Especificação de Uma Abordagem para Desenvolvimento Distribuído de Software Utilizando SPEM

Gislaine Camila Lapasini Leal, Elisa Hatsue Moriya Huzita
Tania Fatima Calvi Tait e César Alberto da Silva

Universidade Estadual de Maringá, Maringá, Paraná, Brasil
gclleal2@uem.br,emhuzita,tait[@din.uem.br],cesaralberto91@hotmail.com
http://www.din.uem.br

Resumo O desenvolvimento distribuído de software tem proporcionado às empresas maior competitividade, como por exemplo: redução de custos; acesso à mão de obra e recursos; avanços na infraestrutura; vantagens de novos mercados; rápida formação de equipes virtuais; e, melhoria do time-to-market. Contudo, essa abordagem necessita de novas tecnologias, processos e métodos compatíveis. Assim, este trabalho apresenta a especificação de uma abordagem integrada de desenvolvimento e teste de software para apoiar o desenvolvimento com equipes distribuídas.

Palavras chave: processo de software, modelo, desenvolvimento distribuído

1 Introdução

O processo de software é definido como um conjunto ordenado de atividades para o gerenciamento, desenvolvimento e manutenção de software, e deve estar alinhado com as condições organizacionais [Fuggetta, 2000]

O Desenvolvimento Distribuído de Software (DDS) é uma abordagem para o desenvolvimento dos projetos de software que vem ao encontro das necessidades da globalização, que são: aumento de produtividade, melhoria de qualidade e redução de custos [Herbsleb et al., 2000]. Essa nova configuração adicionou ao desenvolvimento de software desafios relacionados às diferenças culturais, dispersões geográficas, coordenação e controle, comunicação e espírito de equipe, os quais intensificam alguns dos problemas enfrentados durante o ciclo de vida do projeto.

No cenário atual, vivenciamos uma crescente necessidade de software com qualidade. Diante disso, muitas formas de melhoria dos processos de desenvolvimento de software são utilizadas e diversos artefatos e ferramentas buscam melhorar o desenvolvimento, como a utilização de métodos formais para descrever especificações, *design* e testes [Abdurazik, 2000].

Segundo [Alves et al., 2008] a integração dos processos de desenvolvimento e teste pode trazer diversos benefícios, dentre eles: redução do custo de desenvolvimento; alcance de um alto nível de automação no desenvolvimento e geração de

casos de testes; facilidade para realizar alterações nos requisitos, devido à rastreabilidade provida; e, menores custos de manutenção. Desse modo, esse trabalho formaliza uma abordagem integrada de desenvolvimento e teste de software que apóia o desenvolvimento distribuído de projetos.

Este texto encontra-se estruturado em 5 Seções, além desta introdutória. Na Seção 2 são apresentados os conceitos relacionados a processos de software e técnicas de modelagem. Na Seção 3 é apresentada uma visão geral da abordagem integrada de desenvolvimento e teste de software. A formalização da abordagem é mostrada na Seção 4. Por fim, na Seção 5 são realizadas as considerações finais.

2 Processo de Desenvolvimento de Software

Um processo de software envolve uma série de etapas constituídas por um conjunto de atividades, métodos, práticas e tecnologias que são utilizadas desde o desenvolvimento até a manutenção do software e produtos relacionados.

Segundo [Berger, 2003], o uso de um processo padrão permite aos gerentes de projeto definir planos em conformidade com os padrões de qualidade e procedimentos da organização. Uma equipe de desenvolvimento que trabalha sem um processo bem definido acaba funcionando de maneira ad hoc, o que compromete o seu sucesso, criando uma situação insustentável. Por outro lado, organizações maduras empregando um processo bem definido podem desenvolver sistemas complexos de maneira consistente e previsível, independente de quem o produziu.

