

Mapeamento dos Conceitos do guia do CMMI em notações do SPEM no contexto da Definição do Processo de Software

SANDRO RONALDO BEZERRA OLIVEIRA ^{1,2}
ALEXANDRE MARCOS LINS DE VASCONCELOS ¹
RODRIGO CAVALCANTE MENDES ¹

¹Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Caixa Postal 7851 – 50732-970 – Recife – PE – Brasil
Fone / Fax: (+55 81) 2126-8430

²Centro de Ciências Exatas e Tecnologia – Universidade da Amazônia (UNAMA)
Av. Alcindo Cacela, 287, 66060-902 – Belém – PA – Brasil
Fone: (+55 91) 4009-3000

(srbo, amlv, rcm2)@cin.ufpe.br

Resumo. Este trabalho propõe a modelagem dos conceitos do guia do CMMI (Maturity Levels, Discipline, Specific Goals, e outros) a partir das notações definidas pelo SPEM, como base para a modelagem de processos de software definidos em um Ambiente de Desenvolvimento de Software. Este mapeamento visa capturar informações sobre a aderência e compatibilidade do CMMI no SPEM, capturando os componentes do processo de software e os seus relacionamentos baseando-se na análise deste modelo. A motivação deste trabalho visa à composição de um meta-modelo de processo de software que possibilite definições de processo aderentes ao CMMI e SPEM. Uma área de processo do CMMI foi usada para a representação deste mapeamento.

Palavras-Chave: CMMI, SPEM, Processo de Software, Qualidade de Processo de Software, PSEE.

Mapping of CMMI Guide Concepts on SPEM Notations from Software Process Definition Context

Abstract. This paper considers the modeling of the CMMI guide concepts (Maturity Levels, Discipline, Specific Goals, and others) from the notations defined for the SPEM, as base for the modeling of defined software processes in a Software Development Environment. This mapping aims to capture information on the tack and compatibility of the CMMI in the SPEM, being captured the software process components and its relationships being based on the analysis of this model. The motivation of this work aims to the composition of a software process meta-model that makes possible process definitions adherent to the CMMI and SPEM. A CMMI Process Area was used for the representation of this mapping.

Keywords: CMMI, SPEM, Software Process, Quality Software Process, PSSE.

(Received March 23, 2006 / Accepted October 04, 2006)

1. Introdução

A importância da engenharia de software para o sucesso de um projeto de software tem se tornado

cada vez mais evidente. Áreas como Qualidade têm se tornado grandes parceiras na melhoria de processos de software e aumento da produtividade das empresas de desenvolvimento visto que grande

parte dos projetos não consegue entregar os produtos dentro dos padrões de qualidade, de cronograma e de custo estimados.

A inserção de processos de software com metodologias, procedimentos e práticas para a melhoria da qualidade e produtividade do desenvolvimento de sistemas, vêm se tornando um setor de mais investimento em organizações que desejam melhorar sua competitividade no mercado [15].

A aquisição de certificações em processo de softwares como a ISO 9000-3 [7] tem agregado valor competitivo às organizações essencialmente no panorama nacional e tem a finalidade de padronizar o processo, reduzindo custos. Apesar da ISO 9000-3 possuir uma boa aceitação no cenário brasileiro, empresas de tecnologia que desejam uma visibilidade internacional têm aderido a normas mais conceituadas e já difundidas no mercado como o CMM (Capability Maturity Model) [12]. Porém este modelo está entrando em descontinuidade devido a uma versão mais completa e robusta motivada pela composição de várias práticas, modelos e normas existentes, intitulado CMMI (Capability Maturity Model Integrated) [3].

O CMMI descreve o quê deve ser feito para melhoria gradual de processos, definindo níveis de maturidade que são organizados por áreas de processo que possuem objetivos que são alcançados por práticas que resultam em produtos de trabalho.

Este conjunto de modelos/normas de qualidade para o processo de software torna difícil avaliar a aderência e aspectos relevantes entre diferentes processos institucionalizados, o que poderia contribuir de forma significativa à melhoria dos processos como dos próprios modelos.

Na busca de contribuir para a melhoria do processo de software, surgiu a idéia de promover o desenvolvimento de um ambiente de implementação de processos de software, proposto por [10], desde sua concepção até a instanciação contemplando um projeto real. O ambiente visa a adaptação do meta-modelo de processo de software¹ para cada projeto e organização, apresentando sugestões, tanto de modelo

de ciclo de vida, quanto de atividades, procedimentos e ferramentas.

O objetivo deste trabalho é contribuir para a definição deste meta-modelo de processo, fazendo uma avaliação da aderência do modelo proposto pelo CMMI ao modelo SPEM – Software Process Engineering Metamodel Specification, e a estrutura de processo proposto pelo ambiente [9]. O SPEM é um modelo utilizado para especificar, definir processos e seus componentes. O modelo foi construído a partir de um subconjunto, chamado de SPEM Foundation, do metamodelo da UML 1.4.

