Uma Proposta de Formalização das Práticas Específicas de Gerência de Requisitos do CMMI usando Regras de Negócio

Ingrid Gesser

Universidade Federal do Rio de Janeiro

Instituto de Matemática

Núcleo de Computação Eletrônica

Mestrado em Informática

Orientador: Éber Assis Schmitz, Ph. D.

Co-orientadora: Priscila Machado Vieira Lima, Ph. D.

Rio de Janeiro - Brasil

Dezembro de 2004

G392 Gesser, Ingrid.

Uma Proposta de Formalização das Práticas Específicas de Gerência de Requisitos do CMMI usando Regras de Negócio / Ingrid Gesser - Rio de Janeiro, 2004.

155 f.; il.

Dissertação (Mestrado em Informática) – Universidade Federal do Rio de Janeiro, Instituto de Matemática, Núcleo de Computação Eletrônica, 2004.

Orientador: Eber Assis Schmitz

Co-orientadora: Priscila Machado Vieira Lima

1. Processos de Negócios – Teses. 2. Regras de Negócios - Teses. 3. Processo de Desenvolvimento de Softwares – Teses. 4. CMMI – Teses 5. Formalização de Regras de Negócios I. Éber Assis Schmitz (Orient.). II. Priscila Machado Vieira Lima (Co-orient.). III. Universidade Federal do Rio de Janeiro. Instituto de Matemática. Núcleo de Computação Eletrônica. IV.Título.

CDD

Uma Proposta de Formalização das Práticas Específicas de Gerência de Requisitos do CMMI usando Regras de Negócio

Ingrid Gesser

Tese submetida ao corpo docente do Instituto de Matemática / Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro – UFRJ, como parte dos requisitos necessários à obtenção do grau de Mestre.

Aprovada por:

_____

Prof. Éber Assis Schmitz, Ph. D. (Orientador)

_____

Profa. Priscila Machado Vieira Lima, Ph. D. (Co-orientadora)

_____

Prof. Antônio Juarez Sylvio de Menezes Alencar, D. Phil.

_____

Prof. Márcio de Oliveira Barros, DSc.

Rio de Janeiro - Brasil

Dezembro de 2004

DEDICATÓRIA

Dedico esta dissertação à minha **mãe Iêda** e à minha falecida **avó Ocila**, mulheres batalhadoras, corajosas e confiantes, cujo exemplo de vida me inspiram sempre a enfrentar desafios e a prosseguir na busca dos verdadeiros valores humanos.

Dedico também ao meu **pai Paulo**, pela sua orientação, compreensão e apoio que me são inesquecíveis.

Por fim, dedico meu trabalho à nova geração da família: ao meu **filho Ricardo**, e aos meus **sobrinhos Carol, Cacá** e ao **bebê da Simone (que ainda vai chegar)**, para que minha dedicação e afinco focalizados no crescimento profissional e no enriquecimento intelectual seja para eles um exemplo a ser seguido.

AGRADECIMENTOS

Agradeço a **Deus** a luz e a força que me são indispensáveis na busca do caminho certo.

Agradeço aos meus **orientadores Éber e Priscila** pelo apoio e estímulos constantes, pela paciência e compreensão e também por me forçarem a procurar mais, pesquisar mais, escrever melhor, por efetivamente participar do meu crescimento intelectual.

Agradeço à **UFRJ**, e ao **NCE** em particular, a oportunidade de galgar mais esse degrau da minha vida acadêmica e profissional, ao **Corpo Docente** com o qual estive em contato, à **Dona Deise** e, em especial aos **professores Marcos, Renata e Flávia**.

Não poderia deixar de mencionar meus atuais **colegas de trabalho da Redecard**, que me apoiaram em momentos de tensão com suas palavras de encorajamento e carinho.

Minha gratidão especial àqueles que foram os responsáveis pela minha introdução na Engenharia de *Software* e primeiros encorajadores na minha decisão de realizar esse mestrado: aos meus **amigos Cappelli e Gualter**.

Agradeço às minhas **irmãs Andréa e Simone** pelo apoio nesse ano de 2004 e pela sua eterna "gozação" com meu jeito "Instituto Ingrid" de ser.

Agradeço ao meu **grande amigo (e ex-marido) Pedro** pelo apoio emocional, financeiro, logístico e pela amizade que compartilhamos.

Resumo

GESSER, Ingrid. **Uma Proposta de Formalização das Práticas Específicas de Gerência de Requisitos do CMMI usando Regras de Negócio.** Orientador: Éber Assis Schmitz. Rio de Janeiro: UFRJ/IM/NCE, 2004. Dissertação (Mestrado em Informática).

Observa-se atualmente uma tendência cada vez maior no sentido de estabelecer Processos e Regras de Negócio na gestão organizacional. Apesar de pouco utilizada, a formalização das regras pode trazer grandes benefícios às organizações, como: diminuição de ambigüidade da linguagem natural, facilidade de documentação, divulgação e de atualização, além de clareza e objetividade na definição dos processos de negócio correspondentes. Para o negócio de desenvolvimento de *software*, consideramos útil a formalização de regras extraídas de um modelo de maturidade de *software* – CMMI – *Capability Maturity Model Integration*. Este trabalho propõe a utilização de uma linguagem estruturada como subconjunto de uma linguagem formal – o Português Estruturado – e a aplicação de modelos de sentenças pré-definidos. Com a análise e interpretação das recomendações de práticas da área de processo de Gerência de Requisitos do CMMI sob a perspectiva de regras de negócio, foram elaboradas as traduções das práticas em termos e regras em Português Estruturado. Conjuntamente à análise foram elaborados modelos de processos correspondentes às regras formalizadas e aderentes às práticas originais do CMMI, na forma de diagramas de atividades da UML (*Unified Modeling Language).*

Palavras-Chave: Processo de Negócio, Regras de Negócio, Desenvolvimento de *Software*, CMMI, Gerência de Requisitos, Formalização de Regras de Negócio, Modelagem de Processos.

Abstract

GESSER, Ingrid. **A Proposition to Formalize CMMI´s Requirements Management Specific Practices using Business Rules.** Thesis Advisor: Éber Assis Schmitz. Rio de Janeiro: UFRJ/IM/NCE, 2004. Dissertation (Master Degree in Information Systems).

A current trend regarding organizational management is the establishment of Business Processes and Business Rules. Although it is not widely employed, the adoption of formalization of business rules can bring great benefits to organizations such as: reduction of natural language ambiguity, documentation capabilities, greater awareness, more dissemination of information, updating capabilities, and also ease-of use and objectivity in defining the activities of related business processes. When the business is software development, we consider advisable the use of a formalized set of rules extracted or based on a software maturity model such as the CMMI – Capability Maturity Model Integration. In this work we propose the use of a structured language as a subset of a formal language – Structured Portuguese – and the implementation of pre-defined sentence models. By way of viewing as business rules the recommended practices of CMMI process area that refers to Requirements Management, the corresponding terms and rules were formalized in Structured Portuguese. As an additional result of the analysis, the corresponding set of processes was modeled in compliance to the original practices from CMMI, using UML (Unified Modeling Language) activity diagrams.

Keywords: Business Process, Business Rules, Software Development, CMMI, Requirements Management, Formalization of Business Rules, Process Modeling.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

BD	Base de Dados
BRG	*Business Rules Group*
CM	*Configuration Management*
CMM	*Capability Maturity Model*
CMMI	*Capability Maturity Model Integration*
ERP	*Enterprise Resource Planning*
GP	*Generic Practices*
IEEE	*Institute of Electrical and Electronics Engineers*
LPO	Lógica de Primeira Ordem
MER	Modelo de Entidades e Relacionamentos
PP	*Project Planning*
PMC	*Project Monitoring and Control*
PMI	*Project Management Institute*
PROLOG	*Programming in Logic*
SEI	*Software Engineering Institute*
SP	*Specific Practices*
SRS	*Software Requirements Specification*
RD	*Requirements Development*
REQM	*Requirements Management*
RSKM	*Risk Management*
TI	Tecnologia da Informação
TS	*Technical Solution*
UML	*Unified Modeling Language*

SUMÁRIO

1. Introdução

1.1. Motivação

Atualmente as organizações vêm buscando revisar seus modelos de gestão, motivados pela necessidade de sobrevivência num mercado altamente competitivo, pelos desafios trazidos pela globalização e pela abertura de mercados. A Engenharia de Processos de Negócio propõe-se a auxiliar as organizações a identificar as soluções para os problemas relacionados aos seus processos de trabalho, como também as dificuldades encontradas ao lidar com as regras de negócio. Possibilita o entendimento dos processos de negócio por meio do conhecimento detalhado das suas atividades e das informações manipuladas. Os seus resultados buscam agregar valor e têm como objetivo o planejamento, projeto, estruturação e avaliação dos processos [VERNADAT, 1996].

As regras de negócio estabelecem as diretrizes para a gestão de negócios, então é imperativo que elas sejam usadas na definição dos processos de negócio associados. As regras de negócio possuem como principal característica o dinamismo. Estão permanentemente sujeitas a transformações, de forma a se adaptarem continuamente aos desafios competitivos típicos do ambiente empresarial atual. Neste contexto, observa-se uma tendência cada vez maior para a definição detalhada das regras de negócio. Infelizmente, na maioria dos casos, a representação das regras de negócio é feita por meio de linguagem natural, o que leva a ambigüidades e pode gerar erros e interpretações indevidas. Surge então a necessidade de uma garantia mínima de que as regras de negócio, após serem estabelecidas, sejam corretamente interpretadas e implementadas nos respectivos processos de negócio.

É freqüente observar que o diferencial competitivo de uma empresa apóia-se fortemente no uso de *software*. Desta forma, podemos entender facilmente a importância da adequação do *software* aos negócios, a eficácia no seu desenvolvimento e a eficiência de sua utilização. Segundo HUMPREY [2002], "*Every Business is a Software Business*".

Motivados pela relevância do *software* para os negócios, neste trabalho voltamos a nossa atenção para o negócio de desenvolvimento de *software*.

Em nossa interpretação, as regras de negócio mais alinhadas a um processo de *software* devem ter a capacidade de aprimorar as características do processo de *software* e promover ciclos de melhoria contínua do mesmo. Dessa forma, dentre as normas, modelos de qualidade e padrões conhecidos e disponíveis, escolhemos o **CMMI** – *Capability Maturity Model Integration* [CHRISSIS *et al.*, 2003]. O motivo dessa escolha foi sua atualidade – foi lançado em 2003 - por ser específico e estar direcionado às melhores práticas de desenvolvimento e manutenção de *software*, e por estar consolidado mundialmente.

1.2. Proposta de Solução

A definição formal de regras de negócio apresenta vantagens de explicitação das mesmas em linguagem não-ambígua e documentada. Além disso, possibilita maior grau de detalhamento para a definição do processo de negócio em questão.

Este trabalho propõe o uso de um método de formalização, que tem como objetivo principal eliminar, ou reduzir os problemas de interpretação das regras e dos processos de negócio, para que seja obtida uma diminuição significativa da ambigüidade da linguagem natural. Esse método constitui-se em análises progressivas das regras envolvidas, efetuando sua tradução para Português Estruturado na forma de um grupo pré-definido de modelos de sentenças. Essas sentenças devem se constituir nas atividades de um processo de negócio, organizados na forma de diagramas de atividades. O método prevê revisões sucessivas dos elementos durante todas as etapas de formalização. Por fim, o trabalho propõe a utilização do *framework REGULA* [DIAS *et al*., 2004], ferramenta de gerenciamento de regras de negócio, baseada na estruturação proposta pelo *Business Rules*

Group [BRG, 2001] que permite o armazenamento e conversão das regras de negócio para a Lógica de Primeira Ordem e depois para a linguagem PROLOG.

Nesse trabalho é apresentada a aplicação da formalização proposta às práticas específicas de uma área de processos de nível 2 de maturidade do CMMI – a área de processo **REQM** – *Requirements Management* - Gerência de Requisitos.

1.3. Contribuições do Trabalho

A abordagem do trabalho procurou analisar as práticas de uma área de processo do CMMI sob a ótica de regras de negócio de um processo de produção de *software*, o que trouxe uma perspectiva que nos permite vislumbrar as vantagens de sua formalização. A tradução das práticas e 'subpráticas' para uma linguagem estruturada em modelos de sentença pré-definidos agregou objetividade às recomendações do CMMI na medida em que tornou sua aplicação mais direta e objetiva.

A adoção de um modelo de maturidade da capacidade de *software* por uma organização forçosamente a leva a interpretar as recomendações e conjugá-las à sua própria cultura organizacional, políticas e formas de gestão, com o objetivo de elaborar os seus processos e definir sua estratégia de utilização na produção de *software*. Nesta interpretação é exigido dela um enorme esforço inicial de entendimento e formulação da sua abordagem, além do conseqüente treinamento e esforço de institucionalização em todas as equipes envolvidas.

A utilização de um modelo de processo de *software* já definido, aliado a um conjunto de regras formalizadas, e até mesmo de uma ferramenta de automação de regras pode colaborar no sentido de reduzir o esforço de interpretação, facilitando a adaptação das regras formalizadas, já que, além da tradução para a língua portuguesa das práticas e 'subpráticas' específicas já será fornecida, grande parte da ambigüidade natural do texto

original do CMMI terá sido reduzida pela formalização. Para isso a formalização proposta deve estar aderente e deve respeitar criteriosamente as recomendações originais.

Neste trabalho procuramos aplicar todo o rigor na interpretação e formalização das práticas de forma a garantir a aderência ao CMMI. Entre os objetivos a serem atingidos com o resultado desse trabalho, está a redução efetiva do esforço de implementação do modelo de maturidade da capacidade proposto pelo CMMI. Para isso, adicionalmente à formalização das 'subpráticas', estão apresentados um modelo de processo de *software* e sugestões de documentos referentes ao processo, e que não estão presentes no texto original do CMMI. Concluímos que essa contribuição é significativa por ser pragmática, já que as regras formalizadas apresentam um menor grau de análise para entendimento e interpretação.

Outra contribuição que ressaltamos é seu caráter inovador, já que no campo da formalização, notamos alguns esforços e estudos no sentido de sua aplicação às regras de negócio. Alia-se a isso a adoção de um conjunto de regras do modelo CMMI, somente recentemente divulgadas [CHRISSIS *et al*., 2003] agregando atualidade à pesquisa realizada.

1.4. Estrutura da Tese

Esta dissertação foi estruturada em sete capítulos. Este primeiro capítulo introduz o tema abordado, apresentando a motivação, a proposta de solução, as contribuições do trabalho e a estruturação do trabalho.

No segundo capítulo é apresentada uma revisão da literatura sobre processos e regras de negócio. São abordadas questões sobre o contexto organizacional atual, as tendências e os problemas enfrentados. São introduzidos o processo e as regras de negócio

a serem tratados por este trabalho: o desenvolvimento de *software* e o modelo de maturidade **CMMI**.

No terceiro capítulo é feita uma explanação sobre o modelo de maturidade **CMMI**. São avaliadas as recomendações e a abordagem do mesmo para o desenvolvimento de *software* e seu alinhamento como padrão de regras a nortear a construção de *software* sob o ponto de vista da engenharia.

No quarto capítulo é apresentada a revisão de literatura sobre o uso da lógica e de ferramentas baseadas em formalidade matemática para as propostas de formalização. São apresentadas também: a estruturação proposta pelo BRG (*Business Rules Group*), o Português Estruturado, a Lógica de Primeira Ordem e o PROLOG. A ferramenta *REGULA*, cuja adoção é sugerida por este trabalho como etapa de armazenamento e solução de formalização, conjuga todos esses elementos e se constitui numa possibilidade viável para o cadastramento e formalização de regras de negócio.

No quinto capítulo é apresentada a formalização das regras de negócio do processo REQM do CMMI, onde é aplicada a primeira fase de formalização – a tradução das práticas e 'subpráticas' específicas da área de processo de Gerência de Requisitos para o Português Estruturado, analisando as recomendações do CMMI e identificando o(s) modelo(s) de processo aderente.

O sexto capítulo apresenta uma discussão onde são explanadas as análises das práticas e 'subpráticas' da área de processo estudada e também dos termos e regras identificadas, realizada sob diversos pontos de vista.

Finalmente, no sétimo capítulo são apresentadas as conclusões finais do trabalho, com as sugestões para trabalhos futuros.

2. Processos e Regras de Negócio

O aumento da complexidade dos negócios trazido pela globalização, pela abertura dos mercados e o alto grau de competição entre as organizações estimulam pesquisadores e empresários a buscar novas maneiras de gerenciar e administrar os negócios nos dias atuais. Além disso, devemos levar em conta a diversificação dos produtos oferecidos nos mercados, indicativo do aumento do grau de exigência e sofisticação dos clientes, que esperam sempre produtos melhores e custos menores. Os empregados e os clientes de uma empresa estão envolvidos nos seus processos de negócio – manifestações das atividades econômicas das empresas, e que são intricadas, dinâmicas e estão em constante alteração. Os primeiros tratam de por em prática as atividades que permitem a oferta dos produtos e serviços exigidos pelos clientes. A organização enfrenta então o constante desafio de elaborar, rever e refinar estratégias de negócio capazes de viabilizar seu objetivo final: o lucro.

No esforço de perceber e entender o seu próprio funcionamento, as organizações necessitam identificar os obstáculos existentes para o cumprimento de seus objetivos. Identificam-se, assim, discrepâncias naturais entre o que organização pensa que é e o que ela é na realidade. Com o conhecimento sobre o modo de funcionamento do negócio da organização, obtém-se uma percepção mais clara de sua estrutura, de seus objetivos etc. Como conseqüência, viabiliza-se o planejamento estratégico para a evolução da organização em geral e do negócio em particular. Para esta evolução, a modelagem de processos pode se constituir numa contribuição significativa [CRUZ, 2002].

Diante deste cenário, não se trata somente de obter vantagens competitivas, mas da efetiva sobrevivência no mercado que faz surgir no meio empresarial outras disciplinas a serem consideradas como: engenharia empresarial, gestão de qualidade total, *Six Sigma*, análise de cadeia de valor, gerência da cadeia de fornecimento, estratégia direcionada ao cliente etc [SMITH, FINGAR, 2003]. Nesse contexto, a gestão de processos de negócio

vem sendo adotada por várias organizações mundialmente, seja em organizações nacionais ou multinacionais, de grande, médio ou pequeno porte.

2.1. Processos de Negócio

De alguma forma, os processos de negócio sempre estão presentes no dia-a-dia de uma empresa. O importante é garantir que esses processos estejam sendo adequadamente gerenciados. É imprescindível para uma organização conhecer seus "processos chave", ou seja, aqueles que exercem o maior impacto sobre os clientes, como, por exemplo, os processos de atendimento, entrega de pedidos e de faturamento. Além disso, a estrutura organizacional da empresa deve privilegiar o funcionamento desses processos. Em segundo plano, mas não menos importantes, estão os processos de apoio ao negócio, que mantém a empresa em funcionamento, como, por exemplo, os processos de administração dos recursos humanos e de suprimentos. Como fator de aperfeiçoamento, a adoção e utilização de um ciclo de avaliações e otimizações, ou seja, a melhoria contínua dos processos, é indicada como a forma mais efetiva de tornar os processos de negócio melhores, mais rápidos e eficientes.

A Engenharia de Processos de Negócio propõe-se a auxiliar as organizações na identificação das soluções para os seus problemas. Possibilita o entendimento dos processos de negócio por meio do conhecimento detalhado de seus fluxos de atividades e das informações manipuladas por eles. Objetiva a agregação de valor pelo planejamento, projeto (*design*), modelagem, estruturação e avaliação dos processos de negócio [VERNADAT, 1996].

Os processos de negócio são realizados pela combinação dos processos de trabalho, e suas atividades detalhadas são realizadas no nível mais elementar de uma organização. A Engenharia de Processos de Negócio é apoiada pela modelagem de processos, e segundo VERNADAT [1996], possui os seguintes objetivos:

- Uniformizar o entendimento sobre a operacionalização do trabalho, melhorando a integração;
- Analisar e melhorar o fluxo das informações entre as atividades;
- Explicitar o conhecimento sobre os processos;
- Auxiliar a análise organizacional com a utilização de indicadores de desempenho;
- Viabilizar simulações para o apoio à tomada de decisões.

Segundo DAVENPORT [1994] "Um processo é uma ordenação de atividades de trabalho através do tempo e do espaço, com um início, um fim e um conjunto claramente definido de entradas e saídas". Dessa forma, podemos perceber claramente que o principal componente de um processo de negócio é a atividade, ou o conjunto de atividades que o compõem. Um exemplo tradicional de um processo de negócio em uma empresa de serviços financeiros é o processo de avaliação de crédito, constituído de várias atividades de análise de risco, avaliações de histórico etc.

O foco no processo provê a infra-estrutura necessária para lidar com um mundo em constante transformação e para maximizar o uso de recursos humanos e de tecnologia para serem mais competitivos. Na realidade, o foco no processo pode viabilizar o foco no cliente. Por outro lado, os empresários desejam soluções confiáveis que os ajudem a fazer negócios de uma forma melhor, mais rápida e mais pró-ativa no relacionamento com os seus clientes.

A gerência de processos de negócio de uma empresa é inseparável, tanto do próprio negócio como da tecnologia. Teorias alternativas e uma abordagem pragmática de inovação e mudanças – a da gestão de processos de negócio – constituem o passo de sucessão natural da última onda gerencial – a reengenharia. Muitas tendências convergem para a terceira onda da gerência de processos de negócio: utilização de sistemas de *workflow* (sistemas de controle de fluxos de trabalho), modelagem de processos, gestão de qualidade, gestão de mudanças, computação distribuída etc.

O pioneiro da reengenharia, Michael Hammer [SMITH, FINGAR, 2003, HAMMER, 2001], observou que:

> "... as empresas sabem fazer muitas coisas que podem ser entendidas como processos, ..., por outro lado, converter uma descrição geral de um processo num processo executável por ações é difícil para muitas empresas, pois não é algo em que elas têm muita experiência".

Segundo SMITH, FINGAR [2003],

> "os processos são a principal propriedade intelectual e o diferencial competitivo manifestado em todas as atividades de negócio, e as empresas devem tratá-los com alto grau de habilidade e cuidado".

