Um Modelo para Definição de Metodologia de Desenvolvimento de Software Baseado em Pessoas

Rodrigo L. M. Mota, Pablo B. S. Lima, Breno L. Romano

Departamento de Computação e Matemática – Instituto de Ciências Exatas –
Universidade Federal de Itajubá (UNIFEI)
Caixa Postal 50 – 37.500-903 – Itajubá – MG – Brazil

{rodrigolmmota,pablobslima,blromano}@gmail.com

Abstract. *This article begins by presenting the software engineering and some of its development methods. Then, we present approaches that handle with human and organizational factors. The research focuses on proposing a model-setting methodology based on organizational and procedural aspects that enables analysis of these factors focusing people, and thoroughly, pointing out flaws and an ideal situation. The feasibility of the proposed situation is delayed with the use of hybrid methods that combine traditional and agile methods.*

Resumo. Este artigo se inicia abordando a engenharia de software e alguns de seus métodos de desenvolvimento. Em seguida, são apresentadas abordagens que tratam fatores humanos e organizacionais. A pesquisa foca em propor um modelo de definição de metodologia baseando em aspectos organizacionais e procedimentais que possibilite analisar esses fatores focando as pessoas, e de forma meticulosa, apontar falhas e uma situação ideal. A viabilidade da situação postergada é proposta com o uso de métodos híbridos, que unem práticas dos métodos clássicos e ágeis.

1. Introdução

A Engenharia de Software foi criada para formalizar o processo de desenvolvimento de software [Pressman 2006], e inicialmente foram criados os métodos clássicos que se baseiam em modelos industriais da época. Ao perceber-se a defasagem desses métodos, foram criados métodos ágeis que partem de uma outra visão. Para melhorar os processos, empresas de software visualizaram a possibilidade de adaptar as abordagens de ambos os métodos para criar métodos híbridos de desenvolvimento de software.

Apesar de ser positivo o uso de métodos híbridos, é verificado aqui a ausência de padronização dos modelos. Diante disso, foca-se, neste artigo, nos aspectos diretamente relacionados com as pessoas envolvidas, visando definir um modelo que permita às empresas aplicar as abordagens mais adequadas aos seus processos.

Investigou-se também, na comunidade científica e tecnológica, alguns trabalhos que se relacionam diretamente com esta pesquisa. Um exemplo é o trabalho de Treccani e Souza (2010), que aborda o uso de metodologias de desenvolvimento de software que visam aumentar a qualidade do desenvolvimento colaborativo, proporcionando um modelo que aborda questões relevantes aos interesses das pessoas envolvidas na equipe.

Este artigo encontra-se organizado da seguinte maneira: a Seção 2 aborda os métodos de desenvolvimento de software; a Seção 3 foca na qualidade desses métodos, destacando os aspectos baseados em pessoas; a Seção 4 apresenta os aspectos organizacionais e procedimentais; o modelo de definição de desenvolvimento de

software proposto aqui é reportado na Seção 5; e finalmente a Seção 6 que apresenta algumas recomendações e sugestões para trabalhos futuros.

2. Métodos de Desenvolvimento de Software

A história tem indicado que métodos têm trazido um nível de estrutura para o trabalho, fornecendo um roteiro razoavelmente efetivo para as equipes. Sendo assim, eles definem um conjunto de atividades, ações, tarefas, marcos e produtos de trabalho necessários para aplicar a Engenharia de Software com qualidade e confiabilidade [Pressman 2006]. Esses métodos fornecem estabilidade, controle e organização sobre uma atividade e definem também os passos em um fluxo de processo, e deles surgiram três categorias: clássicas, ágeis, e uma híbrida delas, as quais descrevemos a seguir.

Existem vários métodos que possuem características clássicas e eles têm como base um método tradicional contendo as seguintes atividades: comunicação, planejamento, modelagem, construção e implantação [Pressman 2006] [Sommerville 2003]. Os métodos principais que possuem essas características são: cascata, incremental, evolucionário, prototipagem, espiral e RUP (*Rational Unified Process*).

Quanto aos ágeis, eles foram definidos para ser uma forma alternativa aos métodos clássicos, com a finalidade de reduzir gastos com documentação e enfatizar a comunicação e colaboração com o cliente e a equipe de desenvolvimento [Beck 2001]. A ideia é manter o foco em atividades que proporcionem valor imediato na produção de software com qualidade [Sato 2007]. Eles compartilham os seguintes valores: indivíduos e interação entre eles mais do que processos; software em funcionamento mais do que documentação abrangente; colaboração com cliente mais do que negociação de contratos; responder a mudanças mais do que seguir um plano [Sato 2007]. Os métodos ágeis mais utilizados são: *Crystal Methods*, *Extreme Programming*, *Lean Development* e Scrum.

