across time zones to achieve a single project outcome [11]. The main objective of FTS is to reduce the time-to-market in GSD environments [12].

FTS is an important research area. However, it is relatively understudied within Software Engineering [7]. The success cases in the industry using FTS are insufficient [10]. Carmel, Espinosa and Dubinsky [7] claim that there is few documented success cases in industry. Thus, aiming alleviate the challenges presented in the FTS strategy, this paper shows a proposal process to be used during the development phase of the SDLC. This process focuses in the handoff process between the team that finishes its day work and the team that is beginning its day. The Process was named FTSProc and it was created based in the Composite Personae presented by [15] and the process called 24hr Design and Development, presented by [13]. It also uses the Test-driven development (TDD) technique. Using the created process, this research also presents an experiment used to validate the proposed process. The findings during the experimental process brought evidences that a project that uses the FTSProc is more efficient then a project that does not use it.

To achieve these objectives, this paper is structured as follow: in the Section 2 it is presented the related works used during this research. In Section 3, it is presented the proposed process, named FTSProc. In the Section 4, the whole experimental process is shown, including the definition, objective, execution and a result discussion. Finally, the Section 5 presents the conclusions gathered during this study, including its limitations and future works.

II. RELATED WORK

The literature lacks studies related to the FTS strategy. The publications that present ways to use the FTS strategy are still scarce. However, some studies that deal with a theme similar to the theme showed in this study, i.e., ways to alleviate challenges during the work handoff are described below.

The work proposed by Lindemann et al. [13] shows ways to speed up a project development. To accomplish this, the authors distributed teams across different time zones, and made use of the FTS strategy. They created a handoff process from one site to another. This process consisted of allocating thirty minutes (the team finishing the shift and that is starting) to prepare information to be used during the handoff. At this simultaneously work moment, all the artifacts are delivered, as well as any relevant information to continue the work. This communication is done synchronously, using conference calls. Right after that, the team that started the work day, held a brainstorm, where the current work state is discussed. Based on the work that still needs be done, tasks are allocated to all resources within the team. Towards the end of the day, this process is repeated. This cycle ends when all the requirements are all developed.

The work published by Taweel et al. [14] presents the results of an experiment to evaluate the feasibility of using a sequential process of collaborative software engineering for distributed environments in different time zones. In this experiment was developed a calculator with simple functions. The project was divided into three phases: set-up, where the work to be developed was presented with all the requirements for all teams, along with the distribution of work and the deadline for completion; execution, which occurred the implementation using distributed teams; finishing, where the data gathered from the experiment were collected. The evaluated process was based on sending e-mails between the teams with the current status of the project, containing all information relating to the work. The study shows that, although dealing only with simple tasks, the results demonstrate the feasibility of such kind of process.

The work presented by Denny et al. [15] introduces the concept of Composite Personae (CP). This concept shows how distributed teams can work as one virtual team. For that, the authors state that it is important to have a cohesive team, spread across different time zones. Thus, the work can be passed from one site to another, and the same is continued. All work is based on handoff, where a team finishes its work day and another begins. However, some problems may occur during this transfer. Therefore, this paper shows a simple way to handoff the work. This transition is based on stand-up meetings, coming from the Scrum methodology. Reaching the end of a work day, developers must add their results to the code repository and fill out an automated form called handoff tool. This form should answer three basic questions of a stand-up meeting:

i. What have you done since last meeting?

ii. What are you planning to do until next meeting?

iii. Is there any impediments or blocker?

After completing this information, the work is considered delivered to the next team. The next site begins its work day collecting the information provided by the previous site and defining what should be done, using as main reference, the answers to the questions *i, ii* and *iii*. This work highlights the importance of having staff equivalent in all distributed sites. This equivalence is not related to the number of resources at each site, but in deliverability and troubleshooting capabilities.

Denny et al. [16] present a process of knowledge transfer, created especially for the knowledge factory concept usage in distributed environments. This process was created based on the Personal Software Process (PSP) [24]. This process is designed to facilitate the knowledge transfer from one team to another at the end of each day (handoff). This work also shows some ways to facilitate the work understanding among distributed teams. One of these ways is through the Test-Driven Development technique (TDD). According to the authors, TDD indicates the use of automated unit testing for defect reduction and quality control. In this technique, the test cases are written in order to validate that all requirements are implemented correctly. The test cases become a documented record of understanding the requirement and the solution to achieve them [16].

