Intercursos Revista Científica
Ciências Exatas

CAPACIDADE DE AVALIAÇÃO DE UM SOFTWARE UTILIZANDO O MODELO CMM

Capacity evaluation of a Software Using the Model CMM

Walteno Martins Parreira Júnior, Renner Augusto Alves Lima,
Roberto Guimarães Dutra de Oliveira e Cássio Borges Silva Matsumoto

RESUMO

A intensa evolução dos microcomputadores fez com que o desenvolvimento de *softwares* seja cada vez mais complexo e que se utilizam mais recursos dos computadores. O desenvolvimento de um *software* de alto nível requer da organização que o desenvolve, técnicas avançadas para que o produto final seja de qualidade aceitável no mercado. O modelo de maturidade de capacitação de *software* fornece às organizações um guia de como obter controle em seus processos para desenvolver e manter o *software*, assim como evoluir em direção a uma cultura de engenharia de *software* e excelência de gestão.

Palavras-chave: Modelo de maturidade de capacitação; Qualidade de *software*; Engenharia de *software*.

ABSTRACT

The intense development of computers has made the software development is increasingly complex and use more computer resources. But the development of a high-level software requires the organization that develops, advanced to the final product is of acceptable quality in the market. The capability maturity model software provides organizations with a guide on how to control their processes for developing and maintaining software, and evolve toward a culture of software engineering and management excellence.

Keywords: Capability maturity model; Software quality, Software engineering.

INTRODUÇÃO

Atualmente, existem cada vez mais sistemas controlados por *software*, fazendo com que a economia de vários países seja muito dependente da qualidade dos *softwares,* por eles usados, justificando um investimento significativo nesse setor.

Há alguns anos, o *software* era desenvolvido de maneira artesanal, ou seja, sem muita preocupação de como ficaria o resultado final. A partir de uma definição dos requisitos do *software*, iniciava-se seu desenvolvimento. Hoje, a grande maioria busca soluções profissionais para seu desenvolvimento. Se considerarmos o desenvolvimento de um programa de pequeno porte, que não exige muito esforço e nem técnicas muito avançadas, a forma artesanal não trará grandes problemas para desenvolvê-lo. Considerando um *software* de maior porte, podem acontecer problemas graves na implementação, levando a erros que podem comprometer todo o projeto.

A evolução dos computadores, a facilidade de se obter um computador potente e de última geração, tem feito com que a demanda por softwares mais complexos e que exijam mais processamento, aumente. A demanda por programas mais complexos reflete diretamente na quantidade de problemas que possam aparecer no produto final.

Para tentar solucionar o problema de *softwares* de baixa qualidade, foi proposto que seu desenvolvimento deixasse de ser uma arte artesanal e passasse a basear-se em princípios da Engenharia de *Software*, seguindo uma estrutura bem definida e métodos específicos. Um dos conceitos da Engenharia de Software que mais reflete na sua qualidade é o controle de qualidade do software que, segundo Sommerville (2003, p. 466), “envolve supervisionar o processo de desenvolvimento de software, a fim de assegurar que os procedimentos e os padrões de garantia de qualidade sejam seguidos”.

O termo qualidade no contexto organizacional é, em geral, relacionado a uma série de aspectos, tais como normalização e melhoria de processo, medições, padrões e verificações. Uma definição mais abrangente e completa para qualidade de software seria a proposta por (Bartié, 2002): “Qualidade de

software é um processo sistemático que focaliza todas as etapas e artefatos produzidos com o objetivo de garantir a conformidade de processos e produtos, prevenindo e eliminando defeitos". Outra definição sobre qualidade é sugerida por (Hernaski, 2010), que escreve que a qualidade do produto "é a rigorosa definição das características relevantes do produto, estabelecendo os atributos e as variáveis que deve conter, cuja dimensão deve ser assegurada. A especificação é o documento que formalizará essas definições".