Finalmente, os métodos híbridos surgiram quando ágeis e clássicos foram postos em questionamento pelas organizações, que afirmam que essas abordagens específicas não atendem, de forma adequada, a cultura da empresa. O uso de métodos híbridos, que fazem a fusão de paradigmas dos métodos ágeis e clássicos, pode resolver esse problema. Os projetos conduzidos aqui são aplicados por equipes que utilizavam um determinado método e que sentiram alguma deficiência ou perceberam que implantar métodos diferenciados pode resultar em processos mais adequados [Mangold 2004].

3. Processo de Desenvolvimento de Software Baseado em Pessoas

O uso dos métodos abordados anteriormente tem como finalidade garantir qualidade ao produto resultante dos processos. Na Engenharia de Software, o termo qualidade de software é definido como um processo sistemático com foco nos artefatos produzidos a cada etapa, tendo como objetivo principal melhorar a qualidade do software produzido. Para atingir a este objetivo, os processos de desenvolvimento são totalmente dependentes das pessoas [DeMarco e Lister 1999].

Historicamente, os trabalhadores em geral são divididos em dois principais grupos, trabalhadores manuais e trabalhadores intelectuais. O primeiro é aquele que exerce atividades que dependem basicamente de suas habilidades manuais, já o segundo, produz com base no uso intensivo do conhecimento [Teles 2005].

O desenvolvimento de software, apesar de ser reconhecido como um trabalho intelectual, é habitualmente conduzido de maneira contrária, e uma evidência disso é o

fato de diversas empresas gerenciarem seus processos de forma semelhante aos processos industriais. Portanto, torna-se necessário aplicar práticas que evidenciem mais o trabalho como intelectual, eis elas: definir claramente as tarefas a serem realizadas; assegurar que a equipe tenha autonomia e responsabilidade sobre o que produz; possibilitar que a equipe inove os processos de maneira segura e eficiente; proporcionar a troca de conhecimento; reconhecer a qualidade como um fator mais importante que a quantidade; e fazer com que os profissionais se sintam ativos no processo [Teles 2004].

Estudos tratam o gerenciamento da produtividade em projetos com foco nas pessoas, exemplificando que se "se um pedreiro constrói um muro em 4 dias, cem pedreiros não o construirão em 10 minutos". Este pensamento não reflete a realidade de trabalhos exercidos por trabalhos manuais, tampouco trabalhos intelectuais [Salanova et. al. 1996]. Apesar de parecer absurdo, isso era aplicado há algum tempo, para resolver questões como definição de prazos, tornando necessário analisar diferentes tipos de equipes relacionados aos projetos, e diversos estudos apresentam tipos de equipes, categorizando-as por fatores. Para este artigo, foram utilizados os tipos de equipe apresentados na Tabela 1 [França 2009].

Tabela 1. Classificação dos tipos de equipe.

Propósito	Característica Dominante	Ênfase
1. Resolução de Problemas	Confiança	Foco nos problemas
2. Criativo	Autonomia	Explorar possibilidades e alternativas
3. Tático	Direção	Clareza, papéis bem definidos, foco na tarefa.

De acordo com França (2009), é necessária uma análise nos propósitos das equipes, checando se as mesmas encontram-se alinhadas ao domínio do projeto, minimizando assim possíveis riscos internos.

4. Aspectos organizacionais e procedimentais

Conforme citado na Seção 2, o uso de métodos clássicos e ágeis é questionado por não atenderem a diversos fatores, viabilizando o uso de métodos híbridos. Diante disso, este artigo visa conceber um modelo de análise de situação e com base nos valores presumidos dos aspectos analisados, e indicar ajustes nos processos na própria organização. Portanto, a Figura 1 mostra que o modelo encontra-se dividido em fatores organizacionais e procedimentais.

Figura 1. Divisão dos Fatores considerados no Modelo Proposto neste Artigo

4.1 Parâmetros Procedimentais

Além de ordem cronológica nas atividades, os processos definem fatores muito importantes para um projeto, e desta forma, torna-se plausível analisar alguns aspectos pertinentes a este grupo, que são apresentados na Figura 2 e detalhado na Tabela 3.

