Um Modelo de Processo de Software para o Desenvolvimento *Follow the Sun*

Josiane Kroll, Jorge L. N. Audy

Faculdade de Informática (FACIN)
Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
90.619-900 – Porto Alegre – RS – Brasil

josiane.kroll@acad.pucrs.br, audy@pucrs.br

Abstract. *Follow the Sun development is seen as a potential software development strategy for global software projects. Follow the Sun can help with reducing the software development life cycle duration. However, while this concept looks promising in theory, it appears to be difficult to put into practice. Many software companies have tried to implement Follow the Sun, but have abandoned it after a while because of this difficulty to put it into practice. In this paper, we present a software process model for Follow the Sun adoption in global software projects. The process model is called the FTS-SPM (Follow the Sun Software Process Model). The FTS-SPM comprises 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.*

Resumo. *O desenvolvimento Follow the Sun é visto como uma potencial estratégia para o desenvolvimento de projetos de software globais. Esse tipo de desenvolvimento de software ajuda a reduzir o ciclo de vida de desenvolvimento de software. Entretanto, enquanto esse conceito é descrito como promissor na teoria, ele é difícil de ser aplicado em prática. Muitas empresas de software tentaram aplicar o desenvolvimento Follow the Sun, mas depois abandonaram pela dificuldade de colocá-lo em prática. Neste, artigo, é apresentado um modelo de processo de software para a adoção do desenvolvimento Follow the Sun em projetos de software globais. O modelo foi chamado FTS-SPM (Follow the Sun Software Process Model). Ele é composto por seis subprocessos e vinte e uma boas práticas de software. A sua adoção contribui para aumentar a probabilidade de sucesso com a implementação do desenvolvimento Follow the Sun em empresas de software e também contribui para enfrentar os diferentes desafios do desenvolvimento global de software.*

1. Introdução

O desenvolvimento global de software ou *Global Software Development (GSD)* permite que novas abordagens sejam introduzidas para o desenvolvimento de software, na qual recursos, investimentos, usuários e equipes de desenvolvimento podem estar distribuídos em diferentes locais e fusos horários [Sato, Huzita e Leal 2011]. Além disso, o GSD também permite que empresas criem subsidiárias em outros países e configurem a distribuição de processos de software nas vinte e quatro horas de um dia

de trabalho [Casey, Deshpande e Richardson 2008]. Essa estratégia de desenvolvimento de software é chamada de *Follow the Sun* (FTS).

O desenvolvimento FTS ainda está nos estados iniciais de pesquisa e são encontrados muitos desafios relacionados à comunicação, coordenação e cultura para a sua prática [Hess e Audy 2012]. Segundo Carmel e Espinosa (2011), há poucos casos de sucesso documentados com a aplicação do desenvolvimento FTS na indústria de software e faltam estudos teóricos sobre as suas principais características.

Tem-se observado um grande interesse da indústria de software na prática do desenvolvimento FTS. Prikladnicki e Carmel (2013) reportam que empresas de software estão ansiosas para implementar o desenvolvimento FTS. No entanto, a falta de estudos teóricos combinados com a definição de práticas de software, processos e modelos faz a implementação do FTS difícil. Assim, poucos projetos de GSD conseguem perceber os reais benefícios do FTS. O desenvolvimento FTS visa maximizar a velocidade do desenvolvimento de software, mas ainda não há um suporte empírico rigoroso para essa prática [Colazo e Fang 2010]. Adicionalmente, se o desenvolvimento FTS não é aplicado corretamente, ele resulta em falhas e aumenta os custos do projeto.

O principal objetivo desse artigo é propor um modelo de processo de software para a adoção do desenvolvimento FTS em projetos de GSD. O modelo proposto é chamado de FTS-SPM (*Follow the Sun Software Process Model*). Ele é composto de seis subprocessos que são implementados com base em 21 boas práticas de software. O modelo foi construído baseado em resultados obtidos com os métodos de revisão sistemática da literatura, estudo de caso e painel com especialistas. Neste artigo, são apresentados os passos realizados para a construção do modelo, o detalhamento do modelo e as contribuições obtidas.

1.1 Motivação e Relevância

Na Engenharia de Software a descrição de modelos, processos e práticas são importantes porque elas oferecem prescrições (como o desenvolvimento de software deve ser executado) ou descrições (como o desenvolvimento de software é feito na prática). A construção de processos e a discussão dos seus sub-processos contribui para um melhor entendimento da lacuna entre a prática e a teoria [Pressman 2010].

Empresas de desenvolvimento de software precisam de apoio para implementar seus projetos de GSD [Richardson et al. 2010]. Dessa forma, o GSD como uma área de pesquisa dentro da Engenharia de Software também adota métodos, processos, e práticas para solucionar problemas e apoiar o desenvolvimento de projetos de software.

A principal relevância desse trabalho na área de pesquisa da Engenharia de Software é a definição de um modelo de processo de software para a adoção do desenvolvimento de software FTS em projetos de GSD. Um modelo de processo de software contribui para manter o nível de consistência e qualidade em produtos e serviços produzidos por diferentes pessoas [Pfleeger 2004]. Além disso, a definição de um modelo pode colaborar para o uso com sucesso do desenvolvimento FTS na indústria de software. Empresas interessadas em adotar o FTS para o desenvolvimento de projetos podem fazer uso do modelo proposto. Além disso, organizações podem se beneficiar do resultados desse trabalho obtendo mais informações sobre o desenvolvimento FTS.

A definição de um modelo para FTS também contribuirá para o desenvolvimento de novas teorias. Novos estudos sobre FTS podem ser desenvolvidos a partir desse estudo.