The main difference between the related works and this study is when and how the work transfer should be performed. The proposed process is focused exclusively on the development phase of SDLC, since according to [17], it is not recommended to use the FTS strategy in one single way in all phases of the SDLC. While the related works are not focused on a single phase. Another important difference is the work proposed by [14] where the tasks that each distributed center will develop are defined *a priori*, instead of treating the entire team as a single virtual team. Thus, the development center that starts the work does not continue the work where it was left off from previous site, but only develop different features in parallel.

III. FTSProc

FTSProc was the name given to the proposed process. It aims to mitigate the challenges of coordination, synchronization and communication during the handoffs in the development phase of SDLC. In this sense, the main objectives of this process are:

a. When a team starts a working day (shift), it should simply have the perception of the work that must be developed and the work already done by previous development center.

b. Avoid the needed for synchronous communication between distributed teams.

c. Ensure that the handoff from a development center to another occurs without problems, and that the work can be continued from the point where the previous development center left off.

This kind of processes is still insufficient in the literature. However, some studies show that this kind of process should be "light" [16, 14], it means that should not cause an overhead on a typical work day [16].

The proposed process is based on the Composite Persona (CP) presented by [15] and the process called 24hr Design and Development, presented by [13].

Regardless the FTSProc acts only during the development phase, some pre-conditions from the previous phases (requirements definition and design) support the process:

1. Requirements definition: this phase has as output artifact, the documentation with the requirements of the system to be developed. For the proper functioning of the process it is important that requirements are defined as specifically as possible [18], preferably using the concept of User Stories [19], which splits a requirement in small features to reduce the complexity of each task [13, 15], that usually are developed in a single work day. It is important that the User Stories have well-defined acceptance criteria. It facilitates the understanding of the requirements and according to [14], is critical that the whole development team has full understanding of the work to be performed.

2. Design: the artifacts that this phase will produce are directly related on how the features will be implemented. The diagrams needed for the system understanding, as well as the class and activities diagrams are some output examples in this phase. Furthermore, based on acceptance criteria of each requirement from the previous phase, unit tests should be created, and then use the Test-Driven development technique (TDD). The TDD technique usage is still related to maintaining a documented understanding of the requirement and the solution to be used to develop them [16,18]. Yet, according to [20] before the implementation begins, the TDD can act as part of the specification and, upon the application completion, the TDD becomes the knowledge of how the application was developed [20].

In addition to these pre-conditions, during the process two artifacts developed for FTSProc will be used, as follows:

Artifact 1:

Hand-off form: represents the current work state and should be filled with information about the work. All necessary information is contained in this artifact. This information is required to the next shift start from the point where the previous team left off.

Artifact 2:

Unit test Report: all unit tests that are not covered will be in this report. The importance of this artifact is to assist on planning a work day (shift), as can be seen in step 2 of FTSProc.

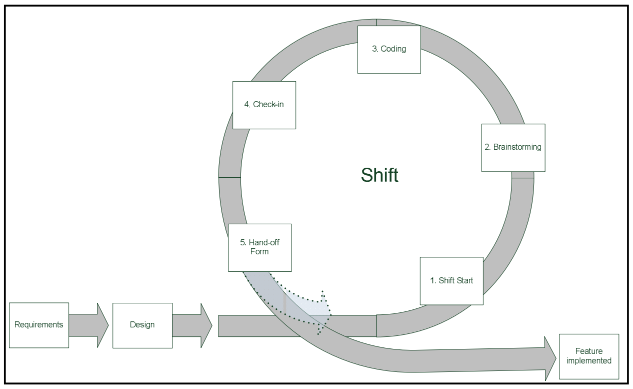

Figure 1. FTSProc: Proposed Process

As shown in Figure 1, the development phase starts at this point. Steps 1, 2, 3, 4 and 5 illustrated in this figure represent a single shift (a work day) for each development team. This process is iterative and these five steps will be repeated every shift, for each distributed development team [13, 18]. Each step can be described as follows:

1. Shift Start:

This state represents the beginning of a shift for each team. The following steps are part of this state:

i. The development center that is starting its shift downloads the latest source code.

ii. Generates a report with the covered tests and not covered ones. If a unit test is "passing", it means that this acceptance criterion is already covered, and it is not necessary to work on it anymore. This report is represented by the FTSProc Artifact 2.

iii. Generates a report with all information provided by the team who worked on the previous shift. This report is based on stand-up meetings, coming from the Scrum methodology [15, 18]. Each developer whom worked on each shift must complete this form (Artifact A). Therefore, this report is composed by all the information given by all developers.

2. Brainstorming:

After reviewing the information provided by the previous site, the team that is starting its shift should do a planning meeting to assign the tasks (daily schedule). This assignment should consider the report with information regarding the previous shift, as well as the result from the unit tests report provided by the previous site [15, 13, 16]. After performing this planning, this step of the process is finished. At this point, all the developers who are starting a shift, already know the point where the previous team left off and how the work should be continued.

3. Coding:

This step represents the requirements implementation phase, following the assignments agreed during the brainstorm. At this stage, the team focuses on the feature development. It is the longest stage of the process because it is where project development is actually done. When the work is finished, every developer must follow to the next state to continue the process.

4. Check-in:

After finishing the implementation, each team member must perform the check-in of the work done during the day, providing all necessary information to the next team to continue the work where it was left off. After performing the check-in and ensure that the latest source code is in the repository, this step is finished.

5. Hand-off form:

Towards the end of the day, each team member must take time to fill out the hand-off form (Artifact 1 of the FTSProc), with all the necessary information to the next site. This form is based on the stand-up meetings format, coming from the scrum and will be used to formalize the hand-off. The following information should be added [14, 15, 18, 13]:

i. What has been done during the last period?

ii. How the work should be continued?

iii. Is there any obstacle blocking the team?

iv. What unit tests have been covered during this shift?

This state represents the end of a shift. At this point, new acceptance criteria are covered by the work done, the latest source code is in the repository and the documentation required for the next team that will start its work is available.

Once all activities carried out in the process, the handoff is completed [7, 12, 14,13].

These steps are repeated until all acceptance criteria are met, i.e., all tests created during the design phase (TDD) are covered. After achieving all the acceptance criteria ends the development phase and the whole feature have been implemented [13, 18].

IV. The Experimental Process

After finalizing the transfer process proposal, it was necessary to carry out an experiment to assess whether the FTSProc met its goals. For this, the objective of the experiment was to compare two projects, one called Adhoc and another named FTSProc. The project Adhoc was used as control, because it did not use a defined process, only made use of the FTS strategy. While the project FTSProc used the proposed process. The requirements to be implemented were the same in both projects. These requirements were composed of a simple mathematical system. This scenario was chosen because of its ease, and probably known to all participants. The experiment was divided into five steps, as follow:

1. Definition

In this phase was used the Goal Question Metric (GQM) approach [21] which defines objectives (conceptual level) to establish questions (operational level) and then identify metrics (quantitative level). In the experiment context this approach assists the objectives definition phase [22].

The overall objective of this experiment is to investigate which approach is more efficient in projects that use the FTS approach: using the FTSProc or without use this process (as known as Adhoc) and thus, identify which approach allows delivering the highest number of requirements implemented in a given time interval.

To achieve the objective of this study sought to answer the following question: "Projects that uses the FTSProc has the same efficiency then adhoc projects carried out in a distributed environment?". The metric associated with this question corresponds to the efficiency of the method, calculated from the sum of the requirements correctly implemented by the participants in each of the two approaches. In this study were defined as correct requirement those with the following characteristics:

- Requirement developed in the Java language, accordingly to the system description, provided to the experiment participants;
- Each requirement has several acceptance criteria, where for the calculation criteria are:
    - All acceptance criteria are covered, the requirement is completely implemented;
    - Any acceptance criteria is not covered, the requirement is partially implemented;

2. Planning

During the planning phase, was defined the experiment participants' needed skills, how would be simulated the work shifts, the definition of hypotheses and definition of requirements to be implemented.