Podemos observar, então, que os processos estão passando para o centro das atenções na arquitetura dos negócios e dos sistemas que os implementam e apóiam.

Segundo DRUCKER [1999], em todos os ramos de atividade, a abrangência da gerência deve considerar todo o processo:

> " ... o ponto de partida para a gerência não pode mais ser seus próprios produtos ou serviços e nem mesmo seu mercado conhecido e seus usos finais conhecidos para seus produtos e serviços. Ele precisa ser aquilo que os clientes consideram valor ... ".

Na era do conhecimento as organizações devem focalizar seus esforços no aumento da efetividade pelo estabelecimento de uma base forte para a gestão do conhecimento que inclui, não apenas a tecnologia da informação, mas que se apóia no conhecimento individual e coletivo de seus colaboradores, nos processos de negócio, nas práticas de trabalho e na cultura organizacional. A gestão do conhecimento e a automação de processos de negócio estão fortemente ligadas, mas dificilmente são vistas como disciplinas

relacionadas – o que pode ser explicado pelas variadas perspectivas de negócio nas quais podem ser tratadas [MOORE, 2004]. Muitas empresas organizam, automatizam e gerenciam o trabalho por um contexto funcional – sob uma ótica da era industrial, onde o trabalho é dividido em partes discretas para maximizar a eficiência. Uma percentagem menor de empresas vem organizando seus empregados em equipes colaborativas responsáveis por processos completos de negócio, do início ao fim. As empresas que se organizaram para apoiar processos de negócio de forma a permear as funções necessárias podem perceber que isso é fundamental para compartilhar o conhecimento relacionado aos produtos com o conhecimento relacionado aos processos, através de suas fronteiras funcionais. A figura 01 apresenta as diferenças entre essas duas perspectivas.

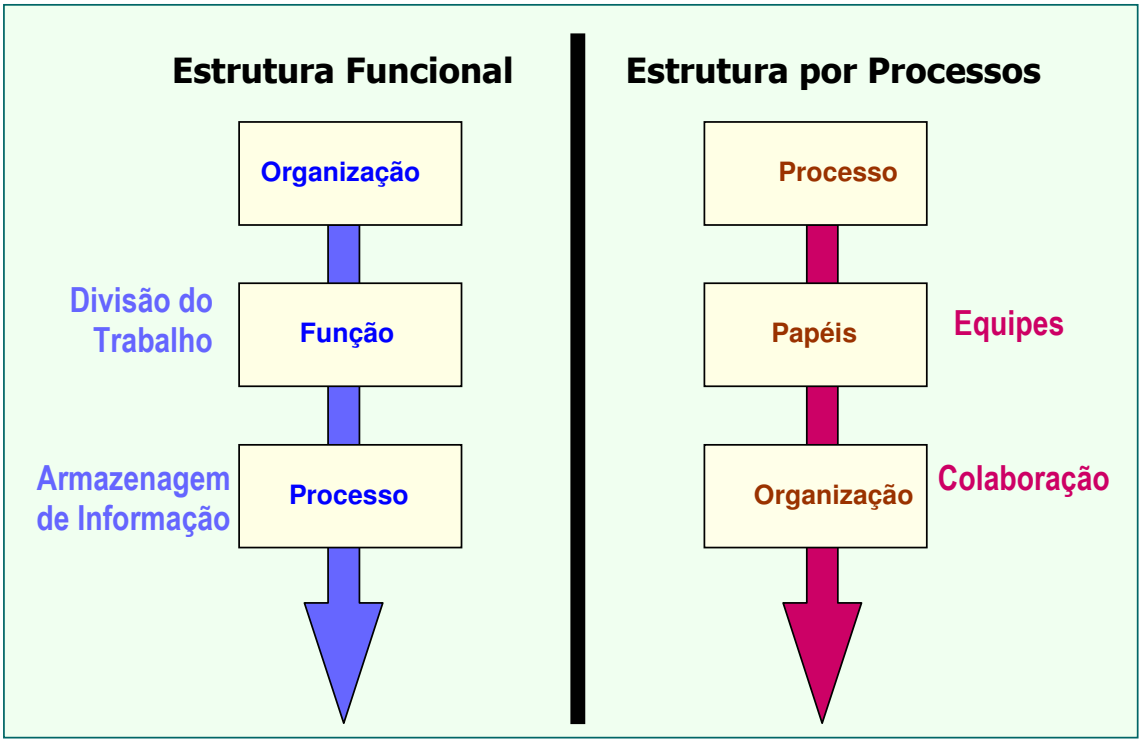

Figura 01 - Comparação das Estruturas Organizacionais: Funcional e por Processos

Os processos de negócio, então, constituem-se de atividades encadeadas, executadas a partir de determinados insumos de informação, por profissionais designados para essas atividades e que levam à produção da saída desejada. Como diretriz maior para as

atividades do processo encontram-se as regras de negócio, estabelecidas tanto pelas estratégias organizacionais definidas, como também pelas condições do mercado e da concorrência.

2.2. Regras de Negócio

Sob a perspectiva do negócio, as regras de negócio podem ser entendidas como a diretriz de condução, ação, prática ou procedimento de uma atividade em particular. Segundo o BRG - *Business Rules Group* [BRG, 2004], podemos citar duas importantes características para as regras de negócio:

- ✓ deve existir uma motivação explícita para a existência da regra;
- ✓ a regra deve ser posta em prática de tal forma que sejam estabelecidas as conseqüências de seu não-cumprimento.

Sob a perspectiva dos sistemas de informação, as regras de negócio estabelecem as definições ou restrições de algum aspecto do negócio. Seu objetivo é definir a estrutura do negócio, ou controlar e influenciar o comportamento do mesmo.

Como fator principal devemos ressaltar a sua característica dinâmica, pois, em qualquer organização, as regras de negócio estão permanentemente sujeitas a transformações, de forma a se adaptarem continuamente aos desafios competitivos típicos do ambiente empresarial atual. Neste contexto, observa-se uma tendência cada vez maior na definição detalhada das regras de negócio. Infelizmente, na maioria dos casos, a representação é feita por meio de linguagem natural, o que leva a ambigüidades e pode gerar erros e interpretação indevida das mesmas. Em muitos casos não há nem mesmo a descrição documentada das regras do negócio, que, na verdade, acabam por ficar embutidas em códigos fonte de programas computacionais, estruturas de bancos de dados ou documentações de sistemas.

As regras de negócio estabelecem as diretrizes para a gestão de negócios, então é imperativo que elas sejam usadas na definição dos processos de negócio associados. Surge aí a necessidade de uma garantia mínima de que as regras de negócio, após serem estabelecidas, sejam corretamente interpretadas e implementadas nos respectivos processos de negócio.

Sob o ponto de vista dos sistemas de informação, as regras de negócio podem ser entendidas e consideradas como os requisitos de um projeto, sejam eles alocados ao *software*, no caso de regras implementadas nas aplicações informatizadas e que compõem uma parte da operação de um negócio, ou não diretamente alocados ao *software* (requisitos de sistema) – como os requisitos computacionais de desempenho e de produção dos sistemas.

Destacamos então a enorme vantagem obtida com o estabelecimento de formalizações e documentações das regras de negócio. Com a adoção dessa formalização, observamos um maior grau de facilidade na inserção de novas regras, na alteração das regras existentes e na supressão de outras. Além disso, pela formalização pode-se identificar mais diretamente as possíveis inconsistências e prevenir os conseqüentes erros. A formalização e a documentação das regras de um 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, além do aumento do entendimento e o tratamento uniforme e consistente do negócio por esses profissionais.

2.3. Qualidade

Qualidade. [Do lat. *qualitate*] S. f. **2**. Numa escala de valores, qualidade que permite avaliar e, conseqüentemente, aprovar, aceitar ou recusar, qualquer coisa.

Dicionário Aurélio Século XXI

Qualidade pode ser definida como o grau em que um conjunto de características inerentes satisfaz requisitos.

Segundo MARSHALL *et al*. [2003],

> "Atualmente, a gestão da qualidade abrange uma visão macro da existência humana, influenciando modos de pensar e de agir. Num sentido amplo, o conceito de qualidade passou a significar modelo de gerenciamento que busca a eficiência e a eficácia organizacionais".

Philip B. Crosby [CROSBY, 1967] considera que a qualidade está associada aos conceitos de "defeito zero" e de "fazer certo da primeira vez", como padrões de desempenho. Para ele, qualidade significa conformidade com especificações, que, por sua vez, variam com as necessidades do cliente. O objetivo da qualidade é viabilizar que se produza de acordo com o estabelecido, para encorajar as pessoas a melhorarem continuamente.

CROSBY [1967] ressalta que a cultura da qualidade desejada deve levar em consideração quatro princípios de gestão da qualidade:

1. A qualidade é definida como conformidade aos requisitos.
2. O sistema que leva à qualidade é a prevenção.
3. O padrão de execução é o defeito zero.
4. A medida da qualidade é o preço da não-conformidade.

Em nosso trabalho apresentamos uma proposta de formalização e adoção de metodologias e modelos de maturidade como forma de obtenção de maior conformidade dos processos de negócio às estratégias da empresa, e conseqüente aumento do grau de satisfação dos clientes. A qualidade também pode ser aprimorada pela adoção de

formalização das regras de negócio como forma de garantir a conformidade em relação aos processos de negócio e aos objetivos finais das organizações.

2.4. Sobre o Negócio Foco do Trabalho

Os desafios competitivos que as organizações de todos os ramos de atividades enfrentam na atualidade envolvem sua maturidade empresarial, visão de negócio, conhecimento do mercado, e também a forma de condução de seus processos no âmbito organizacional interno. Muitas vezes suas atividades envolvem ambientes tecnológicos que precisam estar bem preparados no tratamento das informações. É freqüente observar que o diferencial competitivo de uma empresa apóia-se fortemente no uso de *software*. Até mesmo as organizações que não são tipicamente empresas de *software* encontram-se fortemente envolvidas na produção e utilização de *software* no exercício de suas atividades de negócio. Desta forma, podemos entender facilmente a importância da adequação do *software* aos negócios, a eficácia no seu desenvolvimento e a eficiência de sua utilização.

Tudo isso nos conduz à alta relevância que deve ser dada à adoção de padrões, metodologias, guias e modelos de qualidade de *software* que aprimorem a forma de desenvolvimento, manutenção e produção das aplicações computacionais, e que conseqüentemente viabilizam uma boa condução dos negócios. Essa abordagem pode ajudar a organização a alcançar os seus objetivos empresariais.

2.4.1. Engenharia de *Software*

A engenharia de *software* é o campo da ciência da computação que lida com a construção de sistemas de *software*. Segundo GHEZZI *et al*. [1991], esses sistemas podem ser grandes e complexos, e podem envolver uma ou mais equipes de engenheiros. Freqüentemente, esses sistemas de *software* existem em múltiplas versões e são usados por muitos anos. Durante sua vida, um *software* pode sofrer várias alterações por variados motivos: correção de erros, melhoria de funcionalidades, inclusão ou remoção de funcionalidades, ou adaptação a novos ambientes e plataformas computacionais. Um engenheiro de *software* está envolvido na construção de componentes de *software*, que, quando combinados com outros componentes construídos por outros engenheiros, formam um sistema de aplicação voltado para o atendimento de um negócio ou solução de um problema. Componentes de *software* podem ser modificados por engenheiros diferentes de seus criadores originais e podem ser usados por outros para a criação de versões diferenciadas a serem usadas em outras aplicações. Essencialmente, a engenharia de *software* é uma atividade que envolve equipes de trabalho.

Roger Pressman [PRESSMAN, 2001] comenta a definição de engenharia de *software* proposta por Fritz Bauer:

> "*Software Engineering is the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines*". [BAUER, 1969]

PRESSMAN [2001] questiona:

> "a) quais princípios estáveis da engenharia podem realmente ser aplicados ao desenvolvimento de *software*?; b) como criar economicamente um *software* confiável?; c) o que é necessário para criar *software* eficiente não apenas em uma, mas em várias máquinas reais e diferentes?".

PRESSMAN [2001] considera que a definição do IEEE (*Institute of Electrical and Electronic Engineers, Inc.*) é mais apropriada, a qual estabelece:

> *"Software Engineering:*
>
> 1. *The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.*
> 2. *The study of approaches as in (1)."*

A engenharia de *software* foi criada no final da década de 1960, quando se percebeu que as lições aprendidas sobre programação não estavam sendo efetivas para a construção de melhores sistemas de *software*. Ocorria, então, uma grande dificuldade para a construção de sistemas grandes e complexos. O necessário nesses casos complexos era uma abordagem da engenharia clássica: definir claramente os problemas a serem resolvidos, e desenvolver ferramentas e técnicas para solucioná-los.

Nas disciplinas da engenharia clássica existem ferramentas e maturidade matemática definidas para a especificação de propriedades de produtos, utilizadas ao longo de seu projeto. No caso da engenharia de *software*, os engenheiros se apóiam em experiências passadas e no seu julgamento profissional, ao invés de utilizar técnicas matemáticas. Ferramentas formais de análise são necessárias para a boa prática da engenharia de *software*.

A atividade da engenharia de *software* é parte de uma atividade maior de um projeto, no qual os requisitos de *software* devem ser balanceados contra os requisitos de outras partes do sistema em construção. Como exemplo, podemos citar um sistema de telefonia: ele é composto de computadores, linhas e cabos telefônicos, aparelhos de telefone e outros *hardwares* como satélites e, finalmente, um *software* de controle dos vários componentes. É a combinação de todos esses componentes que deve atender os requisitos do sistema [GHEZZI *et al*., 1991].

Para efetuar corretamente a engenharia de *software* é necessária uma visão mais ampla do problema, tal que inclua a engenharia do sistema [GHEZZI *et al*., 1991]. É necessário que o engenheiro de *software* esteja envolvido na definição e elaboração dos requisitos. Ele deve compreender a área de aplicação do projeto ou negócio. Envolve, além disso, o compromisso dos profissionais, como todas as disciplinas de engenharia. Devem ser claramente definidas as características de implementação do *software*, que envolvem flexibilidade, bem como as características do *hardware*, que envolvem desempenho. Também devem ser tomadas decisões sobre o nível de automação desejado, pelo estabelecimento das funcionalidades a serem automatizadas, e aquelas que permanecerão manuais.

Um engenheiro de *software*, além de estar preparado para lidar com diversas abordagens de projeto (estruturas de dados, linguagens, plataformas computacionais etc) deve ser capaz de transformar requisitos vagos em especificações precisas, e tratar com os usuários em termos do domínio de aplicação do negócio.

O ciclo de vida do *software* envolve atividades, desde o surgimento de uma idéia, necessidade ou oportunidade de negócio, sua implementação e entrega ao usuário, e prossegue com sua produção e manutenção, finalizando somente quando o mesmo é descontinuado. Um modelo clássico de ciclo de vida "cascata" envolve as seguintes fases:

a. ***Análise e especificação de requisitos*:** é usualmente a primeira fase do projeto. Devem ser definidos o escopo de funcionalidades, os custos e prazos. O principal produto desta fase é a identificação e documentação detalhada dos requisitos do *software*, estabelecendo o seu escopo. A partir dessa fase podem ser estimados os custos e prazos e é viabilizada a negociação entre a equipe de desenvolvimento e o cliente.

b. ***Modelagem e especificação*:** a partir da documentação dos requisitos, os engenheiros de *software* elaboram o projeto do sistema de forma a atendê-los. Compõe-se de duas subfases: definição de arquitetura e projeto detalhado.

c. ***Construção*:** elaboração do código a ser entregue ao usuário para a produção do sistema. Estão incluídos os teste unitários dos diversos componentes de *software* construídos.

d. ***Verificação de sistema e integrada*:** Todos os componentes construídos devem ser testados em conjunto (teste de sistema) e também com os demais sistemas envolvidos (teste integrado). Esta verificação deve ser acompanhada pelo usuário, responsável pela aprovação da verificação do *software* (teste de homologação).

e. ***Implantação e manutenção*:** Entrega do *software* ao cliente ou usuário e produção do sistema. Inicia-se então a fase de manutenção do *software* para atender à dinâmica do negócio, ou para ajustá-lo a diferentes circunstâncias.

A figura 02 apresenta uma visão gráfica do ciclo de vida, e ilustra a origem da nomenclatura “cascata”. Cada fase produz o que é usado na fase seguinte, e, idealmente, o processo ocorre de forma linear e ordenada.

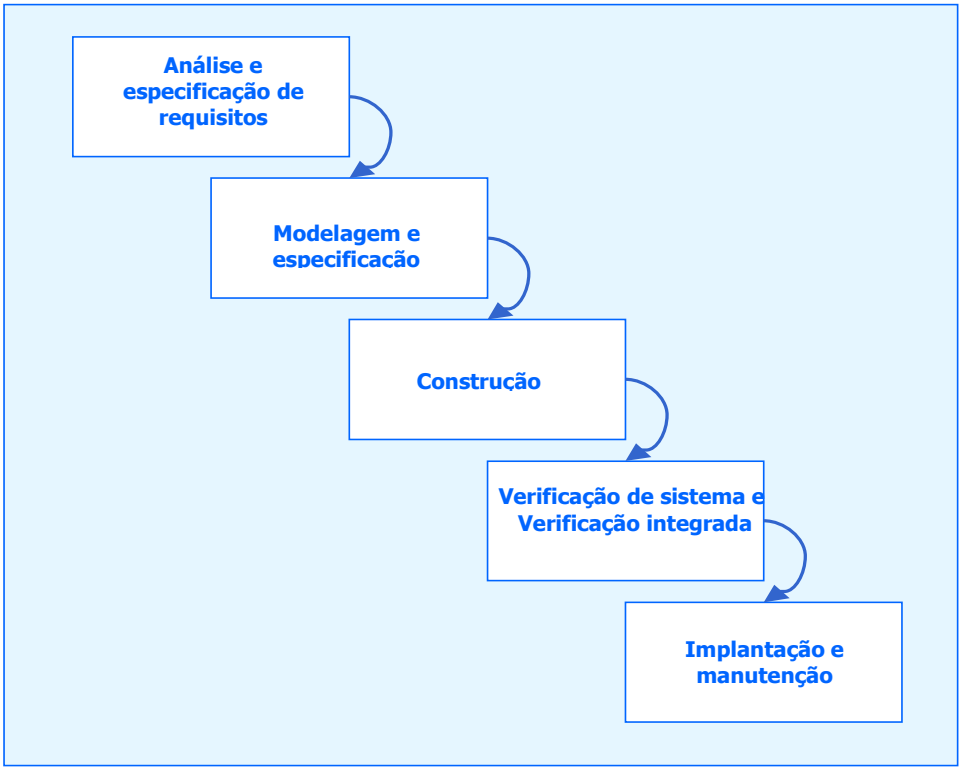

Figura 02 - O Modelo de Ciclo de Vida "Cascata"

2.4.2. Processo de *Software*

Um processo de *software* pode ser definido como o conjunto de atividades, métodos, práticas e transformações que as pessoas empregam para desenvolver e manter o *software* e os produtos associados, por exemplo: planos de projeto, documentos de modelagem, código, casos de teste e manuais de usuário [PAULK *et al*., 1994].

Um processo de *software* engloba várias fases, desde o surgimento do problema até a entrega do *software* ao cliente ou usuário para a produção e seu término só ocorre realmente quando o mesmo é descontinuado. Os processos de produção e manufatura de qualquer produto são extensivamente estudados. O objetivo de aprimoramento de um processo de produção é torná-lo confiável, *predictable* (que se possa predizer) e eficiente.

Pelo estabelecimento de um processo de *software*, é possível colher benefícios do uso de padronização. É necessário considerarmos duas características inerentes ao *software*: em primeiro lugar, a construção de *software* é uma atividade intelectual, e por isso é de difícil automação; em segundo lugar, *software*s são altamente instáveis: os requisitos mudam constantemente, e por isso os produtos devem ser capazes de evoluir.

Diferentes modelos tentam capturar esse processo, através do ciclo de vida do *software*, que deve ser planejado e controlado de forma a agregar as qualidades desejadas no produto, ou seja, no *software*. A adoção de padrões e metodologias viabiliza a incorporação dessas qualidades ao *software*. As questões de gerenciamento do processo de *software* agregam os aspectos econômicos ao desenvolvimento e à manutenção de *software*.

2.4.3. Desenvolvimento e Qualidade de *Software*

No nosso trabalho, o negócio em questão refere-se ao desenvolvimento e manutenção de *software*. As organizações referidas podem ser as chamadas de "consultorias de informática" ou "fábricas de *software*", cuja atividade fim constitui-se do desenvolvimento e manutenção de sistemas de *software*. Podem também ser os "departamentos de informática", ou mesmo "diretorias de tecnologia de informação", de organizações cuja atividade fim não é o desenvolvimento de *software*, mas que estão amplamente apoiadas na sua utilização, como é o caso das empresas do ramo de atividades financeiras, entre outras. O processo de construção de *software* é o processo pelo qual, a partir da especificação do problema por um usuário, um *software* é produzido para atender e solucionar este problema e viabilizar a gestão de um negócio.

Podemos mesmo afirmar que atualmente todos os negócios de alguma forma estão sempre relacionados a *software*. Eles foram usados inicialmente como ferramentas de pesquisa científica pelos pesquisadores e como ferramenta de auxílio administrativo nas empresas. Atualmente, o uso de sistemas especialistas, de aplicativos dirigidos aos

negócios e a disseminação das aplicações ERP (*Enterprise Resource Planning*), levam-nos a afirmar com tranqüilidade que o *software* está efetivamente no centro da maioria dos negócios nas organizações de todo o mundo. Segundo HUMPHREY [2002], "*Every Business is a Software Business*" e, de acordo com sua ótica, existem três princípios que norteiam a atividade de gerência de software, dos quais destacamos o segundo que nos diz que a qualidade deve ser a prioridade principal:

> *"In software work, quality problems overwhelm everything else. Quality is critical, and when quality is not managed, entire software projects are unmanageable. There are known ways to manage software quality, but they require proper training and disciplined engineering methods. The key need is for you to make a commitment to quality. You must make software quality the top priority".* [HUMPHREY, 2002]

Para um processo de *software*, as regras de negócio mais adequadas são metodologias, modelos de maturidade, normas de qualidade de *software*, corpos de conhecimento, ou seja, os métodos e procedimentos que devem ser utilizados para desempenhar de forma efetiva, econômica e madura a atividade de construir *software*. Relacionamos a seguir alguns dentre os vários modelos de qualidade e normas de padronização existentes que vêem sendo utilizados mundialmente.

