Recomendações para a Gestão do Desenvolvimento de Software com Equipes Distribuídas

Gislaine Camila Lapasini Leal[1 2], **César Alberto da Silva**[2], **Elisa Hatsue Moriya Huzita**[2], **Tania Fátima Calvi Tait**[2]

[1]Departamento de Engenharia de Produção – Universidade Estadual de Maringá (UEM)

[2]Departamento de Informática - Universidade Estadual de Maringá (UEM)
Maringá, PR – Brasil

gclleal2@uem.br, cesaralberto91@hotmail.com, {emhuzita, tait}@din.uem.br

Abstract. *Distributed Software Development aims at increase productivity, reduce costs and improve quality. However, this scenario adds new challenges to the development process related to communication, coordination and control. These factors directly influence the quality of software produced. In this way, it is necessary to treat them properly. The purpose of this paper is to present a set of recommendations to offer support to the project manager when distributed software development approach is adopted.*

Resumo. *O Desenvolvimento Distribuído de Software visa o aumento de produtividade, redução de custos e melhoria na qualidade. Porém, esse cenário adiciona novos desafios ao processo de desenvolvimento relacionados à comunicação, coordenação e controle. Esses fatores influenciam diretamente a qualidade do software produzido, dessa forma, é necessário tratá-los adequadamente. Este artigo têm como objetivo apresentar um conjunto de recomendações para apoiar o gerente de projetos quando adotada a estratégia distribuída para o desenvolvimento do software.*

1. Introdução

A crescente globalização do ambiente de negócios e da economia, juntamente com o avanço tecnológico, alta competitividade, aumento do tamanho das equipes, dificuldade em reunir especialistas e aumento da complexidade do software, têm afetado, diretamente, o desenvolvimento de software. Em busca de vantagem competitiva, diversas organizações optaram por distribuir o processo de desenvolvimento de software caracterizando o Desenvolvimento Distribuído de Software (DDS) [Huzita et al. 2008].

O principal objetivo dessa descentralização consiste em otimizar os recursos para desenvolver produtos de maior qualidade e a um custo menor. No entanto, essa dispersão adicionou novos desafios ao desenvolvimento relacionados à comunicação, coordenação e controle, os quais podem afetar negativamente a produtividade e, por conseguinte, a qualidade do software. Esses fatores influenciam a maneira pela qual o software é projetado, desenvolvido, testado e entregue aos clientes, afetando assim os estágios correspondentes do ciclo de vida do software.

De acordo com [Mockus and Herbsleb 2001] e [Damian 2002], os principais desafios encontrados no DDS são relacionados às diferenças culturais, dispersões geográficas, coordenação e controle e comunicação. A diversidade cultural implica em dificuldades de compreensão da linguagem natural utilizada nos documentos de requisitos e a convergência entre diferentes interesses [Layman et al. 2006]. E as distâncias geográficas agravam os problemas de coordenação e controle, de forma direta ou indireta, por meio dos efeitos negativos que causam na comunicação [Hargreaves and Damian 2004].

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.

Em [Jiménez et al. 2009] é possível observar a evolução dessa abordagem de desenvolvimento em função do aumento de trabalhos disponíveis na literatura. Além disso, nota-se que os maiores esforços estão centrados em processos organizacionais, os quais tratam dos recursos humanos, gestão organizacional, alinhamento de infraestrutura e gestão de projetos. Isto evidencia uma lacuna em relação ao nível de projetos e aspectos técnicos o que requer novos processos e ferramentas.

Os processos de engenharia de software possuem artefatos, responsabilidades e atividades bem definidas, possibilitando adequação às necessidades específicas. No entanto, estes processos são, geralmente, voltados para o desenvolvimento de projetos coalocados, não considerando as peculiaridades de coordenação, controle e comunicação do desenvolvimento com equipes geograficamente dispersas. [Rocha et al. 2008] enfatizam a necessidade de processos adequados a essa estratégia de desenvolvimento. De acordo com [Audy and Prikladnicki 2008], em um ambiente de desenvolvimento distribuído, é fundamental um processo de desenvolvimento comum à equipe, visto que uma metodologia auxilia diretamente na sincronização, fornecendo a todos os membros da equipe uma nomenclatura comum de tarefas e atividades, e um conjunto comum de expectativas aos elementos envolvidos no processo.