Figura 2. Parâmetros Procedimentais

Tabela 2. Parâmetros Procedimentais

Seção	Parâmetros	Finalidade	Método que mais enfatiza
Requisitos	Levantamento Prioridade Mudança Presença do Cliente	Definir o que vai ser produzido e em qual sequência.	Métodos clássicos, porém os ágeis são mais eficientes com relação à mudança de requisitos e acompanhamento.
Divisão de Papéis	Alocação de Profissionais Definições dos Papeis Interação dos Profissionais Reestruturação da Equipe	Definir a alocação dos profissionais nas tarefas pertinentes aos processos.	Métodos clássicos.
Erros/Falhas	Análise de Riscos Detecção/Rastreabilidade Atribuição/Responsabilidade Tratamento	Analisar como são tratados os riscos, a detecção a atribuição das falhas e o tratamento das mesmas.	Métodos ágeis, por tratar erros de forma mais pedagógica, porém clássicos são melhores em relação à análise de risco.
Documentação	Manutenção Qualidade Quantidade Abrangência	Auxiliar no entendimento do software através de diagramas e manuais.	Ambos, sendo os ágeis focados em documentar o mínimo possível e os clássicos de forma um pouco mais abrangente.
Qualidade	Produtividade/Qualidade Produção/Volume Garantia Controle/Acompanhamento	Aplicar conceitos de medição, controle e análise da qualidade. Definir se equipe dará mais ênfase na quantidade ou qualidade.	Métodos ágeis e o método RUP, que possui um controle eficaz do planejamento.

4.2 Parâmetros Organizacionais

Além dos parâmetros procedimentais, as organizações têm demonstrado interesse em outros aspectos que não relacionam diretamente com os processos, tratados aqui como parâmetros organizacionais, que são apresentados na Figura 3 e detalhados na Tabela 3.

Figura 3. Parâmetros Organizacionais

Tabela 3. Parâmetros Organizacionais

Seção	Parâmetros	Finalidade	Método que mais enfatiza
Motivação	Disciplina Cultura Determinação Perspectiva	Analisar abordagens que permita uma motivação na organização.	De maneira geral os métodos ágeis (Scrum), que proporcionam uma melhor motivação. Porém a disciplina é mais estabelecida nos métodos clássicos.
Comunicação	Relacionamento com o cliente Flexibilidade Formalização Comunicação Interna	Aplicar práticas que permitam aos membros da equipe uma melhor comunicação.	Ágeis, com o XP, por estabelecerem práticas como a programação em par, fazendo com que haja troca de informação.
Melhoria Continua	Integração da Equipe Treinamentos Jornada Transparência	Proporcionar um constante aprendizado técnico entre os integrantes da organização.	Métodos ágeis, por tratar os erros de forma mais pedagógica, porém os clássicos são mais eficazes com relação à análise de risco.
Estrutura	Ambiente Recursos Humanos Rotatividade Credibilidade no Mercado	Fornecer ferramentas e condições de trabalho necessárias para o desempenho das tarefas.	Apesar de ser extremamente importante, tanto os ágeis quanto os clássicos não enfatizam isso, apenas estimulam questões relacionada às mesas e mobílias.
Equipes	Distribuição de Papeis Produtividade Liderança Interdependência	Definir o formato de liderança e alocar os profissionais nas diversas funções da organização.	Nesses aspectos os métodos ágeis demonstram ser mais eficazes, por propor uma hierarquia menos fechada e tradicional.

5. Modelo de Definição de Metodologia de Desenvolvimento de Software

Para que uma empresa otimize seus processos de desenvolvimento, é necessário primeiramente que ela reconheça através de algum modelo, que os métodos em uso atualmente não funcionam da maneira mais apropriada e que a forma como a equipe é conduzida, inibe ou limita o aumento do conhecimento técnico. De acordo com Cukier (2007), um modelo pode ser visto como um documento que descreve um problema juntamente com uma possível solução. Sendo assim, o objetivo principal deste artigo é fornecer um modelo que possibilite uma análise da situação dos processos, ajustes necessários, os impactos que esses ajustes causarão na organização e possíveis dificuldades na adoção dos novos processos.

Como forma de se obter relevância no mapeamento dos processos, o modelo proposto neste artigo aborda inicialmente a análise da situação atual da empresa juntamente com um ponto ideal (situação almejada).

Na análise, os aspectos são agrupados em seções e divididos entre os parâmetros procedimentais e organizacionais, conforme apontado nas Figuras 2 e 3. A análise deve ser realizada por todos os profissionais envolvidos, para que se obtenha valores mais precisos com a realidade e possíveis de serem alcançados. Cada aspecto deve receber valores, de forma que a soma em cada seção, totalize entre 0 e 100, tanto na situação atual da empresa quanto no ponto ideal (almejado), conforme demonstrado na Tabela 4.

