Práticas do CMMI® como regras de negócio

Gisele P. Morgado
Ingrid Gesser
Denis S. Silveira
Fernando S. P. Manso
Priscila M. V. Lima
Eber A. Schmitz
UFRJ

Resumo

A busca pela qualidade de *software* leva à adoção de boas práticas no processo do desenvolvimento de *software*, como as práticas especificadas pelo modelo de maturidade do CMMI. A implantação do CMMI, entretanto, constitui um processo penoso e demorado. Além disso, a qualidade deste processo afeta diretamente os resultados obtidos. Este artigo explora a possibilidade de formalização das práticas do CMMI através do seu mapeamento para Regras de Negócio. A viabilidade desta proposta é estudada através do mapeamento da parte do CMMI relativa à área Gerência de Requisitos. As vantagens desta abordagem encontram-se tanto no aumento da qualidade do processo de certificação/auditoria do CMMI, quanto na melhoria da qualidade do sistema de informação que apóia o processo de desenvolvimento de *software*.

Palavras-chave

Qualidade, Regras de Negócio, CMMI, gerência de requisitos.

CMMI® Practices as Business Rules

Abstract

The search for software quality results in the adoption of best practices in the software development process, such as the practices specified by the CMMI maturity model. However, the CMMI implementation constitutes a difficult and slow process. Besides, the quality of such a process directly affects the results obtained. This article explores the possibility of formalizing the CMMI practices via their mapping into Business Rules. This proposal viability was studied through the mapping of the CMMI part that concerns the Requirements Management area. This approach implies in an improvement of the CMMI certification/audit process and of the information system that supports the software development process.

Key words

Quality, Business Rules, CMMI, requirements management.

INTRODUÇÃO

"*Every business is a software business*" (HUMPHREY, 2002). Esta frase evidencia a importância do *software* dentro de uma organização e nos leva a uma forte preocupação quanto à qualidade do processo de desenvolvimento de *software*. Para obter esta qualidade, vários conjuntos de recomendações e normas vêm sendo propostos. Dentre estes, destacam-se as normas *ISO* (*International Organization for Standardization*) (MARSHALL, 2003), o *PMBOK* (*Project Management Body of Knowledge*) (PMI, 2004), o *CMM* (*Capability Maturity Model for Software*) (PAULK *et al.*, 1995) e o *CMMI* (*Capability Maturity Model Integration*) (CHRISSIS; KONRAD; SHRUM, 2003).

A adoção das práticas recomendadas pelo CMMI por uma organização implica na implantação, implícita ou explícita, de novas Regras de Negócio.

O *CMMI* é um modelo de maturidade para o desenvolvimento e manutenção de *software* e dos serviços que abrangem o ciclo de vida do produto, desde sua concepção até a sua entrega e manutenção. Este modelo dá ênfase às disciplinas de engenharia de sistemas e também à engenharia de *software* e à integração necessária para construir e manter os produtos de forma abrangente. Este modelo oferece um conjunto de boas práticas agrupadas de acordo com áreas de atividades correlatas e níveis de maturidade. Estes níveis correspondem a etapas progressivas de eficácia gerencial e se apresentam como um caminho evolucionário para qualquer organização que pretenda melhorar seus processos de desenvolvimento e manutenção de *software* (CHRISSIS; KONRAD; SHRUM, 2003). Implantar um determinado nível do *CMMI* dentro de uma organização constitui, entretanto, um processo penoso e demorado. Além disso, a qualidade da implementação do *CMMI* afeta diretamente os benefícios obtidos.

Do ponto de vista organizacional, as empresas vêm buscando cada vez mais explicitar os princípios que as governam através das chamadas Regras de Negócio. As Regras de Negócio são uma forma de se expressar o conhecimento organizacional (VON HALLE, 2002). Elas determinam como o negócio deve operar, mantendo a estrutura do negócio, controlando ou influenciando algum aspecto do negócio (BRG, 2000). Segundo (LEITE; LEONARDI, 1998), sob o ponto de vista do desenvolvimento de *software*, as Regras de Negócio podem ser consideradas como requisitos de um projeto, pois deverão ser implementadas nas aplicações informatizadas que dão suporte às operações de um negócio. Assim, a abordagem ao desenvolvimento de *software* baseada em Regras de Negócio consiste, basicamente, em tratá-las como requisito prioritário.

Esta abordagem tem duas motivações principais. Em primeiro lugar, a explícita consideração das Regras de Negócio como requisito prioritário tende a assegurar o alinhamento entre os objetivos da organização e os seus sistemas (ERIKSSON; PENKER, 2000). Em segundo lugar, as Regras de Negócio são naturalmente mais familiares aos clientes e usuários do que os modelos usuais de captura de requisitos. Tal familiaridade tende a aumentar sua participação e responsabilidade na especificação dos requisitos e, ao mesmo tempo, tende a reduzir as falhas de tradução dos mesmos.

A adoção das práticas recomendadas pelo *CMMI* por uma organização implica na implantação, implícita ou explícita, de novas Regras de Negócio. Este artigo explora uma promissora combinação destas duas iniciativas, o *CMMI* e as Regras de Negócio, propondo a formalização das práticas do primeiro através de um conjunto de Regras de Negócio.

Para apoiar esta abordagem baseada em Regras de Negócio foi desenvolvida uma ferramenta chamada *RÉGULA* (DIAS *et al.*, 2004; ARAÚJO *et al.*, 2006). A ferramenta provê uma forma estruturada para capturar, armazenar e recuperar as informações sobre as Regras de Negócio, facilitando o trabalho de especificação e implementação destas regras.

Este artigo pretende, em síntese, mapear o *CMMI* para um conjunto de Regras de Negócio através da ferramenta *RÉGULA*. E, como as Regras de Negócio podem ser tratadas como requisitos, este artigo pretende também construir uma especificação de um sistema de gerência de desenvolvimento e manutenção de *software* de acordo com as práticas do *CMMI*. O principal objetivo é contribuir para a disseminação do *CMMI*, facilitando sua adoção e certificação.