Este trabalho apresenta um conjunto de recomendações relacionadas à processo de desenvolvimento de software, quando da adoção da estratégia distribuída, as quais podem nortear o Gerente de Projetos. O texto encontra-se estruturado em quatro seções, além da introdução. Na Seção 3 são tratados aspectos do processos de desenvolvimento. Na Seção 4 são apresentadas as recomendações relacionadas ao processo de desenvolvimento. Por fim, na Seção 5 são realizadas as considerações finais.

2. Metodologia

A pesquisa conduzida neste trabalho se caracteriza como sendo do tipo exploratória. O estudo objetiva proporcionar uma maior familiaridade com o problema com vistas a torná-lo explícito [Silva and Menezes 2000].

O objetivo principal deste trabalho foi identificar dificuldades, soluções e fatores críticos relacionados ao processo de desenvolvimento de software com equipes distribuídas. Desse modo, com base na literatura disponível, a principal finalidade foi elaborar um conjunto de recomendações que tem como objetivo a criação de novas teorias ou amadurecimento das já existentes.

3. Processos de Desenvolvimento

Um processo de software pode ser definido como um conjunto coerente de políticas, estruturas organizacionais, tecnologias, procedimentos e artefatos necessários para conceber, desenvolver, implantar e manter um produto de software [Fuggetta 2000]. 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.

O processo de desenvolvimento de software tem passado por intensas transformações nos últimos anos. E nesse contexto, novas abordagens de desenvolvimento têm surgido objetivando que muitos dos problemas encontrados nos projetos anteriores não sejam repetidos. Dentre os problemas mais comuns aos projetos de software podem ser destacados: altos custos para evolução, inconsistência entre documentação e sistema final, pouca ou quase que total falta de portabilidade e baixa confiabilidade. É sabido que o sucesso de um projeto de desenvolvimento de software inicia-se no devido planejamento e na escolha de uma metodologia compatível com as características do mesmo, o que reforça a necessidade de um processo de desenvolvimento que contemple de forma efetiva as necessidades do DDS.

Segundo [Rocha et al. 2008], quando o contexto é Desenvolvimento Distribuído, o cenário muda com relação ao desenvolvimento de software tradicional, pois as variáveis e os riscos aumentam. Então, se não houver uma metodologia adequada para o processo de desenvolvimento, o projeto terá boas chances de não corresponder ao planejamento inicial. Em sua primeira percepção sobre o uso e estudo de DDS, [Rocha et al. 2008] ressaltam a necessidade de processos mais adequados, pois, com base nas pesquisas realizadas, não foi fácil a identificação de modelos de referência de processos para o ambiente DDS.

4. Conjunto de Recomendações

Em [Leal et al. 2010] os processos RUP, XP, AUP, *Extended Workbench* e LAGPRO foram analisados sob a perspectiva do DDS. Com isso, pode-se evidenciar algumas lacunas em relação às novas demandas geradas pela abordagem distribuída.

Com base nessas lacunas identificadas apresenta-se um conjunto de recomendações que tem por objetivo auxiliar o processo de desenvolvimento distribuído de software tanto no nível estratégico (aspectos de gerência) como no nível de projetos e aspectos técnicos. E, pretende-se permitir, no futuro, a elaboração de um processo adequado às características do DDS. Além disso, essas recomendações podem ser facilmente empregadas durante um projeto dada a sua natureza prática [Siqueira and Silva 2006].

As recomendações relacionadas aos aspectos gerenciais são:

- definir a configuração do desenvolvimento e teste (*onshore insourcing, outsourcing, offshoring* ou *internal offshoring*) a ser adotada. De acordo com [L'Erário 2009] é importante explorar, estrategicamente, as alianças entre organizações para obter vantagem competitiva a partir da produtividade distribuída. Pode-se optar por terceirizar os testes de integração, sistemas e de aceitação para empresas que sejam especialistas na área.

Segundo [Bazman 2007] a terceirização da atividade de testes pode ser motivada pela redução de custos, velocidade, melhoria dos testes e aquisição de instalações de ambientes de teste. E como, benefícios tem-se: retorno do investimento; melhoria da confiabilidade do software; realização de uma maior gama de testes; contratação de equipe de teste eficiente e qualificada num curto período de tempo; e, maior eficiência.