A qualidade pode ser medida através do grau de satisfação que os usuários avaliam em determinado produto ou serviço. No entanto, estes podem ter qualidade para algumas pessoas e para outras nem tanto, ou seja, a qualidade é algo subjetivo. Para que exista uma padronização e uma maneira de avaliar se um *software* é de boa qualidade ou não, foram criadas normas para avaliação.

O CMM (*Capability Maturity Model*) em português, "modelo de maturidade de capacitação", é um dos modelos de avaliação de *software* mais importante e popular do momento. Esta metodologia é uma iniciativa do SEI (*Software Engineering Institute*) para avaliar e melhorar a capacitação de empresas que produzem *software*. Seu projeto foi apoiado pelo Departamento de Defesa do Governo dos Estados Unidos. Embora não seja uma norma emitida por uma instituição internacional como a ISO ou o IEEE, tem tido uma grande aceitação, até fora do mercado americano.

O modelo de maturidade de capacitação tem influenciado muito no sentido de convencer a comunidade de engenharia de *software* em geral a considerar seriamente as melhorias de processo. Ele descreve os estágios de maturidade por que passam as organizações, enquanto evoluem no seu ciclo de desenvolvimento de *software*, através de avaliação contínua, identificação de problemas e ações corretivas, dentro de uma estratégia de melhoria dos processos. Também fornece às organizações orientação sobre como ganhar controle do processo de desenvolvimento e como evoluir para uma cultura de excelência na gestão de *software*.

Nesse trabalho busca-se identificar se realmente o CMM contribui para a melhoria da qualidade do desenvolvimento do *software* em empresas desenvolvedoras através de uma pesquisa bibliográfica em casos disponíveis na literatura.

FUNDAMENTAÇÃO TEÓRICA

O modelo CMM foi desenvolvido pelo SEI (Software Engineering Institute) da Universidade *Carnegie Mellon*, em *Pittsburgh* nos Estados Unidos, composto por um grupo de profissionais de *software*, sendo a primeira versão lançada em agosto de 1991. Seu surgimento baseou-se na necessidade de atender a uma demanda do governo federal dos Estados Unidos, de criação de um método para avaliar a capacitação de seus fornecedores de *software*.

Em setembro de 1987, o SEI lançou uma breve descrição de um ambiente de maturidade de processo *de software* e desenvolveu dois métodos, sendo o primeiro de avaliação do processo e o segundo a avaliação da capacidade de software, e um questionário de maturidade para avaliar a maturidade do processo. A avaliação do processo tem o objetivo de determinar o nível do processo atual de desenvolvimento de *software* de uma organização, enquanto a avaliação da capacidade tem como objetivo identificar fornecedores qualificados para o desenvolvimento de software.

> A capabilidade [ou capacidade] do processo de software descreve a gama de resultados esperados que podem ser alcançados com a aplicação do processo de software. A capabilidade do processo de software de uma organização fornece um meio de se prever os resultados mais prováveis a serem esperados no próximo projeto a ser empreendido pela organização (GONÇALVES; VILLAS-BOAS, 2001, p. 15).

A melhoria contínua do processo se baseia mais em pequenos passos evolutivos do que em inovações revolucionárias que formam bases sucessivas para a melhoria contínua do processo, elevando sua qualidade. Hernaski (2010) escreve que a qualidade de processo “é a rigorosa especificação dos processos que serão realizados na produção de um bem ou serviço, incluindo as faixas de tolerância desejada dos resultados”.

Dentro dos passos evolutivos de melhoria de processo, o CMM possui cinco níveis de maturidade, sendo identificados como: inicial, repetível, definido, gerenciado e otimizado. De acordo com Gonçalves e Villas-boas (2001, p. 15), “a maturidade do processo de *software* é a extensão para a qual um processo específico é explicitamente definido, gerenciado, medido, controlado e efetivado”. Esses níveis de maturidade definem uma escala para medir a maturidade do processo de *software* de uma organização e para avaliar a capacidade de seu processo. Definir um nível de maturidade significa estabelecer diferentes componentes do processo, que resultam num crescimento da capacidade do processo de uma organização.

