ATENDENDO À KPA REQUISITOS DO CMM ATRAVÉS DO RUP E FERRAMENTAS RATIONAL

** Fábio Luiz Gambarotto*
*** Ademir Morgenstern Padilha*

RESUMO

O CMM, Modelo de Capacidade e Maturidade, desenvolvido pela Carnage Mellon University, estabelece práticas de Engenharia de *Software* relacionadas com aspectos gerenciais, organizacionais e técnicos. Quando essas práticas são seguidas rotineiramente, as organizações se capacitam para atingir metas estabelecidas de controle de custos, cronograma e produtividade [2]. Já o RUP é um *framework* de processo de *software* que oferece uma abordagem, baseada em disciplinas, para atribuir tarefas e responsabilidades dentro de uma organização de desenvolvimento [3]. O objetivo do presente artigo é avaliar como o RUP atende à KPA requisitos do CMM-SW e, a partir dessa avaliação, realizar um mapeamento detalhado das disciplinas, artefatos e ferramentas necessários para atingir as metas, compromissos, habilidades, atividades, medições e análises, e verificação da implementação, necessários para atender aos requisitos desta KPA.

PALAVRAS-CHAVE: CMM; KPA; Requisitos; RUP.

ABSTRACT

CMM, the Capacity Maturity Model, powered by Carnage Mellon University, establishes practices of Software Engineering related to managerial, organizational, and technical aspects. When these practices are routinely followed, the organizations are enabled to reach established goals of cost control, schedule and productivity [2]. The RUP, on the other hand, is a software process framework that offers an approach based on disciplines to attribute tasks and responsibilities within a development organization [3]. The objective of this work is to evaluate how the RUP matches the KPA requirements of the SW-CMM and, based on

* Graduado em Tecnologia de Processamento de Dados (TEC) pelo Centro Universitário Filadélfia – UniFil. Especialista em Engenharia de *Software* com UML. *E-mail:* faluga@bol.com.br
**Docente do Centro Universitário Filadélfia – UniFil. Mestre em Ciência da Computação pela Universidade de Campinas – UNICAMP. Consultor em Tecnologia da Informação. Orientador da pesquisa. *E-mail:* ademir@forti.com.br

such evaluation, to carry out a detailed mapping of the disciplines, devices, and tools necessary to reach the goals, commitments, abilities, activities, measurements and analyses, and implementation verification, necessary to match the KPA requirements.

KEY-WORDS: CMM; KPA; Requirement; RUP.

1. INTRODUÇÃO

Ao ganharem rapidamente maturidade e profissionalismo, os serviços de desenvolvimento de *software* e de componentes em regime de fábrica, registram um significativo crescimento na demanda, tornando-se competências corriqueiras nos portifólios de numerosos provedores de soluções de Tecnologia da Informação. [1]

É nesse cenário de aquecimento, que as metodologias de gestão mais avançadas, visando à obtenção do máximo de qualidade nos projetos, como o modelo CMM[1], emergem como um importante diferencial competitivo dentro da comunidade de engenharia de *software*, balizando cada vez mais as atividades do setor.

É assim que o modelo CMM, sistematizado pelo SEI no esforço de mapear as melhores práticas mundiais de desenvolvimento de *software*, é apontado pelos especialistas como referência no preenchimento dos requisitos de qualidade. Em síntese, o objetivo desse modelo é fazer com que o contingente de profissionais alocados em um dado projeto trabalhe sob processos cíclicos, padronizáveis e mensuráveis, os quais, independentemente da qualidade do produto final, sustentem, por si só, os níveis de serviços. O CMM confere, por conseguinte, maior previsibilidade aos processos, comprovando a sua maturidade. Este modelo visa ser um guia para avaliação e melhoria de processos de *software*, divididos em 18 áreas-chave de processo, distribuídos em 5 níveis de maturidade.

O presente artigo propõe-se a abordar a primeira área-chave de processo do nível 2 do CMM (Gerenciamento de Requisitos) e como satisfazê-la através do RUP[2] e ferramentas Rational.