Através do mapeamento entre os componentes, é possível verificar quais os pontos em que os modelos têm componentes semanticamente parecidos e outros que, para uma melhor aderência, requerem alguma condição, restrição ou até mesmo composição de mais de um componente do modelo.

Através dos estudos de caso foi possível ter um cenário mais prático de como os modelos se relacionam, permitindo analisar o grau de aderência e identificar componentes que não possuem equivalência em outro modelo. A partir dos resultados deste trabalho é possível extrair informações relevantes para uma proposta que satisfaça as principais práticas e normas da engenharia de software no âmbito de um modelo para processos de softwares, facilitando a proposição de ferramentas para extração de métricas e desenvolvimento de sistemas e aplicações, como o ambiente proposto.

O ambiente proposto por [10] tende a possibilitar a especificação dos processos de acordo com o domínio do projeto específico e das características da organização; a instanciação do processo de software para propriedades dos projetos; sua simulação a partir dos parâmetros de configuração (prazo, pressões, custo, recursos, etc.); uma execução (automação) mais próxima do que se espera para um processo organizacional; e uma avaliação a partir da coleta de métricas desta execução.

Além desta seção introdutória, o artigo apresenta outras quatro seções. A seção 2 aborda as propriedades que compõem o ambiente de implementação de processo de software. Na seção 3 tem-se o mapeamento dos componentes do CMMI no SPEM, tendo como referências a estrutura do processo de software definido ao ambiente. A seção 4 contém uma visão de uma parte do estudo de caso de uma área de processo do CMMI usando as notações

¹ Componente (*framework*) definido para unificar as terminologias das características (atividades, artefatos, perfis, etc.) que definem um processo de software, baseado em modelos/normas de qualidade para processo de software.

do SPEM. Finalmente, a seção 5 apresenta as considerações finais.

2. ImPProS – Um Ambiente de Implementação Progressiva de Processo de Software

O surgimento da tecnologia CASE (*Computer Aided Software Engineering*) - Engenharia de Software Auxiliada por Computador, exerceu um enorme impacto sobre a área. A idéia de utilizar software para auxiliar a produção de software foi bem recebida pelos desenvolvedores. As ferramentas CASE proporcionam uma sólida estrutura às metodologias e métodos de desenvolvimento de software. Os ambientes integrados de desenvolvimento de software, ou simplesmente ambientes de desenvolvimento de software (ADSs) representam uma evolução do conceito de CASE, definindo mecanismos de integração entre as ferramentas, evoluindo para apoiar todas as etapas do ciclo de vida.

Uma evolução significativa nos ADSs foi detectada com a tecnologia de processos de software. A automação do processo de software foi incorporada aos ADSs mais recentes tornando-os ADSs centrados em processo (ou orientados a processo), também conhecido na literatura como PSEE - Process-Centered Software Engineering Environment [13]. Estes ambientes constituem uma nova geração de ADS que suportam além da função de desenvolvimento de software, também as funções associadas de gerência e garantia da qualidade durante o ciclo de vida do software.

No entanto, em alguns casos, percebe-se ao longo da execução do processo a partir destes ambientes de desenvolvimento que sua implementação nem sempre perfaz a realidade das características da organização ou do projeto desenvolvido por esta. Assim, para ajudar uma organização na implementação progressiva de um processo de software, é útil fornecer apoio automatizado por meio de um ambiente capaz de suportar as fases que a literatura especializada propõe como necessárias. O termo “progressiva” decorre do fato de que a implementação do processo é aperfeiçoado com as experiências aprendidas na sua definição, simulação, execução e avaliação.

O ambiente proposto em [10], como projeto de Tese de Doutorado do CIn/UFPE, está sendo concebido (atualmente em desenvolvimento) com o objetivo principal de apoiar a implementação de um processo de software em uma organização. Dentro deste contexto podem ser caracterizados como seus objetivos específicos:

- Especificar um meta-modelo de processo de software a fim de definir uma terminologia única entre os vários modelos de qualidade de processo de software existentes, para uso do ambiente em seus serviços providos;
- Apoiar a definição de um processo de software para organização;
- Permitir a modelagem e instanciación deste processo;
- Permitir a simulação do processo a partir das características instanciadas para um projeto específico;
- Dar apoio à execução do processo de software tomando como base uma máquina de inferência;
- Possibilitar a avaliação dos critérios do processo de software;
- Apoiar a melhoria contínua do processo de software e o reuso através da realimentação e coleta das experiências aprendidas.

Vale ressaltar que todos os objetivos listados acima foram adaptados a partir da estrutura que compõe o meta-processo de software descrito em [14], das características propostas para a implementação de um processo de software [2] e do ciclo de vida para melhoria contínua de processo definido pelo Modelo IDEAL [8]. Para alcançar estes objetivos o ambiente foi concebido para adotar a arquitetura apresentada na Figura 1. Os componentes definidos na arquitetura do ambiente encontram-se resumidamente descritos nas subseções a seguir e seu detalhamento pode ser encontrado em [10].