- identificar as equipes envolvidas no projeto, bem como seu líder (Gerente de Projetos Local) para estimular a confiança e compromisso entre os membros. Na formação das equipes deve-se considerar, além do idioma, questões de afinidade e habilidades dos membros. Segundo [Cibotto et al. 2009], sempre que possível, envolva no projeto equipes que possuem o mesmo idioma nativo para amenizar os problemas de comunicação. Deve-se, também, estabelecer a equipe central, que será responsável pela sincronização das atividades entre as diversas equipes;
- definir uma infraestrutura para comunicação e colaboração, tanto interna (equipes envolvidas no projeto) quanto externa (clientes). Para que essa comunicação ocorra de forma adequada, é necessário que a infra-estrutura seja definida corretamente. Geralmente, são necessários diversos meios, tais como: viagens, telefone, e-mail, videoconferência, ferramentas de gerência de projetos on-line, *instant messengers* e *wiki* [Al-Asmari and Yu 2006];
- definir um idioma para formalização do processo e interação entre as equipes;
- distribuir as atividades entre as equipes envolvidas e seus membros. [Weissleder and Schlingloff 2008] apresentam duas estratégias que podem ser utilizadas para a distribuição das atividades: minimizar a colaboração necessária entre as equipes, visto que isso minimiza os impactos negativos dos problemas de comunicação e colaboração; e, minimizar a diferença entre as equipes, reduzindo o deslocamento de tempo (fuso horário) que afeta a comunicação síncrona ou as diferenças culturais;
- comunicar o que será realizado por cada equipe, bem como a responsabilidade de cada membro, para que todos tenham consciência do que está acontecendo, das dependências, quem está realizando determinada atividade e em que local ela está sendo executada. [Audy and Prikladnicki 2008] destacam que essa consciência é importante, principalmente, sob a perspectiva da documentação e do código fonte. Segundo [Carmel and Tija 2005], para aumentar a percepção podem ser utilizadas algumas técnicas, tais como, o uso de repositórios comuns, sistemas para controle de projetos, ambientes de desenvolvimento integrados, reuniões de status, cronograma das equipes e descrição dos processos;
- gerenciamento híbrido (centralizado-descentralizado) em relação ao controle das atividades, com a definição de um líder para estimular a confiança e compromisso entre os membros ([Mullick et al. 2006] e [Avritzer et al. 2008]);
- encorajar a comunicação entre desenvolvimento e equipe de teste de integração oferecendo artefatos com informações relevantes para as duas equipes [Vanzin et al. 2005];
- especificar a granularidade em que será realizada a distribuição (disciplina, componente, caso de uso, classe ou método). Para estabeler a estratégia de distribuição os Gerentes de Projeto Global e Local podem considerar alguns pontos, tais como:
    1. uma arquitetura de software modular (baixo acomplamento e alta coesão) pode ser utilizada para distribuição dos esforços entre as equipes pois, re-

duz a complexidade e permite um desenvolvimento em paralelo simplificado [Audy and Prikladnicki 2008];
2. a proximidade de uma equipe do cliente pode ser considerada para a distribuição da disciplina, no caso Requisitos.
3. as habilidades e competências das equipes envolvidas (modelagem, teste, implementação).

Cabe ressaltar, que a estratégia de distribuição adotada tem impacto direto sobre a intensidade, frequência e o tipo de coordenação necessária entre os participantes do projeto [Sangwan et al. 2006]

- desenvolvimento iterativo e com entregas frequentes para fornecer uma maior visibilidade do projeto aos gerentes ([Paasivaara and Lassenius 2004] e [Taylor et al. 2006]).
- definir infraestrutura para o controle de versão dos artefatos e da documentação;
- definir um repositório comum à todas as equipes envolvidas no projeto. Os Gerentes de Projeto Global e Local devem definir os níveis de acesso de cada integrante com base nas atividades que estes desempenham e na responsabilidade que lhe é conferida;

O gerente de projetos em conjunto com os desenvolvedores, também deve:

- detalhar as especificações dos casos de uso por meio de restrições utilizando a *Object Constraint Language* (OCL) que é uma linguagem de expressões textuais precisas, utilizada na descrição de restrições em modelos orientados a objeto, como forma de complementar a parte gráfica dos modelos, para descrever restrições, que não conseguem ser diagramaticamente representadas. A OCL é uma linguagem formal baseada em modelos, que adota uma sintaxe simples, e que utiliza símbolos matemáticos mais simples da teoria de conjuntos e lógica, numa proposta de ser precisa, porém de fácil compreensão (escrita-leitura) por quem não possui conhecimento matemático aprofundado. Um dos mais importantes aspectos de OCL é que se tornou parte do padrão *Unified Modeling Language* (UML). No contexto de DDS o uso de OCL permite reduzir a ambiguidade gerada nos artefatos devido aos fatores culturais.
- padronizar os artefatos para reduzir a comunicação entre as equipes e especificação de *templates* para facilitar a efetiva comunicação entre os membros das equipes.
- adotar metodologias de especificação de fácil compreensão e que possuam semântica para transmitir informações aos desenvolvedores, no caso a UML por ser reconhecida tanto no meio acadêmico como industrial [Sengupta et al. 2006];
- especificar os casos de teste utilizando mecanismo formal para amenizar os problemas de ambiguidade e reduzir a necessidade de comunicação [[Mullick et al. 2006], e [Avritzer et al. 2008]]. Para tanto recomenda-se o uso do Perfil UML 2.0 de Testes (U2TP - UML 2.0 *Testing Profile*), que define uma linguagem para projetar, visualizar, especificar, analisar, construir e documentar os artefatos de teste de sistemas. O 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. Além disso, as restrições OCL realizadas nos métodos das classes podem ser utilizadas como oráculo.

5. Considerações Finais

A busca por maior competitividade tem levado as empresas a adotarem o DDS. Tentando realizar desenvolvimento a baixo custo, empresas têm atravessado fronteiras, formando um mercado global. O desenvolvimento distribuído pode ser motivado, também, por questões de qualidade, gerenciamento do controle de produção, domínio de tecnologias empregadas, redução da necessidade de grandes contratações imediatas ou para transferir conhecimento a uma filial [L'Erário 2009].

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 [Sangwan et al. 2006]. 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.

Neste trabalho foi identificado um conjunto de recomendações relacionados ao processo de desenvolvimento de software com equipes distribuídas. Essas recomendações tem como objetivo nortear os envolvidos em DDS no que se refere aos processos de comunicação, coordenação e controle por meio de atividades que contemplam as peculiaridades dessa estratégia de desenvolvimento. Como trabalhos futuros estão previstos: condução de um estudo de caso; avaliação experimental das recomendações; e, definição de um processo de desenvolvimento que considere as características do DDS.

References

Al-Asmari, K. and Yu, L. (2006). Experiences in distributed software development with wiki. In *Software Engineering Research and Practice*, pages 389–293.

Audy, J. and Prikladnicki, R. (2008). *Desenvolvimento Distribuído de Software: Desenvolvimento de software com equipes distribuídas*. Elsevier, Rio de Janeiro, RJ.

Avritzer, A., Paulish, D., and Cai, Y. (2008). Coordination implications of software architecture in a global software development project. In *WICSA '08: Proceedings of the Seventh Working IEEE/IFIP Conference on Software Architecture (WICSA 2008)*, pages 107–116, Washington, DC, USA. IEEE Computer Society.

Bazman (2007). The benefits of outsourced testing. http://bazman.tripod.com/outsourced.html.

Carmel, E. and Tija, P. (2005). *Offshoring Information Technology: Sourcing and Outsourcing to a Global Workforce*. Cambridge University Press, New York.

Cibotto, R. A. G., Pagno, R. T., Tait, T. F. C., and Huzita, E. H. M. (2009). Uma análise da dimensão sócio-cultural no desenvolvimento distribuído de software. In *V Workshop Um Olhar Sociotécnico sobre a Engenharia de Software (WOSES)*, Ouro Preto, MG.

Damian, D. (2002). Workshop on global software development. In *ICSE '02: Proceedings of the 24th International Conference on Software Engineering*, pages 667–668, New York, NY, USA. ACM.

Damian, D. and Lanubile, F. (2004). The 3rd international workshop on global software development. In *ICSE '04: Proceedings of the 26th International Conference on Software Engineering*, pages 756–757, Washington, DC, USA. IEEE Computer Society.