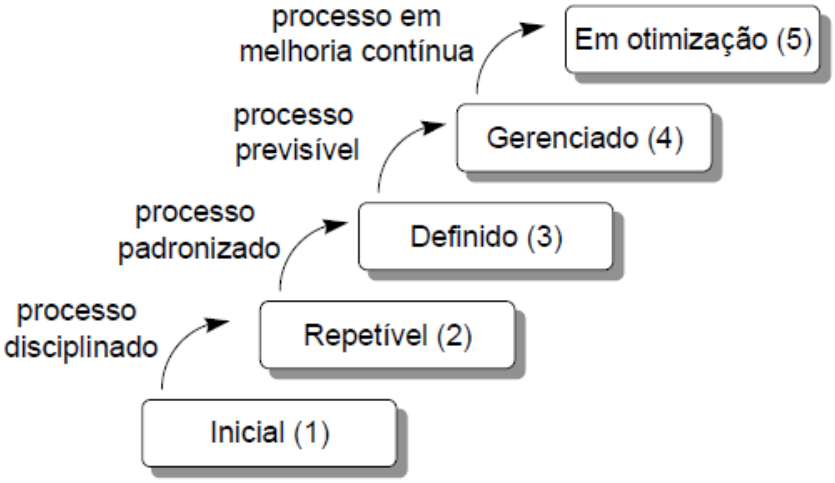

Figura 1 – Níveis de maturidade.
Fonte: Côrtes (1998, p. 11).

A Figura 1 apresenta os cinco níveis de maturidade e a característica principal que permite a progressão para o nível imediatamente acima.

> A cada nível de maturidade corresponde um conjunto de práticas de software e de gestão específicas, denominadas áreas-chave do processo. Estas devem ser implantadas para que a organização possa atingir o nível de maturidade desejado (PARREIRA JÚNIOR, 2010, p. 103).

No primeiro nível, o inicial, uma organização não tem procedimentos eficazes de gerenciamento ou planos de projeto, ou seja, não dispõe de um ambiente estável. O sucesso nessas organizações depende da competência e esforço individual dos seus funcionários e não no uso de processos

estruturados. Se existirem procedimentos formais para o controle de projeto, não existirão os mecanismos organizacionais para garantir que eles são utilizados de modo consistente, fazendo com que as características do *software* e o processo sejam imprevisíveis.

No nível dois, o repetível, a organização tem procedimentos de gerenciamento formal, garantia de qualidade e controle de configuração já implantada. Recebe o nome de nível repetível porque a organização pode repetir projetos do mesmo tipo com sucesso. Contudo, não existe um modelo de processo formal, sendo que o sucesso do projeto depende de os gerentes individuais motivarem uma equipe e de cultura organizacional atuar como uma descrição intuitiva do processo.

No nível três, o definido, uma organização determinou seu processo e, assim, tem uma base para sua melhoria qualitativa de processo, sendo usado para estabelecer uma consistência dentro da organização. Os procedimentos formais estão implantados, a fim de assegurar que o processo definido seja seguido em todos os projetos de *software*.

No penúltimo nível, o gerenciado, a organização tem um processo definido e um programa formal de coleta de dados quantitativos. As métricas de processo e de produto são coletadas e fornecidas nas atividades de melhoria de processo. A principal diferença entre os níveis de maturidade três e quatro é a previsibilidade do desempenho do processo, sendo que no nível quatro, o desempenho do processo é controlado usando técnicas estatísticas e quantitativas e é previsível quantitativamente. Já no nível três, os processos são somente previsíveis qualitativamente.

No último nível, o da otimização, a organização já adquiriu todas as metas específicas das áreas de processo dos níveis anteriores e as metas genéricas dos níveis dois e três. Processos são continuamente melhorados com base no entendimento quantitativo das causas comuns de variação inerente a eles. Nesse nível o foco é o contínuo progresso do desempenho dos processos, através da introdução de melhorias de inovação tecnológica e

incremental e os efeitos da melhoria da revisão deles são medidos e acompanhados, utilizando-se processos de melhoria de qualidade.