2.1. Mecanismo de Interação com o Usuário

Neste mecanismo o foco está em prover aos usuários diferentes visões da mesma informação sendo definida e especificada, provendo interação para diferentes usuários do ambiente, ou seja, trabalha com as características da usabilidade no ambiente.

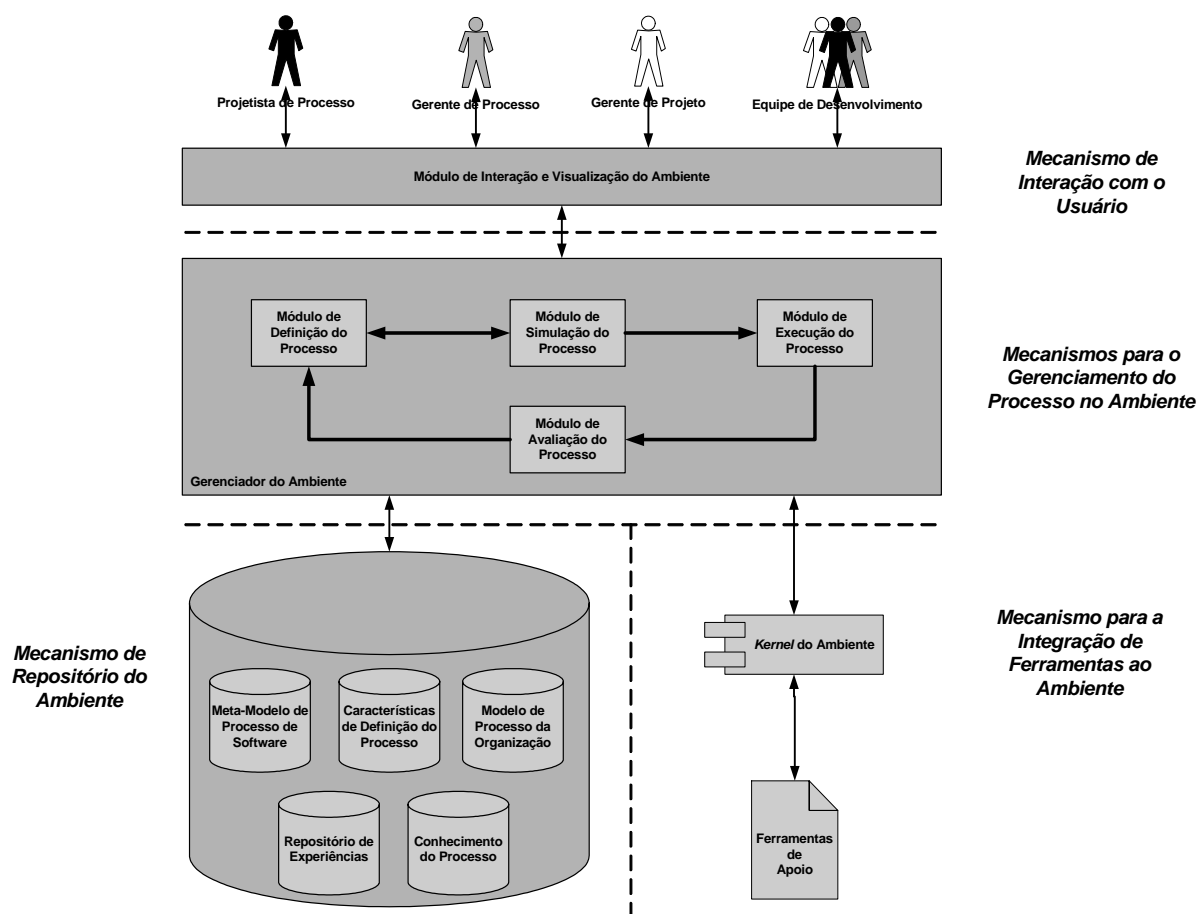


Figura 1. Arquitetura do Ambiente de Implementação de Processo de Software

2.2. Mecanismo para o Gerenciamento do Processo no Ambiente

Este mecanismo possui a responsabilidade de prover os serviços (módulos de definição, simulação, execução e avaliação do processo de software) especificados ao ambiente de forma automatizada, ou seja, possibilitar que os usuários do ambiente executem suas funções tendo como referencial um guia.

2.3. Mecanismo de Repositório do Ambiente

O foco deste mecanismo está em prover ao ambiente o sistema de gerenciamento dos seus objetos a partir de bases de dados que provejam o controle de evolução e manutenção dos componentes do processo de software.

2.4. Mecanismo para Integração de Ferramentas ao Ambiente

Este mecanismo provê a integração do ambiente com outras ferramentas de apoio ao processo de software e à execução do projeto de software, possibilitando a automação de atividades definidas no processo e a execução de alguns módulos do ambiente.

3. Mapeamento do CMMI nas notações do SPEM

Na definição de processos de software do ImPProS, adaptada do modelo definido por [15], inicialmente encontra-se o meta-modelo de processo de software, composto de componentes e dos relacionamentos entre esses que são oriundos do mapeamento de algumas normas e modelos de qualidade para processo de software (CMMI [3], SPICE – ISO 15504 [5], ISO 12207 [6]). O objetivo deste meta-modelo é determinar uma terminologia única para a definição de processos de software no ImPProS.