The Participants for this experimental process were Computer Science Master Students from PUCRS University, along with professionals from the PUCRS' technological park (TECNOPUC). We selected eight participants, and they were divided into two teams, which would hold the two projects: Adhoc e FTSProc. Within each of these teams, the four participants were re-divided, to simulate two different development centers. The Figure 2 bellow illustrates this division:

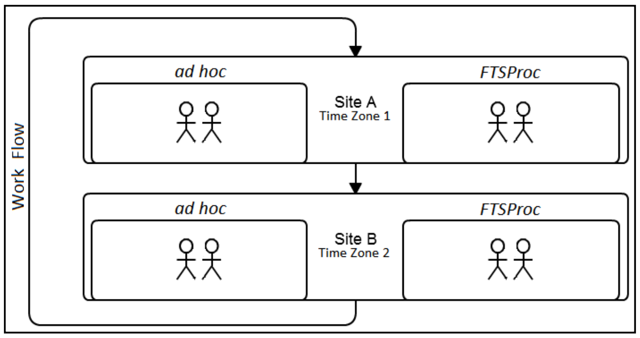

Figure 2. Team distribution

To perform the work shifts, we chose to simulate a time difference of more than 8 hours, i.e., no period of work concurrently. To do so, due the simplicity of the tasks to be performed, and the experience and availability of participants, each work shift (a day of work) were simulated in twenty minutes. During the experiment, it was performed two work shift for each team. It should be noted that these settings were used in the same way for both projects: Adhoc and FTSProc.

Regarding the experiment hypotheses, the following was defined:

- Null Hypothesis (H0): The efficiency of a project that uses the FTSProc is equal than a project developed in an adhoc way.
    - H0: $\sum \text{reqImpleFTSProc} = \sum \text{reqImpleAdHoc}$
- Alternative Hypothesis (H1): The efficiency of a project that uses the FTSProc is greater than a project developed in an adhoc way.
    - H1: $\sum \text{reqImpleFTSProc} > \sum \text{reqImpleAdHoc}$
- Alternative Hypothesis (H2): The efficiency of a project that uses the FTSProc is smaller than a project developed in an adhoc way.
    - H2: $\sum \text{reqImpleFTSProc} < \sum \text{reqImpleAdHoc}$

The requirements to be implemented were the same in both projects. These requirements were composed of a simple mathematical system. This system were composed by several small requirements, for example: sum of values, values subtraction, division, multiplication, factorial, square root, volume of a sphere, volume of a cube, area of a sphere, area of a cube, etc [13]. These requirements were described in detailed document provided to the participants during the experiment execution. This document also contained all the information related to the acceptance criteria for each requirement. Both projects received the same requirements document.

3. Execution

At this phase, the researcher provided the requirements that should be implemented, along with all the necessary instructions for the experiment execution. With all these materials, each team analyzed the requirements for 5 minutes and started the implementation. The time available to each project, adhoc and FTSProc, were 20 minutes for each shift.

At the end of the execution phase, the teams have provided all the source code developed during execution of the experiment along with the data of each handoff collected through the developed support tool. At this stage, no unforeseen happened and Table 1 and Table 2 shows the time spent to complete this step for each project:

- FTSProc project

TABLE I. NEEDED TIME FOR FTSPROC PROJECT.

Activity	Start time	End Time	Total Time
Requirements analysis	18:05	18:10	5 Minutes
Shift 1	18:12	18:32	20 Minutes
Shift 2	18:40	19:00	20 Minutes
Shift 3	19:06	19:26	20 Minutes
Shift 4	19:27	19:34	7 Minutes

- Adhoc project

TABLE II. NEEDED TIME FOR ADHOC PROJECT.

Activity	Start time	End Time	Total Time
Requirements analysis	17:52	17:57	5 Minutes
Shift 1	18:02	18:22	20 Minutes
Shift 2	18:22	18:42	20 Minutes
Shift 3	18:45	19:05	20 Minutes
Shift 4	19:06	19:26	20 Minutes

4. Data analysis

As there was no evidence that the usage of the proposed process would result in a productivity gain, the main research method used was the experiment with quantitative approach. However, due to the number of participants not allow the use of a statistical analysis, the study was supplemented with a quantitative analysis to achieve the results of the experiment.