[1]CMM: Capability Maturity Model – Modelo de Capacidade e Maturidade, desenvolvido pelo SEI (*Software* Engineering Institute), sediado na Carnegie Mellon University, em Pittsburgh, Pennsylvania, Estados Unidos.

[2]RUP: Rational Unified Process – Processo Unificado Racional.

2. C M M

O CMM estabelece práticas de Engenharia de *Software* relacionadas com aspectos gerenciais, organizacionais e técnicos. Quando estas práticas são seguidas rotineiramente, as organizações se capacitam a atingir metas estabelecidas de controle de custo, cronograma e produtividade. [2]

Esse modelo, basicamente, fornece 5 níveis de maturidade para processos de *software* (Inicial, Repetitivo, Definido, Gerenciado e Em Otimização). Cada um desses níveis apresenta fundamentos sucessivos para a melhoria contínua do processo. Os níveis de maturidade fornecem prioridades claras, as quais orientam a seleção de algumas atividades de melhoramento que, quando implementadas, possibilitam a evolução da maturidade do processo de desenvolvimento.

No nível Inicial, o processo de desenvolvimento é desorganizado e até caótico. Poucos processos são definidos e o sucesso depende de esforços individuais e heróicos.

No nível Repetitivo, os processos básicos de gerenciamento de projeto estão estabelecidos e permitem acompanhar custos, cronograma e funcionalidade. É possível repetir o sucesso de um processo utilizado anteriormente em outros projetos similares.

No nível Definido, tanto as atividades de gerenciamento quanto de engenharia do processo de desenvolvimento de *software* estão documentadas, padronizadas e integradas em um padrão de desenvolvimento da organização. Todos os projetos utilizam uma versão aprovada e adaptada do processo padrão de desenvolvimento de *software* da organização.

No nível Gerenciado, são coletadas medidas detalhadas da qualidade do produto e processo de desenvolvimento de *software*. Tanto o produto quanto o processo de desenvolvimento de *software* são entendidos e controlados quantitativamente.

No nível Em Otimização, o melhoramento contínuo do processo é conseguido através de um "*feedback*" quantitativo dos processos e pelo uso pioneiro de idéias e tecnologias inovadoras.

Exceto no nível 1, todos os níveis são detalhados em KPAs[3]. Essas áreas são, exatamente, aquilo em que a organização deve focar para melhorar o seu processo de desenvolvimento de *software*. Para que uma empresa possa se qualificar em um determinado nível de maturidade CMM, deve estar realizando os processos relacionados às áreas-chave daquele determinado nível.

[3]KPA: Key Process Area – Área Chave de Processo.

3. RUP

O RUP é um *framework* de processo de *software*. Ele oferece uma abordagem baseada em disciplinas para atribuir tarefas e responsabilidades dentro de uma organização de desenvolvimento. [3]

Seu objetivo é transformar os requisitos do usuário em sistema de *software*. Sua característica fundamental é a de ser baseada em componentes, ou seja, o *software* desenvolvido é formado por componentes de *software* que se comunicam através de interfaces bem definidas.

O desenvolvimento de *software* efetuado através do RUP é incremental e cada incremento é desenvolvido utilizando-se de quatro fases: iniciação, elaboração, construção e transição. A isto se chama ciclo de desenvolvimento. Após a fase de transição, o produto pode voltar a percorrer cada uma das fases, constituindo a fase de evolução, na qual se produz uma nova geração do produto.

- **Iniciação**: fase de compreensão do problema e da tecnologia através da definição dos casos de uso mais críticos;
- **Elaboração**: fase de descrição da arquitetura do *software* na qual os requisitos que mais impactam na arquitetura são capturados em forma de casos de uso;
- **Construção**: fase na qual o *software* é construído e preparado para a transição para os usuários. Além do código propriamente dito, também são produzidos os casos de teste e a documentação;
- **Transição**: fase de treinamento dos usuários e transição do produto para utilização.