O presente artigo encontra-se organizado da seguinte forma. A segunda seção apresenta o contexto no qual este trabalho se insere: o modelo *CMMI* e as Regras de Negócio. A terceira seção trata das representações das Regras de Negócio e da ferramenta *RÉGULA*. A quarta seção apresenta a metodologia utilizada para o mapeamento das práticas da área de Gerência de Requisitos do *CMMI* para as Regras de Negócio. A quinta seção apresenta o resultado deste mapeamento. E, finalmente, a sexta seção apresenta as conclusões e as perspectivas de trabalho futuro.

CONTEXTUALIZAÇÃO

Antes de descrever detalhes mais específicos do trabalho, alguns conceitos envolvidos necessitam ser discutidos. Os itens *CMMI – Capability Maturity Model Integration* e Regras de Negócio apresentam resumidamente estes conceitos.

CMMI – Capability Maturity Model Integration

O *CMMI* consiste nas melhores práticas direcionadas ao desenvolvimento e à manutenção de produtos e dos serviços, abrangendo todo o ciclo de vida do produto, desde sua concepção até a sua entrega e manutenção (CHRISSIS; KONRAD; SHRUM, 2003). Muitas organizações no mundo todo têm adotado este modelo com o objetivo de possibilitar a elevação da maturidade da capacidade de suas equipes nas atividades relacionadas ao *software*.

O modelo de maturidade *CMMI* descreve um caminho evolucionário, que começa com processos imaturos (inicial) e segue até um processo maduro e disciplinado (otimizado), onde é possível o controle do processo de produção de *software* por meio de métricas e modelos estatísticos. No nível 2 de maturidade de *software* (gerenciado), os projetos da organização garantem que os requisitos são gerenciados e que os processos são planejados, executados, medidos e controlados. A disciplina do processo refletida pelo nível de maturidade 2 ajuda a garantir que as práticas existentes sejam mantidas mesmo em situações de *stress*.

Neste nível de maturidade, o estado dos produtos de trabalho e a entrega de serviços são visíveis em pontos predeterminados. Os compromissos são estabelecidos entre os interessados (*stakeholders*) relevantes e são revisados sempre que necessário. Os produtos de trabalho são controlados adequadamente e satisfazem o processo especificado, as descrições, os padrões e os procedimentos. As práticas recomendadas agrupam-se segundo suas interdependências em áreas de processo. As áreas de processo referentes ao nível 2 de maturidade são: Gerência de Configuração (*Configuration Management*), Medição e Análise (*Measurement and Analysis*), Monitoramento e Controle de Projeto (*Project Monitoring and Control*), Planejamento de Projeto (*Project Planning*), Controle de Qualidade (*Process and Product Quality Assurance*), Gerência de Requisitos (*Requirements Management*) e Gerência de Fornecedores (*Supplier Agreement Management*).

Este trabalho restringe-se à área de Gerência de Requisitos do nível 2. A escolha deve-se em parte à relativa independência da área, uma vez que suas recomendações referem-se à fase inicial do projeto e a sua relativa abrangência, uma vez que a área exerce controle sobre os requisitos ao longo de todo o processo de desenvolvimento e manutenção. O mapeamento das práticas dessa área em Regras de Negócio é apresentado em Mapeando as práticas da área de Gerência de requisitos.

Regras de Negócio

Uma Regra de Negócio é uma sentença que define ou qualifica algum aspecto do negócio, representando o conhecimento dos especialistas do negócio (BRG, 2000). Através das Regras de Negócio é possível garantir a estrutura do negócio ou influenciar o comportamento do mesmo.

Para apoiar esta abordagem baseada em Regras de Negócio foi desenvolvida, por nosso grupo de pesquisa, uma ferramenta chamada RÉGULA.

Uma outra forma de entender as Regras de Negócio é classificá-las. Dentre as diversas propostas de classificação de Regras de Negócio, a mais utilizada encontra-se originalmente em BRG (1997). Segundo este modelo, as Regras de Negócio podem ser agrupadas em seis categorias: termos, fatos, cálculos, derivações, restrições e habilitações de ação. Os termos constituem os elementos básicos da linguagem utilizada para expressar as Regras de Negócio, onde a própria definição de um termo é considerada como uma regra. Fatos descrevem a natureza ou estrutura operacional de um negócio, relacionando os termos uns aos outros. Os cálculos e derivações determinam como um conhecimento ou informação pode ser transformado em outro, através de fórmulas ou mudanças de estado. Já as restrições, conforme o nome indica, restringem algum comportamento do negócio, estando relacionadas a decisões sobre quais dados podem ou não ser atualizados. As habilitações de ação podem ser vistas como regras dedutivas, de raciocínio encadeado à frente, representadas através de um par contendo uma condição e respectiva ação. Vale notar que tal classificação foi ligeiramente modificada em sua versão posterior (BRG, 2000).

A documentação e a formalização das Regras de Negócio constituem um importante ativo estrutural e intelectual para a organização, pois, desta forma, as Regras de Negócio podem ser mais facilmente divulgadas aos profissionais envolvidos, como também são aumentados o entendimento e tratamento uniforme e consistente do negócio por esses profissionais. Para facilitar/viabilizar o gerenciamento das

Regras de Negócio é utilizada a ferramenta *RÉGULA*, apresentada em A ferramenta RÉGULA.

REPRESENTAÇÃO E MANIPULAÇÃO DAS REGRAS DE NEGÓCIO

Uma das questões fundamentais sobre Regras de Negócio é a forma como elas são representadas. A seguir descrevemos duas linguagens, com diferentes níveis de abstração, para a representação das regras. E, na seção seguinte, apresentamos a ferramenta *RÉGULA*, que utiliza estas duas linguagens para permitir a manipulação das Regras de Negócio.

Representação em Português Estruturado e em Prolog

Dentre todas as alternativas de representação existentes para Regras de Negócio, a linguagem natural (texto livre) mostra-se como a forma mais simples e mais difundida. Entretanto, sua propensão a ambigüidades e imprecisões dá margem a interpretações dúbias ou mesmo incorretas. Assim, alguns autores (ROSS, 2000; VON HALLE, 2002; MORIARTY, 2002) argumentam que uma maneira adequada de lidar com as Regras de Negócio é aliar a fluência das linguagens naturais à precisão das linguagens formais. Nesse sentido, a abordagem mais interessante é a adotada por BRG (1997), que se utiliza de modelos de sentença.