[Humphrey and Kellner, 1989] destacam que diversos fatores levam a padronização dos processos, mas as principais razões são: (i) possibilitar treinamento, gerenciamento, revisões e ferramentas de suporte; (ii) pode contribuir para a melhoria dos processos na organização a cada experiência de projeto; (iii) fornecer uma base estrutural para medição; (iv) definir processo demanda tempo e esforço o que torna impraticável novas definições de processo para cada projeto. Além disso, o processo deve possibilitar a melhoria da qualidade de serviço, de engenharia e de projeto; a diminuir custos por meio do aumento da previsibilidade e capacidade de mitigar riscos, bem como melhorar da eficiência e produtividade da organização.

2.1 Técnicas de Modelagem de Processo de Software

O modelo de um processo de software é a representação formal dos elementos e características envolvidos neste processo. Segundo [Ben-Shaul and Kaiser, 1994], um modelo de processo deve especificar os pré-requisitos e consequências de cada tarefa, bem como a sincronização entre as mesmas.

A modelagem de um processo de software deve ser conduzida de modo a possibilitar o entendimento e a padronização do processo. [Humphrey and Kellner, 1989] ressaltam que a representação do processo possibilita a comunicação e o entendimento efetivo do processo, facilita o reuso, apoia a evolução, facilita o gerenciamento do processo e é importante na avaliação, evolução e melhoria contínua

do processo. A estrutura de um modelo de processo de software é, geralmente, composta por quatro elementos:(i) ator: representa as entidades que executam um processo ou assumem um papel durante a execução de uma tarefa; (ii) papel: grupo de responsabilidades que executam uma atividade específica do processo de software; (iii) artefato: porção representativa de informação do produto que resulta de uma atividade e pode ser utilizado posteriormente como matéria-prima para gerar novos artefatos; (iv) atividade: consome artefatos (de entrada) e gera novos artefatos (de saída).

O SPEM (*Software Process Engineering Metamodel Specification*) é um metamodelo proposto pela OMG para a descrição de um processo concreto de desenvolvimento de software ou uma família relacionada de processos de desenvolvimento de software. Sua notação é baseada na UML e, por este motivo, oferece para a modelagem de processo os mesmos diagramas que são usados para modelar sistemas.

Os principais elementos estruturais do SPEM, para a descrição de um processo são: Produto de Trabalho, Tipo de Produto de Trabalho, Definição de Trabalho, Atividade, Papel, Executor e Passos. Um Produto de Trabalho, também chamado de Artefato, é toda informação produzida, consumida ou modificada por um processo. Um Tipo de Produto de Trabalho descreve uma categoria para os artefatos. A Definição de Trabalho é um tipo de operação que descreve o trabalho executado em um processo. Atividade é a principal subclasse da Definição de Trabalho e descreve uma parte de trabalho executado por um Papel no Processo. Uma Atividade pode ser constituída por um conjunto de elementos atômicos, os Passos. O Executor no Processo define um executor para um conjunto de Definições de Trabalho em um processo e apresenta uma especialização chamada Papel no Processo, que define responsabilidades sobre artefatos específicos e define os papéis que executam ou auxiliam em atividades específicas.

Em um processo, as atividades com temas comuns podem ser agrupadas em Disciplinas. A OMG define alguns elementos que auxiliam na definição de como o processo será executado. Sob a perspectiva da execução, um processo é visto como uma colaboração entre papéis para alcançar determinada meta ou objetivo. Para guiar esta execução pode-se considerar as restrições para ser a ordem em que as atividades devem ser executadas, chamadas de pré-condições. Fase é uma especialização da Definição de Trabalho tal que sua pré-condição define os critérios de entrada da fase, os critérios de saída da fase são suas metas, os chamados *milestones*. Ciclo de Vida é definido como uma sequência de Fases que buscam uma meta específica. Iteração é uma Definição de Trabalho com *milestones* menores.