Cada uma das quatro fases do RUP é adicionalmente dividida em iterações e finalizada com um ponto de checagem que verifica se os objetivos daquela fase foram alcançados. Toda iteração é organizada em disciplinas, que são conjuntos de atividades realizadas por responsáveis que produzem artefatos. As principais disciplinas são descritas a seguir:

- Modelagem de Negócios: provê um entendimento comum entre os envolvidos no projeto, sobre quais os processos de negócio que devem ser apoiados.
- Requisitos: objetiva capturar os requisitos que serão atendidos pelo produto de *software*. Nas fases de iniciação e elaboração, a ênfase será maior nesta disciplina.

- Análise e *Design*: objetiva compreender mais precisamente os *use cases* definidos na disciplina de requisitos, produzindo um modelo já voltado para a implementação, que deverá estar adequado ao ambiente de implementação.
- Implementação: objetiva a organização do código em termos de implementação de subsistemas, implementa as classes e objetos em termos de componentes, testa os componentes em termos de unidades e integra os códigos produzidos.
- Teste: objetiva analisar, através de testes, se os requisitos foram atendidos e certificar de que os defeitos serão removidos antes da implantação.
- Implantação: objetiva produzir *releases* do produto e entregá-los aos usuários finais. Isto pode incluir atividades de beta-teste, migração de dados ou *software* existente e aceitação formal.

A Figura-1 mostra a página inicial do RUP, onde são exibidas as quatro fases de desenvolvimento e suas disciplinas.