Um modelo de sentença é uma seqüência padronizada de termos usados para montar Regras de Negócio. Nesse padrão, existe uma certa estrutura de termos textuais que deve ser seguida. Alguns dos termos são fixos (não podem ser alterados) enquanto outros podem ser alterados pelo usuário (de acordo com algumas normas). Cada tipo de regra deve ter um modelo de sentença particular característico. A tabela 1 apresenta os modelos de sentença utilizados para a representação das Regras de Negócio. Ressaltemos, no entanto, que o modelo de sentença aqui ilustrado constitui uma variante do proposto pelo BRG (1997), levando-se em

Tabela 1: Modelos de sentença utilizados pelo RÉGULA.

CATEGORIA	MODELO DE SENTENÇA
Fato	<termo> É CATEGORIA BÁSICA
	<termo1> É SINÔNIMO DE <termo2>
	<termo1> É SUBTIPO DE <termo2> [QUE <verbo_restrição1> <termo_relacionado_restrição1>, <verbo_restrição2> <termo_relacionado_restrição2> ... E <verbo_restriçãoN> <termo_relacionado_restriçãoN>]
	<termo1> TEM COMO ATRIBUTO <termo2>
	<termo> POSSUI COMO DOMÍNIO <domínio>
	<termo1> TEM COMO PARTE <termo2>
	<termo1> ESTÁ RELACIONADO A <termo2> POR <verbo_associação> [COM GRAU <M>..<N>]
	<termo> É DEFINIDO POR <lista_de_nomes>
Cálculo	<termo> É CALCULADO COMO <função>
Derivação	SE <condição>, ENTÃO <termo> É CONSIDERADO COMO <estado>
Habilitadora de ação	SE <condição>, ENTÃO EXECUTAR <ação>
Permissão	<termo1> TEM PERMISSÃO PARA <verbo> {<prep>} {<artigo>} <termo2>
	<termo1> TEM PERMISSÃO PARA <verbo> <comp> <valor> <termo2>
	<termo1> TEM PERMISSÃO PARA <verbo> <termo2> <comp> <valor>
Proibição	<termo1> NÃO TEM PERMISSÃO PARA <verbo> {<prep>} {<artigo>} <termo2>
	<termo1> NÃO TEM PERMISSÃO PARA <verbo> <comp> <valor> <termo2>
	<termo1> NÃO TEM PERMISSÃO PARA <verbo> <termo2> <comp> <valor>
Obrigação	<termo1> DEVE OBRIGATORIAMENTE <verbo> {<prep>} {<artigo>} <termo2>
	<termo1> DEVE OBRIGATORIAMENTE <verbo> <comp> <valor> <termo2>
	<termo1> DEVE OBRIGATORIAMENTE <verbo> <termo2> <comp> <valor>
	<termo1> DEVE OBRIGATORIAMENTE SER <comp> <valor>

consideração observações complementares realizadas na literatura da área, mais especificamente em (MORGAN, 2002). Os modelos de sentença encontram-se separados por categorias e a categorização utilizada é baseada na proposta BRG (1997).

Esta forma de representação estruturada tem, por um lado, a aparência da linguagem natural, sendo familiar aos usuários e, por outro lado, é uma representação consistente e não ambígua do conhecimento. Assim, torna-se possível a conversão das Regras de Negócio para uma representação mais formal e computacionalmente processável, como a linguagem de programação *Prolog* (BRATKO, 1990). O *Prolog* é uma linguagem que se baseia no subconjunto da lógica de primeira ordem composto pelas cláusulas de *Horn* (HOGGER, 1994). Sua utilização no tratamento das Regras de Negócio permite a recuperação das regras de maneira sistemática, a verificação parcial de inconsistências e a validação das regras. A seguir apresentamos a ferramenta *RÉGULA*, que dispõe de um mecanismo para a tradução automática das regras em Português Estruturado para o *Prolog*.

A ferramenta RÉGULA

O *RÉGULA* é uma ferramenta gratuita que foi desenvolvida para o gerenciamento das Regras de Negócio. Resumidamente, o *RÉGULA* possui as seguintes funcionalidades: captura de Regras de Negócio, consulta a permissões, proibições e obrigações, geração do modelo de informação e geração de regras executáveis (DIAS *et al.*, 2004; ARAÚJO *et al.*, 2006).

O módulo de captura de Regras de Negócio se divide em duas partes: a dos termos e fatos e a das demais regras. A parte da captura dos termos e fatos permite a sua definição através do estabelecimento dos sinônimos, das heranças, das restrições, dos atributos, das partes e dos relacionamentos entre os termos. Já a parte da captura das demais regras permite a definição dos cálculos, das derivações, das habilitações de ação, das permissões, das proibições e das obrigações do negócio. Nas duas partes, a captura é feita utilizando o Português Estruturado como forma de Representação (Figura 1).

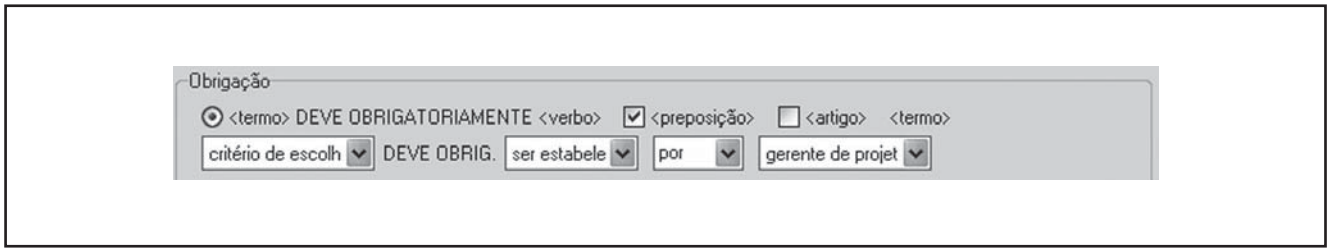
A fim de facilitar o processo de definição das regras, o *RÉGULA* dispõe também de um módulo de consultas onde é possível fazer inferências sobre as regras capturadas, possibilitando uma análise do comportamento entre os termos de acordo com as regras de permissões, proibições e obrigações definidas na ferramenta. As consultas deste módulo funcionam realizando pesquisas na base de regras por meio de uma interface entre o *RÉGULA* e uma máquina de inferências em *Prolog*.