2. Desenvolvimento de Software *Follow the Sun*

Follow the Sun (FTS) é uma estratégia de desenvolvimento de software aplicada para o contexto de projetos de GSD [Carmel e Espinosa 2011]. Ela se caracteriza pelo desenvolvimento de software vinte e quatro horas contínuas com equipes geograficamente distribuídas [Visser e Solingen 2009].

Em FTS, os membros de uma equipe são distribuídos de maneira que quando um local de produção encerra o seu dia de trabalho, membros da equipe localizados em um diferente local de produção e fuso horário assumem as tarefas iniciando a sua jornada de trabalho [Treinen e Miller-Frost 2006]. O desenvolvimento FTS considera as diferenças de tempo e a comunicação contínua para definir turnos e horários de trabalho, com reuniões regulares e relatórios de progresso, a fim de identificar e aplicar estratégias adequadas do início ao final do projeto [Carmel e Espinosa 2011].

A produção diária gerada por uma equipe FTS é enviada para o próximo local de produção, o qual está em um diferente fuso horário para dar continuidade ao trabalho [Visser e Solingen 2009]. A continuidade do trabalho envolve ciclos de troca de tarefas entre as equipes que são separados por *handoffs* diários. O *handoff* é um termo utilizado na literatura para designar o processo de transição de tarefas entre equipes de uma unidade de trabalho para o próximo local de produção [Carmel e Espinosa 2011].

O principal objetivo da estratégia FTS é reduzir o ciclo de desenvolvimento do projeto ou *time-to-market*. Segundo Carmel e Espinosa (2011), a estratégia FTS não oferece outras vantagens além da redução do tempo de desenvolvimento do projeto. Dessa forma, a estratégia FTS se diferencia do GSD tradicional quando quatro critérios são satisfeitos:

1. O principal objetivo é aumentar a velocidade do desenvolvimento reduzindo o tempo de duração do projeto. Esse critério distingue o FTS das outras práticas e configurações do GSD tradicional. O FTS não oferece outras vantagens sobre outras configurações, além da velocidade.
2. Os locais de produção são distantes e estão em diferentes fusos horários. Esse critério diferencia o FTS do GSD tradicional pela aceleração das táticas de produção.
3. Em qualquer ponto no tempo apenas um local de produção possui o produto final (inacabado). Esse critério diferencia o FTS a partir das configurações convencionais globais em que vários locais de produção podem possuir diferentes partes do produto.
4. *Handoffs* são realizados diariamente no final de cada turno de trabalho. Este critério diferencia o FTS das configurações convencionais globais que minimizam as dependências de *handoffs* entre os locais de produção.

Em FTS, são identificados centros que podem especializar-se em fases bem definidas do ciclo de desenvolvimento de software. Há uma tendência do FTS concentrar-se nas fases de implementação permitindo que artefatos possam ser

transferidos entre locais de desenvolvimento. No entanto, devido à diversidade entre estes locais, podem existir muitas dificuldades relacionadas à comunicação, coordenação e cultura para a aplicação do FTS.

Na literatura, muitas vezes FTS também é referenciado por meio de conceitos similares tais como, *24-hour development model, 24-Hour Knowledge Factory Paradigm (24HrKF)* e *round-the-clock*. Embora esses termos sejam usados como sinônimos na literatura, eles possuem diferentes definições. O conceito de FTS é focado na velocidade e na redução da duração do projeto, enquanto o conceito *round-the-clock* e outros, abordam o desenvolvimento de software 24 horas, mas executando operações em todos os turnos de trabalho. Todos esses conceitos usam as diferenças de fusos horários para projetar turnos de trabalho, mas para diferentes propósitos e com diferentes tipos de tarefas [Carmel e Espinosa 2011].

3. Metodologia de Pesquisa

O desenvolvimento desse estudo foi organizado em três fases de pesquisa: (1) Exploratória, (2) Desenvolvimento e (3) Avaliação e Evolução, como mostra a Figura 1. Cada fase é descrita a seguir.

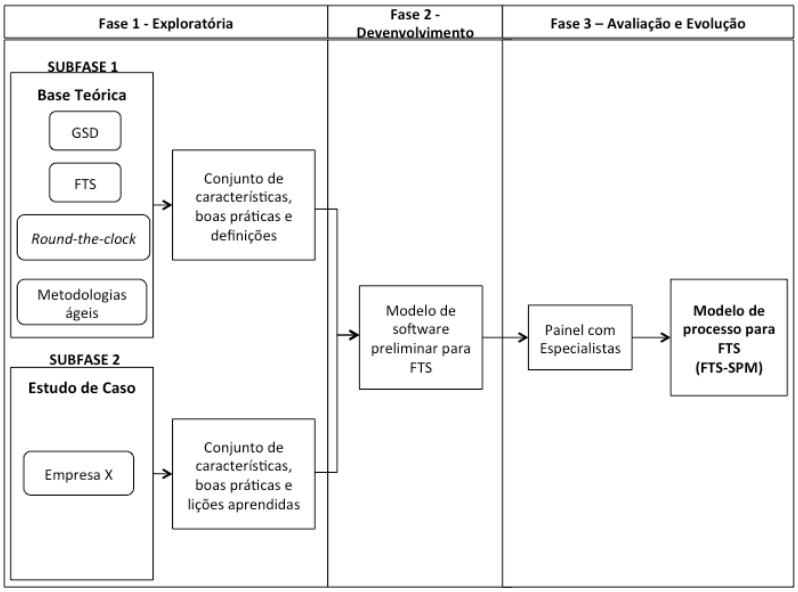

Figura 1. Desenho de pesquisa.