- **Normas e Padrões ISO**: a ISO foi criada em 1947 por representantes de vinte e seis países, com a denominação ***I**nternational **O**rganization for **S**tandardization*, com o objetivo de facilitar, em nível mundial, a coordenação e a unificação de normas industriais. O escopo da ISO sobre normalização está estabelecido em todos os campos do conhecimento, com exceção da área de engenharia eletrônica e elétrica, que possui normalização específica. Em 1987, foi lançada a família de normas ISO 9000, fortemente baseadas nas normas britânicas de qualidade. Em sua versão de 2000, foram

incluídas as normas necessárias aos setores de serviços, incluindo os serviços relacionados à tecnologia da informação [MARSHALL *et al*., 2003].

- **CMM–SW** (***C**apability **M**aturity **M**odel for **S**oftware*): Em 1986, o SEI (*Software Engineering Institute*) – ligado à *Carnegie Mellon University,* da Pensilvânia nos EUA, elaborou um *framework* para a maturidade do processo de *software*. Após quatro anos de experiência com esse *framework*, o SEI evoluiu para alguns modelos de maturidade, sendo o CMM-SW o mais conhecido, que é voltado exclusivamente para o desenvolvimento e manutenção de *software*. Esse modelo foi inicialmente usado pela comunidade de *software* entre 1991 e 1992. A última versão 1.1 do CMM-SW foi lançada no mercado em 1994 na forma do livro de Mark C. Paulk *et al*.: *The Capability Maturity Model: Guidelines for Improving the Softyware Process* [PAULK *et al*., 1994].

- **CMMI** (***C**apability **M**aturity **M**odel **I**ntegration*): Além do CMM-SW, outros modelos voltados para a engenharia de sistemas, aquisição de *software*, gerência e desenvolvimento de pessoas, e desenvolvimento integrado de produtos e processos foram também elaborados pelo SEI. O projeto de criação do *CMM Integration* veio adequar o uso integrado dos modelos variados voltados para as diferentes disciplinas. Foi lançado no mercado em 2003 na forma do livro de Mary Beth Chrissis, Mike Konrad e Sandy Shrum - *CMMI Guidelines for Process Integration and Product Improvement* [CHRISSIS *et al*., 2003].

- **SPICE** (***S**oftware **P**rocess **I**mprovement and **C**apability d**E**termination*): o SPICE é uma iniciativa que se organizou para apoiar o desenvolvimento de um padrão internacional para avaliação de processo de *software* (*International Standard for Software Process Assessment*) e que tem três principais objetivos: 1) desenvolver uma proposta de padrão; 2) conduzir testes do padrão na indústria; 3) promover a

transferência de tecnologia de avaliação de processo de *software* para a indústria de *software* mundialmente. A versão 1 do padrão foi lançada em junho de 1995 [SPICE, 2004].

- **ITIL** (***I**nformation **T**echnology **I**nfrastructure **L**ibrary*): Na década de 1980, por solicitação do governo britânico, o CCTA (*Central Computer and Telecommunications Agency*, agora *Office of Government Commerce*, OGC), iniciou o desenvolvimento de uma abordagem para o uso efetivo e eficiente de recursos de tecnologia da informação, independente de fornecedores e fabricantes de equipamentos. Esse esforço resultou numa coleção de melhores práticas observadas na indústria de serviços de tecnologia da informação. O seu objetivo foi promover a experiência em práticas de gerência de serviços de TI [ITSMF, 2002].

- **PMBOK** *(**P**roject **M**anagement **B**ody **o**f **K**nowledge)*: O PMI (*Project Management Institute*) se estabeleceu em 1969 na Pensilvânia, EUA, como uma associação mundial de profissionais de gerência de projeto que atualmente congrega mais de 125.000 membros. Com a evolução da associação foram lançados programas educacionais e publicações para a disseminação de melhores práticas e estado da arte das várias disciplinas de gerência de projeto, incluindo o *GUIDE to the PMBOK* [PMI, 2004].

- **IEEE STD 830-1998** (***I**nstitute of **E**lectrical and **E**letronics **E**ngineers, Inc – **St**an**d**ard **830-1998***): Os padrões IEEE são desenvolvidos por Sociedades IEEE e pelo Comitê Coordenador de Padrões IEEE. Representam o consenso de larga experiência no assunto e descrevem a recomendação de prática na abordagem de especificação de requisitos de *software*. É baseado num modelo no qual o resultado do processo de especificação de requisito de *software* é um documento de especificação completo e sem ambigüidades [IEEE, 1998].

Com a análise dos padrões, das sugestões de práticas e das metodologias apresentadas observamos que alguns modelos apresentam práticas de gestão com diferentes níveis de objetividade, o que nos leva a considerar alguns deles como menos indicados para a aplicação de uma heurística de formalização. Dessa forma, o modelo de maturidade de *software* escolhido em nosso trabalho foi o **CMMI – *Capability Maturity Model Integration***, de 2003.

Consideramos como positivo e vantajoso a fato de o **CMMI** ser atual, específico e direcionado às melhores práticas de desenvolvimento e manutenção de *software*, e estar consolidado mundialmente. Ressaltamos que, apesar de ter sido lançado no mercado em 2003, o **CMMI** constitui uma evolução dos variados modelos do CMM, já em utilização pela indústria de *software* há mais de dez anos. Além disso, como fator relevante na escolha deste modelo, consideramos que ele propõe recomendações de práticas, que o torna mais pragmático. Outra característica positiva do **CMMI** que foi considerada é a possibilidade que oferece - a concessão de um "selo de qualidade" de reconhecimento internacional - por meio de verificações formais (*assessments*). No atual mercado competitivo, este "selo de qualidade" constitui-se em diferencial competitivo tanto no mercado nacional, como para a exportação do *software* brasileiro.

3. CMMI – Capability Maturity Model Integration

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 [CHRISSIS *et al*., 2003]. 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.

O CMMI apresenta recomendações de melhores práticas específicas e genéricas, por área de processo, a serem adotadas tanto por organizações cuja atividade fim é o desenvolvimento, manutenção e produção de *software*, como também por departamentos de TI (Tecnologia da Informação) de organizações cujo ramo de atividade não é a criação de *software*, mas que utilizam intensamente a informática no seu dia-a-dia. O objetivo do CMMI é possibilitar a elevação da maturidade na capacidade de uma equipe de profissionais nas atividades relacionadas ao desenvolvimento de *software* [CHRISSIS *et al*., 2003].

O modelo de maturidade CMMI descreve um caminho evolucionário, desde processos *ad-hoc*, imaturos, até um processo maduro e disciplinado. O modelo engloba práticas de planejamento, engenharia e gerência do desenvolvimento e manutenção de *software*. Tem como objetivo tornar possível o controle do processo de produção de *software* por meio de métricas e modelos estatísticos. Segundo HUMPHREY [1987] "Para se obter uma melhoria consistente é necessário aperfeiçoar o processo, porém se o processo não está sob controle estatístico, um progresso sustentado não será possível".

Muitas organizações vêm adotando modelos de maturidade de *software*, desde o modelo inicial - CMM, até mais recentemente o CMMI. Em [BASTOS, 1996] podemos avaliar a adoção do modelo CMM numa empresa do setor financeiro brasileiro, com a apresentação da forma de adoção, seus fatores de sucesso, as dificuldades encontradas e também as sugestões de alternativas de implementação.

3.1. Sobre Modelos de Maturidade de *Software* do SEI

O SEI (*Software Engineering Institute*) é uma organização ligada à Universidade Carnegie Mellon na Pensilvânia, EUA. Em 1986, por solicitação do governo federal dos EUA, foi iniciado o esforço de desenvolver um método para a avaliação da capacidade de fornecedores de *software*. O SEI, com a assistência de vários colaboradores, inclusive Watt S. Humphrey com seu livro *Managing the Software Process*, elaborou um *framework* para a maturidade do processo de *software*. Após quatro anos de experiência com esse *framework*, o SEI evoluiu para alguns modelos de maturidade, sendo o CMM-SW [PAULK *et al*., 1994] o mais conhecido e que é voltado exclusivamente para o desenvolvimento e manutenção de *software*. Esse modelo foi inicialmente usado pela comunidade de *software* entre 1991 e 1992. Sua última versão foi lançada no mercado em 1994.

O projeto de criação do *CMM Integration* foi lançado no mercado em 2003 [CHRISSIS *et al*., 2003], realizado em colaboração pelo SEI, membros do governo e da indústria de *software*. Com a combinação dos modelos foi criada uma estrutura (*framework*) para uso pelas organizações que buscam a melhoria de seus processos organizacionais relacionados à construção, aquisição e produção de *software*.

Com o avanço na utilização, o SEI identificou três dimensões críticas nas quais as organizações podem focalizar esforços com o objetivo de aprimorar seus negócios:

1) Pessoas;
2) Procedimentos;
3) Métodos e Ferramentas.

O que engloba essas três dimensões é o processo de trabalho. Os processos permitem que a organização alinhe a sua forma de fazer negócios. Permitem que conhecimentos sejam incorporados para aprimorar o próprio processo. Processos possibilitam o nivelamento dos recursos e o exame das tendências do mercado. CHRISSIS

et al [2003] considera que "a qualidade de um sistema ou produto é altamente influenciada pela qualidade dos processos usados no seu desenvolvimento e manutenção", e diante disso, o SEI procurou elaborar modelos que incorporassem essa premissa. Essa atitude é confirmada mundialmente pelos movimentos em direção à adoção de modelos de qualidade como Six Sigma [CROSBY, 1979], Total Quality [CROSBY, 1979], ISO [CROSBY, 1979].

Os modelos de maturidade de *software* foram adotados, desde sua criação, por organizações visando principalmente o desenvolvimento de novos projetos de *software*. A aplicação das recomendações dos modelos na fase de manutenção de *software*, após o seu desenvolvimento e implantação também pode ser altamente vantajosa e até mesmo fundamental durante toda a vida útil do *software*, ou seja, até que o *software* seja descontinuado. Alguns cuidados devem ser tomados na adaptação das práticas para processos de manutenção, especialmente nas consideradas emergenciais ou corretivas, quando não são consideradas como novas versões do *software*. Segundo CAPPELLI [2000]:

> "Um modelo de maturidade deve ser designado também para apoiar os processos de manutenção no *software* que passam a ocorrer logo após a sua primeira implantação".

3.2. As Áreas de Processo do CMMI

Os corpos de conhecimento englobados pelo CMMI são:

- Engenharia de Sistemas e de *Software*
- Desenvolvimento Integrado de Produtos e Processos
- Fornecedores

As práticas que compõem esses corpos de conhecimento estão apresentadas em **Áreas de Processo** que agrupam metas, práticas específicas e práticas genéricas que, na sua adoção devem ser adequadas à realidade da organização. As práticas específicas estão ligadas às atividades que devem ser executadas para que a meta estabelecida seja alcançada e subdividem-se em 'subpráticas'. As práticas genéricas dizem respeito às características e compromissos da organização e da equipe, como habilidades para executar (as atividades), compromisso da organização em prover infra-estrutura (recursos, ambiente etc), verificações para identificar a utilização correta e indiscriminada do processo definido.

As áreas de processo do CMMI estão organizadas de acordo com as categorias apresentadas na tabela 01:

Tabela 01- Estrutura de Áreas de Processo por Categoria

Categorias	Áreas de Processo
Gerência de Processos	OPF – Foco Organizacional no Processo
	OPD – Definição Organizacional do Processo
	OT – Treinamento Organizacional
	OPP – Desempenho Organizacional do Processo
	OID – Implementação e Inovação Organizacional
Gerência de Projeto	PP – Planejamento de Projeto
	PMC – Controle e Monitoração de Projeto
	SAM – Gerência de Acordo de Fornecedor
	IPM – Gerência Integrada de Projeto
	RSKM – Gerência de Risco
	IT – Coordenação Integrada de Equipes
	ISM – Gerência Integrada de Fornecedor
	QPM – Gerência Quantitativa de Projeto

Categorias	Áreas de Processo
Engenharia	REQM – Gerência de Requisitos
	RD – Desenvolvimento de Requisitos
	TS – Solução Técnica
	PI – Integração de Produto
	VER – Verificação
	VAL - Validação
Support	CM – Gerência de Configuração
	PPQA – Garantia de Qualidade de Processo e Produto
	MA – Medições e Análise
	DAR – Análise de Decisão e Solução
	OEI – Ambiente Organizacional para Integração
	CAR – Análise Causal e Solução

3.3. Os Níveis de Capacidade e de Maturidade

Os níveis de maturidade usados no CMMI descrevem um caminho evolucionário para a organização que deseja a melhoria de seus processos de desenvolvimento e manutenção de produtos e serviços relacionados ao *software*. Um nível de maturidade é um plano bem definido de evolução para se alcançar um processo de *software* maduro. Cada nível de maturidade compreende um conjunto de metas e recomendações de práticas

de processo que, quando alcançadas, estabilizam um importante componente do processo de *software* [PAULK *et al*., 1994].

A adoção das práticas de uma área de processo pode ser implementada por duas formas de representação: contínua ou nivelada. A representação contínua é uma abordagem flexível para a melhoria dos processos. A organização pode escolher as áreas de processo que considere mais relevantes para os seus objetivos e adequar a sua implementação. A escolha das áreas de processo adotadas pode levar a organização a ser avaliada em diferentes níveis em diferentes áreas. Na representação nivelada, a abordagem é a mesma sugerida pelo CMM-SW, ou seja, todas as áreas de processo de um nível de maturidade devem ser adotadas para que a organização seja considerada como em pleno exercício naquele nível de maturidade.

As duas representações para a adoção das práticas, a contínua e a nivelada, permitem que a organização aprimore seus processos de forma gradual. Quando a representação escolhida é a contínua, as áreas de processos adotadas na organização são escolhidas de forma direcionada aos objetivos da organização. Nesse caso, ao ser efetuada uma avaliação (*assessment*) da conformidade ao modelo CMMI, a mesma é voltada para uma classificação segundo o seu **Nível de Capacidade**, os quais são classificados de zero a cinco. Na adoção da representação nivelada, todas as áreas de processo referentes ao nível devem ser adotadas e sua avaliação é segundo seu **Nível de Maturidade**, numa escala de um a cinco. Apesar da nomenclatura diferenciada, o conceito de níveis é o mesmo: definem a mudança do estado de situações menos maduras para mais maduras, medidas pelas informações quantitativas sobre as melhorias em direção ao alcance dos objetivos de negócio. Para atingir um determinado nível, a organização deve satisfazer todas as metas da(s) área(s) de processo alvo(s) da melhoria.

Nas duas representações o mesmo conteúdo essencial e os mesmos componentes do modelo deverão ser seguidos. As estruturas das representações contínua e nivelada são apresentadas nas figuras 03 e 04, respectivamente.

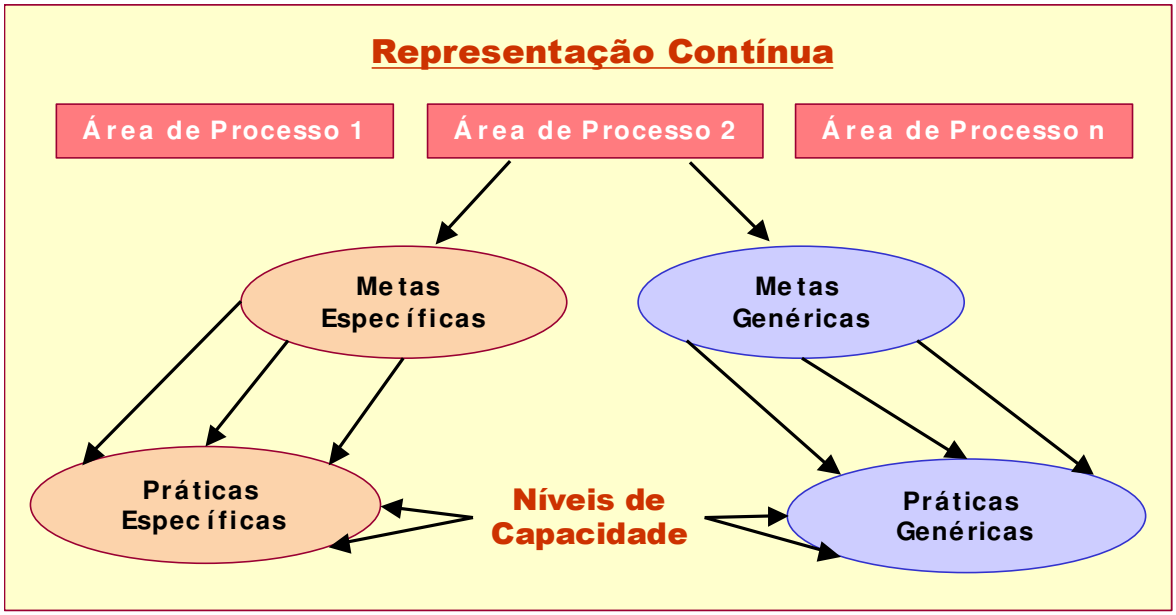

Figura 03 – Estrutura da Representação Contínua

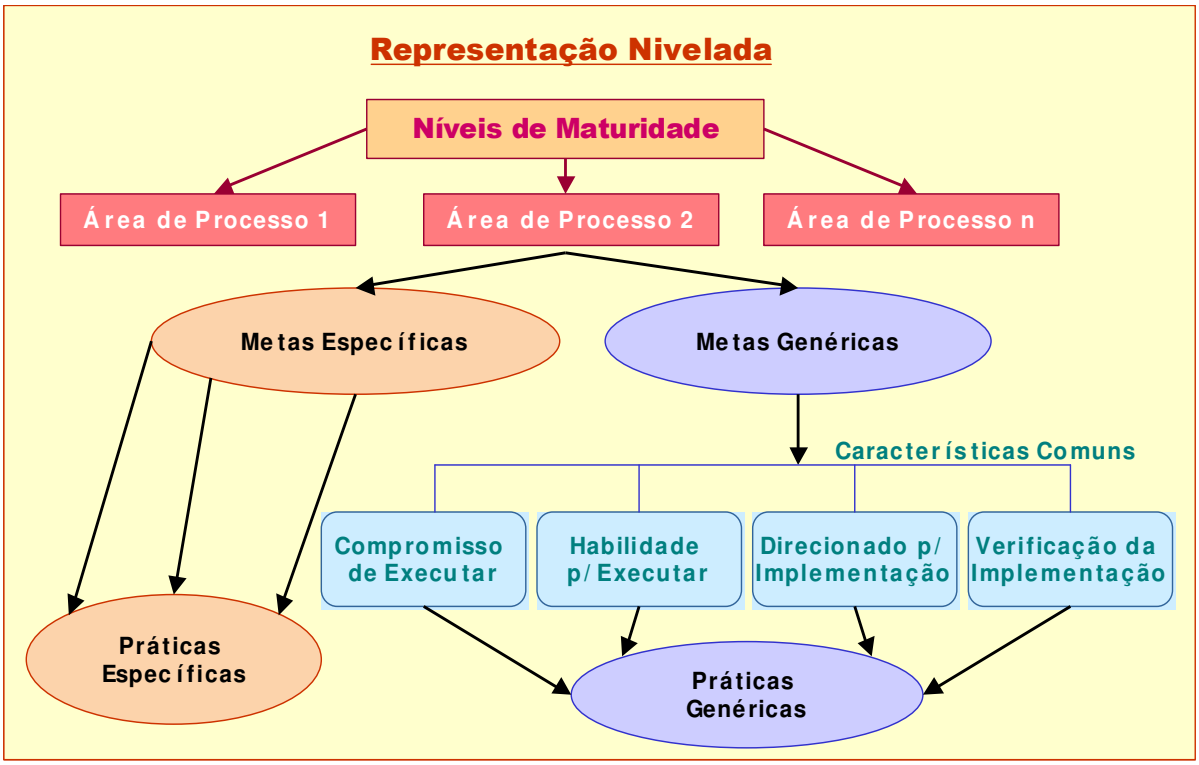

Figura 04 – Estrutura da Representação Nivelada

Na comparação das duas representações podemos observar similaridades. No caso da representação nivelada, ressaltamos a subdivisão das metas genéricas em características comuns - *common features* – que, quando agrupadas, compõem as práticas genéricas.

A tabela 02 apresentada a seguir ilustra uma comparação dos seis níveis de capacidade aos cinco níveis de maturidade.

Tabela 02 – Comparação de Níveis de Capacidade e de Maturidade

Nível	***Representação Contínua*** **Níveis de Capacidade**	***Representação Nivelada*** **Níveis de Maturidade**
0	Incompleto	Não se aplica
1	Executado	Inicial
2	Gerenciado	Gerenciado
3	Definido	Definido
4	Gerenciado Quantitativamente	Gerenciado Quantitativamente
5	Otimizado	Otimizado

3.4. As Áreas de Processo de Nível 2 de Maturidade

No nível 2 de maturidade de *software*, 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 são mantidas mesmo em situações de stress. Quando estas práticas estão em vigor, os projetos são realizados e gerenciados de acordo com seus planos, procedimentos e documentações.

Neste nível de maturidade o estado dos produtos de trabalho e a entrega de serviços são visíveis em pontos pré-determinados (p.ex. marcos temporais importantes e ao final de grandes etapas de tarefas). Os compromissos são estabelecidos entre os *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 áreas de processo referentes ao nível 2 de maturidade de *software* são:

- **CM** – Gerência de Configuração
- **MA** – Medições e Análise
- **PMC** – Controle e Monitoração de Projeto
- **PP** – Planejamento de Projeto
- **PPQA** – Garantia de Qualidade de Processo e Produto
- **REQM** - Gerência de Requisitos
- **SAM** – Gerência de Acordo de Fornecedor

3.5. Sobre a Área de Processo Foco do Trabalho

Neste trabalho escolhemos como alvo de nossa implementação a seguinte área de processo: **REQM – *Requirements Management* – Gerência de Requisitos**.