O principal objetivo deste projeto é contribuir para a disseminação do CMMI, facilitando sua adoção e certificação.

As consultas funcionam da seguinte forma: o usuário digita uma pergunta em português estruturado e o módulo de consulta mapeia a pergunta para a sua representação interna em *Prolog*. Com o auxílio da hierarquia de classes e das permissões, proibições e obrigações cadastradas na base de regras, a questão é processada e o resultado obtido é traduzido de volta ao usuário no formato textual padrão. De posse dos resultados obtidos, pode-se ter um melhor entendimento sobre as regras, facilitando a administração das regras existentes e a construção de novas regras. Um exemplo de consulta pode ser visto na Figura 2 a seguir.

Para facilitar o trabalho do analista de sistemas, o *RÉGULA* possui um gerador automático do modelo de informação. Este módulo funciona gerando um arquivo na versão 1.1 do *XMI* (*XML Metadata Interchange*) (OMG, 2002) a partir da definição dos termos do negócio. Do arquivo *XMI* gerado podemos extrair o diagrama de classes de domínio, uma vez que o *XMI* é um padrão da *OMG* (*Object Management Group*) que permite o intercâmbio de metadados entre ferramentas de modelagem baseadas na *UML* (*Unified Modeling Language Specification*) (BOOCH; RUMBAUGH; JACOBSON, 1999).

Figura 1: Fragmento da tela de construção de regras do RÉGULA.



Além do gerador do modelo de informação, o *RÉGULA* dispõe também de um gerador de regras executáveis. Este gerador realiza a exportação das regras cadastradas para um arquivo *Prolog* que permitirá a execução destas regras. A utilização deste gerador ocorre da seguinte forma: as Regras de Negócio são escritas em Português Estruturado e traduzidas automaticamente para *Prolog*. O usuário do *RÉGULA* pode, a qualquer momento, exportar esta base de regras em *Prolog* e acessar esta mesma base para implementar sistemas que utilizem as Regras de Negócio cadastradas no *RÉGULA*.

O RÉGULA é uma ferramenta gratuita que foi desenvolvida para o gerenciamento das Regras de Negócio.

MÉTODO UTILIZADO NO MAPEAMENTO PARA REGRAS DE NEGÓCIO

O principal objetivo do mapeamento do *CMMI* para as Regras de Negócio é a diminuição da ambigüidade da linguagem natural, possibilitando a análise, descrição e revisão das mesmas, a partir de um padrão estruturado de sentenças. Na primeira etapa deste trabalho foi adotada uma abordagem de análise gradual das práticas e subpráticas da área de processo de Gerência de Requisitos do *CMMI* (CHRISSIS, KONRAD; SHRUM, 2003).

Após o estudo do modelo do *CMMI*, cada prática em conjunto com as respectivas subpráticas foi cuidadosamente analisada com o objetivo de explorar seu significado. Para isso, partindo-se do texto em inglês, foi elaborada a sua tradução para o português. Em seguida, foi identificado o modelo de sentença do Português Estruturado que melhor correspondia ao texto da subprática. Em muitos casos, para uma subprática original foi necessário utilizar mais de uma sentença em Português Estruturado, para que o seu significado integral fosse considerado. Além disso, devemos considerar como paradigma da formalização a tradução na forma atômica de todos os termos e regras extraídas de um texto em linguagem natural. Isso significa que os termos e regras sempre devem conter informações simples. Isso nos leva, na maioria das vezes, a formular mais de uma sentença formalizada para exprimir todo o significado de uma sentença em linguagem natural.

A partir dos modelos de sentença escolhidos, foram identificados os termos presentes em cada subprática. Estes termos foram sendo relacionados e, para cada termo identificado, foi elaborada uma definição. Sempre que possível, foram usadas as definições encontradas nos glossários fornecidos na literatura, e como fontes podemos citar os livros do CMM (PAULK *et al.*, 1995), do *CMMI* (CHRISSIS; KONRAD; SHRUM, 2003) e os padrões IEEE (1998). Em alguns casos, porém, as definições foram elaboradas com o auxílio de dicionários ou formuladas a partir da experiência de profissionais da área de engenharia de *software*. Para cada definição de termo, foi feita a verificação da mesma no contexto da subprática para evitar a introdução de inconsistências que tornassem a subprática incoerente.

Em seguida, todos os termos e regras obtidos como resultado da tradução das subpráticas foram cadastrados na ferramenta *RÉGULA*. O cadastramento levou à validação das sentenças obtidas durante a tradução, uma vez que a interface do construtor de sentenças do *RÉGULA* permite apenas a construção de sentenças em Português Estruturado. Além disso, o cadastramento tornou possível a execução das primeiras

Figura 2: Fragmento da tela de consulta do RÉGULA.

etapas em direção à construção de um sistema de gerência de requisitos aderente às regras que regem o *CMMI*. Estas etapas são: a geração automática do modelo de informação a partir dos termos definidos no *RÉGULA* e a geração automática das regras em *Prolog* que serão executas pelo sistema que implementará estas regras.

Por fim, foi realizada uma revisão dos termos e das sentenças. Para realizar a revisão, tivemos o auxílio do módulo de consultas do *RÉGULA*. As consultas foram executadas para permitir a análise do comportamento entre os termos de acordo com as regras cadastradas na ferramenta. Esta análise possibilitou a verificação de omissões, redundâncias e inconsistências, o que levou a refinamentos sucessivos tanto das sentenças, quanto dos seus termos.

MAPEANDO AS PRÁTICAS DA ÁREA DE GERÊNCIA DE REQUISITOS

O objetivo específico da área de Gerência de Requisitos é gerenciar a captura, trajetória, implementação e consistência dos requisitos. A área contém cinco práticas específicas. A primeira prática trata da recepção dos requisitos na organização desenvolvedora e sua validação para posterior atendimento. A segunda prática trata da organização do atendimento aos requisitos. A cada novo requisito ou mudança de requisito, o plano e os compromissos do projeto são revistos. A terceira prática trata das mudanças de requisitos. A quarta prática encarrega-se da rastreabilidade entre a fonte dos requisitos e sua implementação. A quinta e última prática trata da consistência entre os requisitos e o projeto (seu plano e seus produtos) (CHRISSIS; KONRAD; SHRUM, 2003).