• *Fase 1 – Exploratória:* essa fase foi dividida em duas subfases. Na subfase 1, foi conduzida uma revisão da base teórica, a qual envolveu a condução de uma revisão sistemática da literatura (RSL) em FTS. O objetivo dessa subfase foi construir um conjunto de características, boas práticas e definições para a preparação de um modelo de processo para FTS. No total foram identificados17 desafios e 36 boas práticas para FTS [Kroll et al. 2013].

Na subfase 2 foi conduzido um estudo de caso na empresa *Infosys Technologies* localizada em Bangalore, Índia. O estudo conduzido na *Infosys* visou examinar a viabilidade e os resultados obtidos com a aplicação da abordagem FTS para o desenvolvimento de um projeto de software interno da empresa. Este estudo focou na fase de desenvolvimento de um projeto de software. A equipe alocada para o projeto foi

composta de seis desenvolvedores e um gerente de projeto. Dois desenvolvedores no México, dois desenvolvedores na Austrália e outros dois desenvolvedores e o gerente do projeto na Índia. O desenvolvimento do projeto foi estimado para aproximadamente um mês de duração. Durante este estudo foram coletados dados empíricos para a construção do modelo [Kroll e Audy 2012] [Kroll et al. 2013].

• *Fase 2 - Desenvolvimento:* nesta fase foi proposto um modelo de processo de software preliminar para FTS. Estudos conduzidos na fase 1 forneceram um melhor entendimento das características do desenvolvimento FTS. Além disso, os resultados obtidos na fase 1 forneceram informações sobre boas práticas de software e lições aprendidas para a construção do modelo. Nessa fase também foi feita a avaliação do *design* do modelo preliminar por especialistas do grupo de pesquisa Lero (*The Irish Software Research Centre*) [Kroll, Richardson e Audy 2014].

• *Fase 3 – Avaliação e Evolução:* especialistas no processo de avaliação ajudaram a refinar o modelo e faze-lo aplicável na indústria de software. Vinte especialistas em GSD foram entrevistados para avaliar o modelo. Os participantes forneceram detalhes da sua experiência na prática para suportar as boas práticas e subprocessos incluídos no modelo. Duas questões foram escritas baseadas em cada boa prática, cada subprocesso e considerando o modelo como um todo. Dessa forma, o questionário foi composto de 64 questões. Também foi criado um plano para distribuir as questões entre os especialistas. Cada questão foi respondida 3 vezes por diferentes especialistas. Cada especialista respondeu 8 ou 10 questões. No final foram coletadas 192 respostas para avaliar o modelo. Como resultado, foi obtido um modelo de processo de software para FTS bem definido, com um conjunto de recomendações para a aplicação do modelo e a importância de cada de elemento no processo para o desenvolvimento FTS.

4. O Modelo FTS-SPM

O modelo de software proposto foi chamado FTS-SPM (*Follow the Sun Software Process Model*). Ele é composto por seis subprocessos como apresentado na Figura 2: *SP01 – Configuração da equipe, SP02 – Planejamento do projeto, SP03 – Protocolo de comunicação, SP04 – Treinamento cultural, SP05 – Alocação de tarefas* e *SP06 – Sessões de handoff*.

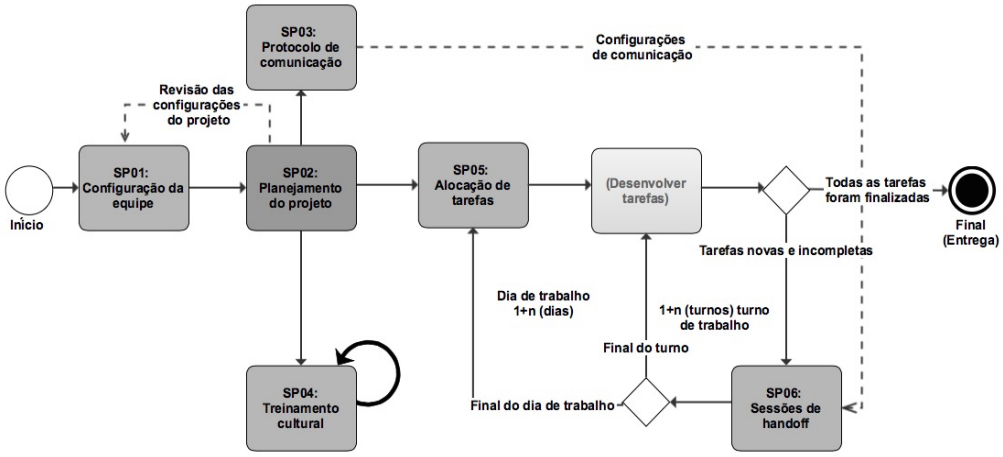

Figura 2. FTS-SPM: Modelo de processo de software para FTS.

O modelo FTS-SPM tem um estado inicial e outro final. O estado inicial dá início ao processo com o SP01 e o estado final finaliza o processo quando todas as tarefas já foram finalizadas e nesse ponto há a entrega do software.

O SP01 inicia o processo. Ele visa identificar sites e equipes disponíveis para o projeto. É importante verificar se há restrições de custo, recursos humanos ou de escopo em cada site. Essas restrições e outras relacionadas aos objetivos do projeto podem ser consideradas para definir prioridades na seleção de locais apropriados para o desenvolvimento do projeto.

SP02 é iniciado após o SP01. O SP01 fornece a informação necessária para o desenvolvimento do plano do projeto. SP02 interage com o SP01 e vice-versa para incluir e excluir novos sites no projeto.