A área de processos de Gerência de Requisitos está contida no corpo de conhecimentos de Engenharia do CMMI, que envolve terminologia e disciplinas técnicas voltadas para a engenharia de *software* e engenharia de sistemas. Esta área garante o estabelecimento dos requisitos do projeto e a definição de seu escopo. Descreve as

atividades para a obtenção dos requisitos e o controle de suas alterações, que podem afetar todas as demais áreas de processo de engenharia ao longo do ciclo de vida de desenvolvimento do *software*. Também prevê o mapeamento dos requisitos aos planos, processos, componentes e produtos de trabalho e de *software*, que permite o rastreamento e controle dos mesmos durante todas as fases do ciclo de vida de desenvolvimento. Freqüentemente a gerência de requisitos é uma seqüência recursiva de eventos. A gerência de requisitos é uma área dinâmica e fundamental para um processo controlado e disciplinado de engenharia. A transcrição parcial da parte II do livro "*CCMI – Guidelines for Process Integration and Product Improvement*" [CHRISSIS *et al*., 2003], referente às metas, práticas e 'subpráticas' específicas da área de processos de Gerência de Requisitos está apresentada no Anexo 1.

O objetivo da Gerência de Requisitos é gerenciar os requisitos dos produtos e componentes de produtos de um projeto e identificar inconsistências entre esses requisitos e os planos e produtos de trabalho. Isso inclui tanto os requisitos recebidos pelo projeto quanto os gerados no projeto. Estão englobados todos os tipos de requisitos: técnicos, não-técnicos, requisitos organizacionais e as regras particulares de um negócio. A equipe de desenvolvimento e os representantes do cliente são estabelecidos com seus respectivos papéis (aprovador, provedor de requisito, representante nos testes, 'desenvolvedor' etc). O projeto deve executar atividades adequadas para garantir que um conjunto acordado de requisitos estabeleça o escopo do projeto. Todas as questões e conflitos dos requisitos devem ser analisados pelos provedores autorizados de forma a prevenir mal-entendidos logo no início do projeto. Na ausência dessa formalização de escopo, problemas podem ser identificados em fases mais avançadas do projeto, acarretando custos e repetição de trabalho desnecessários, provocando freqüentemente impactos no cronograma estabelecido.

A Gerência de Requisitos, por meio da definição clara dos mesmos, possibilita à equipe responsável pelo desenvolvimento do projeto o estabelecimento de compromissos em termos de custos e prazos do projeto. Com o controle e documentação das alterações dos requisitos são gerados menores impactos de custos e cronograma, além de viabilizar a negociação de sua aplicação mediante aprovação dos clientes (aprovadores, provedores de

requisitos). O rastreamento bi-direcional entre as fontes de requisitos e os produtos ou componentes de produtos garante a implementação de todos os requisitos solicitados no projeto e dá aos *stakeholders* a visibilidade sobre a sua evolução ao longo do ciclo de vida de desenvolvimento do *software*.

A escolha dessa área de processo pode ser justificada pela análise de algumas de suas características. Em primeiro lugar devemos considerar que as áreas de processo do CMMI possuem interligações entre si, estando algumas práticas sujeitas à implementação em conjunto, de forma a otimizar a sua aplicabilidade. Como exemplo consideremos as práticas das áreas **PMC** – *Project Monitoring and Control* e **PP** – *Project Planning*. É altamente recomendável que a implementação dessas duas áreas de processos seja adotada em conjunto, pois as recomendações de acompanhamento e controle de projeto perdem grande parte de sua efetividade na melhoria de processos sem que sejam adotadas também as recomendações de planejamento de projeto. No caso da área de processo de Gerência de Requisitos, por conter recomendações referentes à fase inicial de um projeto de desenvolvimento de *software* – levantamento preliminar e definição de escopo do projeto, torna-se uma área de aplicabilidade menos dependente das outras áreas. Por outro lado, esta área permite o controle das alterações dos requisitos ao longo do desenvolvimento do *software*, o que a torna uma área fundamental a todas as outras áreas de processo. Também é importante ressaltar que se trata de uma área de processo de nível de maturidade 2 (dois) ou nível de capacidade 3 (três), ou seja, necessária à maioria dos processos de melhoria, pois constitui o aprimoramento básico das práticas de organizações imaturas ou pouco maduras.

Ressaltamos, porém, que o escopo do trabalho refere-se à formalização das quatro primeiras práticas específicas desta área de processo, que totalizam quatorze ‘subpráticas’. As práticas em questão são:

(1.1) *Obtain an understanding of requirements*;

(1.2) *Obtain commitment to requirements*;

(1.3) *Manage requirements changes*;

(1.4) *Maintain bidirectional traceability of requirements*.

A prática específica (1.5) *Identify Inconsistencies between project work and requirements* não foi implementada, pois o estudo de caso inclui apenas a fase inicial do projeto, referente ao levantamento dos requisitos e estabelecimento do escopo. Os demais documentos das fases subseqüentes do processo de *software* não estão incluídos neste estudo de caso. Dessa forma, tornou-se inviável a formalização da quinta prática específica.

4. Formalização de Regras de Negócio

> **Formalismo** [De *formal* + *-ismo*] S. m. **5**. *Filos*. Tendência a priorizar, em qualquer domínio do conhecimento, as características e relações formais dos objetos.
>
> **Formalizar** [De *formal* + *-izar*] V. t. d. **3**. Executar conforme as regras ou cláusulas.
>
> **4**. *Lóg*. Substituir os conceitos e as relações dos sistemas por símbolos sujeitos a regras operatórias bem definidas.
>
> Dicionário Aurélio Século XXI

Uma das formas mais estudadas de representação do conhecimento humano e do raciocínio é o uso da Lógica [BRATKO, 2001]. As principais virtudes da lógica em geral, e da lógica de primeira ordem em particular, para a representação de conhecimento são: sua semântica bem definida e o fato de ser um formalismo declarativo.

A busca por maior grau de estruturação e formalismo, justifica-se pela efetiva diminuição do grau de ambigüidade da linguagem natural, possibilitando a análise, descrição e revisão do objeto formalizado contra um padrão estruturado de sentenças. No contexto empresarial isso pode prevenir problemas e uniformizar o entendimento dos empregados e colaboradores sobre as regras e procedimentos, além de viabilizar a documentação do negócio levando em conta a cultura organizacional.

4.1. Lógica

O estudo da lógica é o estudo dos métodos e princípios usados para distinguirmos o raciocínio correto do incorreto. Podemos considerar que uma pessoa, dotada de seu intelecto inato, com o estudo da lógica tem maior probabilidade de raciocinar corretamente

do que aquela que não aprofundou seus conhecimentos nos seus princípios gerais. O problema central que a lógica se incumbe em tratar é a distinção entre o raciocínio correto e o incorreto [COPI, 1961]. Os métodos e as técnicas do lógico foram desenvolvidos, primordialmente, com a finalidade de elucidar essa distinção. O lógico está interessado em todos os raciocínios, independentemente do seu conteúdo, mas só a partir desse ponto de vista especial. COPI [1961] nos relembra que a interrogação do lógico é sempre "... a conclusão a que se chegou deriva das premissas usadas ou pressupostas?". Se as premissas fornecem bases ou boas provas para a conclusão, se a afirmação da verdade das premissas garante a afirmação de que a conclusão também é verdadeira, então o raciocínio é correto.

Vale esclarecermos aqui alguns dos termos usados pelo lógico. A inferência é o processo pelo qual se chega a uma proposição, afirmada na base de uma ou mais proposições aceitas como ponto de partida do processo. As proposições podem ser afirmadas ou negadas, podem ser consideradas verdadeiras ou falsas. Uma proposição pode também ser chamada de "enunciado" ou "declaração". Um argumento é qualquer grupo de proposições tal que se afirme ser uma delas derivada das outras, as quais são consideradas provas evidentes da verdade da primeira. Premissa e conclusão são termos relativos: uma única proposição pode ser premissa num argumento, como também pode ser conclusão em outro [COPI, 1961].

4.2. Formalização – Estruturação BRG

O projeto do *GUIDE Business Rules Project* [BRG, 2001] foi organizado em 1993 com o objetivo de formalizar uma abordagem para a identificação e articulação de regras que definem a estrutura e o controle da operação de uma organização. Usualmente as regras que definem a estrutura e as funções de uma empresa são documentadas até certo ponto. Outras regras não são tão bem articuladas, muitas vezes nem isso. O projeto do BRG (*Business Rules Group*) [2001] propõe-se a definir uma forma de aprimorar a

articulação das regras e, dessa forma, corrigir as falhas de regras de negócio que não foram adequadamente documentadas no passado.

À medida que a tarefa de definir as regras de negócio seja melhor compreendida, o BRG [2001] acredita que técnicas e ferramentas serão desenvolvidas para apoiar os elementos ausentes dessa tarefa. As técnicas poderão incluir métodos formais de descrição rigorosa de regras, com ferramentas para traduzir esse formalismo diretamente em código de programas computacionais ou em outros componentes de implementação das aplicações de negócio. Podemos constatar que, apesar de já existirem há algum tempo várias técnicas de modelagem gráfica para descrever estruturas de dados e funcionalidades, os métodos formais para a descrição de regras são relativamente novos e ainda necessitam ser refinados.

As técnicas de análise de sistemas têm evoluído ao longo do tempo e vêm provendo métodos de descrição de vários aspectos de negócios. Atualmente é possível desenhar modelos que descrevem a maneira como a informação flui pela organização, a seqüência de ações em que a organização funciona, a estrutura operacional da informação, e muitos outros aspectos. De alguma forma, tudo isso constitui as "regras do negócio", mas um importante aspecto continua a descoberto: o conjunto de regras que determina como um negócio funciona – que previne, causa ou sugere que coisas aconteçam.

Segundo o BRG [2001], o contexto de implementação adotado pelo seu esforço corrente é voltado especificamente para a estrutura computacional das informações de um negócio com suas restrições e outros aspectos relacionados à motivação de um sistema de informação para uma organização. Ou seja, a preocupação do BRG [2001] é concernente às regras do negócio que afetam o armazenamento de valores persistentes de dados, descritos de uma forma tecnologicamente neutra. O estágio da versão (1.3) do projeto do BRG [2001] não está preocupado com as regras de negócio que não constituem um componente de um sistema de informações.

As regras de negócio são declarações que definem ou restringem alguns aspectos de um negócio. No projeto *GUIDE Business Rules Project* [BRG, 2001], as regras de negócio devem ser atômicas, ou seja, não podem ser subdivididas em outras regras. Isso pode ser

visto sob duas perspectivas: da organização, que se aplica ao comportamento das pessoas na empresa; e dos sistemas de informação, que se refere aos fatos que são registrados como dados e restrições na mudança dos valores desses fatos. Ou seja, quais dados devem ou não ser registrados nos sistemas de informação.

Uma declaração de regra de negócio deve ser classificada em uma das quatro categorias:

- **Definição de termos do negócio**: o elemento mais básico de uma regra de negócio é a linguagem usada para expressá-la. A definição de um termo em si próprio é uma regra de negócio que descreve como as pessoas pensam e falam sobre as coisas, ou seja, como compreendem o termo no contexto da empresa, levando em conta a sua cultura.

- **Fatos que relacionam os termos entre si**: a natureza ou estrutura operacional de uma organização pode ser descrita pelos fatos que relacionam os termos de negócio uns aos outros. Os fatos podem ser documentados na forma de sentenças em linguagem natural ou como relacionamentos, atributos e estruturas de generalização em modelos gráficos.

- **Restrições ou declarações de ação:** toda empresa restringe comportamentos de alguma forma, e isso está fortemente relacionado às restrições sobre a escolha dos dados que devem ou não ser atualizados. Para evitar que um registro seja feito, muitas vezes é necessário evitar que uma ação ocorra.

- **Derivações:** as regras de negócio (incluindo as leis da natureza) determinam como o conhecimento em uma forma pode ser transformado em outro conhecimento numa forma diferente.

Segundo CORRÊA [2002], podemos acrescentar à definição de categorias do BRG [2004] os seguintes esclarecimentos:

a. Os fatos implicam em possibilidade.

b. As restrições implicam em obrigatoriedade ou impossibilidade.

c. As restrições também podem ser entendidas como habilitações de ações, ou seja, regras de negócio que levam à execução de uma ação na ocorrência de um determinado evento (pares de "condição-ação").

d. As derivações também podem ser entendidas como fórmulas de cálculo.

Alguns princípios básicos se aplicam à definição de regras de negócio [BRG, 2001]:

- **Expressões explícitas** – é necessário explicitar as declarações das regras de negócio, seja na forma gráfica, seja como linguagem formal baseada em lógica.
- **Representação coerente** – como a formalização de regras de negócio vem se tornando parte do processo de análise de um sistema de informação, é desejável que seja estabelecida uma representação coerente e única para todos os tipos de regras de negócio.
- **Extensão evolucionária** – conceitualmente, a representação de restrições pode ser entendida como uma extensão de um diagrama de entidades e relacionamentos (MER – Modelo de Entidades e Relacionamentos), adicionando-se as restrições aos elementos estruturais do diagrama. Apesar disso, o objetivo do projeto BRG [2001] é descrever a natureza das regras de negócio independente da sua forma de representação.
- **Natureza declarativa** – uma regra é declarativa, não é procedimental. Descreve um estado possível e desejado que é sugerido, requerido ou proibido. Pode ser condicional, mas não descreve as etapas de transição de um estado a outro, ou para coibir a transição.

4.3. Estruturação de Regras em Português Estruturado

Como etapa inicial de formalização de regras de negócio, sugerimos a adoção de uma proposta de linguagem, que representa um subconjunto da língua portuguesa, a qual chamamos **Português Estruturado**. Esta linguagem possibilita uma especificação legível das regras para os usuários e é uma representação consistente e não ambígua do conhecimento, quando passada para a lógica de primeira ordem.

A representação do português estruturado se baseia em modelos de sentença como forma de viabilizar a representação das regras de negócio. Conforme a categorização proposta pelo BRG [2001] para as regras de negócio e complementada por CORRÊA [2002], para cada categoria de regras foi elaborado um ou mais modelos de sentença. Estes modelos devem ser suficientemente expressivos para permitir a representação do conhecimento a respeito do domínio de forma eficiente. Evidenciamos os principais benefícios diretos dessa proposta:

- eliminar as ambigüidades;
- permitir a verificação de inconsistências;
- facilitar a circulação das regras entre os envolvidos na organização, sem a necessidade de documentações adicionais;
- permitir a validação das regras.

As categorias de regras podem ser expressas em um padrão em linguagem Português Estruturado, que possibilita organização e consistência. A esse padrão pode ser acoplado um grupo de modelos de sentenças de forma a padronizar as expressões e facilitar sua posterior tradução para Lógica de Primeira Ordem.

A proposta de linguagem, pertencente ao escopo da ferramenta RÈGULA [DIAS *et al*., 2004], e reformulada para a sua segunda versão descrita em CORRÊA *et al*. [2001] são ambas baseadas nos modelos de sentença propostos pelo BRG [2001]. Apresentamos a

seguir as notações utilizadas para os modelos de sentença da linguagem Português Estruturado.

Tabela 03 - Notação Utilizada para a Especificação da Linguagem Português Estruturado

Nome Completo	Abreviatura	Domínio
Artigo	artigo	o, a, um, uma
Verbo	verbo	verbo
Preposição	prep	preposição
Termo comparativo	comp	=, >=, >, <, <=, ...
Valor	valor	qualquer valor
Domínio	domínio	domínio
Procedimento	procedimento	procedimento
Condição	condição	condição
Termo	termo	objeto a ser tratado
Definição Textual	definição textual	texto que exprime o sentido desejado para um termo

São os seguintes os modelos de sentença para cada categoria de regra de negócio:

- **Termos:**

  (I) <termo> É UM(A) <definição textual>

  (II) <termo> É O (A) <definição textual>

  Ex: “Requisito de projeto de *software* É UMA condição que deve ser atendida pelo *software*, necessária a um cliente ou usuário para resolver um problema ou alcançar um objetivo”.

- **Cálculos:**

  (I) <termo> É CALCULADO COMO <fórmula>

  (II) <termo> É CALCULADO COMO <procedimento>

  Ex: “Valor total da compra É CALCULADO COMO a soma dos valores dos itens de compra.”

- **Derivações:**

  (I) SE <condição>, ENTÃO <termo> É CONSIDERADO COMO <estado>

  Ex: “SE for estabelecido um acordo informal sobre os requisitos do projeto de *software*, ENTÃO o requisito de projeto de *software* É CONSIDERADO COMO aceito”.

- **Fatos:**

  (I) <termo1> TEM PERMISSÃO PARA <verbo> {<prep>} {<artigo>} <termo2>

  (II) <termo1> TEM PERMISSÃO PARA <verbo> <comp> <valor> <termo2>

  (III) <termo1> TEM PERMISSÃO PARA <verbo> <termo2> <comp> <valor>

  (IV) <termo1> É UM ELEMENTO DE <termo2>

  (V) <termo1> É SUBTIPO DE <termo2> QUE <verbo> <termo3> [, <verbo> <termo4> ... E <verbo> <termoN>]

  (VI) <termo1> TEM COMO ATRIBUTO <termo2>

  (VII) <termo1> TEM COMO PARTE <termo2>

  (VIII) <termo1> ESTÁ RELACIONADO A <termo2> POR <verbo> [COM GRAU <M> : <N>]

  (IX) <termo2> POSSUI COMO DOMÍNIO <domínio>

  Ex: “Os participantes do projeto de *software* TÊM PERMISSÃO PARA consultar o registro do requisito de projeto de *software*”.

- **Habilitadores de Ação:**

  (I) SE <condição>, ENTÃO EXECUTAR <ação>

  Ex: "SE saldo menor que zero, ENTÃO EXECUTAR cálculo de juros".

- **Restrições:**

  (I) <termo1> DEVE OBRIGATORIAMENTE <verbo> {<prep>} {<artigo>} <termo2>

  (II) <termo1> DEVE OBRIGATORIAMENTE <verbo> <comp> <valor> <termo2>

  (III) <termo1> DEVE OBRIGATORIAMENTE <verbo> <termo2> <comp> <valor>

  (IV) <termo1> NÃO TEM PERMISSÃO PARA <verbo> {<prep>} {<artigo>} <termo2>

  (V) <termo1> NÃO TEM PERMISSÃO PARA <verbo> <comp> <valor> <termo2>

  (VI) <termo1> NÃO TEM PERMISSÃO PARA <verbo> <termo2> <comp> <valor>

  (VII) <termo1> DEVE OBRIGATORIAMENTE SER <comp> <valor>

  Ex: "O requisito de projeto de *software* DEVE OBRIGATORIAMENTE ser registrado pelo gerente de projeto".

4.4. Formalização em Lógica de Primeira Ordem

A Lógica Proposicional [CASANOVA *et al*., 1987] é considerada a forma mais simples da Lógica e baseia-se no fato de que uma proposição só pode assumir dois valores possíveis: falso ou verdadeiro. A **LPO** - Lógica de Primeira Ordem, ou Lógica de Predicados, é considerada uma extensão da Lógica Proposicional.

Suas principais vantagens para a representação do conhecimento são a sua semântica bem definida e seu formalismo declarativo. Seus elementos fundamentais são os objetos e seus predicados. Os predicados são utilizados para representar os conhecimentos

sobre um determinado objeto, levando em consideração a estrutura interna das sentenças. Os predicados podem estar relacionados com mais de um objeto, e também podem estar conectados por meio de operadores lógicos.

A LPO utiliza palavras ou símbolos especiais, chamados de quantificadores que tornam as proposições mais exatas ou bem-definidas. Temos então, além dos conectivos do cálculo proposicional e os parênteses, os seguintes símbolos:

- Variáveis: x, y, z, ...
- Constantes: a, b, c, ...
- Símbolos de Predicados: P, Q, R, S, ...
- Quantificadores: $\forall$ (universal) e $\exists$ (existencial)

A LPO aplica-se, ainda que de forma limitada, à representação da linguagem natural, uma vez que pode representar conteúdos expressos em frases declarativas. Assim, dada uma frase em português, pode-se associar uma fórmula da linguagem do cálculo proposicional, atribuindo símbolos sentenciais às partes da sentença que não possuem nenhuma conjunção ou palavras que possam ser associadas aos símbolos lógicos.

4.5. Linguagem de Programação em Lógica – PROLOG

Como vantagem adicional do uso da Lógica de Primeira Ordem, vale ressaltar a existência de linguagens de programação baseadas em lógica, como, por exemplo, a linguagem PROLOG [CASANOVA *et al*., 1987], [BRATKO, 2001]. A ferramenta RÉGULA [DIAS *et al*., 2004] faz uso dessa linguagem na implementação de suas funcionalidades. Programas em PROLOG são baseados no subconjunto da lógica de

primeira ordem, composto pelas cláusulas de Horn, onde cada cláusula só pode conter no máximo um literal positivo na cabeça da cláusula.

Segundo CASANOVA *et al*. [1987], um programa em lógica é um modelo de um determinado problema ou situação expresso por meio de um conjunto finito de sentenças lógicas. Um programa em lógica não é a descrição de um procedimento para obter soluções de um problema, mas assemelha-se mais a um banco de dados, pois assim como nestes, num programa em lógica são estabelecidas relações entre objetos. A programação em lógica exemplifica um estilo mais fundamental, que pode ser chamado de programação declarativa (não-procedimental), em contraste com a programação declarativa ou imperativa. A programação declarativa engloba ainda a programação funcional (linguagem LISP) e as linguagens de consultas a bancos de dados, como o SQL. Apesar da programação funcional já ser conhecida desde 1960, a programação em lógica só ganhou ímpeto a partir de 1973 com a criação da linguagem PROLOG – *Programming in Logic* na Universidade de Marselha, na França.