Nas subseções a seguir, são apresentados os principais resultados da aplicação do método de mapeamento das práticas da área de Gerência de Requisitos, apresentadas em (CHRISSIS; KONRAD; SHRUM, 2003), para Regras de Negócio. Por considerações de espaço e escopo o mapeamento da última prática será omitido.

Prática 1: desenvolver um acordo sobre o significado dos requisitos com os provedores de requisitos.

Esta primeira prática diz respeito à recepção dos requisitos pela equipe de analistas de requisitos do projeto. A prática envolve dois conjuntos de critérios: um primeiro para selecionar provedores de requisitos e um segundo para avaliar requisitos. Embora não esteja explícito, é óbvio que apenas os provedores de requisitos poderão provê-los por parte da organização cliente. A prática visa desenvolver um acordo sobre o significado dos requisitos para que os mesmos possam ser passados para os projetistas. As regras e suas respectivas subpráticas estão relacionadas na Tabela 2.

Prática 2: obter o compromisso dos participantes do projeto para com os requisitos

A segunda prática diz respeito à passagem dos requisitos já devidamente aceitos e acordados entre os analistas de requisitos e os provedores de requisitos para os projetistas, com o objetivo de dar seqüência ao atendimento dos requisitos. A prática também se aplica às mudanças de requisitos. Obviamente, a cada momento, os projetistas estão compromissados com o atendimento aos requisitos anteriores. Antes, portanto, dos projetistas se comprometerem com o atendimento de um novo requisito ou de uma mudança de um requisito, o impacto do novo requisito (ou da mudança) nos compromissos já firmados deve ser avaliado. O atendimento do novo requisito pode ser incompatível com os compromissos já firmados. Neste caso, o conjunto de compromissos de projeto deve ser renegociado. As regras que regem esta prática são mostradas na Tabela 3.

Ao dispor das práticas do CMMI formalizadas como Regras de Negócio, o processo de auditoria e certificação torna-se menos subjetivo.

Prática 3: gerenciar as mudanças de requisitos ao longo do projeto

A terceira prática trata da gerência das mudanças de requisitos. As mudanças podem ser originadas na organização cliente ou geradas pelo próprio projeto. No primeiro caso as mudanças são tratadas como requisitos novos, ou seja, a elas se aplicam as práticas 1 e 2. No segundo caso, quando estas práticas não se aplicam propriamente, as mudanças são avaliadas por esta terceira prática. Além dessa avaliação, a prática 3 ocupa-se da completeza, do registro e da disseminação pelo projeto dos requisitos e suas mudanças. A Tabela 4 apresenta as regras que regem esta terceira prática.

Prática 4: manter rastreabilidade bidirecional entre os requisitos e os planos e produtos do projeto

A quarta prática trata da rastreabilidade da trajetória dos requisitos, desde sua formulação original como uma "necessidade do cliente" até sua implementação. Requisitos podem ser decompostos em requisitos derivados. Os requisitos folhas da árvore resultante são alocados a produtos ou a componentes de produtos que os implementam. Por outro

Tabela 2: Regras que regem a prática 1.

SUBPRÁTICA	REGRA EM PORTUGUÊS ESTRUTURADO	REGRA EM PROLOG
Estabelecer critérios para selecionar provedores de requisitos.	Um critério de escolha de participantes da provisão de requisitos DEVE OBRIGATORIAMENTE *ser estabelecido* pelo gerente de projeto.	obligation(critério_de_escolha_de_participantes_da_provisão_de_requisitos, gerente_de_projeto, ser_estabelecido, regra1).
Estabelecer critérios objetivos para a aceitação de requisitos.	Um critério de aceite de requisitos DEVE OBRIGATORIAMENTE *ser estabelecido* pelo gerente de projeto.	obligation(critério_de_aceite_de_requisitos, gerente_de_projeto, ser_estabelecido, regra2).
	O critério de aceite de requisitos DEVE OBRIGATORIAMENTE *ser cumprido por* um requisito do projeto de *software*.	obligation(critério_de_aceite_de_requisitos, requisito_do_projeto_de_software, ser_cumprido, regra3).
Analisar requisitos para assegurar que os critérios estabelecidos são atendidos.	Um requisito de projeto de *software* DEVE OBRIGATORIAMENTE *ser analisado* pelo gerente de projeto.	obligation(requisito_do_projeto_de_software, gerente_de_projeto, ser_analisado, regra4).
Chegar a um acordo sobre os requisitos com os provedores de requisitos para que os participantes do projeto possam se comprometer com eles.	SE for estabelecido um acordo informal sobre os requisitos do projeto de *software* ENTÃO o requisito de projeto de *software* É CONSIDERADO COMO aceito.	change_state(X, acordo_informal_sobre_os_requisitos_do_projeto_de_software, estabelecido) :- check_type_bring_data(X, acordo_informal_sobre_os_requisitos_do_projeto_de_software, D), check_state(acordo_informal_sobre_os_requisitos_do_projeto_de_software,estabelecido).
	SE um requisito de projeto de *software* é considerado aceito, ENTÃO o compromisso dos participantes do projeto É CONSIDERADO COMO estabelecido.	change_state(X, requisito_do_projeto_de_software, considerado_aceito) :- check_type_bring_data(X, requisito_do_projeto_de_software, D), check_state(requisito_do_projeto_de_software,considerado_aceito).

Tabela 3: Regras que regem a prática 2.