A. *Quantitative analysis*

The most important result coming from the experiment is directly related to the amount of requirements developed by the teams in each approach. This result will be used to verify the proposed hypothesis regarding the efficiency of the FTSProc. Table 3 shows the results obtained regarding the number of requirements correctly and partially implemented in each project.

TABLE III. IMPLEMENTED REQUIREMENTS.

Implemented requirements	*FTSProc*	*Adhoc*
Correctly	12	4
Partially	0	8
Not implemented	0	0
Total	12	12

Analyzing the results in Table 3, it can be seen that the efficiency of the team using FTSProc is greater than the team that carried out the project on an adhoc way. The team that used the FTSProc implemented a greater number of correct requirements than the adhoc project team. In addition, the team that used the FTSProc obtained a higher rate of work correctly done (amount of correct requirements *100/amount of defined requirements):

- The team that used the FTSProc delivered 100% (12) of the requirements correctly implemented, based in the 12 requirements.
- The team that did not use the FTSProce, i.e., the adhoc project had 33,3% (4) of the requirements correctly implemented and 66,6% (8) of the requirements partially implemented, based in the 12 requirements.

Importantly, as can be seen in Table 1, the team that used the FTSProc needed only three shifts and seven minutes of the fourth shift, to ensure that all requirements were implemented correctly. Meanwhile, another team used all four full shifts, to complete only 4 requirements correctly.

With these results it is seen that the correct amount of requirements delivered rate is higher when using the FTSProc for running distributed projects using the FTS strategy. These data provide evidence for accepting the alternative hypothesis H1 ("The efficiency of a project that uses the FTSProc is greater than a project developed in an adhoc way.").

B. *Qualitative Analysis*

At the end of the experiment execution, the Researcher requested to participants to respond a questionnaire regarding their perceptions about the method they used: FTSProc or adhoc. For each question applied to the two groups, we sought to compare the participants' perceptions in the two approaches. The results obtained in this step are shown in the following tables, where, for each question it is shown the count of positive and negative responses to each teams. Right after, it is presented an analysis of these results.

- The work handoff from one center to another occurred in an appropriate way?

	FTSProc	*Adhoc*
Yes	4	3
No	0	1
Total	4	4

- At the beginning of each shift, you could see directly how the work should be continued?

	FTSProc	*Adhoc*
Yes	4	0
No	0	4
Total	4	4

- Do you believe that the work handoff from a development center to another led to a significant work overhead?

	FTSProc	*Adhoc*
Yes	0	1
No	4	3
Total	4	4

It is noted that, in the participants' perception, the work handoff from one center to another happened in an appropriate way in both approaches. However, it is clear that the identification of the point where the work should be continued was not directly in the adhoc team. But in the team that used the FTSProc, this identification was facilitated. This result is consistent with one of the objectives of the proposed process, which is to facilitate the identification of the point where the work should be continued. Finally, we can identify that in the perception of the participants, the overhead caused by the FTSProc usage was not significant. These findings are consistent with the literature, which shows that this kind of process should be "light", i.e., cannot cause a large increase in the workload (overhead) in a typical working day [16, 14].

Additionally, other questions were applied to all participants for the identification of positives, negatives and improvement opportunities for both approaches.

The strengths in relation to the adhoc method cited by the participants of the experiment include the low complexity of the tasks, since the requirements of the experiment were created for this purpose. Another positive point mentioned was how the participants attempted to show the other team for the current state of work. To do that, the participants used comments in the source code and in the code repository for each check-in.

For the negative points mentioned related to the adhoc method showed especially problems that the lack of a process can cause. Among these problems, lack of awareness of where the work had stopped on the previous shift and how this should be continued were the most cited by participants. It should be noted that these points are consistent with literature. The main problems highlighted by the literature are related to the challenges of coordination, synchronization and communication, especially during the work handoff from one development center to another [7, 9, 10]. The FTSProc try to alleviate these challenges.

Suggestions for improvements listed by the participants that use the adhoc show the necessity of using a standard process for the work handoff, like the FTSProc. Yet, the TDD technique usage was cited as a possible facilitator for the synchronization between the teams.