SP03, SP04 e SP05 são iniciados em paralelo com o SP02. O SP03 define os recursos de comunicação e a agenda para a comunicação síncrona entre os sites. O gerente de projeto pode sugerir tecnologias ou ferramentas já utilizadas em outros projetos. SP04 desenvolve sessões de treinamento cultural para estabelecer a confiança entre membros da equipe. O SP04 pode ser desenvolvido várias vezes durante o projeto para reestabelecer a confiança entre os membros da equipe (flecha com um laço).

No início e no final de cada dia de trabalho, o SP05 é desenvolvido. Ele fornece tarefas para o dia de trabalho. O SP06 inicia na sequencia do SP05. O SP06 objetiva receber e transferir tarefas em progresso, novas tarefas e o *status* do projeto para o próximo site que irá iniciar o dia de trabalho. O SP06 é desenvolvido no início e no final de cada turno de trabalho.

O processo termina quanto todas a tarefas forem realizadas. O subprocesso "Desenvolver tarefas", é um subprocesso interno de cada organização. Cada organização define como ele será executado. Esse subprocesso no modelo FTS-SPM representa como ele está relacionado com os outros subprocessos.

No primeiro diamante, o processo finaliza se todas as tarefas estiverem finalizadas ou então SP06 inicia se houverem tarefas inacabadas. No segundo diamante, uma novo turno de trabalho inicia se é final do turno ou SP05 inicia se é final do dia de trabalho. As setas mostram o fluxo de sequência entre subprocessos. Uma seta adicional está incluída entre SP03 e SP06 para indicar a relação entre esses subprocessos. As configurações de comunicação definidas na SP03 são usadas em SP06 .

Os subprocessos são desenvolvidos com base em um conjunto de boas práticas de software. No total, o modelo inclui 21 boas práticas distribuídas nos seis subprocessos. As boas práticas de software foram incluído nos modelo baseadas em recomendações da literatura e lições aprendidas no estudo de caso.

O modelo está estruturado em 3 níveis. O nível 1 denota os seis subprocessos (SP01 até SP06), enquanto nível 2 corresponde ao conjunto de boas práticas incluídas nos subprocessos (BP01 até BP21). Adicionalmente, algumas boas práticas incluem recomendações que formam o nível 3 do modelo. Para cada boa prática incluída em um subprocesso, também é apresentada sua importância para o desenvolvimento FTS. Neste estudo, são apresentados os níveis 1 e 2 do modelo. O detalhamento do nível 3 do modelo é descrito por Kroll (2014). A tabela 1 apresenta as boas práticas incluídas nos subprocessos. Na sequencia são descritos os subprocessos e boas práticas.

Tabela 1. Subprocessos e boas práticas do modelo FTS-SPM.

Sigla do subprocesso (SP)	Nome do subprocesso (SP)	Boa prática (BP)	Nome da boa prática (BP)
SP01	Configuração da equipe	BP01	Ajuste das horas de trabalho para um bom *overlap*
		BP02	Distribuição dos membros da equipe em 2 ou 3 locais
SP02	Planejamento do projeto	BP03	Uso de metodologias ágeis para o gerenciamento do projeto
		BP04	Uso de abordagens de desenvolvimento de software incrementais
		BP05	Aplicação do FTS para as fases de teste e desenvolvimento
		BP06	Uso de padrões de código
SP03	Protocolo de comunicação	BP07	Uso de tecnologias para a troca diária do *status* do projeto
		BP08	Uso do compartilhando da área de trabalho do computador para a troca de conhecimento
		BP09	Calendário de sessões de *handoff* devem ser claramente definidas
		BP10	Uso de tecnologias de tempo real para compartilhar conhecimento
		BP11	*Wikis* e fóruns *on-line* para compartilhar conhecimento entre equipes
		BP12	Janela de tempo
		BP13	Uso de tecnologias corporativas para a interação entre equipes
		BP14	Modelos de e-mail e mensagens eletrônicas
SP04	Treinamento cultural	BP15	Reuniões entre equipes para construir confiança
		BP16	Treinamento de sensibilização cultural
SP05	Alocação de tarefas	BP17	CPro conceito
		BP18	Distribuição de tarefas por sequencia ou dependência
SP06	Sessões de *handoff*	BP19	*Daily stand-up meetings*
		BP20	*Handoffs* diários de 30 minutos de duração com cada local de desenvolvimento
		BP21	Uso de um FTP Server (ou repositório de dados) para a troca de código e documentos

4.1 Subprocesso: SP01 – Configuração da equipe

O SP01 objetiva identificar locais de desenvolvimento e alocar recursos humanos para o projeto. A configuração da equipe em um projeto FTS deve permitir a criação de um ciclo de desenvolvimento entre os locais de desenvolvimento. O SP01 inclui duas boas práticas, as quais são descritas a seguir.

4.1.1 BP01: Ajuste das horas de trabalhos para um bom overlap

O gerenciamento de fusos horários é necessário para o ajuste das horas de trabalho da equipe para um bom *overlap* [Holmstrom et al. 2006]. A escolha dos locais de

produção para um bom *overlap* nem sempre é possível. Dessa forma, as diferenças de fusos horários se tornam gerenciáveis quando é possível ajustar as horas de trabalho da equipe.

4.1.2 BP02: Distribuição dos membros da equipe em 2 ou 3 locais

Equipe distribuídas em diferentes locais de desenvolvimento possuem diferentes habilidades de trabalho, experiências, culturas, infraestrutura de trabalho e horas de *overlap* entre locais de produção. Essas diferenças associadas as características do FTS podem resultar em falhas e aumento dos custos do projeto. Assim, a BP02 recomenda a distribuição dos membros da equipe em 2 ou 3 locais de desenvolvimento. Cabe ressaltar, que o desenvolvimento FTS requer pelo menos 2 locais de desenvolvimento. O estudo realizado por Solingen e Valkema (2010) descreve que não há ganhos significativos em termos de acurácia e velocidade de desenvolvimento com equipes distribuídas em mais de 3 locais de produção.