SUBPRÁTICA	REGRA EM PORTUGUÊS ESTRUTURADO	REGRA EM PROLOG
Avaliar o impacto dos requisitos nos compromissos existentes.	O impacto de um requisito de projeto de *software* DEVE OBRIGATORIAMENTE *ser avaliado* em relação aos compromissos de projeto de *software* existentes.	obligation(impacto_de_um_requisito_de_projeto_de_software, compromisso_de_projeto_de_software_existente, ser_avaliado, regra7).
Negociar e registrar os compromissos.	Os compromissos do projeto de *software* existentes DEVEM OBRIGATORIAMENTE *ser registrados* pelo gerente de projeto.	obligation(compromisso_de_projeto_de_software_existente, gerente_de_projeto, ser_registrado, regra8).
	Os compromissos do projeto de *software* DEVEM OBRIGATORIAMENTE *ser negociados* pelos participantes do projeto de *software*.	obligation(compromisso_de_projeto_de_software_existente, participante_do_projeto_de_software, ser_negociado, regra9).

lado, decomposições podem implicar acoplamentos, se um mesmo requisito ou requisitos relacionados forem alocados a produtos diferentes. Estes possíveis acoplamentos podem ser identificados pela rastreabilidade horizontal das trajetórias. Basicamente, a prática visa manter um esquema do conjunto dessas trajetórias de decomposição e alocação. As regras que regem esta prática são mostradas na Tabela 5.

Definição dos termos presentes nas práticas

Para expressar as regras relacionadas às práticas apresentadas neste exemplo, foram utilizados 17 (dezessete) termos. Para cada um destes termos foi criada uma definição através da construção de regras do tipo fato. Para ilustrar os resultados obtidos neste trabalho de definição dos termos, apresentamos a seguir uma parte dos termos e dos fatos utilizados na sua definição (Tabela 6). Apresentamos também o modelo de informação que corresponde a estes fatos (Figura 3).

CONSIDERAÇÕES FINAIS

Partindo da premissa de que processos de qualidade geram produtos de qualidade, este artigo propõe uma abordagem

Tabela 4: Regras que regem a prática 3.

SUBPRÁTICA	REGRA EM PORTUGUÊS ESTRUTURADO	REGRA EM PROLOG
Capturar todos os requisitos e mudanças de requisitos que sejam dados ou gerados pelo projeto.	O requisito de projeto de *software* DEVE OBRIGATORIAMENTE *ser capturado* pelo gerente de projeto.	obligation(requisito_do_projeto_de_software, gerente_de_projeto, ser_capturado, regra10).
	O requisito de projeto de *software* DEVE OBRIGATORIAMENTE *ser registrado* pelo gerente de projeto.	obligation(registro_do_requisito_de_projeto_de_software, gerente_de_projeto, ser_registrado, regra11).
	Uma alteração de requisito de projeto de *software* DEVE OBRIGATORIAMENTE *ser capturada* por um projeto de *software*.	obligation(alteração_de_requisito_de_projeto_de_software, projeto_de_software, ser_capturado, regra12).
	A alteração de requisito de projeto de *software* DEVE OBRIGATORIAMENTE *ser registrada* pelo gerente de projeto.	obligation(alteração_de_requisito_de_projeto_de_software, gerente_de_projeto, ser_registrado, regra13).
Manter o histórico da mudança de requisitos com justificativa das mudanças.	O racional das alterações de requisitos de projeto de *software* DEVE OBRIGATORIAMENTE *ser registrado* pelo gerente de projeto.	obligation(racional_das_alterações_de_requisitos_de_projeto_de_software, gerente_de_projeto, ser_registrado, regra14).
	O histórico das alterações de requisitos de projeto de *software* DEVE OBRIGATORIAMENTE *ser registrada* pelo gerente de projeto.	obligation(histórico_das_alterações_de_requisitos_de_projeto_de_software, gerente_de_projeto, ser_registrado, regra15).
Avaliar o impacto das mudanças de requisitos da perspectiva dos clientes e usuários (*stakeholders*) relevantes.	A análise de impacto das alterações de requisitos de projeto de *software* DEVE OBRIGATORIAMENTE *ser realizada* pelo gerente de projeto.	obligation(análise_de_impacto_das_alterações_de_requisitos_de_projeto_de_software, gerente_de_projeto, ser_realizado, regra16).
Tornar os dados sobre os requisitos e mudanças disponíveis para o projeto.	O registro do requisito de projeto de *software* DEVE OBRIGATORIAMENTE *ser realizado* pelo gerente de projeto.	obligation(registro_do_requisito_de_projeto_de_software, gerente_de_projeto, ser_realizado, regra17).
	O registro da alteração do requisito de projeto de *software* DEVE OBRIGATORIAMENTE *ser realizado* pelo gerente de projeto.	obligation(registro_da_alteração_do_requisito_de_projeto_de_software, gerente_de_projeto, ser_realizado, regra18).

inovadora para a implementação de modelos de melhoria na qualidade do processo de desenvolvimento de *software*, através da utilização de Regras de Negócio. Como prova de conceito, é apresentado o mapeamento de parte das práticas do *CMMI* da área de Gerência de Requisitos para Regras de Negócio, mais especificamente, para as representações do *RÉGULA*. Tal mapeamento implica em dotar as práticas, seus termos e fatos, de algum formalismo; o que permite a remoção de ambigüidades, inconsistências e redundâncias. Além disso, o mapeamento torna possível a utilização de algumas funcionalidades providas pelo *RÉGULA*, como o gerador de regras em *Prolog*, o gerador do modelo de informação e o mecanismo de consulta às regras.

A abordagem apresentada acarreta, principalmente, dois tipos de melhoria: na qualidade do processo de certificação e na qualidade do sistema de informação de suporte ao processo de desenvolvimento de *software*. Ao dispor das práticas do *CMMI* formalizadas como Regras de Negócio, o processo de auditoria e certificação torna-se menos subjetivo. Adicionalmente, sendo as Regras de Negócio parte fundamental dos requisitos de sistemas de informação, a formalização facilita a construção de um sistema de informação de apoio ao processo de desenvolvimento de *software*.

Em andamento encontra-se o mapeamento do conjunto de práticas relativas à área de Planejamento de Projeto do nível 2 do *CMMI*. Além disso, a ferramenta *RÉGULA* passa por um processo de revisão da tradução das Regras de Negócio para o *Prolog*, juntamente com a implementação de melhorias na sua máquina de inferência.

Como trabalhos futuros, apontamos a formalização de todas as áreas de nível 2 do modelo *CMMI* e a utilização destas Regras num estudo de caso de certificação e no uso destas na especificação de um sistema de informação de apoio ao processo *CMMI* nível 2. Sua expansão para os níveis subseqüentes será, então, um caminho natural. Finalmente, vale ainda ressaltar que esta técnica pode ser aplicada a qualquer conjunto de normas ou recomendações.