> Uma distinção crítica entre os níveis 4 e 5 é o tipo de variação do processo. No nível 4, o interesse é verificar as causas especiais de variação de processo e fornecer resultados estatísticos. No nível 5, o foco são as causas comuns de variação de processo e a introdução de mudanças de modo a melhorar a performance do processo, atingindo objetivos quantitativos (PARREIRA JÚNIOR, 2010, p. 106).

Assim, a melhoria dos processos é gradual e a cada progresso alcançado, pode-se identificar a sua contribuição para atender aos objetivos quantitativos propostos e a sua progressão quanto à maturidade.

APLICAÇÕES

O Processo de maturidade para *software* significa quanto um determinado processo está definido, gerenciado, medido, controlado e efetivo. Maturidade implica no potencial de crescimento da capacidade de um processo, além de indicar os detalhes dos processos e a consistência com que são aplicados dentro da organização, mostrando que a produtividade e a qualidade resultantes destes processos podem ser consistentemente melhoradas ao longo do tempo.

Apesar de engenheiros e administradores conhecerem os problemas existentes de forma bem detalhada, normalmente ocorrem divergências sobre quais melhorias são mais importantes. Sem uma estratégia organizada para enfrentar os problemas, é muito difícil chegar a um consenso entre a administração e a produção sobre quais ações tomar. Para que os esforços de melhoria forneçam resultados duradouros é necessário desenhar uma rota evolucionária que aumentará a maturidade da organização de forma escalonada, onde cada estágio serve de base para implementar as melhorias necessárias para atingir o estágio seguinte, criando um ciclo constante de melhoria dos processos. Este sistema não fornece soluções rápidas para projetos com problemas, mas identifica as deficiências e baliza o avanço.

Para as equipes de processo de engenharia de *software*, ou para outros grupos que estão tentando melhorar seus processos de *software*, o CMM tem um valor particular nas áreas de planejamento de ações, implementação de planos de ação e definição de processos. Durante o planejamento de ações, os membros da equipe de processos, utilizando seu conhecimento sobre as questões relativas a seus processos de *software* e ambiente de negócios, podem comparar suas práticas correntes com as metas das áreas-chave de processos do CMM. As práticas-chave deveriam ser examinadas com relação às metas corporativas, às prioridades de gerenciamento, ao nível de desempenho da prática, ao valor de implementar cada prática para a organização e à habilidade da organização em implementar a prática à luz de sua própria cultura.

A equipe de processo deve então determinar as melhorias de processo que são necessárias, como executar a mudança, obtendo o necessário convencimento das pessoas. O CMM ajuda nessa atividade fornecendo um ponto de partida para discussões sobre melhoria de processos e também levantando suposições discrepantes sobre as práticas de engenharia de *software* comumente aceitas. Na implementação do plano de ação, o CMM e as práticas-chave podem ser utilizados pelas equipes de processo para desenvolver partes do plano de ação operacional e definir o processo de *software*.

O modelo tem como principal objetivo a melhora dos processos internos para o desenvolvimento de *software* por parte da organização. Dentro desse conceito, os principais defensores do modelo argumentam que com sua implantação e conforme o nível de maturidade atingido, cada vez mais o CMM trará muitos benefícios para as organizações tais como a melhoria na execução dos trabalhos (aumentando a produtividade dos recursos), melhoria no controle das atividades desempenhadas (possuindo métricas e projetos bem delineados), melhora na qualidade do produto final (a melhoria proporciona uma melhora significativa na determinação da real necessidade do cliente), melhoria nos relacionamentos internos (quanto mais organizado, menos

conflitos ocorrerão) e melhoria nos relacionamentos externos da organização (maior facilidade de relacionamento com o cliente).

CONSIDERAÇÕES FINAIS

Para que as organizações de desenvolvimento de software alcancem altos níveis de maturidade, é necessário que haja um comprometimento de longo prazo e que as atividades direcionadas para este fim sejam incrementais e contínuas.