Pode-se expressar conhecimento em PROLOG por meio de cláusulas de dois tipos: fatos e regras. Um fato em PROLOG denota uma verdade incondicional, enquanto as regras em PROLOG definem as condições que devem ser satisfeitas para que uma determinada declaração seja considerada verdadeira. Como os fatos e as regras em PROLOG podem ser utilizados em conjunto, nenhum componente dedutivo adicional precisa ser utilizado. Além disso, como são permitidas regras recursivas e não-determinismo, os programadores podem obter descrições muito mais claras, concisas e não-redundantes da informação que desejam representar. Não há distinções entre argumentos de entrada e de saída, qualquer combinação de argumentos pode ser empregada [ARITY, 2002].

Uma visão apropriada da interpretação de um programa PROLOG em termos matemáticos é a seguinte: o sistema PROLOG aceita os fatos e regras como um conjunto de axiomas e a consulta do usuário como um teorema a ser provado. A tarefa do sistema é demonstrar que o teorema pode ser provado com base nos axiomas representados pelo conjunto de cláusulas que constituem o programa.

Os conceitos de chamada (ou consulta) e de resposta na programação PROLOG diferem das noções tradicionais. Uma consulta a um programa em lógica é uma afirmação que exprime as condições a serem satisfeitas por uma resposta correta em presença da informação descrita pelo programa [CASANOVA *et al*., 1987]. A resposta de uma consulta a um programa em lógica não se limita apenas a indicar que uma suposição acerca da informação contida no programa é falsa ou verdadeira. A resposta efetivamente exibe informação extraída do programa e pode vir acompanhada de uma explicação sobre como foi obtida, expressa em termos da refutação que a gerou.

Um programa em PROLOG consiste basicamente de: a) uma declaração de fatos (*facts*) sobre objetos e suas relações; b) definições de regras (*rules*) sobre objetos e suas relações; c) consultas (*queries*) que são feitas sobre objetos e suas relações.

4.6. Sobre a Proposta de Formalização Foco do Trabalho

4.6.1. Primeira etapa de formalização

O principal objetivo da formalização das regras de negócio é a diminuição da ambigüidade da linguagem natural, possibilitando a análise, descrição e revisão das mesmas contra um padrão estruturado de sentenças. A proposta completa de formalização de regras de negócio prevê uma seqüência de etapas que parte de regras em linguagem natural e finaliza com um conjunto de sentenças lógicas em PROLOG. Uma das etapas, a tradução da linguagem natural para a linguagem Português Estruturado, pertence ao escopo da ferramenta RÈGULA [DIAS *et al*., 2004], já reformulada para a sua segunda versão descrita em CORRÊA *et al*. [2001].

Neste trabalho o escopo se compõe da aplicação da primeira etapa de formalização a um conjunto de regras aderentes a um processo de negócio. Com isso, propomos como

método a realização da tradução de regras de negócio em linguagem natural para um conjunto de termos e regras atômicas formuladas em modelos de sentenças pré-definidos do Português Estruturado, cuja principal vantagem que se apresenta é a diminuição da ambigüidade inerente à linguagem natural. As características desejáveis para regras de negócio, e que foram aplicadas na formalização, estão apoiadas nas recomendações de MORGAN [2002]:

- Atômicas: não podem ser subdivididas sem perda de informação;
- Não Ambíguas: possuem apenas uma interpretação;
- Compactas: tipicamente são elaboradas em uma única sentença;
- Consistentes: em conjunto provêem uma descrição unificada e coerente;
- Compatíveis: usam os mesmos termos em todo o modelo de negócio.

Esta primeira etapa de formalização foi aplicada ao conjunto de regras de negócio aderente ao processo foco de nosso trabalho, como mencionado na seção 3.5 (Sobre a área de processo foco do trabalho), ou seja, às práticas e 'subpráticas' da área de processo de Gerência de Requisitos proposta pelo CMMI [CHRISSIS *et al*., 2003].

Como resultado da aplicação da primeira etapa de formalização obtivemos um conjunto de termos e regras em Português Estruturado, que exprimem de forma mais objetiva e detalhada as 'subpráticas' da área de processo de Gerência de Requisitos do CMMI [CHRISSIS *et al*., 2003] Esses termos e regras resultantes configuram-se aptas a serem utilizadas nas etapas seguintes de formalização de regras de negócio na direção da automação. Mesmo que não sejam adotadas essas etapas seguintes, a utilização direta do conjunto de termos e regras formalizadas de uma área de processo do CMMI também se constitui como um valor agregado à área de processo, facilitando a adoção e adaptação das recomendações do CMMI por uma organização de desenvolvimento de software.

4.6.2. Automação com a ferramenta REGULA

Como complemento à nossa proposta de trabalho, sugerimos a utilização de uma ferramenta de automação, em nosso caso o *framework REGULA* [DIAS *et al*., 2004], aplicação informatizada de gerenciamento de regras de negócio. Essa ferramenta é baseada na estruturação proposta pelo BRG [2001] e permite o armazenamento e conversão das regras de negócio já formalizadas em Português Estruturado para a Lógica de Primeira Ordem e depois para a linguagem PROLOG. Apresenta-se como uma versão aprimorada da ferramenta *RULECHECK* [CORRÊA, 2002]. Com essa seqüência de traduções, torna-se possível obter as cláusulas em PROLOG correspondentes às regras de negócio e gerar uma base de conhecimento. Esta ferramenta está em fase de construção, inserida no contexto da Bancada Virtual para Projetos de Negócio [SCHMITZ *et al*., 2004].

O ambiente *REGULA* baseia-se num ideal de construção de sistemas de informação apoiado no uso de regras de negócio [DIAS *et al*., 2004]. A figura 05 apresenta o seu esquema de modelagem.

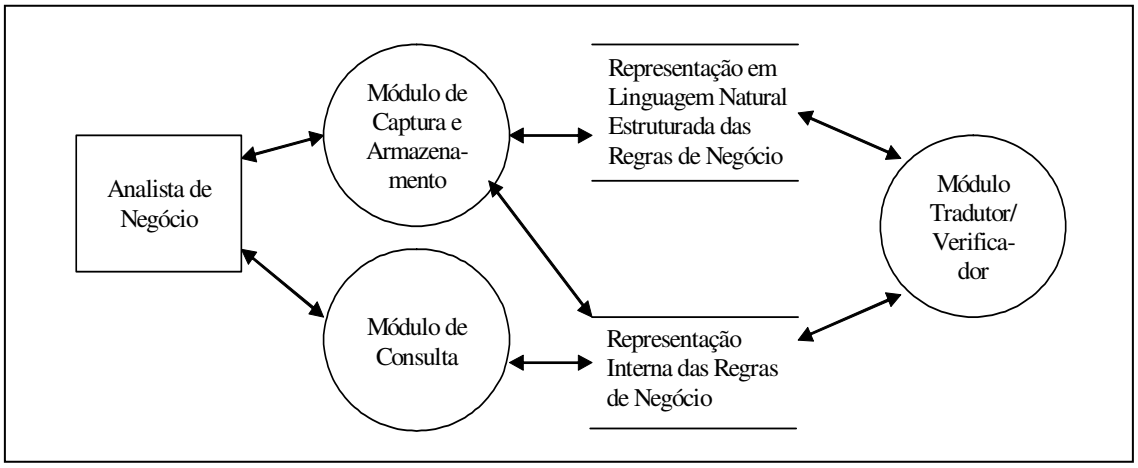

Figura 05 - Esquema de Modelagem do Ambiente REGULA

A ferramenta *REGULA* está sendo construída para se apresentar como uma interface moderna e amigável e possui cadastros de: Negócios, Áreas de Negócio, Termos e Regras

de Negócio. Permite as seguintes funcionalidades: a geração de Base de Conhecimento, Controle de Usuários (em fase de implementação), Emissão de Relatórios, Construção de Regras e Interpretação de Traduções para Lógica de Primeira Ordem (em fase de implementação). As variáveis do negócio devem ser definidas no *REGULA* como termos. A título de ilustração, apresentamos a seguir algumas de suas telas.

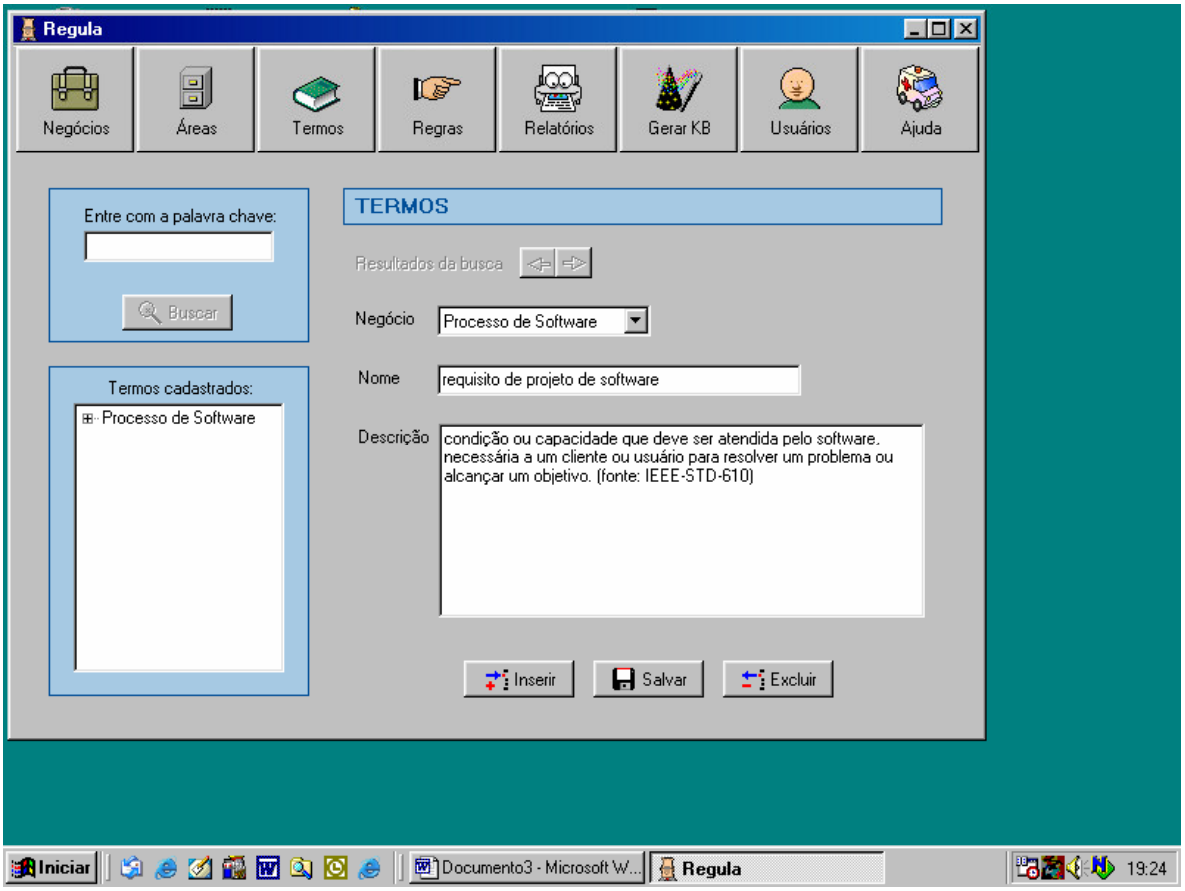

Figura 06 – Tela de Cadastramento de Termos da ferramenta *REGULA*

Em seguida, as regras definidas a partir dos termos cadastrados e do conjunto de modelos de sentenças em Português Estruturado podem ser também cadastradas nos *templates* disponíveis na ferramenta. Internamente o *REGULA* armazena as regras num

formato que consiste de sentenças na linguagem de programação em lógica PROLOG [BRATKO, 2001]. As figuras 07 e 08 a seguir apresentam as telas de cadastramento de regras de negócio com as janelas de detalhamento do modelo de sentença escolhido.

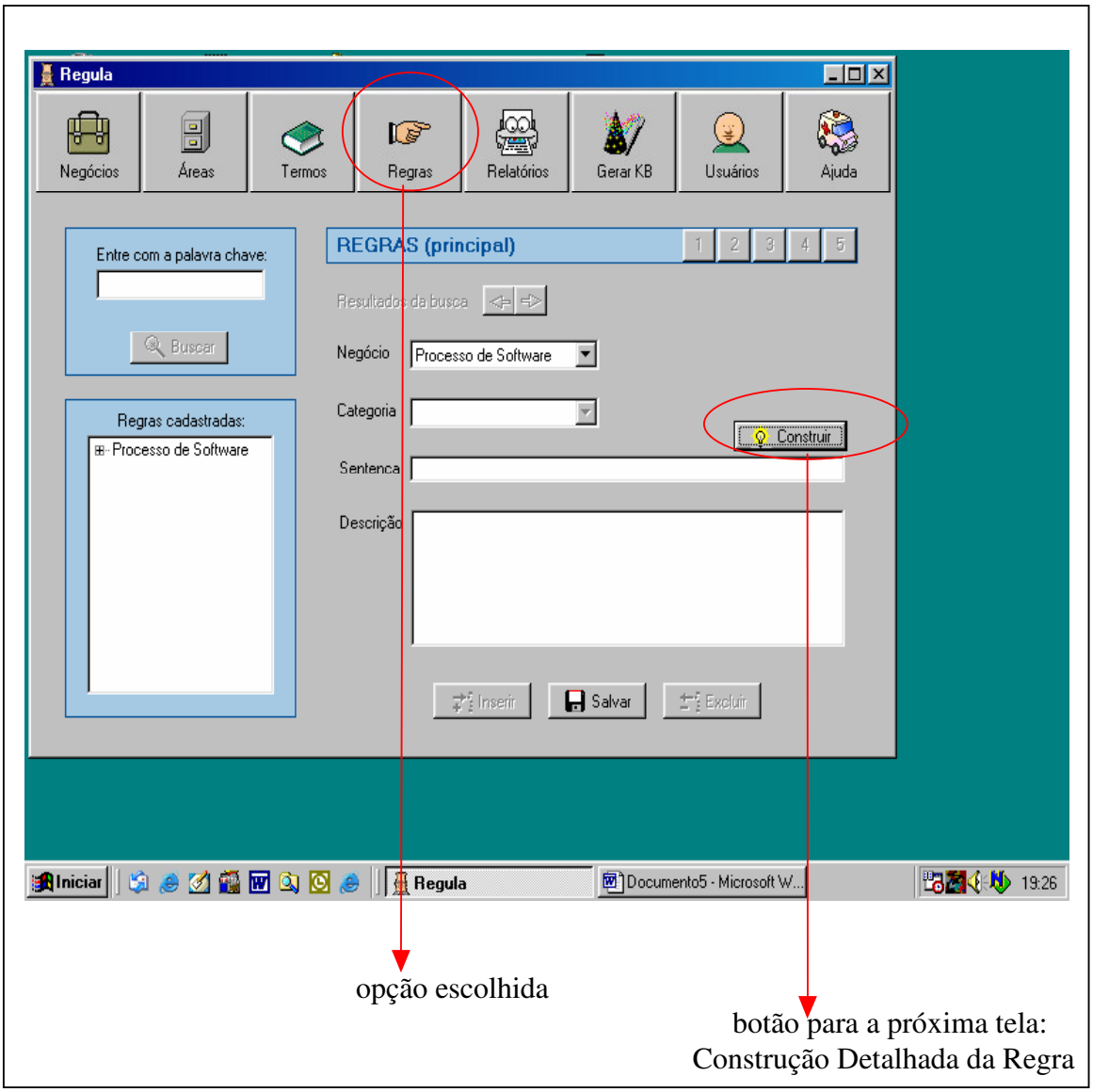

Figura 07 – Tela de Cadastramento de Regras de Negócio da ferramenta *REGULA*

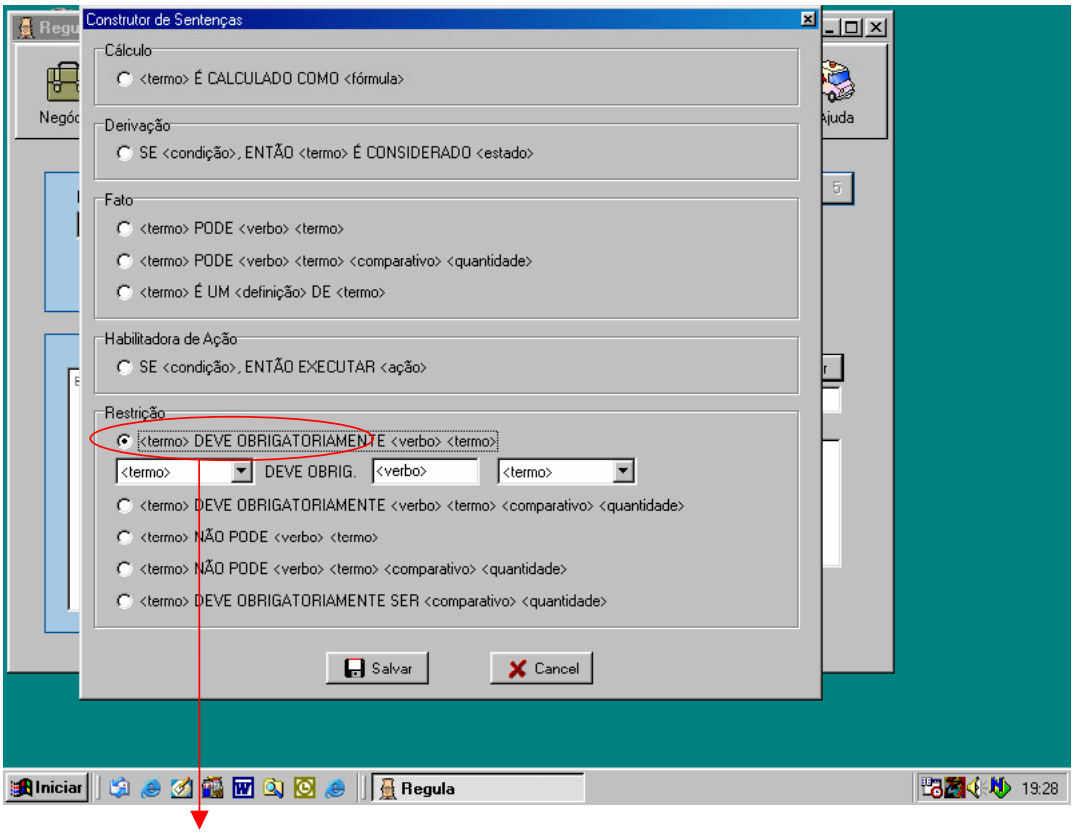

Figura 08 – Tela de Construção Detalhada de Regras da ferramenta *REGULA*

Além de propiciar a documentação e armazenamento de regras de negócio, o ambiente *REGULA* poderá tornar mais eficaz a análise e levantamento de requisitos de um processo de negócio, o que virá a facilitar o desenvolvimento do sistema de informações correspondente. Tal ferramenta também representará um avanço na busca de eficiência e eficácia na gestão de processos e de conhecimento numa organização.

Com os termos e regras de negócio cadastradas no *REGULA*, será possível a um operador (p.ex. um analista de negócios) submeter perguntas em Português Estruturado, utilizando o módulo de consultas às permissões da ferramenta. As perguntas deverão ser estruturadas num dos seguintes formatos:

- <Termo1> TEM PERMISSÃO PARA <verbo> <Termo2>?
- QUEM TEM PERMISSÃO PARA <verbo> <Termo2>?
- <Termo1> TEM PERMISSÃO PARA <verbo> O QUÊ?
- <Termo1> TEM PERMISSÃO PARA FAZER O QUÊ?
- QUEM TEM PERMISSÃO PARA FAZER O QUÊ?

O *REGULA* mapeará a pergunta em sua representação interna em PROLOG e a processará internamente. O resultado obtido será trazido de volta ao operador na forma textual padrão, onde FAZER indica que a consulta irá retornar, em caso de sucesso, um verbo ou expressão verbal. Similarmente, QUEM e O QUÊ resultarão, em caso de sucesso, um (ou mais) termo(s) catalogado(s). Os padrões de resposta para as duas primeiras perguntas são "SIM / NÂO" e, em caso de sucesso, a identificação da regra que satisfaz a pergunta.

Com a utilização do REGULA os operadores envolvidos na formalização das regras de negócio deverão ser capacitados apenas na tradução das mesmas para sentenças em Português Estruturado. Após o cadastramento, o operador poderá validar o conjunto de regras com a mesma estruturação pela utilização do módulo de consultas a permissões, proibições e obrigações. Essa capacitação permitirá que ele, mesmo sem o conhecimento de linguagens técnicas e formais, possa modelar e verificar a consistência das regras de negócio dos diversos domínios de aplicação da organização. As traduções seguintes para LPO e PROLOG serão efetuadas automaticamente pela ferramenta, eliminando as dificuldades que a capacitação dos operadores nessas linguagens trariam à organização, o que resultaria na inviabilidade da formalização, perdendo-se assim suas vantagens.

5. Formalização das Regras de Negócio do Processo REQM do CMMI

O CMMI apresenta recomendações de práticas gerais e específicas a serem adotadas por organizações cuja atividade fim seja o desenvolvimento, manutenção e produção de *software* ou por departamentos de tecnologia da informação de quaisquer organizações (onde TI não é a atividade fim da empresa) com o objetivo de elevar a maturidade de capacidade da atividade relacionada ao software [CHRISSIS *et al*., 2003]. Podemos considerar como implícito nessas recomendações que os atores dessas práticas são os profissionais de tecnologia da informação envolvidos em projetos relacionados à criação de *software*.

Este trabalho concentra-se na área de processos de Gerência de Requisitos, onde suas práticas e 'subpráticas' específicas estão aqui analisadas e formalizadas com o objetivo de identificar possíveis oportunidades de melhoria, eliminação de ambigüidade, identificação e esclarecimento de omissões. Um outro resultado a ser obtido dessa análise é ressaltar os pontos fortes das recomendações e sua contribuição positiva na área de engenharia de *software* e de excelência da atividade de informática para as organizações envolvidas com desenvolvimento de *software*.