Por sua vez, a definição de um processo padrão estabelece uma estrutura comum a ser utilizada pela organização nos seus projetos de software e constitui a base para a definição de todos os seus processos. Dessa forma, estabelece-se um processo básico que servirá como ponto de partida para a posterior definição dos processos de software adequados às diferentes características de cada projeto, permitindo economia de tempo e esforço na definição de novos processos.

Tendo em vista que tipos de software diferentes possuem características distintas e requerem diferentes abordagens de desenvolvimento, o processo de software padrão da organização deverá ser adaptado (especializado) considerando-se as características relacionadas ao tipo de software (por exemplo, sistemas de informação) e ao paradigma de desenvolvimento utilizado (por exemplo, orientação a objetos). Assim, durante a etapa de especialização do processo padrão, atividades poderão ser adicionadas ou modificadas, de acordo com o contexto para o qual se está realizando a especialização.

A instanciação para projetos específicos consiste na adaptação de um processo especializado a um projeto, considerando-se as suas peculiaridades. Nesta etapa, são definidos o modelo de ciclo de vida, os métodos e as ferramentas que serão utilizadas no projeto, os recursos humanos e suas responsabilidades ao longo do processo e os artefatos (produtos) consumidos e gerados.

Estas três etapas de definição do processo (definição do processo padrão, especialização e instanciação do processo) consistem em adaptar o meta-modelo de processo para atender a objetivos específicos, através da análise das características organizacionais e de projetos de software. Esta seção, pretende descrever o mapeamento feito pela notação da modelagem de processo de software usada no ImPProS com o modelo de qualidade do CMMI para que se tenha uma análise do estrutural e comportamental dos conceitos definidos neste modelo de qualidade em relação ao SPEM.

3.1 Estrutura Padrão do Processo de Software do ImPProS

O ImPProS por possuir como uma de suas características a definição do processo de software sob a forma de um modelo de processo de software e sua representação diagramática, possui como estrutura geral de composição dos processos de software o modelo baseado nas definições de ontologias de processo de software de Falbo [4].

Processos são coleções de atividades relacionadas que têm lugar durante o desenvolvimento de um produto. Basicamente, um processo consiste de um conjunto estruturado de atividades e, por conseguinte, toda a infra-estrutura envolvida na realização destas (artefatos, procedimentos e recursos). Por sua vez Atividades são as tarefas ou trabalhos a serem realizados. Uma atividade requer recursos e pode consumir ou produzir artefatos. Para sua realização, uma atividade pode adotar um procedimento. Uma atividade pode ser decomposta em outras atividades. Além disso, atividades, em qualquer nível, podem depender da finalização de outras atividades, denominadas pré-atividades. O conceito de atividade está presente em todos os modelos de processo de software e são consideradas primitivas que geram artefatos a partir de artefatos de entrada auxiliados por recursos. Atividades podem corresponder a diferentes níveis, seja uma tarefa ou uma etapa do processo de desenvolvimento.

Para descrever as etapas do processo e as atividades a serem realizadas em cada etapa, surge o conceito de Modelo de Ciclo de Vida, que estrutura atividades e define abordagem de como organizar um projeto em fases. O ciclo de vida é iniciado quando um software é concebido até quando entra em desuso, ou seja, contém um conjunto de atividades de desenvolvimento, operação e manutenção. Aliado a este conceito temos a Combinação, a qual define a forma como um conjunto de fases de um modelo de ciclo de vida deve ser realizado e especifica o tipo de ordenação em que a estrutura pode ser: sequencial ou iterativa.

Os Artefatos são produtos de software produzidos ou consumidos por atividades durante a sua realização. São exemplos de artefatos: manuais de qualidade, manuais de revisão, diagramas de fluxos de dados, diagramas de objetos, código fonte, etc. Um artefato pode ser decomposto em outros artefatos (composição de artefatos). É a entrada ou produto de uma atividade, podendo ser artefatos de código, documentos ou componentes de software.

Já os Procedimentos são condutas bem estabelecidas e ordenadas para a realização de atividades. Alguns procedimentos podem ser parcialmente automatizados por ferramentas de software. São utilizados para auxiliar a realização das atividades, podendo ser direcionados a um tipo específico de atividade, devendo ser adequados a uma tecnologia de desenvolvimento e a um paradigma. Aliado a este conceito, temos o Padrão de Atividades que um procedimento deve sugerir para a execução

de uma atividade. Representa um comportamento em que decomposições de uma atividade têm em comum.

As pessoas, as ferramentas de software, os equipamentos, ou quaisquer outras infra-estruturas necessárias à execução de uma atividade, recebem o nome de Recurso. Um recurso humano, especificamente, desempenha um papel na execução das atividades do processo. São elementos necessários para a realização de uma atividade, tais como agentes humanos, equipamentos de hardware e ferramentas de software. Apóiam ou atuam na realização da atividade, mas não podem ser consideradas “matérias-primas” para a atividade, ou seja, apenas auxiliam o processo, mas não são incorporados ao produto de software sendo considerados recursos para a atividade.