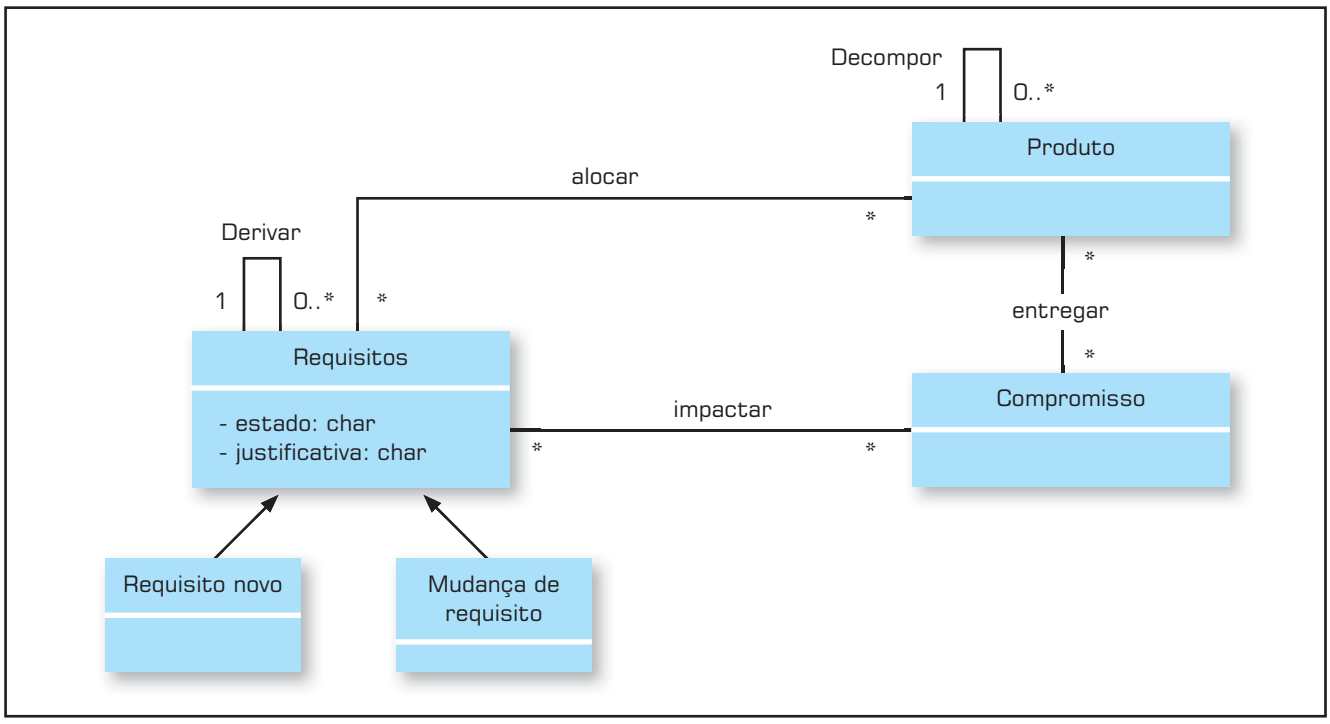
Tabela 5: Regras que regem a prática 4.

SUBPRÁTICA	REGRA EM PORTUGUÊS ESTRUTURADO	REGRA EM PROLOG
Manter a rastreabilidade dos requisitos para assegurar que a fonte dos requerimentos (derivados) de mais baixo nível esteja documentada.	Um rastreamento de requisitos de projeto de *software* DEVE OBRIGATORIAMENTE ser realizado pelo gerente de projeto.	obligation(rastreamento_de_requisitos_de_projeto_de_software, gerente_de_projeto, ser_realizado, regra19).
	O rastreamento de requisitos de projeto de *software* DEVE OBRIGATORIAMENTE *ser mantido atualizado* pelo gerente de projeto.	obligation(rastreamento_de_requisitos_de_projeto_de_software, gerente_de_projeto, ser_mantido_atualizado, regra20).
Manter a rastreabilidade dos requisitos a partir de um requisito para seus requisitos derivados e alocação a funções, objetos, pessoas, processos e produtos.	A documentação da origem de um requisito derivado DEVE OBRIGATORIAMENTE *ser garantida* pelo rastreamento dos requisitos do projeto de *software*.	obligation(documentação_da_origem_de_um_requisito_derivado, rastreamento_de_requisitos_de_projeto_de_software, ser_garantida, regra21).
Manter a rastreabilidade horizontal de função a função e através de interfaces.	A documentação da origem de um requisito derivado DEVE OBRIGATORIAMENTE *ser garantida* pelo rastreamento dos requisitos do projeto de *software*.	obligation(documentação_da_origem_de_um_requisito_derivado, rastreamento_de_requisitos_de_projeto_de_software, ser_garantida, regra22).
Gerar a matriz de rastreabilidade dos requisitos.	Uma matriz de rastreamento de requisitos de projeto de *software* DEVE OBRIGATORIAMENTE *ser gerada* pelo gerente de projeto.	obligation(matriz_de_rastreamento_de_requisitos_de_projeto_de_software, gerente_de_projeto, ser_gerada, regra23).

Tabela 6: Principais termos e sua definição através de fatos.

TERMO	FATOS EM PORTUGUÊS ESTRUTURADO	FATOS EM PROLOG
Requisito	Requisito ESTÁ RELACIONADO A requisito (derivado) POR derivar COM GRAU 0..N.	derivar(X, Y, 0, N) :- element(X, requisito), element(Y, 'requisito derivado').
	Requisito ESTÁ RELACIONADO A produto POR alocar COM GRAU M..N	alocar(X, Y, muitos, muitos) :- element(X, requisito), element(Y, produto).
	Requisito TEM COMO ATRIBUTO estado de requisito E estado de requisito É DEFINIDO POR formulado, aceito ou acordado.	has_attribute(requisito, 'estado de requisito'). has_domain('estado de requisito', 'formulado, aceito ou concordado').
	Requisito TEM COMO ATRIBUTO justificativa de requisito.	has_attribute(requisito, 'justificativa de requisito'). has_domain('justificativa de requisito', texto).
	Requisito ESTÁ RELACIONADO A compromisso de projeto POR impactar.	impactar(X, Y, um, um) :- element(X, requisito), element(Y, compromisso).
Compromisso de projeto	Compromisso de projeto ESTÁ RELACIONADO A produto POR entregar.	entregar(X, Y, um, um) :- element(X, 'compromisso de projeto'), element(Y, produto).
Produto	Produto ESTÁ RELACIONADO A produto (componente) POR decompor COM GRAU 0..N.	decompor(X, Y, 0, N) :- element(X, produto), element(Y, 'produto componente').
Requisito novo	Requisito novo É SUBTIPO DE requisito.	subtype('requisito novo', requisito). element(X, 'requisito novo') :- element(X, requisito).
Mudança de requisito	Mudança de requisito É SUBTIPO DE requisito.	subtype('mudanca de requisito', requisito). element(X, 'mudanca de requisito') :- element(X, requisito).