4.2 Subprocesso: SP02 – Planejamento do projeto

O planejamento do projeto no modelo FTS-SPM é realizado com base na configuração da equipe dada pelo SP01. O SP1 fornece os locais de desenvolvimento e recursos humanos disponíveis para o projeto. Dessa forma, o SP02 interage com o SP01 e vice-versa para a definição do plano do projeto. O SP02 faz parte do gerenciamento do projeto. Ele inclui 4 boas práticas, as quais são descritas a seguir.

4.2.1 BP03: Uso de metodologias ágeis para o gerenciamento do projeto

As metodologias ágeis visam se adaptar rapidamente as mudanças de um novo ambiente de desenvolvimento de software. Uma metodologia ágil enfatiza a comunicação e a colaboração em um processo interativo de desenvolvimento de software [Yap 2005]. Além disso, as metodologias ágeis são indicadas para cenários onde existe uma constante mudança de requisitos e os resultados devem ser entregues em pequenos espaços de tempo. As metodologias são mais flexíveis em relação aos tradicionais e isso é observado com um benefício o desenvolvimento FTS.

4.2.2 BP04: Uso de abordagens de desenvolvimento de software incrementais

Test-driven Development (TDD) é uma abordagem para o desenvolvimento de software de forma incremental. Nessa abordagem, unidades de software são desenvolvidas em pequenas partes.

Em TDD, um desenvolvedor implementa um pedaço de uma funcionalidade pela escrita do código usando o estilo *test-before-code*, onde ele escreve um teste funcional antes do relevante código ser inserido. O TDD não requer detalhes iniciais de *design*, uma vez que as unidades do programa são desenvolvidas incrementalmente [Gupta e Jalote 2007]. Essa abordagem contribui para o desenvolvimento FTS.

4.2.3 BP05: Aplicação do FTS para as fases de teste e desenvolvimento

Segundo Carmel e Espinosa (2011), evidências da indústria de software mostram que o desenvolvimento FTS é efetivo na redução da duração somente em algumas fases específicas do ciclo de vida de desenvolvimento de software. Estas incluem as fases de teste e desenvolvimento. Ainda segundo Carmel e Espinosa (2011), a fase de teste pode trabalhar bem em FTS, pois *handoffs* são estruturados, granulares e com o desenvolvimento de sessões de treinamento, problemas de comunicação, os quais

são muitos frequentes nesta fase podem ser minimizados. Hess e Audy (2012) também mostram que FTS é viável para a fase de desenvolvimento. Segundo esses autores, a aplicação de FTS pode reduzir o tempo gasto com o desenvolvimento.

4.2.4 BP06: Uso de padrões de código

No desenvolvimento FTS, a equipe que finalizou o seu dia trabalho transfere suas tarefas para outra equipe que está iniciando o seu dia de trabalho. A equipe que está iniciando o seu dia de trabalho está em um diferente local de produção e fuso horário. Uma equipe pode simplesmente assumir as tarefas da equipe anterior ou pode realizar algumas tarefas de transferência adicionais. No entanto, na maioria das vezes os membros da equipe irão continuar do ponto em que os membros anteriores pararam.

Para dar continuidade à tarefa, a equipe terá que executar atividades de verificação antes iniciar a tarefa. O uso de padrões de código é uma prática que permite que membros entendam e identifiquem claramente as mudanças feitas no código depois da última sessão de *handoff*. Além disso, o uso de padrões e inserção de comentários no código contribui para evitar o retrabalho [Taweel e Brereton 2006].

4.3 Subprocesso: SP03 – Protocolo de comunicação

A distância temporal que existe entre os locais de desenvolvimento reduz as oportunidades para a comunicação síncrona em projetos FTS [Carmel e Espinosa 2011]. Nesses casos onde as horas de *overlap* não permitem síncrona comunicação, a comunicação assíncrona pode ser adotada. O SP03 visa definir como a comunicação entre equipes será realizada durante o projeto. Ela inclui recursos de comunicação tais como, agenda para comunicação síncrona e assíncrona entre a equipe, ferramentas e tecnologias, e outras. SP03 inclui 8 boas práticas, as quais são descritas a seguir.

4.3.1 BP07: Uso de tecnologias para a troca diária do status do projeto

Ramesh e Dennis (2002) sugerem que a troca diária do *status* do projeto entre os membros da equipe. Essa prática pode ser realizada por meio de tecnologias tais como, telefone, vídeo conferência ou e-mail.

As chamadas telefônicas e as de vídeo conferência permitem a síncrona comunicação para que membros da equipe interajam em tempo real, minimizando mal entendidos que possam vir a ocorrer. Ambas as tecnologias podem ser usadas em conjunto com o e-mail ou outras tecnologias.

4.3.2 BP08: Uso compartilhando da área de trabalho do computador para a troca de conhecimento

O compartilhamento de telas ou da área de trabalho foi uma prática recomendada por Tang et al. (2011). O compartilhamento de telas entre os membros da equipe contribui para transferir e facilitar o entendimento das tarefas. A sua utilização torna fácil o entendimento das informações que estão sendo discutidas pela equipe.

4.3.3 BP09: Calendário de sessões de handoff devem ser claramente definidas

A criação de uma agenda de sessões de *handoff* é uma prática recomendada por Deshpande e Richardson (2009). Segundo esses autores, para uma boa comunicação entre as equipes distribuídas, uma agenda de sessões de *handoff* para que os membros da equipe possam interagir deve ser definida antes de o projeto iniciar.