Podemos encontrar, ainda, os conceitos de Paradigma de Desenvolvimento, que são princípios e conceitos que orientam o desenvolvimento (por exemplo: o estruturado e o orientado a objetos), e a Tecnologia de Desenvolvimento que representa a tecnologia a ser empregada no desenvolvimento do software (é o caso das tecnologias convencional de processamento de dados e de sistemas baseados em conhecimento). Por fim, para limitar e/ou restringir a execução das atividades definidas no processo, tem-se as Restrições, que especificam as regras de definição do processo de software.

3.2 Mapeamento do CMMI versus SPEM

Para representar diagramaticamente cada um dos itens definidos na seção 3.1, o ImPProS faz uso de uma linguagem de modelagem de processos a fim de possibilitar o seu projeto e sua posterior visualização, guia e avaliação. A linguagem usada trata-se do SPEM – Software Process Engineering Metamodel Specification [11], um modelo utilizado para especificar, definir processos e seus componentes. O modelo foi construído a partir de um subconjunto, chamado de SPEM Foundation, do metamodelo da UML 1.4. A linguagem SPEM oferece algumas representações e estereótipos para modelar seus principais elementos em diagramas UML. A descrição dos principais elementos que compõem o SPEM pode ser encontrada em [11].

Por outro lado o propósito do CMMI (Capability Maturity Model Integration) é fornecer guias para o melhoramento de processos e para o gerenciamento do desenvolvimento, aquisição, e manutenção de produtos e serviços [3]. O CMMI possui duas representações: contínua e em estágios. A representação contínua, similar a ISO/IEC 15504 [5],

oferece uma abordagem mais flexível para melhoria do processo, são focadas áreas de processo específicas diretamente relacionadas ao objetivo de negócio da organização. A representação em estágios, similar a SW-CMM [12], oferece um passo a passo detalhado para melhoria do processo, descreve a seqüência de execução das áreas de processo e estas são agrupadas em níveis de maturidade que quando alcançados indicam uma melhoria substancial do processo. O foco deste artigo é na representação em estágios do CMMI. Os componentes da representação em estágio do modelo CMMI são: Níveis de Maturidade, Áreas de Processo, Objetivos Específicos e Genéricos, Práticas Específicas e Genéricas, e Características Comuns. Um detalhamento das definições de cada um destes componentes pode ser encontrado em [3].

Assim, com base na Estrutura Padrão do Processo de Software do ImPProS, nas notações definidas pelo SPEM para representar diagramaticamente o processo e nas características do CMMI, a Tabela 1 apresenta um mapeamento que representa estruturalmente o modelo de referência dos processos de software no ImPProS [9]. Um melhor entendimento comportamental dos conceitos do mapeamento é definido após a tabela.

Tabela 1: Mapeamento da Estrutura Padrão do Processo de Software do ImPProS versus Notações do SPEM versus Componentes do CMMI

Estrutura Padrão do Processo do ImPProS (Modelo de Referência)	Notações do SPEM	Componentes do CMMI
Processo	Process	
	ProcessComponent	
Modelo de Ciclo de Vida	LifeCycle	
	Iteration	
Combinação	Phase ProcessPackage	Maturity Levels Discipline
Atividade	WorkDefinition	Process Area
	Discipline	Process Area
	Activity	Specific Practices Specific Goal
	Step	Subpractices
Artefato	WorkProduct Document UMLModel	Typical Work Product

Procedimento	Guidance Guideline Technique UMLProfile ToolMentor CheckList Template	Subpractices Generic Practice Elaboration Discipline Amplification
Recurso	ProcessPerformer ProcessRole	Stakeholder
Padrão de Atividades	Step	Specific Practice Specific Goal
Paradigma de Desenvolvimento		
Tecnologia de Desenvolvimento		
Restrições	ExternalDescription Goal Precondition ActivityParameter trace refersTo categorizes precedes impacts import governs assist perform	Generic Goal Generic Practice Purpose Statement Introductory Notes Related Process Areas

A representação de Processo no modelo de referência foi mapeada em dois componentes no SPEM, o Process e o ProcessComponent, que possuem uma semântica semelhante que expressam um conjunto estruturado de atividades para. O CMMI não possui nenhum componente equivalente, pois o mesmo sugere áreas de conhecimento e processos para a definição de processos do ciclo de vida de software.

O Modelo de Ciclo de Vida do modelo de referência pode ser representado no SPEM, como a junção de 2 (dois) componentes, o LifeCycle e Iteration, descrevendo a vida do software desde sua concepção até seu desuso, onde o LifeCycle delimita o tipo de ciclo de vida a ser executado (sequencial ou iterativo) e o Iteration especifica como este encontra-se organizado. O CMMI não aborda ciclo de vida no modelo, mas pode-se utilizar em composição com diferentes tipos de ciclos de vida.