Figura 3: Modelo de Informação obtido a partir dos termos e fatos.



Artigo recebido em 16/05/2006
Aprovado para publicação em 25/05/2007

Referências

ARAÚJO, B. *et al*. RÉGULA – Uma Ferramenta para a Captura de Requisitos de Software através de Regras de Negócio. Simpósio Brasileiro de Engenharia de Software, Sessão Ferramentas. Florianópolis. Brasil, outubro 2006.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. *Unified Modeling Language User Guide*. Massachusetts, EUA: Addison-Wesley, 1999.

BRATKO, I. *Prolog Programming for Artificial Intelligence*. 2. ed. Massachusetts, EUA: Addison-Wesley, 1990.

BRG Business Rules Group. *GUIDE Business Rules Project: Final Report, 1997*. revisão 1.2. Disponível em: <http://www.businessrulesgroup.org>. Acesso em: jun. 2005.

BRG Business Rules Group. *Defining Business Rules – What Are They Really*? 2000, revisão 1.3. Disponível em: <http://www.businessrulesgroup.org>. Acesso em: jun. 2005.

CHRISSIS, M. B.; KONRAD, M.; SHRUM, S. *CMMI: Guidelines for Process Integration and Product Improvement*. Pensilvânia, EUA: SEI Software Engineering Institute Addison-Wesley, 2003

DIAS, F. *et al*. *Um Ambiente para Modelagem Organizacional Baseado em Regras de Negócio*. I Simpósio Brasileiro de Sistemas de Informação, Porto Alegre, RS, Brasil, 2004.

ERIKSSON, H.; PENKER, M. *Business Modeling with UML:* Business Patterns at Work. John Wiley & Sons, 2000.

HOGGER, C. J. *Essentials of Logic Programming*. Oxford University Press, 1994.

HUMPHREY, W. S. *Winning with Software – an Executive Strategy*. Massachusetts, EUA: Addison-Wesley, EUA, 2002.

IEEE (1998) Institute of Electrical and Electronics Engineers, Inc. *IEEE Std 830-1998 Recommended Practice for Software Requirements Specifications*, New York, EUA, 1998.

LEITE, J.; LEONARDI, M. *Business Rules as Organizational Policies*. 9th International Workshop on Software Specification & Design, 1998.

MARSHALL, I. *Gestão da Qualidade*. FGV Management – série Gestão Empresarial, 1. ed. Rio de Janeiro, Brasil: Editora FGV, 2003.

MORGAN, T. *Business Rules and Information Systems*. Massachusetts, EUA: Addison-Wesley, 2002.

MORIARTY, T. *Business Grammar Rules Part 1*. 2002. Disponível em: <www.tdan.com/i020fe03.htm>. Acesso em: fev. 2005.

OMG – Object Management Group. *XML Metadata Interchange*, 2002. Disponível em: <http://www.omg.org/technology/documents/formal/xmi.htm>. Acesso em: fev. 2005.

PAULK, M. C. *et al*. *The Capability Maturity Model: Guidelines for Improving the Software Process*. Massachusetts, EUA: Addison-Wesley, 1995.

PMI – Project Management Institute. *History*, 2004. Pensilvânia, EUA. Disponível em: <http://www.pmi.org/info/AP_IntroOverview.asp>. Acesso em: jun. 2004.

ROSS, R. *Principles of the Business Rule Approach*. Massachusetts, EUA: Addison-Wesley, 2000.

VON HALLE, B. *Business Rules Applied*. New York: John Wiley & Sons, Inc., 2002.

Sobre os autores

Gisele P. Morgado
Núcleo de Computação Eletrônica – Instituto de Matemática – Universidade Federal do Rio de Janeiro (UFRJ)
End.: Caixa Postal 2324 – 20001-970 – Rio de Janeiro – RJ – Brasil
E-mail: gpmorgado@gmail.com

Ingrid Gesser
Núcleo de Computação Eletrônica - Instituto de Matemática – Universidade Federal do Rio de Janeiro (UFRJ)
End.: Caixa Postal 2324 – 20001-970 – Rio de Janeiro – RJ – Brasil
E-mail: igesser@uninet.com.br

Denis S. Silveira
IBMEC – Faculdades Ibmec-RJ – Graduação em Administração
Programa de Engenharia de Produção – COPPE – Universidade Federal do Rio de Janeiro (UFRJ)
End.: Centro de Tecnologia Bloco F, Cidade Universitária – Ilha do Fundão – RJ – Brasil
E-mail: denis@pep.ufrj.br

Fernando S. P. Manso
Núcleo de Computação Eletrônica – Instituto de Matemática – Universidade Federal do Rio de Janeiro (UFRJ)
End.: Caixa Postal 2324 – 20001-970 – Rio de Janeiro – RJ – Brasil
E-mail: fmanso@nce.ufrj.br

Priscila M. V. Lima
Núcleo de Computação Eletrônica – Instituto de Matemática – Universidade Federal do Rio de Janeiro (UFRJ)
Caixa Postal 2324 – 20001-970 – Rio de Janeiro – RJ – Brasil
E-mail: priscila@nce.ufrj.br

Eber A. Schmitz
Núcleo de Computação Eletrônica – Instituto de Matemática – Universidade Federal do Rio de Janeiro (UFRJ)
Caixa Postal 2324 – 20001-970 – Rio de Janeiro – RJ – Brasil
E-mail: eber@nce.ufrj.br