3 Abordagem Integrada de Desenvolvimento e Teste de Software para Equipes Distribuídas

A abordagem proposta considera o processo de testes como um conjunto de tarefas à parte que pode ser aplicado ao longo do processo de desenvolvimento, instanciado em paralelo.

A abordagem de desenvolvimento utiliza a *Unified Modeling Language* (UML), que é uma linguagem de modelagem que permite aos desenvolvedores visualizar os produtos de seu trabalho em diagramas padronizados. O uso de especificações UML é atrativa por possuir informações relevantes para testes e estar em uso na academia e na maioria das indústrias de tecnologia, de modo que não há custos extras associados ao treinamento e integração no processo de desenvolvimento de uma aplicação [Hartmannm et al.,2004].

Na abordagem de teste é utilizado o Perfil UML 2.0 (U2TP), que define uma linguagem para projetar, visualizar, especificar, analisar, construir e documentar os artefatos de teste de sistemas [OMG, 2005]. A U2TP é uma linguagem de modelagem de testes que pode ser usada com tecnologias de componentes e linguagens orientadas a objeto, aplicadas em diversos domínios de aplicação e auxilia na documentação, entendimento e ratreabilidade dos artefatos de teste. Os artefatos de teste especificados pela abordagem, são os recomendados pela norma IEEE 829, que descreve um conjunto básico de documentos de teste de software.

Segundo a ISO/IEC 12207 a abordagem pode ser classificada como sendo um processo fundamental relacionado ao desenvolvimento. No entanto, apresenta alguns elementos de processos organizacionais e de apoio. Em relação aos processos organizacionais contempla aspectos de gerência. E, no que se refere aos processos de apoio engloba itens de verificação e validação, os quais fornecem subsídios para a qualidade do produto a ser desenvolvido.

A abordagem encontra-se estruturada em termos de objetivos, atividades, papéis e artefatos. A integração das unidades, independente da granularidade da distribuição (disciplina, componente, caso de uso, classe ou método), deve ser sempre acompanhada, podendo ocorrer dentro de cada Disciplina, como também, entre as Disciplinas.

3.1 Abordagem de Desenvolvimento

A Abordagem de Desenvolvimento é composta pelas seguites Disciplinas:

- **Planejamento:** Responsável pela configuração do processo de desenvolvimento e definição dos aspectos relacionados ao gerenciamento do projeto. Os papéis envolvidos são o Gerente de Projetos Global e o Gerente de Projetos Local. O artefato gerado é o plano de desenvolvimento de software (global e local), que fornece a linha básica de recursos e cronograma.
- **Requisitos:** Descreve como definir uma visão do sistema e traduzi-la em modelos. Como metas, tem-se: estabelecer e manter contato com os *stakeholders* sobre o que o sistema deve fazer, facilitar o entendimento do desenvolvedor sobre os requisitos através dos modelos gerados e delimitar o

escopo do sistema. Os papéis envolvidos nessa são o Engenheiro de Negócios e o Especificador. Os artefatos produzidos são: descrição textual do sistema, modelo do negócio (arquitetura inicial), o modelo de objetos de negócio, os diagramas de casos de uso e a descrição extendida de caso de uso.
- **Desenvolvimento:** Traduz os artefatos gerados na fase de Requisitos numa especificação que descreve como implementar o sistema. Os papéis envolvidos são o Arquiteto e o Projetista. Os artefatos produzidos são: Diagrama de Sequência, Diagrama de Comunicação, Diagrama de Classes e Descrição da Arquitetura.
- **Implementação:** Tem como metas a definição da organização do código fonte, a implementação das classes e objetos e a integração dos resultados alcançados pelas diversas equipes em sistema executável, se essa disciplina, também, for distribuída. O Desenvolvedor é o papel que desempenha as atividades dessa Disciplina. O artefato gerado é o código-fonte.