O componente Combinação é representado por Phase e ProcessPackage no SPEM, onde sua

representação no CMMI está focada em níveis de maturidade, correspondendo a fase de maturidade do processo, em composição com ProcessPackage, que representa a que categoria/família a Área de Processo pertence, ou seja, a área de conhecimento da área de processo.

Uma Atividade no modelo de referência pode ser representada de maneiras diferentes no SPEM e no CMMI, dependendo de sua granularidade. Uma Process Area no CMMI é representada como um WorkDefinition, descrevendo todo o trabalho realizado para alcançar o objetivo. Esta mesma Process Area no CMMI define uma Discipline no SPEM, pois agrupa práticas de acordo com um tema em comum, ou seja, apenas é um informativo sobre o tema a ser tratado no processo de software. Uma Activity no SPEM é considerada uma Specific Practice e uma Specific Goal no CMMI por representar uma parte do trabalho a ser realizado e um objetivo da área de processo usada.

Um Step, do SPEM, por representar uma atividade atômica, identifica dois conceitos no modelo de referência: Atividade, quando esta representar uma única atividade de forma ordenada para especificar como uma macro-atividade pode ser executada, e, assim sendo, no CMMI equivale a Subpractices já que equivale a uma atividade atômica; e Padrão de Atividades, quando esta representar uma coleção de atividades desordenadas que poderão servir como sugestão para a detecção de sub-atividades de uma macro-atividade, quando a esta for associado um método como procedimento, e, assim sendo, equivale no CMMI a Specific Practice e Specific Goal pois engloba práticas e objetivos de uma área de processo usada.

Um Artefato no modelo de referência é considerado um WorkProduct no SPEM e Typical Work Product no CMMI e possui a mesma semântica. No caso do SPEM ele ainda possui componentes especiais para artefatos em padrão de documentos e para modelos em UML, o UMLModel e o Document, no entanto estas representações são subtipos do WorkProduct.

Um Procedimento são guias ou itens que auxiliam o processo, sendo representados pelo componente Guidance e suas extensões como Guideline, Template, Technique, UMLProfile, ToolMentor e CheckList no SPEM, e no CMMI podem ser Subpractices, Generic Practices Elaborations ou Discipline Amplifications, dependendo da semântica da descrição destes componentes.

A representação de Recurso no modelo de referência é mapeado no SPEM como um ProcessRole e um ProcessPerformer, pois são necessários para a realização das atividades. No caso do CMMI são definidos por Stakeholders.

Um Padrão de Atividade é representado como um Step no SPEM, e no CMMI como uma Specific Practice e Specific Goal, por representar um comportamento semelhante em decomposições de uma atividade, como já definido anteriormente.

Paradigmas e Tecnologias de Desenvolvimento do modelo referência não possuem representação em componentes tanto no SPEM como no CMMI, sendo apenas considerados como propriedades informativas no momento da definição do processo.

As Restrições são representadas no SPEM basicamente por componentes do pacote Dependency_Foundation e no CMMI podem ser, dependendo do contexto agregado, por Generic Goals, Generic Practices, Purpose Statement, Introductory Notes, Related Process Areas, por limitarem, restringirem a execução das atividades nas áreas de processo.

4. Estudo de Caso

No estudo de caso relatado nesta seção foi avaliada uma Área de Processo do modelo CMMI – Gerência de Requisitos, pertencente ao nível de maturidade 2 na representação por estágios, verificando sua aderência aos componentes que compõem o SPEM. O estudo de caso completo, contemplando todos os componentes da Área de Processo Gerência de Requisitos pode ser encontrado em [9], bem como um outro estudo de caso usando a área de processo Gerência de Configuração do CMMI.

A Área de Processo de gerência de requisitos pode ser representada no SPEM como mostra a Figura 2. A *Gerência de Requisitos* é considerada no SPEM uma disciplina que pertence ao pacote de processos (ProcessPackage) de Engenharia de Processos e classificada na fase (Phase) do Nível de Maturidade 2.

O propósito (Purpose no CMMI) desta PA é gerenciar os requisitos de produtos dos projetos e dos componentes e identificar inconsistências entre os requisitos, planos de projeto e artefatos. Suas notas introdutórias são consideradas uma ExternalDescription e descreve com mais detalhes a Área de Processo. As Áreas de Processo relacionadas (Precondition no SPEM) são: Desenvolvimento de Requisitos; Solução Técnica; Planejamento de

Projeto; Gerência de Configuração; Monitoramento e Controle de Projeto; Gerência de Riscos.