4.3.4 BP10: Uso de tecnologias de tempo real para compartilhar conhecimento

Muitas tecnologias estão disponíveis para facilitar a troca de conhecimento entre as equipes. Tang et al. (2011) e Gupta et al. (2009) recomendam o uso de *webcams* e programas de mensagens instantâneas para facilitar a comunicação entre os membros da equipe que estão distribuídos em diferentes locais de produção.

4.3.5 BP11: Wikis e fóruns on-line para compartilhar conhecimento entre equipes

A divisão do conhecimento por meio das ferramentas *wiki* e fóruns *on-line* é uma prática de software recomendada por Gupta et al. (2012). Essa prática consiste no uso de um *wiki* interno como base de conhecimento, onde a equipe pode dividir seus problemas e soluções.

Fóruns *on-line* é outra ferramenta que pode funcionar como uma plataforma de comunicação da equipe. Ambos, *wiki* e fóruns *on-line* permitem compartilhar um informal conhecimento para ser armazenado em formato estruturado e acessado por indivíduos em diferentes locais.

4.3.6 BP12: Janela de tempo

A janela de tempo ou *time window* pode utilizada entre membros de uma equipe FTS. Essa prática visa minimizar conflitos na colaboração entre locais de produção, além de fornecer oportunidades para interações síncronas [Lings et al. 2007].

Durante a troca de turno de trabalho, a janela de tempo para interação reduz atrasos na comunicação que podem trazer problemas de coordenação entre os locais de produção. Segundo Tang et al. (2011) essa prática cria oportunidades para uma natural comunicação sem requerer prévio planejamento ou agendamento.

4.3.7 BP13: Uso de tecnologias corporativas para a interação entre equipes

A prática BP13 estabelece a disponibilização de tecnologias, tais como vídeo conferência, compartilhamento de telas e outros recursos corporativos para que membros da equipe possam participar de reuniões em casa. Segundo Tang et al. (2011) a implementação dessa prática permite janelas de interação mais flexíveis aumentando a conectividade entre a equipe.

4.3.8 *BP14: Modelos de e-mail e mensagens eletrônicas*

A construção de modelos de e-mail e mensagens eletrônicas pode ajudar a minimizar problemas de comunicação [Gorton, Hawryszkiewycz e Fung 2006]. Um diferente modelo de mensagem pode fornecer para cada tipo geral de mensagem um propósito, tais como, pedidos técnicos, não técnicos e pedidos referentes ao processo. Os modelos podem conter campos que ajudam a lembrar do tipo de informação que é tipicamente incluída em tal mensagem.

4.4 Subprocesso: SP04 – Treinamento cultural

Os projetos FTS possuem membros das equipes com diferentes culturas. A diversidade cultural torna difícil a coordenação e a comunicação entre as equipes. Dessa forma, o SP04 visa treinar pessoas em aspectos culturais e promover o conhecimento de diferentes culturas e valores religiosos das equipes. Além disso, o SP04 visa também construir confiança entre os membros das equipes.

4.4.1 BP15: Reuniões entre equipes para construir confiança

As reuniões presenciais entre os membros da equipe é uma prática usada estabelecer ou reestabelecer a confiança, se a mesma estiver baixado no decorrer do projeto. Mídias de rica comunicação as que usam a interação *face-to-face*, tentem a ser mais eficientes do que mídias tais como, telefone, e-mail ou *chat* [Setamanit, Wakeland e Raffo 2007].

4.4.2 BP16: Treinamento de sensibilização cultural

A prática BP16 consiste no desenvolvimento de sessões de treinamento cultural. Seu objetivo é desenvolver a conscientização cultural entre os membros da equipe. Ela deve ser realizada no início do projeto para que a equipe possa compreender e respeitar as diferenças de estilo de trabalho e horários, potencialmente eliminando essas questões.

As diferenças culturais são reduzidas com a sensibilização, o que evita que riscos como a ruptura de relações entre os membros da equipe distribuída possam vir a ocorrer [Treinen e Miller-Frost 2007].

4.5 Subprocesso: SP05 – Alocação de tarefas

A alocação de tarefas é difícil de ser realizada no desenvolvimento FTS. As tarefas devem ser alocadas para a equipe e não para membros individuais da equipe. A equipe trabalha nas mesmas tarefas até todas as tarefas serem completadas. Tarefas não finalizadas são transferidas para o próximo local de desenvolvimento durante as sessões de *handoff*. Dessa forma, o SP05 visa implementar a locação de tarefas. O SP05 é realizado no início de cada dia de trabalho. O gerente do projeto aloca tarefas para as 24 horas de um dia de trabalho. O SP05 incluí duas boas práticas, BP17: CPro conceito e BP18: Distribuição de tarefas por sequencia ou dependência.

4.5.1 BP17: CPro conceito

A alocação de tarefas baseada no CPro conceito é descrita por Gupta et al. (2012) e Denny et al. (2008). O CPro é um processo de software ágil que considera a distribuição de tarefas para CPs (*Composite Persona*). As tarefas são alocadas para CPs. Cada CP é formado por um membro da equipe de cada localização, onde a alocação ocorre verticalmente.

4.5.2 *BP18: Distribuição de tarefas por sequencia ou dependência*

A distribuição de tarefas por sequência ou dependência é recomendada por Taweel e Brereton (2006). Na distribuição por sequência, a tarefa, a qual é normalmente feita por uma pessoa, é dividida em duas ou mais pessoas localizadas em diferentes fusos horários. Isto é, uma pessoa transfere a tarefa para uma segunda pessoa localizada em um diferente fuso horário e esta segunda pessoa continuará o trabalho do ponto onde a primeira parou. Com essa prática, pode-se realizar um período de vinte e quatro horas para um único dia de trabalho.