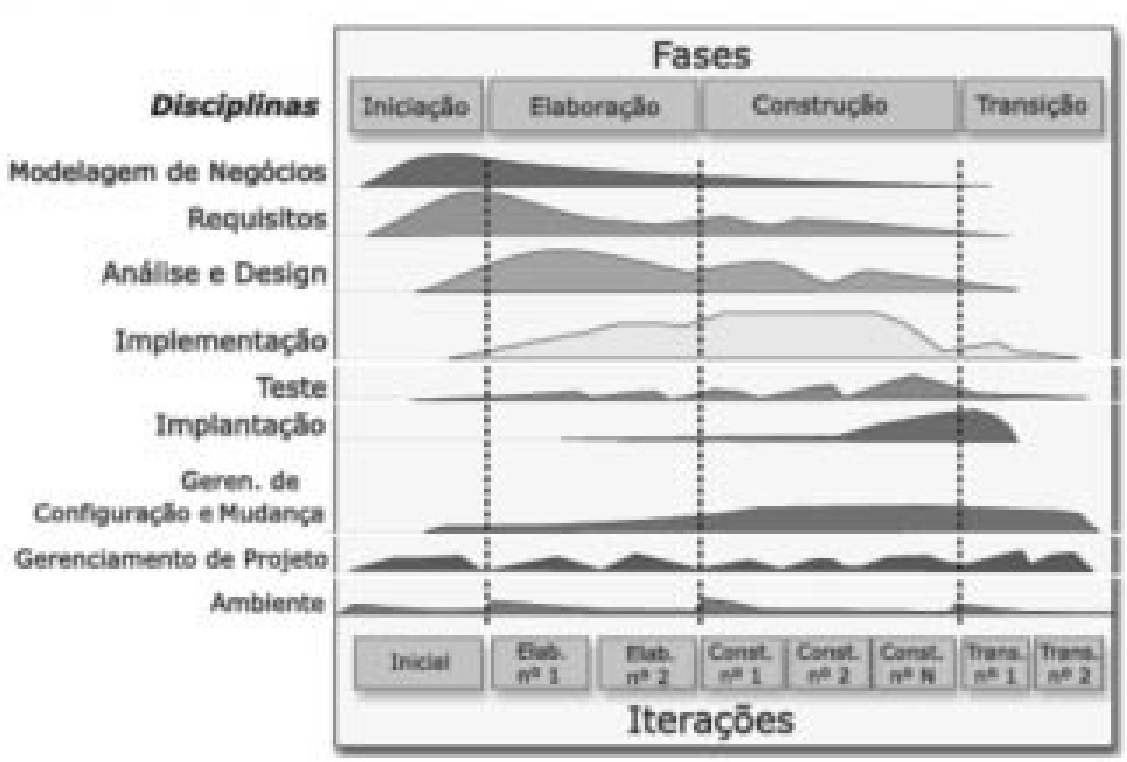


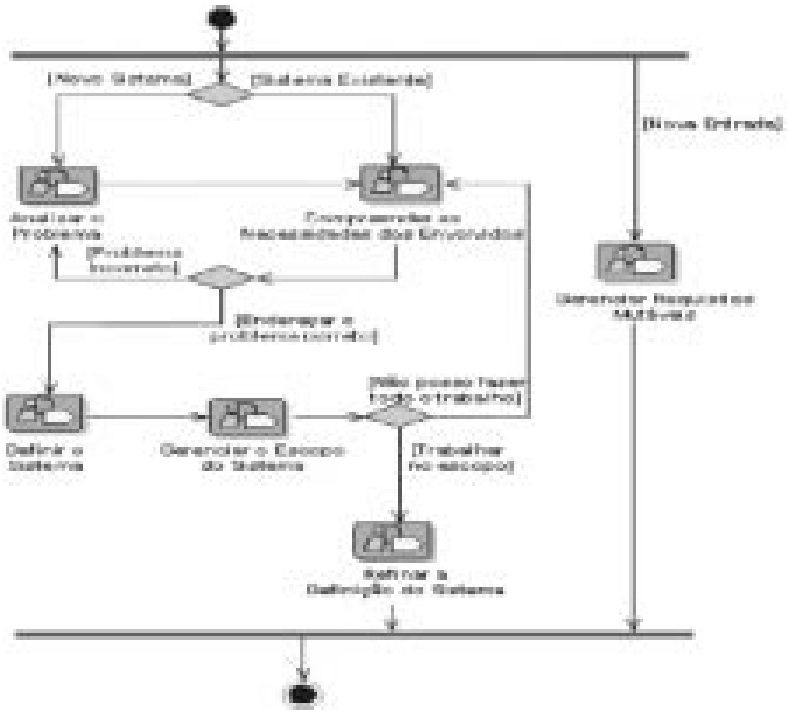
Figura-1: RUP – Fases e Disciplinas.

4. A DISCIPLINA DE REQUISITOS DO RUP

Segundo o RUP, um requisito é definido como uma condição ou uma capacidade com a qual o sistema deve estar de acordo. [4]

Entretanto, requisitos não se referem apenas à funcionalidade desejada para um sistema ou *software* (requisitos funcionais), mas também se referem às questões não funcionais (requisitos não funcionais).

Existem vários tipos de requisitos. Um modo de categorizá-los é descrito como o modelo FURPS+ [5], usando o acrônimo FURPS (Funcionalidade, Usabilidade, Confiabilidade, Desempenho e Suportabilidade) para descrever as principais categorias de requisitos com suas respectivas subcategorias.



A Figura-2 mostra o *workflow* de Requisitos do RUP e apresenta suas atividades, que consomem e produzem artefatos.

5. ATENDENDO A KPA REQUISITOS ATRAVÉS DO RUP

O modelo CMM define, para cada área-chave de processo, metas, compromissos, habilidades, atividades, medições e análises, e a verificação da implementação específica para cada uma delas.

Na tabela seguir, será apresentado um mapeamento detalhado de como o RUP atenderá à KPA Requisitos do CMM.