Fuggetta, A. (2000). Software process: a roadmap. In *ICSE '00: Proceedings of the Conference on The Future of Software Engineering*, pages 25–34, New York, NY, USA. ACM.

Hargreaves, E. and Damian, D. (2004). Can global software teams learn from military teamwork models? In *Proc. of 3rd Int. Workshop on Global Software Development, ICSE'04*.

Huzita, E. H. M., Silva, C. A., Wiese, I. S., Tait, T. F. C., Quinaia, M., and Schiavone, F. L. (2008). Um conjunto de soluções para apoiar o desenvolvimento distribuído de software. In *II Workshop de Desenvolvimento Distribuído de Software*, pages 101–110, Campinas, SP.

Jiménez, M., Piattini, M., and Vizcaíno, A. (2009). Challenges and improvements in distributed software development: A systematic review. *Advances in Software Engineering*, vol. 2009.

Layman, L., Williams, L., Damian, D., and Bures, H. (2006). Essential communication practices for extreme programming in a global software development team. *Information and Software Technology*, 48(9):781–794.

Leal, G. C. L., Silva, C. A., Huzita, E. H. M., and Tait, T. F. C. (2010). Towards a process to information system development with distributed teams. In *12th International Conference on Enterprise Information Systems*, Funchal-Madeira, Portugal.

L'Erário, A. (2009). As alianças estratégicas no desenvolvimento distribuído de software. In *XXIX Encontro Nacional de Engenharia de Produção - A Engenharia de Produção e o Desenvolvimento Sustentável: Integrando Tecnologia e Gestão*, pages 756–757, Salvador, BA.

Mockus, A. and Herbsleb, J. (2001). Challenges of global software development. In *METRICS '01: Proceedings of the 7th International Symposium on Software Metrics*, page 182, Washington, DC, USA. IEEE Computer Society.

Mullick, N., Bass, M., Houda, Z., Paulish, P., and Cataldo, M. (2006). Siemens global studio project: Experiences adopting an integrated gsd infrastructure. In *Global Software Engineering, 2006. ICGSE '06. International Conference on*, pages 203–212.

Paasivaara, M. and Lassenius, C. (2004). Using iterative and incremental processes in global software development. *IEEE Seminar Digests*, 2004(912):42–47.

Rocha, R., Arcoverde, D., Brito, D., Arôxa, B., Costa, C., Silva, F. Q. B., J., A., and Meira, S. R. L. (2008). Uma experiência na adaptação do rup em pequenas equipes de desenvolvimento distribuído. In *II Workshop de Desenvolvimento Distribuído de Software*, pages 81–90, Campinas, SP.

Sangwan, R., Bass, M., Mullick, N., Paulish, D. J., and Kazmeier, J. (2006). *Global Software Development Handbook (Auerbach Series on Applied Software Engineering Series)*. Auerbach Publications, Boston, MA, USA.

Sengupta, B., Chandra, S., and Sinha, V. (2006). A research agenda for distributed software development. In *ICSE '06: Proceedings of the 28th international conference on Software engineering*, pages 731–740, New York, NY, USA. ACM.

Silva, E. L. and Menezes, E. M. (2000). *Metodologia da pesquisa e elaboração de dissertação*. Laboratório de Ensino a Distância da UFSC.

Siqueira, F. and Silva, P. S. M. (2006). Recomendações para a gerência de projetos no desenvolvimento distribuído de software. In *V Simpósio Brasileiro de Qualidade de Software*, Vila Velha, ES.

Taylor, P. S., Greer, D., Sage, P., Coleman, G., McDaid, K., and Keenan, F. (2006). Do agile gsd experience reports help the practitioner? In *GSD '06: Proceedings of the 2006 international workshop on Global software development for the practitioner*, pages 87–93, New York, NY, USA. ACM.

Vanzin, M., Ribeiro, M., Prikladnicki, R., Ceccato, I., and Antunes, D. (2005). Global software processes definition in a distributed environment. In *Software Engineering Workshop, 2005. 29th Annual IEEE/NASA*, pages 57–65. IEEE Computer Society.

Weissleder, S. and Schlingloff, B.-H. (2008). Quality of automatically generated test cases based on ocl expressions. In *ICST '08: Proceedings of the 2008 International Conference on Software Testing, Verification, and Validation*, pages 517–520, Washington, DC, USA. IEEE Computer Society.