Como apoio à formalização foram utilizados elementos e artefatos definidos pela UML (*Unified Modeling Language*) [BOOCH *et al*., 1994], devido à sua relevância e destaque tanto no meio acadêmico, como nos ambientes empresariais. Destacamos o Diagrama de Atividades como a representação adotada para a modelagem de processos, por se constituir num elemento de modelagem simples e eficaz para a descrição de fluxos de trabalho na forma de seqüências de atividades [CRUZ, 2004]. Neste modelo podemos destacar a sua capacidade de representar atividades paralelas e a atribuição de papéis, pela identificação de raias de responsabilidades. Além disso, podem ser representados insumos consumidos, recursos utilizados e produtos gerados, na forma de objetos relacionados às atividades [BOOCH *et al*., 1994].

5.1. Descrição do Método Adotado no Trabalho

Neste trabalho foi adotada uma abordagem de análise gradual das práticas e 'subpráticas' da área de processo de Gerência de Requisitos do CMMI [CHRISSIS *et al*., 2003]. Após o estudo do assunto na literatura referenciada na bibliografia e nos livros do CMM e 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 elaborar 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. Segundo MORGAN [2002], a ênfase na clareza das regras é crucial e o seu real poder vem do fato de que:

> "The combined effect of relatively large numbers of simple statements, so that together, the whole has an impact that's greater than the sum of the individual parts." [MORGAN, 2002]

A partir dos modelos de sentença escolhidos, foram identificados os termos presentes em cada 'subprática'. Estes termos foram relacionados em uma tabela, onde, para cada termo 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*., 1994], do CMMI [CHRISSIS *et al*., 2003] e nos 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. Por fim, a relação de termos foi classificada alfabeticamente e cada termo recebeu uma codificação que permitiu sua identificação única.

Com todas as 'subpráticas' traduzidas para sentenças do Português Estruturado, os conjuntos de 'subpráticas' que formavam cada prática original foram estruturados numa seqüência de atividades, ou seja, em processos, na forma de diagramas de atividade da UML [BOOCH *et al*., 1999]. Estes diagramas têm a capacidade de tornar mais objetivos alguns aspectos das regras/atividades, além de se constituir num rico recurso gráfico permitindo visualizar globalmente a seqüência de atividades, as responsabilidades, as tomadas de decisão, pontos de controle e os objetos envolvidos num processo.

Para a elaboração dos processos, inicialmente foi necessário efetuar a identificação de atores ou mesmo áreas de responsabilidade envolvidas nas atividades. Estes formaram as raias de responsabilidade nos quais as sentenças foram distribuídas, levando-se em conta a seqüência adequada em cada caso. Para cada atividade do processo foram sendo registradas, na forma de comentários, as sentenças de Português Estruturado correspondentes. Durante todo o processo de formalização, foram realizadas revisões das sentenças em função das atividades do processo e vice-versa, ou seja, por meio de refinamentos sucessivos tanto as sentenças e seus termos, como também o próprio diagrama, foram analisados contínua e profundamente.

5.2. A Área de Processos de Gerência de Requisitos - REQM

O objetivo da área de processo de Gerência de Requisitos é assim descrito pelo CMMI:

> "The purpose of Requirements Management (REQM) is to manage the requirements of the project's products and product components and to identify inconsistencies between those requirements and the project's plans and work products". [CHRISSIS *et al*., 2003]

Apenas uma meta específica é estabelecida para essa área de processo: **"Gerenciar Requisitos"**, que deve ser atendida por cinco práticas específicas. Esta área de processo está classificada no nível 2 de maturidade da capacidade de *software*. Algumas práticas de outras áreas de processo estão também relacionadas com esse objetivo ou meta, e algumas estão classificadas no mesmo nível de maturidade. Além disso, existem outras áreas de processo que também tratam de requisitos de projetos, mas que estão classificadas em níveis mais altos de maturidade.

Na figura 09 estão apresentadas as cinco práticas específicas **SP** - *Specific Practices* de Gerência de Requisitos e as relações entre elas e as práticas genéricas **GP** – *Generic Practices*, como também a relação com algumas outras áreas de processos de nível de maturidade 2:

- **PP** - *Project Planning* – Planejamento de Projeto,
- **CM** – *Configuration Management* – Gerência de Configuração,
- **PMC** – *Project Monitoring and Control* – Acompanhamento e Controle de Projeto.

Além disso, estão indicadas as relações com algumas áreas de processo de nível de maturidade 3:

- **RSKM** – Risk Management – Gerência de Risco,
- **TS** – Technical Solution – Solução Técnica,
- **RD** – Requirements Development – Desenvolvimento de Requisitos.

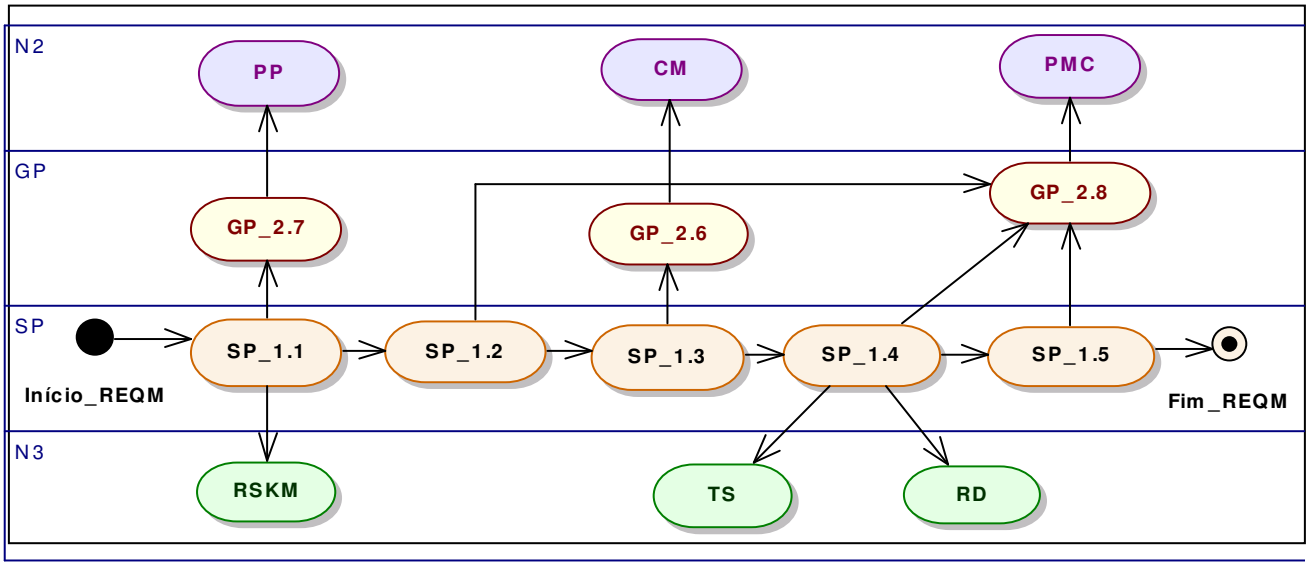

Figura 09 - Relação entre as Práticas Específicas (**SP**) e Genéricas (**GP**) de Gerência de Requisitos (REQM), algumas Áreas de Processo de Nível 2 (**N2**) e de Nível de 3 (**N3**)

As práticas específicas da área de processo de Gerência de Requisitos são apresentadas na Tabela 04, sendo que as práticas estudadas neste trabalho estão assinaladas em negrito com fundo cinza.

Tabela 04 – Práticas Específicas da Área de Processo de Gerência de Requisitos

Prática Específica	Descrição
SP 1.1	**Desenvolver um acordo informal com os provedores de requisitos sobre o significado dos requisitos.**
SP 1.2	**Obter dos participantes do projeto o compromisso com os requisitos do projeto.**
SP 1.3	**Gerenciar as mudanças de requisitos ao longo do desenvolvimento do projeto.**
SP 1.4	**Manter um rastreamento bidirecional entre os requisitos e os planos e produtos de trabalho do projeto.**
SP 1.5	Identificar as inconsistências entre o trabalho do projeto e os requisitos.

As práticas genéricas dizem respeito às características comuns (*common features*) que permeiam todas as áreas de processo. As práticas genéricas de Gerência de Requisitos que estão mais diretamente ligadas às práticas específicas e que, portanto, têm uma relação direta com o processo são aquelas de "direcionamento à implementação" (*directing implementation*), assinaladas em negrito com fundo cinza. Estão apresentadas na Tabela 05.

Tabela 05 – Algumas Práticas Genéricas da Área de Processo de Gerência de Requisitos

Práticas Genéricas	**Descrição**	**Características Comuns**
GP 2.1	Estabelecer uma política organizacional.	Compromisso de Executar
GP 2.2	Planejar o processo.	Habilidade para Executar
GP 2.3	Prover recursos.	Habilidade para Executar
GP 2.4	Alocar Responsabilidades.	Habilidade para Executar
GP 2.5	Treinar pessoas.	Habilidade para Executar
GP 2.6	**Gerenciar configurações.**	**Direcionar à Implementação**
GP 2.7	**Identificar e envolver os *stakeholders* relevantes.**	**Direcionar à Implementação**
GP 2.8	**Monitorar e controlar o processo.**	**Direcionar à Implementação**
GP 2.9	Avaliar a aderência objetivamente.	Verificar a Implementação
GP 2.10	Rever o status com a gerência de alto nível.	Verificar a Implementação

Ressaltamos, porém, que o escopo deste trabalho atem-se à análise das quatro primeiras das cinco práticas da área REQM – *Requirements Management*. No Anexo 1 está apresentada a transcrição parcial da parte II do livro "*CCMI – Guidelines for Process Integration and Product Improvement*" [CHRISSIS *et al*., 2003], referente às metas, práticas e 'subpráticas' específicas da área de processo de Gerência de Requisitos. No Anexo 2 estão transcritas algumas das práticas genéricas desta mesma área de processo.

A seguir apresentamos na figura 10 uma representação gráfica dos principais insumos trocados entre os clientes e/ou usuários e a equipe de projeto, no que diz respeito à Gerência de Requisitos.

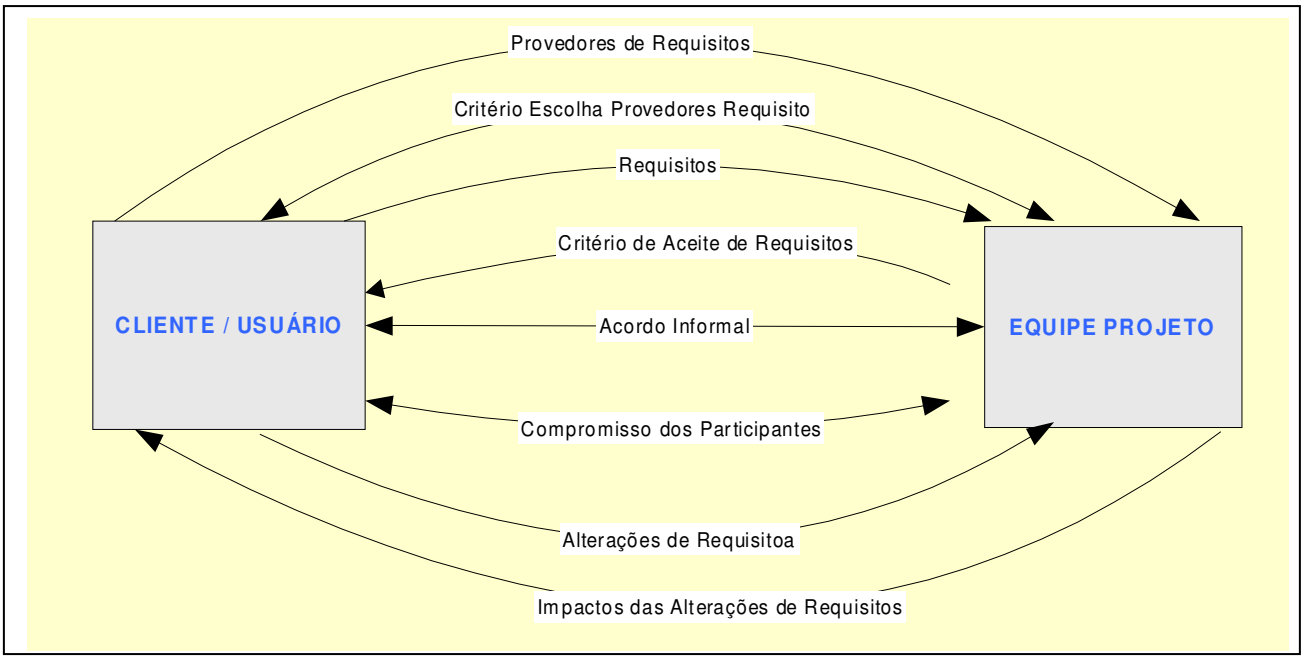

Figura 10 – Diagrama de Contexto da Área de Processo de Gerência de Requisitos

5.3. A Prática Específica REQM SP 1.1

A primeira prática específica de Gerência de Requisitos apresentada pelo CMMI é:

Prática	Descrição
REQM 1.1	**Desenvolver um acordo informal com os provedores de requisitos sobre o significado dos requisitos.**

O processo sugerido para a implementação desta prática específica é apresentado no modelo elaborado segundo o diagrama de atividades da UML, na figura 11:

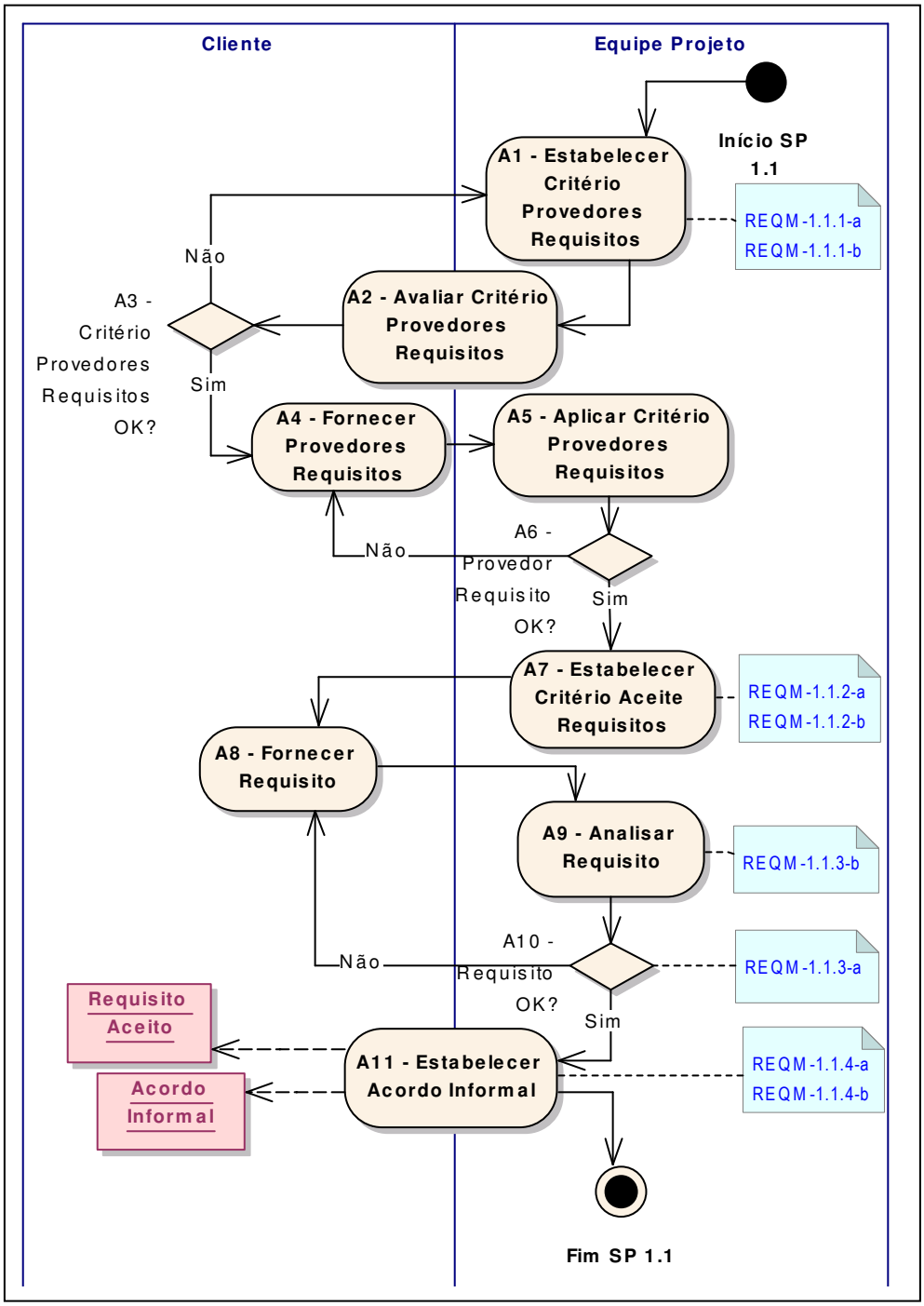

Figura 11 – Modelo de Processo da Prática Específica **REQM SP 1.1**

Para verificar o mapeamento das regras traduzidas em Português Estruturado às atividades do modelo de processo acima vide o Anexo 4.

Na elaboração do diagrama de atividades, o primeiro desafio encontrado foi a identificação dos atores, ou áreas de responsabilidades do processo, que estão mapeadas no diagrama na forma de raias de responsabilidade. As 'subpráticas' que compõem a primeira prática de REQM dizem respeito aos envolvidos na fase de provisão de requisitos. A raia de responsabilidade "Cliente" deve ser compreendida como os representantes do cliente/usuário solicitante do *software* que tiverem sido designados para esta tarefa. No caso da raia "Equipe de Projeto", são considerados aqueles membros da organização 'desenvolvedora' de *software* que receberão e questionarão os requisitos providos, ou seja, devem incluir profissionais de áreas de desenvolvimento de *software*, bem como de áreas de *hardware*, de planejamento de capacidade, avaliação de desempenho de *software* entre outros.

Em seguida, no mapeamento das atividades e seu encadeamento, deparamo-nos com situações onde a responsabilidade pela atividade é conjunta das duas áreas apresentadas. Isso pode ser identificado nas atividades "A2 – Avaliar Critério Provedores Requisitos" e "A11 – Estabelecer Acordo Informal". O significado dessas atividades implica em negociações conjuntas, e por isso as atividades estão apresentadas dividindo-se na fronteira entre as raias envolvidas.

Também foram incluídos pontos de tomada de decisão que não estão explicitamente apresentadas na forma de sentenças, mas que se mostraram indispensáveis no encadeamento lógico das atividades, como por exemplo "A3 – Critério Provedores Requisitos OK?", que está apresentado após a atividade "A2 – Avaliar Critério Provedores Requisitos". A avaliação mencionada pode levar a pelo menos duas alternativas: no caso de avaliação positiva, o processo pode avançar para a próxima atividade - "A4 – Fornecer Provedores Requisitos". No caso de avaliação negativa, o processo deve regredir para a atividade anterior – "A1 – Estabelecer Critério Provedores Requisitos".

Na última atividade – "A11 – Estabelecer Acordo Informal" é indicada a geração de dois objetos: "Requisito Aceito" e "Acordo Informal". Estes objetos representam o registro de informações para utilização posterior nos processos que se seguem, e estão entre os insumos do processo apresentados na figura 10.

Prosseguindo na avaliação da prática REQM SP 1.1, segundo a sugestão do CMMI, o desdobramento da prática requer a verificação de suas quatro 'subpráticas', analisadas e formalizadas a seguir.

5.3.1. A 'Subprática' Específica REQM SP 1.1.1

Esta primeira 'subprática' da prática REQM SP 1.1 recomenda que seja estabelecido, no início do projeto, qual o critério a ser adotado para identificar os provedores de requisitos adequados. Os provedores de requisitos são os responsáveis pela definição e detalhamento dos requisitos e, por conseqüência, responsáveis pela delimitação do escopo do *software* a ser desenvolvido.

'Subprática'	**Descrição**
REQM 1.1.1	Estabelecer um critério para a distinção dos provedores de requisitos adequados ao projeto.

Não identificamos nesta 'subprática' a definição dos responsáveis pela elaboração do critério. Por outro lado, as recomendações de práticas do CMMI são dirigidas aos técnicos e engenheiros envolvidos na construção e manutenção de *software*. Dessa forma, é razoável supor que o critério de distinção dos provedores de requisitos deva ser elaborado inicialmente pelos membros da organização 'desenvolvedora' de *software*. Adicionalmente, consideramos também razoável que os clientes ou usuários solicitantes do *software* sejam envolvidos na decisão sobre o critério e que forneçam a sua concordância formal sobre o critério a ser adotado. Esse envolvimento é inferido em nossa análise, pois, como pode ser verificado a seguir, não existe uma recomendação explícita do CMMI em relação à concordância do cliente e/ou usuário sobre esse critério.

Dessa forma, em nosso modelo de processo podemos observar que a atividade **"A1 - Estabelecer Critério Provedores Requisitos"** está apresentada na raia de responsabilidade da equipe de projeto. Em seguida, está apresentada uma atividade de **"A2 - Avaliar Critério Provedores Requisitos"** na região de interseção de ambas as raias de responsabilidades, sugerindo uma responsabilidade conjunta. A decisão sobre a adequação desse critério **"A3 – Critério Provedores Requisitos OK?"** está sob a responsabilidade do usuário, e em caso de não ser considerado aceito, a equipe de projeto deve voltar a estabelecer o critério, sugerindo um ciclo de negociações. Quando o critério de distinção dos provedores de requisitos for considerado aceito, o usuário deve informar quais são os participantes da provisão de requisitos: **"A4 - Fornecer Provedores Requisitos"** e a equipe de projeto aplica então o critério estabelecido: **"A5 – Aplicar Critério Provedores Requisitos"**. Em seguida, **"A6 - Provedores Requisitos OK?",** implementa a decisão de aceitação ou rejeição dos participantes fornecidos pelo cliente ou usuário para a provisão de requisitos.