Tipo	Requisito	Artefato/Ferramenta/Atividade	Disciplina
Metas	Meta 1 Meta 2	Plano de Gerenciamento de Requisitos, Modelo de Casos de Uso, Visão, *Rational Rose*, *Requisite Pro*, *Clear Quest*	Requisitos
Compromissos	Compromisso 1	Caso de Desenvolvimento	Ambiente
		Plano de Gerenciamento de Requisitos, Modelo de Casos de Uso, *Requisite Pro*	Requisitos
Habilidades	Habilidade 1	Plano de Gerenciamento de Requisitos	Requisitos
	Habilidade 2	Modelo de Casos de Uso, *Requisite Pro*	
	Habilidade 3	Plano de Desenvolvimento de *Software*	Gerência de Projeto
		Plano de Gerenciamento de Requisitos	Requisitos
	Habilidade 4	Guia de Modelagem de Casos de Uso, Guia de Ferramentas	Ambiente
Atividades	Atividade 1	Avaliação de Resultados	Requisitos
	Atividade 2	*Baseline*, Modelo de Casos de Uso, Solicitações dos Principais Envolvidos, *Clear Quest*	
	Atividade 3	Solicitação de Mudança, Plano de Gerenciamento de Configuração	Gerência de Configuração e Mudança
Medições e Análises	Medição 1	Plano de Gerenciamento de Requisitos, Modelo de Casos de Uso	Requisitos
		Plano de Iteração, Plano de Métricas, *Project Console*, Soda, *Requisite Pro*	Gerenciamento de Projeto
Verificação da Implementação	Verificação 1 Verificação 2 Verificação 3	Plano de Gerenciamento de Requisitos	Requisitos
		Registro de Revisão	Gerenciamento de Projeto
		Solicitação de Mudança	Gerência de Configuração e Mudança

O Plano de Gerenciamento de Requisitos tem por objetivo descrever a documentação de requisitos, os tipos de requisitos e seus respectivos atributos, especificando as informações e os mecanismos de controle que devem ser coletados e usados para avaliar, relatar e controlar mudanças nos requisitos do produto.

O Modelo de Casos de Uso é um modelo das funções pretendidas do sistema e seu ambiente, e serve como um contrato estabelecido entre o cliente e os desenvolvedores. O Modelo de Casos de Uso é usado como fonte de informações essenciais para atividades de análise, *design* e teste.

O documento **Visão** define a visão que os envolvidos têm do produto a ser desenvolvido, em termos das necessidades e características mais importantes. Esse documento fornece uma base de alto nível - algumas vezes contratual - para os requisitos técnicos mais detalhados por conter uma descrição dos requisitos centrais pretendidos. Também pode conter uma especificação de requisitos formais.

A Disciplina de Ambiente produz o **Caso de Desenvolvimento**, que documenta a metodologia para o projeto baseado nos padrões e políticas organizacionais. Esta metodologia inclui políticas, ferramentas e técnicas a serem utilizadas em todo o projeto de desenvolvimento de *software*.

A Disciplina de Ambiente apresenta também guias e manuais que são úteis nessa habilidade, como o **Guia de Modelagem de Casos de Uso**, que descreve como modelar os casos de uso, e o **Guia de Ferramentas**, que descreve como usar as ferramentas utilizadas nessa Área-Chave de Processo, abordando informações de instalação, versão, parâmetros de configuração, limitações na funcionalidade e a funcionalidade que o projeto decidiu não usar, artifícios, integração com outras ferramentas, incluindo procedimentos a serem seguidos, *software* a ser usado e princípios a serem aplicados.

A Disciplina de Gerência de Projeto produz o **Plano de Desenvolvimento de *Software***, que é um artefato composto e abrangente que reúne todas as informações necessárias ao gerenciamento do projeto. Ele inclui vários artefatos separados, desenvolvidos durante a Fase de Iniciação, e é mantido durante todo o projeto.

As **Solicitações dos Principais Envolvidos** contém qualquer tipo de solicitação dos clientes, usuários finais, pessoal de *marketing*, em relação ao sistema que será desenvolvido. Ele também pode conter referências a qualquer tipo de fonte externa com a qual o sistema deve estar de acordo.

Uma **Solicitação de Mudança** é um artefato formalmente submetido, que é usado para rastrear todas as solicitações dos envolvidos (inclusive novas características, solicitações de melhoria, conserto de defeitos, mudança de requisitos, etc.), junto com as informações de status relacionadas durante todo o ciclo de vida do projeto. Todo o histórico de mudanças será mantido com a Solicitação de Mudança, o que inclui todas as mudanças de estado, datas e motivos para as mudanças.