A distribuição de tarefas por dependência é realizada em conjunto com a distribuição de tarefas por sequência. Nesse caso, uma tarefa é feita por uma pessoa localizada em um particular fuso horário e a tarefa dependente é subsequentemente realizada por outra pessoa localizada em um fuso horário diferente. Essas tarefas por ser, por exemplo, código e teste.

4.6 Subprocesso: SP06 – Sessões de *handoff*

Em projetos FTS, *handoffs* são realizados diariamente no início e no final de cada dia de trabalho em cada local de produção. O desenvolvimento de *handoffs* diários requer um grande esforço de coordenação, comunicação e colaboração de toda a equipe envolvida no projeto. O SP06 é implementado no FTS-SPM para a realização da transição de tarefas não finalizadas e do *status* do projeto para o próximo local de desenvolvimento. O SP06 inclui 3 boas práticas.

4.6.1 BP19: Daily stand-up meetings

A BP19 recomenda a utilização de *daily stand-up meetings*. As *daily stand-up meetings* são uma prática oriunda da metodologia Scrum, a qual é utilizada para compartilhar a informação [Yap 2005]. Elas também podem ser utilizadas durante as sessões diárias de *handoff* entre membros de um mesmo local de desenvolvimento.

4.6.2 BP20: Handoffs diários de 30 minutos de duração com cada local de desenvolvimento

Hess e Audy (2012) recomendam que *handoffs* diários sejam realizados com 30 minutos de duração com cada local de desenvolvimento. De acordo com esses autores, 30 minutos são suficientes à transferência da tarefa e recapitulação do que deve ser feito pelos membros da equipe que estarão iniciando o dia de trabalho.

4.6.3 BP21: Uso de um FTP Server (ou repositório de dados) para a troca de código e documentos

A BP21 consiste no armazenamento de arquivos do projeto e códigos fonte produzidos pela equipe FTS. Estes podem ser armazenados em um comum repositório de dados. Os membros da equipe devem ser capazes de acessar todo o conteúdo do repositório tendo permissões e restrições gerenciadas a este.

A troca de código e documentos pode ser realizada usando um comum FTP Server ou repositório de dados disponibilizado pela própria organização [Taweel e Brereton 2006] [Ramesh e Dennis 2002].

4.7 Considerações finais sobre o modelo

O modelo FTS-SPM foi construído baseado em três fases definidas no desenho de pesquisa (veja Figura 1). Na fase 1 foram coletados dados para a preparação do modelo. Na fase 2 foi definido um modelo preliminar para FTS. O modelo preliminar foi então avaliado por meio do método painel com especialistas na fase 3. A adoção dos diferentes métodos de pesquisa contribuíram para construir um modelo de processo de software consistente e alinhado com a teoria e prática.

A aplicação da avaliação do *design* contribuiu para identificar melhorias no *design* modelo. Mudanças foram aplicadas para melhor representar o fluxo entre subprocessos e atender as características do FTS. Além disso, o painel com especialistas identificou a importância e relevância de cada prática e subprocesso incluído no modelo.

O modelo FTS-SPM pode ser aplicado por grandes empresas, como também por pequenas empresas. Entretanto, o modelo traz mais benefícios para empresas que já adotam o modelo de negócio *offshore insourcing*. Nesse modelo de negócio, a

organização desenvolve sua própria solução, mas com a subsidiária localizada em outro país [Moe, Smite e Hanssen 2012].

5. Conclusão e Trabalhos Futuros

Esse artigo apresentou um modelo de processo de software para FTS. O modelo FTS-SPM objetiva apoiar a implementação do desenvolvimento FTS em projetos de GSD. Empresas interessadas em adotar o desenvolvimento FTS podem fazer uso do modelo.

Do ponto de vista da indústria de software, o modelo FTS-SPM pode ser combinado com outras boas práticas de software e subprocessos. Todos os seis subprocessos que compõem o modelo FTS-SPM são essenciais para a implementação da abordagem FTS. Entretanto, organizações de software podem adicionar novos subprocessos e boas práticas no modelo. Os benefícios adquiridos com a avaliação do modelo aumentam a probabilidade de sucesso da adoção do modelo na prática.

O modelo FTS-SPM destaca a necessidade de pesquisa na área, principalmente para definir novas práticas para o desenvolvimento FTS. Estudos futuros visarão a aplicação do modelo na indústria de software. Além disso, observa-se a necessidade de investigar tipos de configuração de projetos de software que podem se beneficiar do desenvolvimento FTS.

Também poderão ser investigadas novas práticas de software para FTS ou melhorias poderão ser incluídas em subprocessos ou boas práticas do modelo. Uma vez que o modelo possui elementos chave para a implementação do desenvolvimento FTS, novos elementos podem ser identificados no futuro.

Durante a avaliação do modelo, especialistas também reportaram a necessidade de tecnologias e ferramentas para o suportar a implementação do modelo. Um dos trabalhos futuros será o desenvolvimento de ferramentas para o gerenciamento e suporte de subprocessos e das boas práticas sugeridas no modelo. Essas ferramentas poderão também demonstrar a relação entre várias práticas e como elas podem se beneficiar umas das outras.

Referências

Carmel, E., Espinosa, J. A. (2011) I'm Working While They're Sleeping: Time Zone Separation Challenges and Solutions, Kindle Edition, 188 p.