Tabela 4. Análise da situação atual e ponto ideal

Parâmetros Procedimentais - Seção: Requisitos		
Aspecto	**Situação Atual**	**Ponto ideal**
Levantamento de requisito	30	40
Priorização dos requisitos	25	30
Mudança de requisitos	20	20
Acompanhamento dos requisitos	15	10

A Tabela 4 aponta valores para os aspectos da seção requisitos como consta nos parâmetros procedimentais e apresentados na Tabela 2. Em uma situação como essa, o aspecto levantamento de requisitos representa a maior pontuação entre os demais, porém deve-se analisar que não há nesse caso uma diferença considerável entre a situação atual e o ponto ideal, portanto as mudanças não causariam tanto impacto. Neste caso, sugere-se a adoção de práticas presentes no método RUP.

Para um melhor entendimento e uso do modelo proposto, sugere-se como trabalhos futuros um detalhamento a respeito de como os diferentes métodos tratam os aspectos abordados neste artigo e apresentados nas Tabelas 2 e 3.

A implantação das novas abordagens definidas pelo modelo pode ter algumas dificuldades, principalmente com relação à resistência por parte dos membros. Esse tipo de resistência é comum quando as empresas dão maior enfoque nos aspectos do tipo procedimental. Porém, é necessário que as empresas tenha conhecimento de que os primeiros a serem afetados pelas novas abordagens são as pessoas e, de acordo como elas são afetadas, os projetos podem se suceder de maneira positiva ou negativa. Porém, torna-se possível reduzir essa resistência fazendo um balanceamento entre os aspectos procedimentais e organizacionais, de forma que a equipe esteja em constante aprendizado com relação aos processos. Como forma de otimizar a aceitação por parte da equipe, é recomendável que a empresa introduza as novas abordagens de maneira gradual, analisando o quanto a equipe se sente confortável diante das mesmas.

6. Conclusão

Baseada em modelos industriais, a Engenharia de Software surgiu para colocar ordem no processo de desenvolvimento. Desde então, diversos métodos surgiram e foram sendo aperfeiçoados e um marco foi o surgimento dos métodos ágeis, criados para facilitar o processo de desenvolvimento, tornando-o menos preditivo.

Diversas empresas interessadas em obter vantagens, observaram vários fatores dos métodos ágeis e clássicos e adaptaram às suas necessidades, criando os métodos híbridos. Outra observação é o fato de empresas de software analisar fatores além dos procedimentais em seus projetos para focar também as pessoas.

Este trabalho teve como principal contribuição a criação de um modelo que permite às empresas de software uma análise de diversos aspectos dos procedimentos e da organização para adotar melhores práticas de cada método, para que a empresa otimize seus processos continuamente. Apesar de não ter sido aplicado na prática, o modelo proposto busca sua eficácia no fato do mesmo utilizar-se de maneira flexível de diversos aspectos, comuns em organizações e em processos de desenvolvimento de software, possibilitando para cada empresa a definição de uma metodologia de desenvolvimento mais apropriada e específica.

Para analisar a eficácia do modelo proposto, torna-se necessário viabilizar a sua aplicação em casos reais. Esta aplicação está em andamento, através da elaboração de uma série de questões envolvendo os aspectos abordados no modelo proposto. Após a conclusão do questionário, a pesquisa irá proceder através de consulta a diversas empresas de software, como forma de alinhar os aspectos abordados junto à realidade das empresas a serem consultadas.

7. Referências Bibliográficas

Cukier, D. (2007) Tópicos em Ciência da Computação: Padrões para Introduzir Novas Ideias, IME-USP.

De Marco, T. and Lister, T. (1999) Peopleware: Productive Projects and Teams, Dorset House, 2nd Edition

França, C. C. (2009) Um Estudo sobre Motivação em Integrantes de Equipes de Desenvolvimento de Software. UFPE.

Mangold, P. (2004), TI - Gerenciamento de Projetos, Campus.

Beck, K. (2001) Manifesto for agile software development.

Pressman, R. S. (2006), Engenharia de Software, McGraw-Hill, 6a edição.

Salanova, M., Hontagas, P. e Pieró, J. M. (1996). "Motivation Laboral", In: Peiró, J. M.

Sato, D. T. (2007) Uso Eficaz de Métricas em Métodos Ágeis de Desenvolvimento de Software. IME-USP.

Schwaber, K. and Beedle, M. (2001) Agile software development with scrum. NJ: Prentice-Hall.

Sommerville, I. (2003), Engenharia de Software, Prentice-Hall, 6a edição.

Teles, V. M. (2004), Extreme Programming, Novatec.

Teles, V. M. (2005) Um Estudo de Caso da Adoção das Práticas e Valores do Extreme Programming. IM-NCE/UFRJ.

Treccani, P. J. F. e Souza, C. R. B. (2010) Utilização de Metodologias Ágeis no Desenvolvimento de Software: Resultados de um Estudo Empírico. VII ESELAW - UFG.