Nessa análise e formalização identificamos a necessidade de definir os seguintes termos, que estão apresentados no Anexo 3 – Glossário de Termos:

Identificação	**Termo**	**Definição**
REQM-T09	critério de escolha de participantes da provisão de requisitos	é uma regra que define os participantes da provisão de requisitos.
REQM-T12	gerente de projeto	é: (1) o papel com responsabilidade total do negócio de um projeto inteiro. É o indivíduo que dirige, controla, administra e regula um projeto de construção de um sistema de *hardware*/*software*. O gerente de projeto é o indivíduo responsável, em última análise, perante o cliente. (2) a pessoa responsável pelo planejamento, direção, controle, estruturação e motivação do projeto. O gerente do projeto é responsável pela satisfação do cliente. (fonte: CMMI)

Identificação	Termo	Definição
REQM-T23	projeto de *software*	é o esforço necessário que é focalizado na análise, especificação, design, desenvolvimento, teste e/ou manutenção de componentes de *software* e documentação associada de um sistema. Um projeto de *software* pode ser parte de um projeto de construção de um sistema de *hardware/software*. (fonte: CMM)

Com esses termos definidos, foram então identificadas duas regras de negócio que traduzem a 'subprática' REQM 1.1.1 e que se enquadram nas estruturas de sentença de "Fato IX" e "Restrição I":

Código	Sentença	Estrutura
REQM-1.1.1-a	Um critério de escolha de participantes da provisão de requisitos (REQM-T09) **está relacionado a** um projeto de *software* (REQM-T23) **por** *'possui'* **com grau** 1**:**1.	Fato IX
REQM-1.1.1-b	Um critério de escolha de participantes da provisão de requisitos (REQM-T09) **deve obrigatoriamente** *ser estabelecido* pelo gerente de projeto (REQM-T12).	Restrição I

As duas regras traduzidas acima estão indicadas no diagrama como comentário ligado à atividade "A1 – Estabelecer Critério Provedores Requisitos". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.3.2. A 'Subprática' Específica REQM SP 1.1.2

Prosseguindo em nossa formalização, analisando a segunda 'subprática' da prática REQM SP 1.1, interpretamos que o critério de aceite dos requisitos deve ser estabelecido unicamente pela equipe do projeto, tendo em vista que, em nossa interpretação, o critério de

aceite dos requisitos deve considerar inclusive aspectos técnicos do requisito para sua inclusão no escopo do projeto.

'Subprática'	**Descrição**
REQM 1.1.2	Estabelecer um critério de aceite objetivo para os requisitos.

Dessa forma, a equipe de projeto deve definir o critério de aceite dos requisitos na forma da atividade **"A7 - Estabelecer Critério Aceite Requisitos"**. Em seguida, apresentamos a atividade de **"A8 - Fornecer Requisito"** na raia de responsabilidade do cliente.

Em nossa interpretação, o critério de aceite de requisitos é o critério a ser aplicado a cada requisito para que seja considerado como válido para o projeto e que, portanto, estará incluído no escopo do mesmo. Com esse critério de aceite pode se obter uma garantia de que o requisito que foi aceito e compreendido pela equipe do projeto a um ponto tal que decisões técnicas que afetarão o projeto ao longo de todo o seu processo de desenvolvimento possam ser tomadas. Desta forma, podemos considerar que um critério de aceite válido pode ser a verificação de atributos que o requisito deve possuir para ser considerado aceito no projeto. O exemplo indicado pelo CMMI sugere os seguintes atributos: clareza, completeza, consistência do requisito com os demais, possuir uma identificação única, ser apropriado para a implementação, estar apto a ser verificado ou testado, e ser passível de rastreamento.

Nesse ponto um novo termo (Vide Anexo 3 - Glossário de Termos) foi identificado e definido:

Identificação	**Termo**	**Definição**
REQM-T08	critério de aceite de requisitos	é uma regra que valida um requisito, de forma a ser aceito pelo usuário, cliente ou outra entidade autorizada. (fonte: CMM)

Duas novas regras de negócio traduzem para o Português Estruturado a 'subprática' REQM 1.1.2, na forma das estruturas de sentença de "Fato IX" e "Restrição I":

Código	Sentença	Estrutura
REQM-1.1.2-a	Um critério de aceite de requisitos (REQM-T08) **está relacionado a** um projeto de *software* (REQM-T23) **por** *'possui'* **com grau** 1**:**1.	Fato IX
REQM-1.1.2-b	Um critério de aceite de requisitos (REQM-T08) **deve obrigatoriamente** *ser estabelecido* pelo gerente de projeto (REQM-T12).	Restrição I

As duas regras traduzidas acima estão indicadas no diagrama como comentário ligado à atividade "A7 – Estabelecer Critério Aceite Requisitos". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.3.3. A 'Subprática' Específica REQM SP 1.1.3

A terceira 'subprática' da prática REQM SP 1.1 sugere explicitamente a aplicação do critério de aceite dos requisitos aos requisitos fornecidos pelo cliente:

'Subprática'	Descrição
REQM 1.1.3	Analisar os requisitos para garantir que o critério estabelecido foi atendido.

Cada requisito fornecido deve ser analisado segundo o critério de aceite estabelecido, apresentado na atividade "**A9 - Analisar Requisito"** do nosso modelo. A decisão subseqüente **"A10 - Requisito OK?"**, define se o requisito deve ser aceito ou rejeitado.

Na análise de um requisito quanto aos atributos definidos no critério de aceite, é necessário que todos os atributos sejam atendidos para que o critério seja considerado como

atendido (“A10 - Requisito OK? = Sim”). Quando um requisito não cumpre isso, ele deve ser re-interpretado, analisado e redefinido de forma a cumprir o critério de aceite estabelecido, sob pena de ser considerado rejeitado e ficar fora do escopo do projeto.

Mais um novo termo (Vide Anexo 3 - Glossário de Termos) foi identificado e definido:

Identificação	**Termo**	**Definição**
REQM-T29	requisito de projeto de *software*	é uma condição ou capacidade que deve ser atendida pelo *software*, necessária a um cliente ou usuário para resolver um problema ou alcançar um objetivo. (fonte: IEEE-STD-610)

Duas novas regras de negócio traduzem para o Português Estruturado a ‘subprática’ REQM 1.1.3, na forma da estrutura de sentença de “Restrição I”:

Código	**Sentença**	**Estrutura**
REQM-1.1.3-a	O critério de aceite de requisitos (REQM-T08) **deve obrigatoriamente** *ser cumprido por* um requisito do projeto de *software* (REQM-T29).	Restrição I
REQM-1.1.3-b	Um requisito de projeto de *software* (REQM-T29) **deve obrigatoriamente** *ser analisado* pelo gerente de projeto (REQM-T12).	Restrição I

As duas regras traduzidas acima estão indicadas no diagrama na forma de comentário (em azul), sendo **REQM-1.1.3-a** ligado à atividade “A10 – Requisito OK?” e **REQM-1.1.3-b** ligado à atividade “A9 – Analisar Requisito”. Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.3.4. A ‘Subprática’ Específica REQM SP 1.1.4

A quarta ‘subprática’ da prática REQM SP 1.1 diz respeito à compreensão dos requisitos e aos compromissos dos participantes.

‘Subprática’	**Descrição**
REQM 1.1.4	Estabelecer um acordo informal sobre os requisitos com os provedores dos requisitos, de forma que os participantes do projeto possam se comprometer com eles.

Para os requisitos aceitos, a concordância conjunta da equipe do projeto e do cliente pressupõe o estabelecimento de um acordo informal sobre compromissos em relação ao mesmo.

Neste ponto devemos analisar cuidadosamente o texto original em inglês para evitar uma interpretação indevida, já que pode ser encontrada ambigüidade na formulação original desta ‘subprática’. No texto original do CMMI encontramos a seguinte definição: “*Reach an **understanding** of the requirements with the requirements provider so that the project participants can **commit** to them*” [CHRISSIS *et al*., 2003]. As palavras em negrito foram pesquisadas para identificar seu significado mais correto:

> **Understanding** n [C]. An informal agreement between people.
>
> **Commit** (obj) PROMISE. v. To promise or give (your loyalty or money) to a particular principle, person, or plan of action.
>
> Cambridge International Dictionary of English

Segundo estas definições devemos, então, traduzir da língua inglesa para a portuguesa a palavra “*understanding*” como “acordo informal” e a palavra “*commit*” como “comprometer-se”. Dessa forma, a ‘subprática’ foi traduzida como “Estabelecer um acordo

informal sobre os requisitos com os provedores dos requisitos, de forma que os participantes do projeto possam se comprometer com eles". Analisando a escolha da palavra "*understanding*", entendemos que ela foi propositalmente usada para descaracterizar a formalidade do acordo, já que se o desejado fosse o oposto, o mais apropriado seria utilizar "*agreement* - acordo" ou "*formal agreement* – acordo formal" na elaboração da recomendação no CMMI. Concluímos então que esta 'subprática' recomenda que com esse acordo informal referente ao escopo inicial do projeto, os 'desenvolvedores' já podem *("... can commit ...")* estabelecer a contrapartida de seu esforço e custo e, por conseqüência, o seu compromisso com o projeto. No nosso modelo esta 'subprática' está apresentada pela última atividade **"A11 – Estabelecer Acordo Informal"**. A criação do objeto "**REQUISITOS ACEITOS**" reforça a idéia de que os requisitos aceitos são documentados e de que serão referenciados posteriormente. Da mesma forma pode-se considerar a criação de um documento registrando o acordo informal estabelecido, na forma do objeto "**ACORDO INFORMAL**".

Encontramos aqui uma ligação direta entre esta prática específica da área de processos **REQM** - Gerência de Requisitos com a área de processo de nível 2 de maturidade – **PP** - Planejamento de Projetos. O "acordo informal" sobre os requisitos aceitos no projeto é o subsídio fornecido pela Gerência de Requisitos para o estabelecimento de estimativas iniciais de esforço, tamanho e conseqüentemente de custo, prazo e cronograma. Estas estimativas fazem parte das recomendações da área de processo PP - Planejamento de Projeto, portanto o mapeamento do objeto "acordo informal", que faz parte de um documento desta área de processo (PP), não está foi incluído neste trabalho. A interação entre essas duas áreas de processo está apresentada na figura 9 no início deste capítulo.

Nesta 'subprática' mais uma vez é feita uma referência aos "participantes do projeto" sem o esclarecimento sobre quem são eles. Neste ponto do projeto, em que ainda estão ocorrendo as fases iniciais do trabalho, o envolvimento dos "participantes do projeto" só pode ser entendido como a participação dos membros da organização 'desenvolvedora' de *software* que estão envolvidos na gerência ou liderança técnica de todos os aspectos do

projeto (desenvolvimento de sistemas, suporte técnico, analistas de negócio, áreas de avaliação de capacidade e desempenho produtivo etc). Esses líderes e gerentes podem enriquecer a discussão e elaboração do entendimento dos requisitos com sua participação nesta fase inicial de definição e devem ter autoridade e responsabilidade suficientes para avaliar a viabilidade técnica e operacional e comprometer o esforço de suas equipes no projeto. Consideramos que esses participantes devem ser formalmente estabelecidos com seus respectivos papéis e responsabilidades identificadas.

Convém acrescentar que, apesar de não estarem claramente estabelecidos nestas 'subpráticas', o envolvimento e o compromisso dos participantes são considerados em outra recomendação – a prática genérica **REQM GP 2.7** de "identificar e envolver os *stakeholders* relevantes do projeto". Com isso podemos constatar o inter-relacionamento das práticas recomendadas (vide figura 9). Além disso, como já mencionado, esta prática genérica está fortemente ligada a outra área de processo de nível 2 de maturidade – **PP -** Planejamento de Projetos, onde os papéis e responsabilidades são formalmente relacionados.

Os seguintes novos termos (Vide Anexo 3 - Glossário de Termos) foram identificados e definidos:

Identificação	**Termo**	**Definição**
REQM-T01	acordo informal sobre os requisitos de projeto de *software*	é (1) o resultado obtido da análise conjunta e objetiva do requisito de projeto de *software*. Esse acordo garante que o requisito pode ser incluído num conjunto compatível e factível de requisitos de projeto de *software*. (2) acordo uniforme e compartilhado por todos os participantes do projeto de *software*.
REQM-T07	compromisso dos participantes do projeto	é um acordo livremente assumido, visível e esperado dos participantes envolvidos no projeto de *software*.

Duas novas regras de negócio traduzem para o Português Estruturado a 'subprática' REQM 1.1.4, na forma da estrutura de "Derivação I":

Código	Sentença	Estrutura
REQM-1.1.4-a	**Se** for estabelecido um acordo informal sobre os requisitos do projeto de *software* (REQM-T01) **então** o requisito de projeto de *software* (REQM-T29) **é considerado como** aceito.	Derivação I
REQM-1.1.4-b	**Se** um requisito de projeto de *software* (REQM-T29) **é considerado** aceito, **então** o compromisso dos participantes do projeto (REQM-T07) **é considerado como** estabelecido.	Derivação I

As duas regras traduzidas acima estão indicadas no diagrama como comentário ligado à atividade "A11 – Estabelecer Acordo Informal". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.4. A Prática Específica REQM SP 1.2

A segunda prática específica de Gerência de Requisitos apresentada pelo CMMI é:

Prática	Descrição
REQM 1.2	**Obter dos participantes do projeto o compromisso com os requisitos do projeto.**

O processo sugerido para a implementação desta prática específica é apresentado no modelo elaborado segundo o diagrama de atividades da UML, na figura 12:

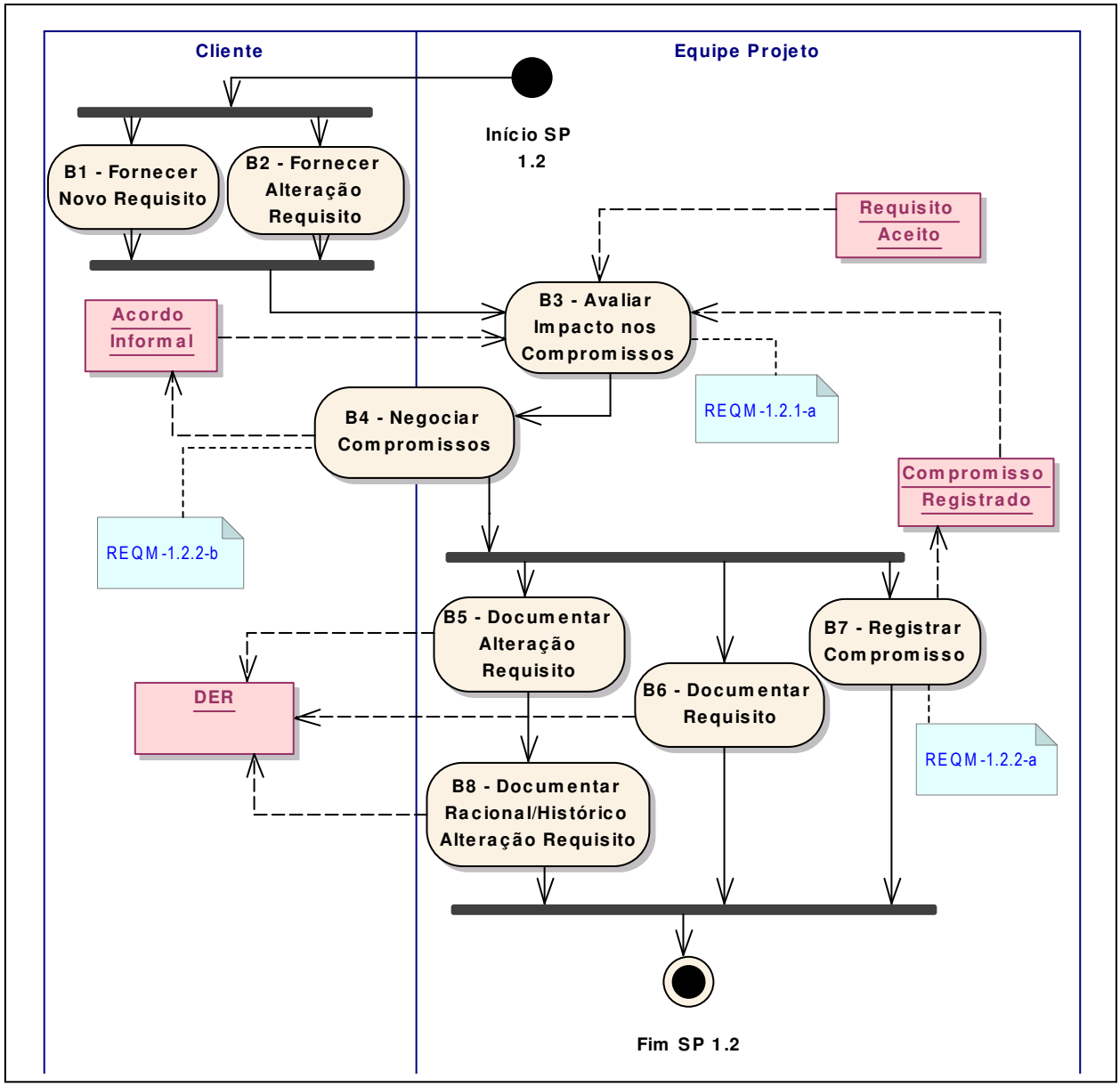

Figura 12 – Modelo de Processo da Prática Específica **REQM SP 1.2**

Para verificar o mapeamento das regras traduzidas em Português Estruturado às atividades do modelo de processo acima vide o Anexo 4.

Neste processo o desafio inicial colocou-se com as atividades “B1 – Fornecer Novo Requisito” e “B2 – Fornecer Alteração Requisito”. É necessário ressaltar que este diagrama representa um processo recorrente ao longo de algumas fases do desenvolvimento

de um *software*. Desde o início de um projeto, ao longo do seu ciclo de vida, novos requisitos ou alterações de requisitos podem ser apresentados à equipe de desenvolvimento para serem considerados no projeto. Isso ocorre para que todas as informações, necessidades e alternativas sejam levadas em conta na elaboração do *software*. O gerenciamento dessas solicitações, a negociação de seus impactos no andamento do trabalho e o controle do que deve ser incorporado ou não ao projeto são os meios sugeridos para minimizar os problemas decorrentes. A forma de apresentar estas alternativas – novos requisitos ou alterações de requisitos – foi apresentá-los entre barras de sincronismo, onde no surgimento de apenas um deles, ou até mesmo de ambos, o processo pode se iniciar.

Novamente registramos a apresentação de uma atividade na fronteira das raias de responsabilidade, já que se trata de uma atividade de negociação e que pressupõe que seja realizada entre pelo menos duas partes – "B4 – Negociar Compromissos".

As atividades "B5 – Documentar Alteração Requisito", "B6 – Documentar Requisito", "B7 – Registrar Compromisso" e "B8 – Documentar Racional/Histórico Alteração Requisito" também estão apresentadas após uma barra de sincronismo, indicativo de que somente poderão se iniciar após a atividade de negociação ("B4"). Além disso, todas convergem para outra barra de sincronismo, pois somente após a sua realização é que o processo pode se finalizar.

Prosseguindo na análise desta prática, entendemos que ela recomenda que seja estabelecido o compromisso de implementação dos requisitos em *software* e envolve o compromisso dos participantes do projeto que pertencem à organização 'desenvolvedora'. Esse compromisso deve garantir que os requisitos acordados reflitam-se nos planos, nas atividades do processo e nos produtos de trabalho. Da mesma forma, as alterações dos requisitos também devem ser consideradas e refletidas nos mesmos planos, atividades e produtos de trabalho.

O CMMI desdobra esta prática específica nas duas 'subpráticas' analisadas a seguir.

5.4.1. A ‘Subprática’ Específica REQM SP 1.2.1

A primeira ‘subprática’ referente à prática REQM SP 1.2 diz respeito à análise do impacto dos requisitos nos compromissos dos participantes já existentes.

‘Subprática’	**Descrição**
REQM 1.2.1	Verificar o impacto dos requisitos nos compromissos existentes.

Observamos que esta ‘subprática’ se inicia mencionando os “compromissos existentes”, nunca antes mencionados. Na prática REQM SP 1.1 foi obtido o acordo informal sobre os requisitos de tal forma que possibilitem o compromisso. Como sugestão de aprimoramento desta prática fica aqui registrada a sugestão de incluir uma nova ‘subprática’ inicial que recomende a formalização dos compromissos não só da organização ‘desenvolvedora’, mas que também seja feito um esforço para se obter o compromisso do cliente e/ou usuário. O compromisso deve ser feito para cada requisito e na análise dos requisitos subseqüentes também deve ser avaliado o impacto do novo requisito sobre os compromissos existentes.

Analogamente, antes de ser aceita no projeto uma solicitação de alteração de requisito, deve ser feita uma análise de impacto da mudança sobre o projeto. As solicitações de alteração de requisitos devem ser formalizadas e, após a análise, os compromissos devem ser revistos e renegociados. Somente após a aprovação formal do cliente é que a solicitação de mudança de requisito pode passar a ser considerada e refletida nos planos, atividades e produtos de trabalho do projeto.

Podemos também observar que como insumos de entrada nesse processo estão presentes os dois objetos criados no processo referente à prática específica anterior: “**REQUISITOS ACEITOS**” e “**ACORDO INFORMAL**”.

Em nosso modelo de processo apresentamos as atividades **“B1 – Fornecer Novo Requisito”** e **“B2 – Fornecer Alteração Requisito”** como as atividades iniciais de cada ciclo de fornecimento de novos requisitos e/ou alteração de requisitos. Essas atividades são

apresentadas em paralelo e, tanto podem ambas ocorrer, como apenas uma delas pode ocorrer em cada ciclo. Em seguida, os impactos decorrentes do novo requisito ou da alteração de requisito devem ser avaliados contra os compromissos existentes, indicados pela atividade **"B3 – Avaliar Impacto nos Compromissos"**.

Os seguintes novos termos (Vide Anexo 3 - Glossário de Termos) foram identificados e definidos:

Identificação	**Termo**	**Definição**
REQM-T14	impacto de um requisito de projeto de *software*	é o efeito que um requisito exerce nos compromissos assumidos, no andamento, nos produtos de trabalho envolvidos e nas tarefas do projeto. Pode significar re-trabalho e provocar re-planejamento do projeto.
REQM-T06	compromissos de projeto de *software* existentes	são os compromissos assumidos pelos participantes do projeto com os requisitos considerados aceitos, num dado momento do projeto de *software*.