3.2 Abordagem de Teste

As Disciplinas que constituem a Abordagem de Teste são:

- **Planejamento:** Responsável pela configuração do processo de teste, trata da definição dos fatores relacionados à gerência. Os papéis envolvidos são o Gerente de Projetos Global e o Projetista de Testes.O artefato produzido é uma versão inicial do Plano de Testes.
- **Preparação:** Descreve o planejamento de todas as atividades envolvidas no teste de um software. O papel envolvido na execução dessa disciplina é o de Projetista de Teste. Essa Disciplina produz uma atualização do Plano de Teste contendo o planejamento para execução do teste, incluindo a abrangência, abordagem, recursos e cronograma das atividades de teste.
- **Projeto:** Detalha os aspectos técnicos a serem adotados na condução das atividades de teste. O papel envolvido é o Projetista de Teste. Esta disciplina produz como artefatos: Especificação de Projeto de Teste, Especificação de Casos de Teste e Especificação de Procedimentos de Teste.
- **Execução:** O objetivo desta Disciplina é a execução e registro das atividades de teste projetadas nas disciplinas anteriores. O papel envolvido nessa disciplina é o de Testador. Os artefatos gerados são: Diários de Teste, Relatório de Incidentes e Resumo de Teste.

4 Especificação da Abordagem Integrada

O modelo da Abordagem utiliza diagramas da UML e propostos pelo SPEM para modelar tantos os elementos estáticos como os dinâmicos do processo.

A visão geral da abordagem integrada, que mostra o fluxo de informações entre as Disciplinas, é apresentada no Diagrama de Atividades da Figura 1. Os artefatos gerados são passíveis de refinamento nas Disciplinas posteriores.

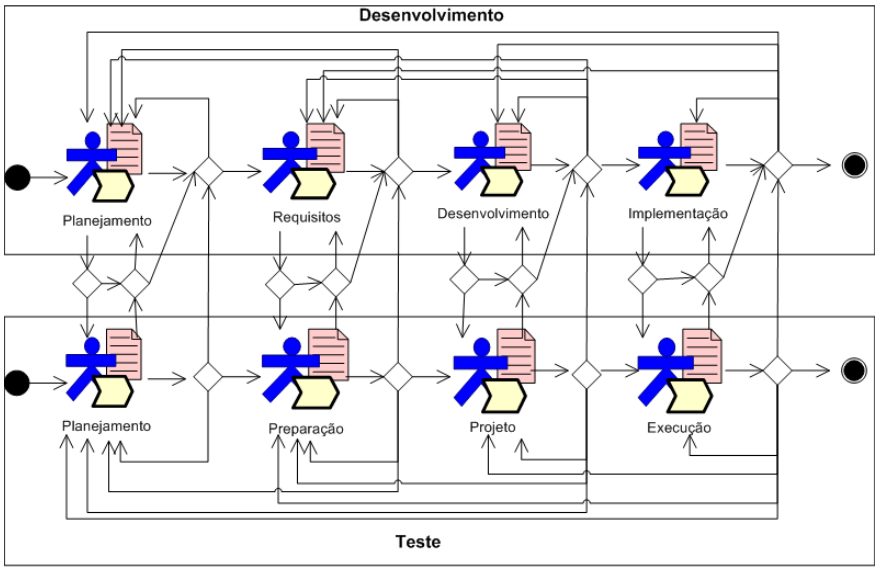

Figura 1. Diagrama de Atividades da Abordagem Integrada de Desenvolvimento e Teste de Software.

A Disciplina de Planejamento da Abordagem de Desenvolvimento será apresentada sob a perspectiva de Diagramas de Pacotes, Casos de Uso, Classes e Atividades.

A Figura 2 apresenta a Definição de Trabalho, mecanismo de divisão semântica para as atividades, que compõe a Disciplina Planejamento.

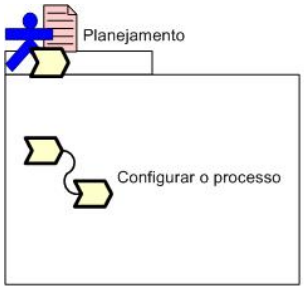

Figura 2. Diagrama de Pacotes da Disciplina Planejamento.