Casey, V., Deshpande, S., Richardson, I. (2008) “Outsourcing Software Development. The Remote Project Manager's Perspective”, The 2nd Global Sourcing Workshop - Services, Knowledge and Innovation, Val D'Isere, France.

Colazo, J., Fang, Y. L. (2010) “Following the Sun: Temporal Dispersion and Performance in Open Source Software Project Teams”, Journal of the Association for Information Systems, In Press.

Denny, N., Mani, S., Nadella, R. S., Swaminathan, M., Samdal, J. (2008) “Hybrid Offshoring: Composite Personae and Evolving Collaboration Technologies”, IRMJ 21(1): 89-104.

Deshpande, S., Richardson, I. (2009) “Management at the Outsourcing Destination - Global Software Development in India”, Fourth IEEE International Conference on Global Software Engineering (ICGSE '09),Washington, DC, USA, 217-225.

Gorton, I., Hawryszkiewycz, I., Fung, L. (1996) "Enabling software shift work with groupware: a case study", Proceedings of the Twenty-Ninth Hawaii International Conference on System Sciences, vol.3, pp.72-81.

Gupta, A., Mattarelli, E., Seshasai, S., Broschak, J. (2009) "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.

Gupta, A., Hu, L., Hedberg, T., Prendergast, C., Crk, I. (2012) "Creating the 24-Hour Knowledge Factory", Available at: http://ssrn.com/abstract=2004791.

Hess, E., Audy, J. L. N. (2012) "FTSProc: a Process to Alleviate the Challenges of Projects that Use the Follow-the-Sun Strategy", International Conference on Global Software Engineering (ICGSE'12), Porto Alegre, Brazil.

Holmstrom, H., Conchuir, E. O., Agerfalk, P. J., Fitzgerald, B. (2006) "Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance," Proceedings of the International conference on Global Software Engineering (ICGSE '06),Washington, DC, USA, pp. 3-11.

Kroll, J. (2014). A Software Process Model for Follow the Sun Development. PhD's thesis, Pontifical Catholic University of Rio Grande do Sul (PUCRS), Computer Science Department, Porto Alegre - RS, Brazil.

Kroll, J., Hashmi, S. I., Richardson, I., Audy, J. L. N. (2013) "A Systematic Literature Review of Best Practices and Challenges in Follow-the-Sun Software Development", In Global Software Engineering Workshops (ICGSEW), pp. 18-23.

Kroll, J., Audy, J. L. N. (2013) "Adopting Agile Methods for Follow-the-Sun Software Development", 19th Americas Conference on Information Systems (AMCIS), Chicago, USA.

Kroll, J., Prikladnicki, R., Audy, J. L. N., Carmel, E., Fernandez, J. (2013) "A Feasibility Study of Follow-the-sun Software Development for GSD Projects", International Conference on Software Engineering (SEKE), Boston, USA.

Kroll, J., Richardson, I., Audy, J. L. N. (2014) "Proposing a Software Process Model for Follow the Sun Development", 26th International Conference on Software Engineering & Knowledge (SEKE), Vancouver, CA.

Lings, B., Lundell, B., Ågerfalk, P. J., Fitzgerald, B. (2007) "A reference model for successful Distributed Development of Software Systems," International Conference on Global Software Engineering (ICGSE), pp. 130-139.

Moe, N. B., Smite, D., Hanssen, G. K. (2012) "From Offshore Outsourcing to Offshore Insourcing: Three Stories", Proceedings of the IEEE International Conference on Global Software Engineering (ICGSE), pp.1-10.

Pfleeger, S. L. (2004) Engenharia de Software: teoria e prática. 2ed. São Paulo - Prentice Hall.

Pressman, R. S. (2010) Software Engineering: a practitioner's approach". New York: McGraw Hill, 7th Edition, 928p.

Prikladnicki, R., Carmel, E. (2013) "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.

Ramesh, V., Dennis, A. (2002) "The object oriented team: Lessons for virtual teams from global software development", Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS), volume 1.

Richardson, I., Casey, V., Burton, J., McCaffrey, F. (2010) "Global Software Engineering: A Software Process Approach", Collaborative Software Engineering, Editors I. Mistrík, J. Grundy, A. Hoek, J. Whitehead, Publisher Springer, pp. 35-56.

Sato, G. Y., Huzita, E. H. M., Leal, G. C. L. (2011) "A process engine for outsourced software development," 6th Iberian Conference on Information Systems and Technologies (CISTI), pp.1-4.

Setamanit, S. O., Wakeland, W., Raffo D. (2007) "Improving Global Software Development Project Performance Using Simulation. Management of Engineering and Technology", Portland International Center, pp. 2458-2466.

Solingen, V. R., Valkema, M. (2010) "The Impact of Number of Sites in a Follow the Sun Setting on the Actual and Perceived Working Speed and Accuracy: A Controlled Experiment", 5th International Conference on Global Software Engineering (ICGSE), pp. 165- 174.

Tang, J. C., Zhao, C., Cao, X., Inkpen, K. (2011) "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, NY, USA, pp. 235-244.

Taweel, A., Brereton, P. (2006) "Modelling software development across time zones," Inf. Softw. Technol. 48, 1, pp.1-11.

Treinen, J. J., Miller-Frost, S. L. (2006) "Following the Sun: Case Studies in Global Software Development", IBM Systems Journal, Vol. 45, Number 4.

Visser, C., Solingen, V. R. (2009) "Selecting Locations for Follow-the Sun Software Development: Towards A Routing Model", Fourth International Conference on Global Software Engineering (ICGSE).

Yap, M. (2005) "Follow the sun: distributed extreme programming development," Agile Development Conference (ADC'05), pp. 218-224.