A Disciplina de Gerência de Configuração e Mudança produz o **Plano de Gerenciamento de Configuração**, que descreve todas as atividades do Gerenciamento de Controle de Configuração e Mudança (CCM) que serão executadas durante o ciclo de vida do produto ou do projeto. Ele detalha o cronograma de atividades, as responsabilidades atribuídas e os recursos necessários, como equipes, ferramentas e computadores.

A Disciplina de Gerenciamento de Projeto produz o **Plano de Iteração**, que é um conjunto de atividades e tarefas divididas por seqüência de tempo, com recursos atribuídos e dependências de tarefas, para a iteração; é um plano sofisticado. É muito utilizado pelo Gerente de Projeto, para planejar tarefas e atividades de iteração, programar necessidades de recursos e acompanhar o andamento em relação ao cronograma.

Também, nessa disciplina, é produzido o **Plano de Métricas**, que define as metas de medição, as métricas associadas e as métricas primitivas a serem coletadas no projeto para monitorar o seu andamento, e o **Registro de Revisão**, que é criado para capturar os resultados da revisão de um artefato de projeto. Um Registro de Revisão é o formulário que é preenchido para cada revisão.

Sobre as ferramentas *Rational*, temos o ***Rational Rose***, que é uma ferramenta gráfica para desenvolvimento e modelagem de componentes, o ***Requisite Pro***, para controle dos requisitos, e o *Clear Quest*, para controle das mudanças dos mesmos.

CONCLUSÕES

O Gerenciamento de Requisitos constitui-se em uma atividade central e complexa para os projetos de desenvolvimento de *software*. O principal objetivo do processo de definição de requisitos é concluir, com êxito, um acordo entre quem solicita e quem desenvolve, estabelecendo, clara e rigorosamente, o que deverá ser produzido.

Este trabalho mostrou que, além de um processo claro de definição de requisitos, é importante o processo de Gerência de Requisitos, haja vista que os mesmos sofrem evoluções (refinamentos e modificações) ao longo do projeto. Observou-se que o CMM apresenta uma clara preocupação com este aspecto, definindo uma KPA especificamente para esta disciplina.

Ao longo deste trabalho, foi evidenciado que o RUP apresenta todas as ferramentas necessárias para o processo de Gerência de Requisitos. Ao contrário do que se pensava inicialmente, apenas as disciplinas de Modelagem de Negócios e Requisitos não são suficientes para atender à KPA Requisitos do nível 2 do CMM. Foram necessários o emprego de ferramentas automatizadas (*ReqPro*, *Rational*

Rose, *Clear Case* e *Clear Quest*) e artefatos de outras disciplinas (Gerência de Projetos, Ambiente e Gerência de Configuração e Mudanças). A tabela apresentada mostrou um resumo do tipo de requisito utilizado para atender à KPA Requisitos: os artefatos, ferramentas e atividades, e a qual disciplina os mesmos pertencem,

Finalmente, pôde-se comprovar que as atividades, papéis e ferramentas apresentadas pelo RUP mostraram-se suficientes para atender à KPA Requisitos do CMM.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] UEHARA, Irineu. *Software*: Qualidade Total. **Revista e-Manager,** São Paulo, p.14-18, nov., 2002.

[2] FIORINI, Soeli T.; STAA, Arndt Von; BAPTISTA, Renan Martins. **Engenharia de *Software* com CMM**. Rio de Janeiro: Brasport, 1998.

[3] MACHADO, Cristina A. Filipack; REINEHR, Sheila dos Santos; CALSAVARA, Alcides; BURNETT, Robert Carlisle. **Aderência do RUP à Norma NBR ISSO/IEC 12207**. Pontifícia Universidade Católica do Paraná, Curitiba, 2000.

[4] RATIONAL SOFTWARE CORPORATION. **RUP**. Versão 2002.05.00. 2001.

[5] GRADY, Robert. *Practical software metrics for project management and process improvement*. Prentice Hall, 1992.