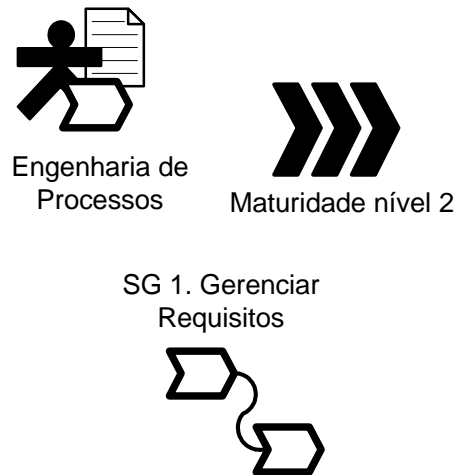


Figura 2: Estrutura da PA de Gerência de Requisitos no SPEM

SG 1. Gerenciar Requisitos

Esta Área de Processo possui na representação de estágios no CMMI apenas um objetivo específico - Specific Goal (SG) - que é chamado Gerenciar Requisitos, sendo este mapeado como um WorkDefinition no SPEM. Este objetivo específico tem como objetivo (Goal) manter um conjunto de requisitos atualizados e aprovados durante a realização do projeto. Ele faz referência (RefersTo) às Áreas de Processo de *Soluções Técnicas*, de *Desenvolvimento de Requisitos* e de *Monitoramento e Controle de Projeto*.

SP 1.2. Obter Concordância dos requisitos

A Figura 3 mostra a representação gráfica dos componentes que participam desta atividade. A SP 1.2 “Obter Concordância dos Requisitos” é considerada uma atividade (Activity) no SPEM e tem como objetivo (Goal no SPEM) que todos os participantes do projeto entrem em acordo em relação aos requisitos.

A amplificação referente ao *Guia para desenvolvimento de Produtos e Processos Integrados*, que ressalta a importância de que diferentes equipes que participam do projeto também devem acordar com os requisitos, é considerada um Guidance no SPEM. Os seguintes artefatos (WorkProducts no SPEM) participam ou são produzidos:

- Avaliações dos impactos dos requisitos;

- Documentar acordos feitos em relação aos requisitos e mudanças.

As sub-práticas para a realização da atividade *Obter Concordância dos Requisitos* são:

- Avaliar o impacto dos requisitos no que já foi acordado, (Technique no SPEM);
- Negociar e registrar o que foi acordado, (Technique no SPEM).

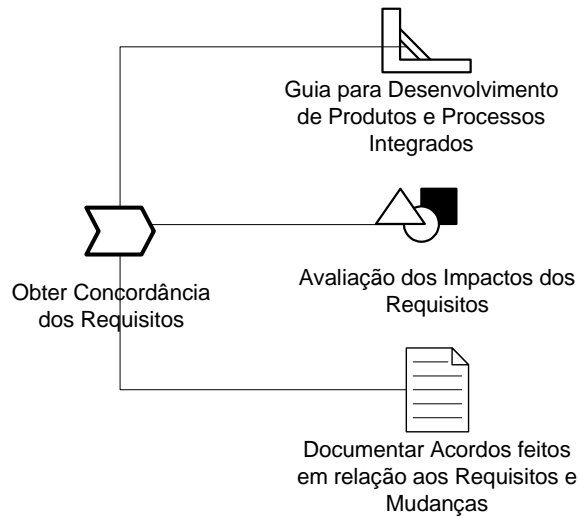


Figura 3: Representação da SP 1.2 no SPEM

SP 1.5. Identificar Inconsistências entre o Plano de Projeto e os Requisitos

A Figura 4 exibe a representação gráfica dos componentes que participam desta atividade. A SP 1.5-1 é considerada uma atividade (Activity) no SPEM e tem como objetivo, Goal no SPEM, identificar inconsistências entre o plano de projeto, artefatos e requisitos. A Área de Processo de Monitoração e Controle de Projeto é referenciada para fornecer mais detalhes sobre monitoramento e controle de artefatos e planos de projeto, portanto uma dependência do tipo *RefersTo* no SPEM.

Nesta prática os seguintes artefatos (WorkProducts no SPEM) participam ou são produzidos:

- Lista de Critérios para diferenciar os requisitos fornecidos;
- Resultado entre a análise e os critérios;
- Critérios de avaliação e aceitação dos requisitos;
- Um conjunto de requisitos aprovados.

Para a realização dessa atividade, foram estabelecidas algumas sub-práticas:

- Estabelecer critérios para diferenciar os requisitos dos fornecedores. (Step no SPEM)
- Estabelecer critérios objetivos para a aceitação dos requisitos. (Step no SPEM)
- Analisar os requisitos para garantir que os critérios estabelecidos foram alcançados. (Technique no SPEM)
- Obter um entendimento dos requisitos dos fornecedores para que os participantes do projeto possam concordar com eles. (Step no SPEM)

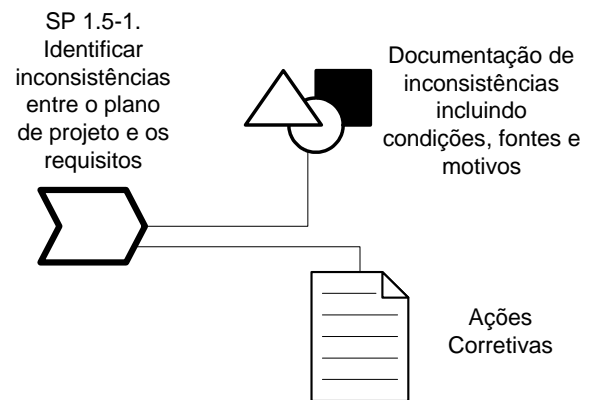


Figura 4: Representação da SP 1.5 no SPEM

Foi fácil perceber, que os estudos de caso realizados possibilitaram uma análise do mapeamento do guia do CMMI nas notações do SPEM no que tange aos aspectos da aderência do meta-modelo do processo proposto pelo ImPProS aos modelos de processo definidos a partir dele, usando como base o modelo CMMI. Este trabalho nos proveu uma base para análise das demais normas/modelos de qualidade constantes no ImPProS, conforme listado no início da seção 3.