The strengths cited by the participants who used the FTSProc are directly related to how the process was created, i.e., the usage of clear requirements, TDD usage and the three questions used as a basis for the work handoff.

As expected at the beginning of the experiment, there were few negative points raised by the FTSProc project participants, if compared with the adhoc project and are not related to work synchronization or coordination. One raised point is related to the vocabulary. As the three questions were answered with text there might happen understanding and interpretation problems between the different development centers. This was the only negative point raised by the participants in the FTSProc project.

Just a suggestion for improving FTSProc was cited and is not directly related to the process, but the used tools. This suggestion refers to the use of an automatic work check-in tool. Thus, if some of the participants of the previous shift did not do the check-in, this tool would avoid the next shift to start without the latest source code in the repository.

5. Results Discussion

After presenting the results and the evidence for the confirmation of the H1 hypothesis, which shows that the efficiency of a project that uses the FTSProc is greater than a project developed in an adhoc way, this section presents other factors that corroborate to this result.

Analyzing the qualitative results, we found that the advantage of the FTSProc lies in the fact that the FTSProc team realizes clearly and quickly how the work should be continued. Thus, the time for this identification is smaller than the adhoc team, resulting in a greater time for requirements development during each working day (shift). This advantage is due two main factors related to the FTSProc: usage of TDD and the main three questions that the process proposed.

The usage of the TDD is effective since the adhoc project implemented 8 requirements partially, i.e., some acceptance criteria were not covered. In a real project, these problems would be identified later, only in a testing phase. Due the adhoc team did not use TDD, they had to implement the feature as well as create tests for that. In this sense, the lack of this technique also affects the time required for implementation. While the FTSProc team had unit tests to ensure the requirements were properly implemented, the adhoc team invested much of its time creating and running the tests. Another factor that the TDD technique helps is on the work progress perception, since verify the tests that are already covered and which ones still need to be worked facilitates the understanding of the work progress, i.e., how close are the project to finalize all the requirements. For this reason, during the experiment, in just seven minutes of the fourth shift, the FTSProc team found that all the work had been done, and there was no more work to be continued.

The three main questions that the process proposed (*(i)* What have you done since last meeting?, *(ii)* What are you planning to do until next meeting?, *(iii)* Is there any impediments or blocker?) also assisted teams on identifying where the work should continue. Easily, the participants checked the answers provided by the previous center and quickly knew what had been implemented. After reading this, the point to start the development was confirmed with the unit testing execution and, based on that, the tasks were distributed inside the team. Thus, in a short time, the team started the development work.

While the FTSProc team quickly and directly identified how the work should be continued, in contrast, the adhoc team did not have this perception. To identify the point that the work should be continued, the adhoc team needed to analyze the source code created or modified by the previous team, which spends a long time. Since there was no time to work simultaneously for the work handoff and there was no support tool, the adhoc team used comments in the source code and in the repository check-ins to report what had been accomplished. However, since there was not a defined structure to pass this information and not even an obligation, were not all participants that used this feature and, the ones who used, did not follow a standardized pattern. At the end of the experiment, these comments were cited as positive points. Once again, this is consistent with what the FTSProc proposed to facilitate the work handoff, using the three questions.

Analyzing the drawbacks raised out by the teams, it is possible to note that the adhoc project indicated several problems. Among these problems, most of them are generated by the lack of a standard handoff process form one center to another. It is possible to note that several issues raised by the adhoc team were not identified in the FTSProc team. This fact is a further evidence that the created process, indeed facilitates the work handoff. Yet, when analyzing the improvements suggestions to the team that did not use the FTSProc was verified that there are indications to use techniques and practices that the FTSProc already uses, such as: unit tests, test-driven development (TDD), a standard way to report what has been done and the definition of a task synchronization system.

Analyzing the negative points rose in the FTSProc team it is noted that there are just some few points, and they are not related to the process itself, but to difficulties found in any software project, even those that are not developed in a distributed environment. One of the points raised is the fact that there is not a standardized vocabulary. In this case, there may happen interpretation problems, since the questions are answered through free text. Another point raised shows that problems generated affect the team as a whole. In this regard, it is noted that this problem is related to the fact that the work is continued shift after shift, i.e., a defects generated and not fixed moves to the next site.