A Definição de Trabalho - Configurar Processo - é detalhada no Diagrama de Pacotes representado na Figura 3, em que é possível visualizar os elementos (papéis, atividades e artefatos) que a constituem.

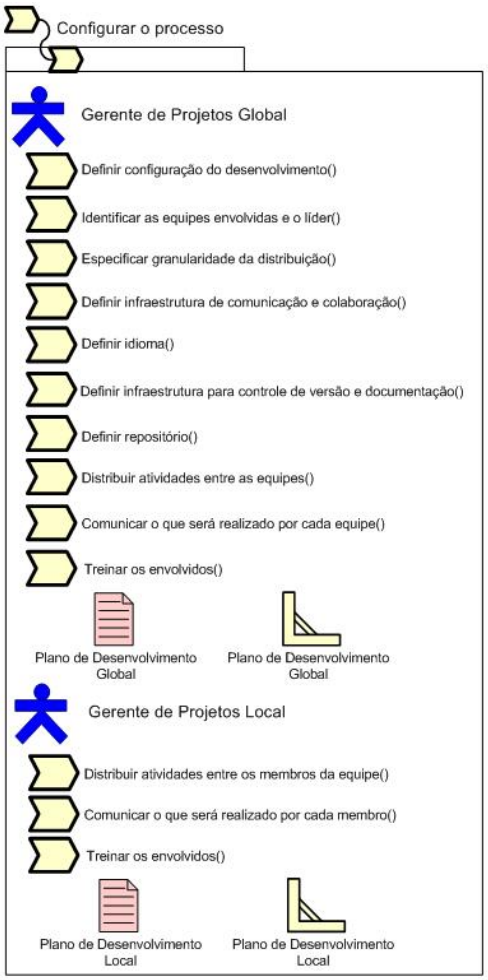

Figura 3. Diagrama de Pacotes da Definição de Trabalho - Configurar o Processo.

Os artefatos gerados são o plano de desenvolvimento de software global e local, que fornece a linha básica de recursos e cronograma. Nesse artefato são especificadas informações sobre o escopo do projeto, restrições, estrutura organizacional, papéis e responsabilidades, recursos de hardware e software e prazos. Para esses dois artefatos foram definidos *templates*, o que facilita a comunicação efetiva entre os membros das equipes.

O Diagrama de Classes da Figura 4 ilustra o relacionamento dos papéis e artefatos envolvidos, definindo assim, os responsáveis pela geração de cada artefato.

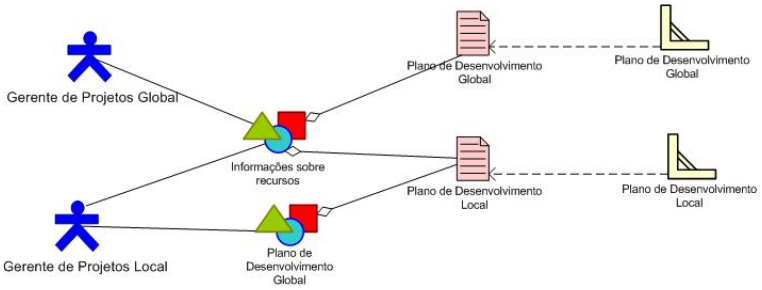

Figura 4. Diagrama de Classes - Definição de Trabalho - Configurar o Processo.

Nessa Disciplina há o Gerente de Projeto Global, que é responsável por configurar o processo de desenvolvimento e o Gerente de Projeto Local. O relacionamento entre os papéis e atividades é ilustrado no Diagrama de Casos de Uso da Figura 5, em que é possível observar que o Gerente de Projeto Global é auxiliado pelos Gerentes de Projeto Local na estruturação do processo.