O CMM não abrange todos os pontos importantes para o sucesso do projeto, como por exemplo, quais as tecnologias e ferramentas utilizadas, porém fornece às organizações orientação sobre como ganhar controle do processo de desenvolvimento de *software* e como evoluir para uma cultura de excelência na gestão de *software*. Com o uso desse modelo, haverá uma melhora no gerenciamento e desenvolvimento de produtos de *software*, levando a uma melhora significativa na qualidade final do produto.

Porém não é tão simples desfazer dos valores intelectuais de uma organização, principalmente as já acostumadas com as turbulências. Para que uma organização esteja inserida no jogo do mercado e conquiste território é preciso seguir um modelo, ou seja, um caminho a ser trilhado, para que exista um mínimo de ordem no desenvolvimento de *software*.

Fica claro, portanto, que, em vez de burocracia sem sentido, padrões e processos bem definidos e usados, pode fazer muito pela qualidade dos processos de desenvolvimento em uma empresa, aumentando a qualidade dos produtos e reduzindo prazos e custos, compensados por menos trabalho e informações devidamente centralizadas e distribuídas.

Conclui-se que julgando todos os fatores que envolvem o modelo de maturidade de capacitação, chega-se a um resultado positivo. Com a certeza de que a qualidade do produto final está diretamente ligada à maneira como as etapas do processo são desenvolvidas, com a utilização do CMM, a qualidade do produto final torna-se acima da média, atingindo padrões elevados de

qualidade e capacitando organizações a terem softwares fortemente conceituados no mercado e atendendo as necessidades dos clientes.

REFERÊNCIAS

BARTIÉ, A. **Garantia da qualidade de software**: adquirindo maturidade organizacional. Rio de Janeiro - RJ: Elsevier, 2002.

CÔRTES, Mario L. **Modelos de Qualidade de SW**. 1998. Disponível em: http://www.ic.unicamp.br/~cortes/mc726/cap5a.pdf . Acesso em: 6 ago. 2011.

HERNASKI, M. **Qualidade do produto VS Qualidade do processo**. out. 2010. Disponível em: http://mauricio.hernaski.com.br/blog/qualidade-do-produto-vs-qualidade-do-processo-2/. Acesso em: 8 ago. 2011.

GONÇALVES, J. M.; VILLAS-BOAS, A. **Modelo de Maturidade de Capabilidade de Software,** 11 ago. 2001 Campinas - SP: CPqD.

PARREIRA JÚNIOR, Walteno M. **Engenharia de software** (Apostila). Ituiutaba - MG: FEIT-UEMG, 2010.

SOMMERVILLE, Ian. **Engenharia de Software**. São Paulo - SP: Pearson Education do Brasil, 2003.

AUTORES

Walteno Martins Parreira Júnior, é professor dos cursos de Engenharia da Computação, Engenharia Elétrica e Sistemas de Informação da UEMG - Campus de Ituiutaba. Especialista de Design Instrucional para EaD e Informática Aplicada à Educação. Mestrando em Educação no PPGED-UFU.
waltenomartins@yahoo.com

Renner Augusto Alves Lima, é acadêmico do curso de Engenharia de Computação da Fundação Educacional de Ituiutaba – FEIT - associada à Universidade do Estado de Minas Gerais – UEMG - Campus de Ituiutaba.
renneramil@gmail.com

Roberto Guimarães Dutra de Oliveira, acadêmico do curso de Engenharia da Computação da Fundação Educacional de Ituiutaba – FEIT – associada à Universidade do Estado de Minas Gerais - UEMG - Campus de Ituiutaba.
robertoguimaraes8@hotmail.com

Cássio Borges Silva Matsumoto, acadêmico do curso de Engenharia da Computação da Fundação Educacional de Ituiutaba – FEIT – associada à Universidade do Estado Estado de Minas Gerais - UEMG - Campus de Ituiutaba.
cassiomatsumoto@hotmail.com