This qualitative analysis allowed us to identify that the FTSProc had several strengths, such as the usage of the three questions that are the base of this process as well as the usage of TDD. Moreover, the analysis of the experiment results and the questionnaire applied to the participants showed evidences of the greater efficiency of projects that use FTSProc then the adhoc projects.

However, since this is an exploratory study on a new research theme, it cannot be considered a final work in this area. Therefore, despite the favorable results found, during this research some limitations were identified, such as the lack of an experiment with a greater number of participants enabling a statistical analysis. Also, during this study were identified some future work, like the needed to expand this research to other phases of the software development life cycle. The next session presents detailed information about on these limitations and future studies.

V. Conclusions

In this research was presented a handoff process proposal to projects that use the FTS strategy. The objectives of the proposed process in this work are focused on reducing the challenges posed by FTS strategy. To evaluate the proposed process, an experiment was conducted, which demonstrated that the FTSProc actually achieves its goals, i.e., alleviates the challenges present in projects that use the strategy FTS.

A. *Contributions*

The contributions of this study are located in two main dimensions: for the theory and for the market:

For the theory area, the major contribution of this research was the creation of a process for the work handoff during the development phase. As shown, the process was proposed with data from the literature. After running the experiment and analyzed its results, we found that the process was indeed effective. This point is another important contribution to the theory, since while the literature does not present a specific process for the development phase, this study proposed a process, performed an experiment, and pointed evidences of the effectiveness of the proposed process.

For the market, currently searching for competitive advantages such as cost reduction and productivity gains, companies are using offshore operations. Thus, this work can contribute 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 the this phase of SDLC. Therefore, the proposed process is a starting point for organizations that work in distributed environment could begin the usage of the FTS strategy. Also, besides the process, the developed support tool is also considered another contribution to the practical point of view, since it offers a huge number of important features.

B. *Limitation*

The first limitation identified in this research is related to the process evaluation used. In the experimental process, due some schedule constraints, the number of participants able to join the experiment was low (8 people). Therefore, it was an impediment to use statistical methods to verify the hypothesis, opting then by a qualitative interpretation to analyze the results. This interpretation presented evidences of the greater efficiency of the project that used the FTSProc, but not allowed to obtain conclusions with a significant confidence level, what could be achieve through experiments with statistical analysis of the results

Moreover, as this work limitation, is considered the specific generalization of the experiment due the fact that the scope of the project was fictitious and created by the researcher. Also, there are the issues related to application of an experimental research method, as the subjective influence of the researcher or the participants in the results.

The support tool created for the FTSProc is still a prototype and therefore can be considered one of the limitations of this research. Despite being implemented all requirements planned, before using the tool in a real environment, it would be necessary to review aspects of usability, performance and reliability of the tool.

C. Future Studies

The experiment results interpretation showed evidences favorable to projects that use the FTSProc strategy, demonstrating that it is more efficient for distributed projects that use the FTS strategy then adhoc projects, commonly used in the companies. This way, with the purpose of substantiating the evidence presented by experimental method, as a future study, it is suggested the experiment replication with a larger number of participants to evaluate the FTSProc, which allows a significant statistical validation for obtaining conclusions on the hypotheses.

Yet, it is important to carry out a study case to evaluate the use of the process created in a real environment, using a project and a real team in a company that uses the distributed software development. Thus, it will be possible to verify the behavior of the process in this kind of environment. So, it is possible to prove the results found in this study, through a controlled experiment, are equivalent in a real environment.

Finally, studies aiming to expand this process to other phases of the SDLC are relevant. Thus, other phases might be contemplated with a process to facilitate the FTS strategy usage. Focusing on specific phases based on different approaches, at the end might be possible to create a process comprised by several sub-processes, which contemplate all phases of SDLC. Thus, the entire software project could be accomplished using the FTS strategy and therefore, reducing the time of construction in all phases of a project.

Acknowledgment

The authors are funded by the PDTI program, financed by Dell Computers of Brazil Ltd. (Law 8.248/91).

References

[1] 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.

[2] Prikladnicki , R. , Audy J., “Process models in the practice of distributed software development: A systematic review of the literature”. Inf. Softw. Technol. Pp. 779-791, August 2010.