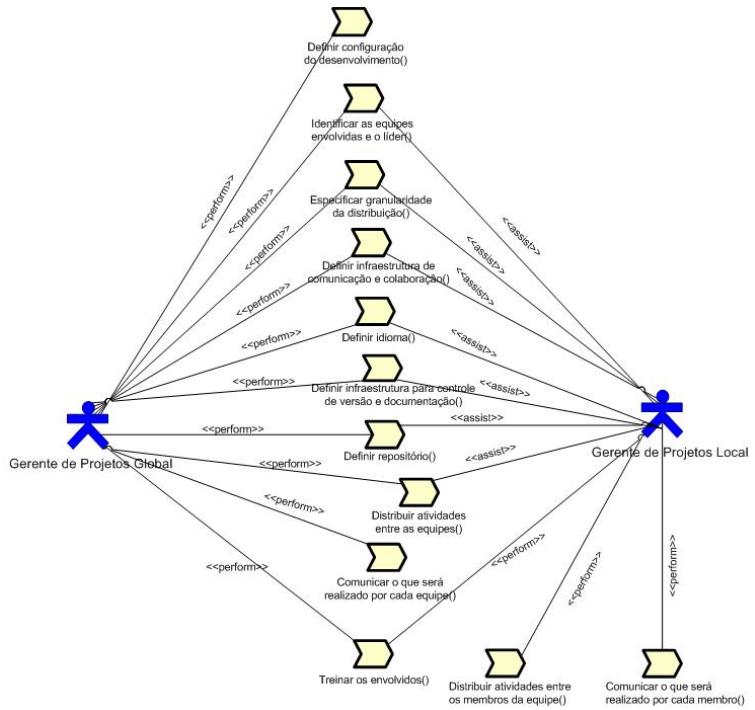

Figura 5. Diagrama de Casos de Uso - Definição de Trabalho - Configurar o Processo.

O fluxo de atividades da Definição de Trabalho - Configurar o Processo - é representado pelo Diagrama de Atividades da Figura 6, que permite visualizar a precedência das atividades, bem como, as que podem ser executadas simultaneamente. O modelo auxilia na percepção de processo, pois possibilita responder questões do tipo: "Como a tarefa daquela pessoa se encaixa na minha? e "O que devo fazer agora?"

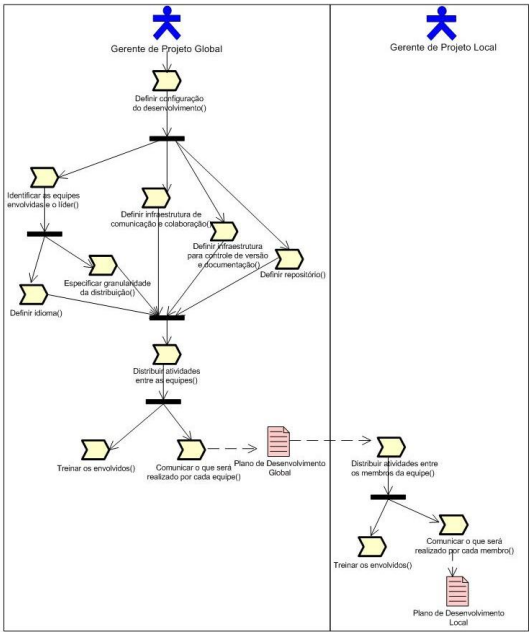

Figura 6. Diagrama de Atividades - Definição de Trabalho - Configurar o Processo.

5 Considerações Finais

A crescente busca por maior competitividade tem levado as empresas a adotarem o DDS. Essa mudança de paradigma tem causado impacto no marketing, na distribuição e na forma de concepção, de produção, de projeto, de teste e de entrega do software aos clientes. Segundo [Damian and Lanubile, 2004] para minimizar esses efeitos e alcançar níveis mais elevados de produtividade são necessárias novas tecnologias, processos e métodos compatíveis com a abordagem de desenvolvimento distribuído.