5. Conclusões

Este trabalho teve como objetivo avaliar a aderência entre o modelo CMMI e o SPEM, fazendo um estudo entre os componentes que constituem cada modelo a partir de suas semânticas e aplicações.

O modelo CMMI por ser mais voltado a parte estrutural do processo, possui carências de

representação quanto à modelagem de processos com um maior grau de especificação, onde o SPEM possui muitos componentes para representar esse aspecto. Em contrapartida, os aspectos comportamentais podem ser modelados de forma mais apropriada no SPEM, mas estruturalmente o CMMI é mais representativo.

A carência de componentes do tipo Nota, possibilitando um melhor detalhamento, no SPEM como componentes de processo e ciclo de vida no CMMI demonstra que apesar de um grau de aderência significativo podem ganhar mais representatividade se seus resultados forem combinados. Esta avaliação entre os modelos contribuiu para a concepção do meta-modelo de processo de software, em uso pelo ImPProS, fornecendo uma visão mais ampla de semelhanças e aspectos particulares de cada modelo.

Um outro trabalho, relacionado à modelagem gráfica de processos de software foi realizado a fim de propor ao ImPProS a representação diagramática dos processos de software mediante ao uso de uma ferramenta CASE de modelagem de processos [1], e a composição de um arquivo XML para a exportação do processo de software gerado pela ferramenta. Isto se torna eficaz quando os processos são representados de forma lógica a fim de que máquinas de processo possam, através de uma base de conhecimento de processos, sugerir processos e melhorias para os já existentes.

Referências Bibliográficas

- [1] Araujo, R. M.: Construção Gráfica de Processos de Desenvolvimento e Geração de uma Ontologia de Processo de Software, Orientador Prof. Alexandre Vasconcelos, Trabalho de Graduação, CIN/UFPE, 2005.
- [2] Balduino, R.: Implementação de um processo de desenvolvimento de software: uma abordagem passo-a-passo, Rational Software White Paper, 2002.
- [3] Chrissis, M. B., Konrad, M. and Shrum, S.: CMMI Guidelines for Process Integration and Product Improvement, Addison-Wesley, 2003.
- [4] Falbo, R. A.: Integração de Conhecimento em um Ambiente de Desenvolvimento de Software”, Orientadora Ana Regina Cavalcanti da Rocha, Tese de Doutorado. COPPE/UF RJ, 1998.
- [5] ISO/IEC TR 15504, Parts 1-9: Information Technology – Software Process Assessment, International Organization for Standardization, 1998.
- [6] ISO/IEC TR 12207: Amendment: Information Technology – Amendment to ISO/IEC 12207, PDAM 3 version, 2000.
- [7] ISO/IEC TR 9000-3: The Application of ISO 9000 Series Standards to Software – Guidelines in Plain English, International Organization for Standardization, 1997.
- [8] Mcfeeley, B.: IDEALSM: A User’s Guide for Software Process Improvement, Software Engineering Institute Handbook. Carnegie Mellon University. CMU/SEI-96-HB-001, 1996.
- [9] Mendes, R. C.: Modelagem e Avaliação do CMMI no SPEM para Definição de um Meta-Processo de Software, Orientador Prof. Alexandre Vasconcelos, Trabalho de Graduação, CIN/UFPE, 2005.
- [10] Oliveira, S., Vasconcelos, A., Rouiller, A. C.: Uma Proposta de um Ambiente de Implementação de Processo de Software, Artigo publicado na Revista InfoComp – Revista de Ciência da Computação da UFLA – vol. 4, n. 1, Lavras-MG, 2005.
- [11] OMG – Object Management Group: SPEM – Software Process Engineering Metamodel Specification, version 1.1, formal/05-01-06, 2005.
- [12] Paulk, M.C., Weber, C.V., Curtis, B., Chrissis, M.B.: Capability Maturity Model for Software Version 1.1”, SEI (Software Engineering Institute) – Carnegie Mellon, USA, 1993.
- [13] Reis, R. Q.: Reutilização de Processos de Software, Orientador Daltro Nunes. Exame de Qualificação do Doutorado. PPGC–UFRGS, 2000.
- [14] Reis, C. A. L.: Ambientes de Desenvolvimento de Software e seus Mecanismos de Execução de Processos de Software, Orientador Daltro Nunes. Exame de Qualificação do Doutorado. PPGC–UFRGS, 2000.
- [15] Rocha, A. R. C., Maldonado, J. C. and Weber, K. C.: Qualidade de software: teoria e prática, São Paulo: Prentice-Hall, 2001.