[3] Lane, M.; Ågerfalk P., “Suitability of Particular Software Development Roles to Global Software Development”. 3rd IEEE International Conference on Global Software Engineering, 2008.

[4] Holmstrom, H., Conchuir, E. O., 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 (ICGSE '06). IEEE Computer Society, Washington, DC, USA, pp. 3-11, 2006.

[5] Herbsleb, J. D., Grinter, R. E., „Architectures, coordination, and distance: Conway's law and beyond”. Software, IEEE , vol.16, no.5, pp. 63-70, 1999.

[6] Treinen, J. J., Miller-Frost, S. L. „Following the Sun: Case Studies in Global Software Development.: IBM Systems Journal”, Volume 45, Number 4, 2006.

[7] Carmel, E., Espinosa, A., Dubinsky, Y. “Follow The Sun Software Development: New Perspectives, Conceptual Foundation, and Exploratory Field Study”. 42nd Hawaii International Conference on System Sciences, Proceedings, 2009.

[8] Lings B., Lundell B., Ågerfalk P. J., Fitzgerald B., ”A reference model for successful Distributed Development of Software Systems”. ICGSE 2007. IEEE International Conference on, pp. 130-139, 2007.

[9] Setamanit, S. O., Wakeland, W., Raffo, D., “Improving Global Software Development Project Performance Using Simulation”. Management of Engineering and Technology, Portland International Center, pp. 2458-2466, 2007.

[10] Solingen, van 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), 2010 5th IEEE International Conference on , pp.165-174, 2010.

[11] A. Cameron, “A Novel Approach to Distributed Concurrent Software Development using a Follow-the- Sun Technique”, Unpublished EDS working paper, 2004.

[12] Carmel, E.; Espinosa, J. Alberto; Dubinsky, Y. "Follow the Sun Workflow in Global Software Development”, Journal of Management Information Systems Vol. 27 No. 1, pp. 17 – 38, 2010.

[13] 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.

[14] Taweel, Adel, Brereton, Pearl: Developing Software Across Time Zones: An Exploratory Empirical Study. Informatica (Slovenia) 26(3): 2002.

[15] Denny, Nathan, Mani, Shivram, Sheshu Nadella, Ravi, Swaminathan, Manish, Samdal, Jamie: Hybrid Offshoring: Composite Personae and Evolving Collaboration Technologies. IRMJ 21(1): 89-104, 2008.

[16] Denny, Nathan, Crk, Igor, Nadella, Ravi Sheshu and Gupta, Amar, Agile Software Processes for the 24-Hour Knowledge Factory Environment (February 27, 2009).

[17] Carmel, E.; Espinosa, A., “I'm Working While They're Sleeping: Time Zone Separation Challenges and Solutions.” Estados Unidos: Nedder Stream Press, 2011. 188 p.

[18] 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, pp. 147-161, 2009.

[19] Haugen, N.C.; , "An empirical study of using planning poker for user story estimation," Agile Conference, 2006 , vol., no., pp.9 pp.-34, 23-28 July 2006.

[20] Meszaros, Gerard, XUnit Test Patterns: Refactoring Test Code, Prentice Hall PTR, Upper Saddle River, NJ, 2006.

[21] Basili, V. R.; Caldiera, G.; Rombach, H. D. “The Goal Question Metric Approach: Encyclopedia of Software Engineering”. Nova Iorque: Wiley- Interscience, 1994, 578 p.

[22] Wohlin, C.; Runeson, P.; Höst, M.; Ohlsson, M. C.; Regnell, B.; Wesslén, A. “Experimentation in Software Engineering: An introduction”. Kluwer Academic Publishers, 2000, 204 p.

[23] A. Gupta, S. Seshasai, and R. Aron, "Research Commentary: Toward the 24-Hour Knowledge Factory - A Prognosis of Practice and a Call for Concerted Research" (November 19, 2006). Eller College Management Working Paper No. 1038-06 Available at SSRN: http://ssrn.com/abstract=946012

[24] Humphrey, W. S. (1995) "Introducing the personal software process", Annals Of Software Engineering volume 1, 1995.