Diante deste cenário o objetivo deste trabalho foi formalizar uma abordagem integrada de desenvolvimento e teste de software para apoiar o desenvolvimento com equipes distribuídas. Essa abordagem foi desenvolvida com o intuito de oferecer suporte adequado aos processos de comunicação, coordenação e controle fornecendo uma nomenclatura comum aos envolvidos, melhorando a comu-

nicação entre as equipes através de artefatos com informações relevantes tanto para desenvolvimento quanto para teste, oferecendo visibilidade sobre as disciplinas em execução, os envolvidos e suas responsabilidades.

A abordagem abrange um conjunto de atividades que devem ocorrer ao longo do ciclo de vida de um projeto e as características essenciais de cada uma. Considera atividades desde o modelo de negócios a ser adotado para atuar em DDS (empresas globais, parcerias estratégicas, entre outros) até a implementação. A modelagem da abordagem foi realizada utilizando a notação SPEM com o intuito de facilitar a comunicação, o entendimento do processo, o reuso, apoiar a sua evolução, facilitar o gerenciamento e com vistas a melhoria contínua do processo.

Referências

[Abdurazik, 2000] Abdurazik, A. *Evaluation of Three Specification-Based Testing Criteria*. In: *ICECCS '00: Proceedings of the 6th IEEE International Conference on Complex Computer Systems*, Washington, DC, USA, (2000).

[Alves et al., 2008] Alves, E. L. G. and Machado, P. D. L. and Ramalho, F. Uma Abordagem Integrada para Desenvolvimento e Teste Dirigido por Modelos. In: *2nd Brazilian Workshop on Systematic and Automated Software Testing*, Campinas, SP, pp. 74–83, (2008).

[Ben-Shaul and Kaiser, 1994] Ben-Shaul, I. Z. and Kaiser, G. E. *A paradigm for decentralized process modeling and its realization in the Oz environment*. In: *ICSE '94: Proceedings of the 16th international conference on Software engineering*, IEEE Computer Society Press, Los Alamitos, CA, USA, pp. 179–188, (1994).

[Berger, 2003] Berger, P. M. Instanciação de Processos de Software em Ambientes Configurados na Estação TABA, Universidade Federal do Rio de Janeiro, Rio de Janeiro - RJ, (2003).

[Damian and Lanubile, 2004] Damian, D. and Lanubile, F. *The 3rd International Workshop on Global Software Development*. In: *ICSE '04: Proceedings of the 26th International Conference on Software Engineering*, IEEE Computer Society, Washington, DC, USA, pp. 756–757, (2004).

[Fuggetta, 2000] Fuggetta, A. *Software Process: A Roadmap*. In *ICSE '00: Proceedings of the Conference on The Future of Software Engineering*, IEEE Computer Society, New York, NY, USA, pp. 25–34, (2000).

[Hartmannm et al.,2004] Hartmannm, J. and Vieira, M. and Ruder, A. *UML-based Test Generation and Execution*. In: *Proceedings of the 21st Workshop on Software Test, Analyses and Verification (GI-FG TAV)*, Berlin, (2004).

[Herbsleb et al., 2000] Herbsleb, James D. and Mockus, Audris and Finholt, Thomas A. and Grinter, Rebecca E. *Distance, dependencies, and delay in a global collaboration*. In: *CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work*, ACM, New York, NY, USA, pp. 319–328, (2000).

[Humphrey and Kellner, 1989] Humphrey, W. S. and Kellner, M. I. *Software Process Modeling: principles of entity process models*. In: *ICSE '89: Proceedings of the 11th international conference on Software engineering*, New York, NY, USA, pp. 331–342, (1989).

[OMG, 2005] OMG. *Software Process Engineering Metamodel Specification. An Adopted Specification of the Object Management Group, Inc.T*, Rio de Janeiro - RJ, (2005).