Uma nova regra de negócio traduz para o Português Estruturado a 'subprática' REQM SP 1.2.1, na forma da estrutura de sentença de "Restrição I":

Código	**Sentença**	**Estrutura**
REQM-1.2.1-a	O impacto de um requisito de projeto de *software* (REQM-T14) **deve obrigatoriamente** *ser avaliado* em relação aos compromissos de projeto de *software* existentes (REQM-T06).	Restrição I

A regra traduzida acima está indicada no diagrama na forma de comentário ligado à atividade "B3 – Avaliar Impacto nos Compromissos". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.4.2. A ‘Subprática’ Específica REQM SP 1.2.2

Prosseguindo, a segunda ‘subprática’ da prática REQM SP 1.2 estabelece:

‘Subprática’	**Descrição**
REQM 1.2.2	Negociar e registrar os compromissos.

Os compromissos, requisitos e alterações de requisitos devem ser negociados entre a equipe do projeto (‘desenvolvedores’) e o cliente, o que está apresentado como a atividade **“B4 – Negociar Compromissos”**. Além do registro dos compromissos, consideramos apropriado introduzir, neste ponto da formalização, a documentação, ou seja, o registro formal dos requisitos, das alterações dos requisitos, seu racional e histórico. Como se pode constatar adiante, isso será necessário para as práticas seguintes, além de também se alinhar com a prática genérica **REQM GP 2.6** de: “gerenciar as configurações”. Essa prática genérica está fortemente ligada à outra área de processo de nível 2 de maturidade – **CM -** Gerência de Configuração, que prevê o registro, armazenamento, controle de versão e de acesso aos produtos de trabalho e componentes de *software* construídos no projeto. Portanto, em nosso modelo apresentamos as atividades **“B5 – Documentar Alteração Requisito”**, **“B6 – Documentar Requisito”**, **“B7 – Registrar Compromisso”** e **“B8 – Documentar Racional/Histórico Alteração Requisito”**.

Vale ressaltar aqui que a atividade “B6 – Documentar Requisito” introduz a criação de mais um objeto “**DER** – DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS”. Em nossa proposta julgamos necessária a existência de um documento de formalização de todas as informações sobre a provisão de requisitos de forma que fiquem registradas e formalmente aprovadas pelos envolvidos: o cliente e a equipe do projeto. A criação desse documento é apoiada pela recomendação do IEEE-STD-830 [IEEE, 1998], que recomenda a criação de um **SRS** – *Software Requirements Specification*, que estabelece que cuidados devem ser tomados sobre os limites do papel do SRS:

a. Deve definir corretamente todos os requisitos de *software*. Um requisito de *software* deve existir pela natureza da tarefa a ser solucionada ou por uma característica especial do projeto;

b. Não deve descrever nenhum detalhe de desenho (*design*) nem de implementação. Estes devem ser descritos posteriormente na fase de desenho.

c. Não devem impor restrições adicionais no *software*. Estas restrições devem ser descritas em outros documentos apropriados como, por exemplo, o plano de garantia de qualidade de *software*.

O IEEE-STD-830 recomenda ainda que uma boa especificação de requisito deve ser: correta, não ambígua, completa, consistente, classificada por importância e/ou estabilidade, verificável, modificável e 'rastreável' (capaz de ser rastreado). Além disso, a recomendação do IEEE-STD-830 estabelece que essa especificação deve ser elaborada em conjunto pela equipe do projeto com o cliente e/ou usuário, pois nenhuma das duas partes é considerada qualificada para esta elaboração [IEEE, 1998] isoladamente. No Anexo 7 pode ser encontrado um modelo do **DER** – Documento de Especificação de Requisitos proposto.

Além disso, existe a criação do objeto "**COMPROMISSOS EXISTENTES**", que tanto recebe neste processo o registro dos novos compromissos, como é usado como insumo no início deste mesmo processo para a verificação de novos requisitos em relação aos compromissos já existentes. Isso reforça a idéia de que todos esses processos são iterativos e repetem-se ao longo do projeto, sempre que for necessário incluir um novo requisito ou alterar um requisito já existente. Ou seja, a Gerência de Requisitos se inicia juntamente com o projeto, mas não se encerra aí, podendo ocorrer ao longo do projeto.

Um novo termo (Vide Anexo 3 - Glossário de Termos) foi identificado e definido:

Identificação	Termo	Definição
REQM-T19	participante do projeto de *software*	é um indivíduo que é afetado ou é responsável pelo resultado de um projeto de *software*. Um participante pode incluir membros da organização de desenvolvimento do projeto, fornecedores, clientes, usuários finais e outros.

Duas novas regras de negócio traduzem para o Português Estruturado a 'subprática' REQM 1.2.2, na forma da estrutura de sentença de "Restrição I":

Código	Sentença	Estrutura
REQM-1.2.2-a	Os compromissos do projeto de *software* existentes (REQM-T06) **devem obrigatoriamente** *ser registrados* pelo gerente de projeto (REQM-T12).	Restrição I
REQM-1.2.2-b	Os compromissos do projeto de *software* (REQM-T06) **devem obrigatoriamente** *ser negociados* pelos participantes do projeto de *software* (REQM-T19).	Restrição I

As regras traduzidas acima estão indicadas no modelo na forma de comentário, sendo REQM-1.2.2-a ligada à atividade "B7 – Registrar Compromisso" e REQM-1.2.2-b ligada à atividade "B4 – Negociar Compromissos". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.5. A Prática Específica REQM SP 1.3

A terceira prática específica de Gerência de Requisitos apresentada pelo CMMI é:

Prática	Descrição
REQM 1.3	**Gerenciar as mudanças de requisitos ao longo do desenvolvimento do projeto.**

O processo sugerido para a implementação desta prática específica é apresentado no modelo elaborado segundo o diagrama de atividades da UML, na figura 13:

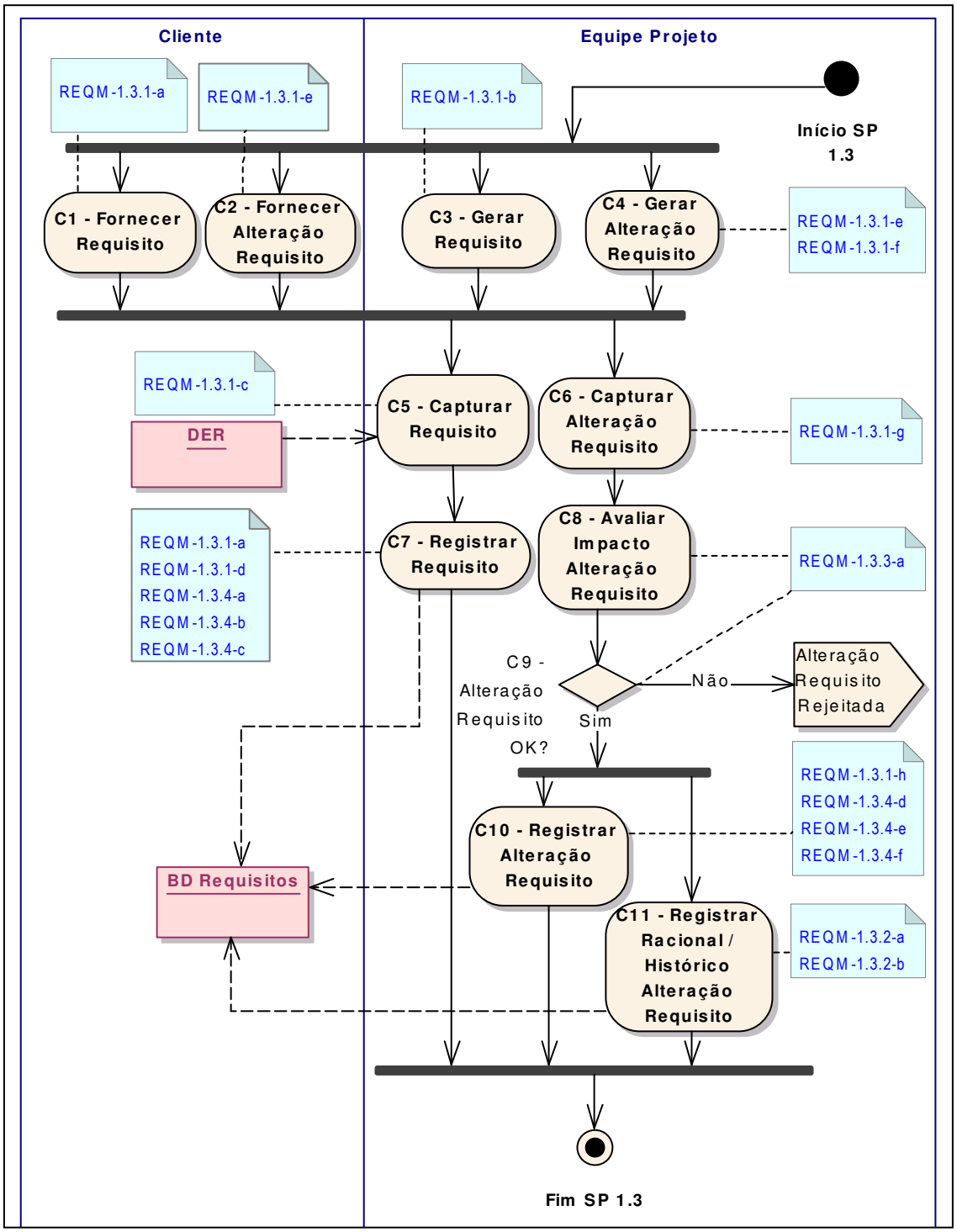

Figura 13 – Modelo de Processo da Prática Específica **REQM SP 1.3**

Para verificar o mapeamento das regras traduzidas em Português Estruturado às atividades do modelo de processo acima vide o Anexo 4.

O principal desafio encontrado na modelagem deste processo foi alinhar todas as atividades de geração de requisitos e alterações de requisitos fornecidos pelo cliente com as a geração dos mesmos insumos pela própria equipe do projeto. Isso pode ocorrer já que ao longo do desenvolvimento, o projeto pode receber requisitos e/ou alterações de requisitos como pode também gerar internamente. A geração interna desses insumos ocorre quando a equipe do projeto identifica derivações de requisitos compostos ou inconsistências nos requisitos acordados. A solução de modelagem foi explicitar o sincronismo destas atividades (estão apresentadas entre barras de sincronismo) que representam que quando qualquer uma das atividades ocorrer o processo se inicia.

O objetivo desta prática específica é a gerência de *baseline* de requisitos ao longo de todas as fases do projeto. Por vários motivos, os requisitos podem sofrer mudanças ao longo do tempo (adaptação às necessidades do mercado, dinâmica do negócio, mudanças legais etc). As alterações devem ser cuidadosamente tratadas e para isso justifica-se conhecer sempre a origem dos requisitos e o racional da sua alteração. Esta prática está fortemente ligada à prática genérica **REQM GP 2.6** de "gerenciar as configurações". Por sua vez, essa prática genérica está aderente à outra área de processo de nível 2 de maturidade – **CM** – Gerência de Configuração, onde são definidos os produtos de trabalho e componentes de *software* que devem ser colocados sob gerência de configuração, ou seja, controle de versão e de acesso. A prática específica **REQM SP 1.3** nos mostra a recomendação implícita do CMMI de que os requisitos e as alterações dos requisitos dos projetos estejam assim controlados, ou seja, devem ser registrados e armazenados, e devem sofrer controle de versões e de acesso para consulta e/ou atualização.

O CMMI desdobra esta prática específica nas quatro 'subpráticas' analisadas a seguir.

5.5.1. A ‘Subprática’ Específica REQM SP 1.3.1

Nesta primeira ‘subprática’ da prática REQM SP 1.3 podemos ressaltar dois aspectos importantes: 1º capturar **todos** os requisitos e alterações de requisitos; 2º que forem entregues ou gerados pelo projeto.

‘Subprática’	**Descrição**
REQM 1.3.1	Captar todos os requisitos e alterações de requisitos que forem entregues ou gerados no projeto.

O primeiro aspecto traz em si uma dificuldade: como garantir que todos os requisitos e alterações de requisitos sejam capturados? Isso, por si só, insere uma dificuldade de formalização, pois não existem meios para verificar, identificar ou garantir que esta recomendação seja cumprida. Dessa forma, a recomendação do CMMI só pode referir-se à preocupação e atenção constantes que a equipe de projeto, principalmente na figura do gerente de projeto, dedique para a obtenção de todos os requisitos possíveis que o cliente sugira, ainda que não explicitamente. O segundo aspecto nos mostra que os requisitos e alterações de requisitos tanto podem ser fornecidos à equipe do projeto pelo cliente, como também podem ser gerados internamente pelo projeto, identificado e fornecido pela própria equipe. Isso pode ser facilmente compreendido quando se considera que um requisito complexo informado pelo cliente pode se desdobrar em vários requisitos pela própria equipe de forma a facilitar sua compreensão e implementação em componentes de *software*.

Em nosso modelo de processo, as atividades **“C1 – Fornecer Requisito”** e **“C2 – Fornecer Alteração Requisito”** indicam, como no processo da prática específica REQM SP 1.2, que o cliente pode (e efetivamente o faz) fornecer requisitos e alterações de requisitos. De fato, ele é o principal provedor dos mesmos. Por esse motivo, estas atividades estão na raia de responsabilidade do cliente. Além destes, estão apresentadas as atividades **“C3 – Gerar Requisito”** e **“C4 – Gerar Alteração Requisito”** que estão na raia de responsabilidade da equipe de projeto, e que, portanto, referem-se aos requisitos e

alterações de requisitos identificados e gerados no próprio projeto e que devem ser igualmente considerados no projeto. Todas estas atividades na verdade podem ocorrer em paralelo e não são todas obrigatórias, ou seja, pode existir um projeto onde não existam alterações de requisitos, ou não haja requisitos gerados pelo próprio projeto, ainda que isso seja improvável.

A seguir são apresentadas em paralelo as atividades **"C5 – Capturar Requisito"** e **"C6 – Capturar Alteração Requisito"**, indicando a disposição da equipe em acolher e considerar todos os requisitos e alterações fornecidas. No caso da atividade **"C5 – Capturar Requisito"**, além de receber requisitos fornecidos pelo cliente e gerados pela equipe do projeto, identificamos que ela utiliza como insumo o objeto "**DER** – DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS", que foi criado no processo anterior.

Nesse ponto, identificamos a necessidade de definir um novo termo, apresentado no Anexo 3 – Glossário de termos:

Identificação	**Termo**	**Definição**
REQM-T02	alteração de requisito de projeto de *software*	é a alteração de um requisito de projeto de *software* previamente elaborado e acordado para o projeto de *software*. A alteração deve ser analisada para a avaliação de seu impacto no andamento do projeto de *software* e deve ser acordada com o cliente ou usuário antes de ser aceita no projeto de *software*.

Dessa forma, podemos traduzir essa 'subprática' em oito novas regras em Português Estruturado que se enquadram nas estruturas de sentença de "Restrição I" e "Fato IX":

Código	**Sentença**	**Estrutura**
REQM-1.3.1-a	Um requisito de projeto de *software* (REQM-T29) **está relacionado a** um projeto de *software* (REQM-T23) **por '***possui'* **com grau** 1..N**:**1.	Fato IX
REQM-1.3.1-b	Um projeto de *software* (REQM-T23) **está relacionado a** um requisito de projeto de *software* (REQM-T29) **por** *'gerou'* **com grau** 1**:**0..N.	Fato IX

Código	Sentença	Estrutura
REQM-1.3.1-c	O requisito de projeto de *software* (REQM-T29) **deve obrigatoriamente** *ser capturado* pelo gerente de projeto (REQM-T12).	Restrição I
REQM-1.3.1-d	O requisito de projeto de *software* (REQM-T29) **deve obrigatoriamente** *ser registrado* pelo gerente de projeto (REQM-T12).	Restrição I
REQM-1.3.1-e	Um projeto de *software* (REQM-T23) **está relacionado a** uma alteração de requisito de projeto de *software* (REQM-T02) **por** *'recebeu'* **com grau** 1**:**0..N.	Fato IX
REQM-1.3.1-f	Um projeto de *software* (REQM-T23) **está relacionado a** uma alteração de requisito de projeto de *software* (REQM-T02) **por** *'gerou'* **com grau** 1**:**0..N.	Fato IX
REQM-1.3.1-g	Uma alteração de requisito de projeto de *software* (REQM-T02) **deve obrigatoriamente** *ser capturada* por um projeto de *software* (REQM-T23).	Restrição I
REQM-1.3.1-h	A alteração de requisito de projeto de *software* (REQM-T02) **deve obrigatoriamente** *ser registrada* pelo gerente de projeto (REQM-T12).	Restrição I

As oito regras acima estão indicadas no diagrama de processos como comentários, sendo REQM-1.3.1-a ligado à atividade "C1 – Fornecer Requisito", REQM-1.3.1-b ligado à atividade "C3 – Gerar Requisito", REQM-1.3.1-c ligado à atividade "C5 – Capturar Requisito", REQM-1.3.1-e e REQM-1.3.1-f ligados à atividade "C4 – Gerar Alteração Requisito", REQM-1.3.1-g ligado à atividade "C6 – Capturar Alteração Requisito", REQM-1.3.1-d ligado à atividade "C7 – Registrar Requisito" e REQM-1.3.1-h ligado à atividade "C10 – Registrar Alteração Requisito". Estas duas últimas atividades serão apresentadas e detalhadas adiante. Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.5.2. A ‘Subprática’ Específica REQM SP 1.3.2

Prosseguindo em nossa formalização, a segunda ‘subprática’ da prática REQM SP 1.3 nos indica a necessidade de manter o registro do racional e do histórico das alterações de requisitos, além de manter o registro das alterações de requisitos propriamente ditas.

‘Subprática’	**Descrição**
REQM 1.3.2	Manter o histórico de alterações de requisitos e o racional das alterações de requisitos.

À medida que um requisito vai sofrendo alterações, os racionais, ou seja, as descrições de motivos, explicações, datas, responsáveis e eventos que justificam as alterações, devem ser registrados, compondo o histórico do requisito. É importante prover os mecanismos necessários de armazenamento, manutenção, acesso e controle desses registros para garantir a integridade e disponibilidade dessas informações aos canais devidamente autorizados para acessos e/ou atualização. Em nosso modelo estão indicadas as atividades **“C11 – Registrar Racional/Histórico Alteração Requisito”**.

Os seguintes novos termos (Vide Anexo 3 – Glossário de Termos) foram identificados e definidos:

Identificação	**Termo**	**Definição**
REQM-T24	racional da alteração de requisito de projeto de *software*	é a explicação ou justificativa formal e documentada sobre a necessidade ou causa da alteração do requisito de projeto de *software*.
REQM-T13	histórico da alteração de requisito de projeto de *software*	é o registro formal sobre as alterações de requisito de projeto de *software* que foram acordadas para os requisitos de projeto de *software*.

Duas novas regras de negócio traduzem para o Português Estruturado a ‘subprática’ REQM 1.3.2, na forma de estrutura de sentença “Restrição I”:

Código	Sentença	Estrutura
REQM-1.3.2-a	O racional das alterações de requisitos de projeto de *software* (REQM-T24) **deve obrigatoriamente** *ser registrado* pelo gerente de projeto (REQM-T12).	Restrição I
REQM-1.3.2-b	O histórico das alterações de requisitos de projeto de *software* (REQM-T13) **deve obrigatoriamente** *ser registrada* pelo gerente de projeto (REQM-T12).	Restrição I

As regras traduzidas acima estão indicadas no diagrama de processo como comentário ligado à atividade "C11- Registrar Racional/Histórico Alteração Requisito". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.5.3. A 'Subprática' Específica REQM SP 1.3.3

A terceira 'subprática' da prática REQM SP 1.3 recomenda que a avaliação de impacto das alterações de requisitos seja feita levando-se em conta o ponto de vista dos *stakeholders* relevantes.

'Subprática'	Descrição
REQM 1.3.3	Avaliar o impacto das alterações de requisitos pelo ponto de vista dos *stakeholders* relevantes.

Segundo o CMM [PAULK *et al*., 1994], os *stakeholders* relevantes podem ser considerados como os participantes do projeto que possuem o maior interesse no produto final, alvo do desenvolvimento do projeto. São elementos do cliente e/ou usuário e também da organização 'desenvolvedora' – equipe do projeto. Sua relevância no projeto pode ser avaliada pelo seu poder de decisão (aprovação/reprovação de recursos, prazos, decisões

técnicas) e pela sua responsabilidade (cumprimento de prazos e cronogramas, entregas, participação em testes, tomadas de decisão etc).

Todas as alterações de requisitos que forem solicitadas para o projeto devem ser avaliadas quanto ao custo/benefício para o produto final e o uso pelo cliente. Também deve ser avaliado o impacto das mesmas no andamento do projeto, assim como avaliar o 'replanejamento' e 'retrabalho' provocado. Um equilíbrio entre o custo e o benefício é desejável e a palavra final sobre a inclusão da solicitação de mudança do requisito no projeto deve ser obtida junto ao cliente/usuário, aliada sempre à renegociação dos compromissos. Constatamos, assim, uma referência nesta 'subprática', à prática específica **REQM 1.2.1** – "Verificar o impacto dos requisitos nos compromissos existentes".

Em nosso modelo de processo esta 'subprática' está representada pelas atividades **"C8 – Avaliar Impacto Alteração Requisito"** e **"C9 – Alteração Requisito OK?"**, ambas apresentadas na raia de responsabilidade da equipe de projeto. Apesar dessa responsabilidade estar apresentada como sendo conjunta da equipe, em última análise, consideramos razoável que ela deva ser atribuída ao gerente de projeto, pois supomos que ele possui o discernimento necessário à análise e o poder de decisão para aceitar ou não a alteração de requisito em questão. Isto pode ser reforçado pela recomendação do próprio CMMI na prática genérica **REQM GP 2.4** "alocar responsabilidades" e **REQM GP 2.5** "treinar pessoas".

Identificamos, então, a necessidade de definir um novo termo (vide o Anexo 3 – Glossário de Termos):

Identificação	**Termo**	**Definição**
REQM-T05	avaliação do impacto da alteração de requisito de projeto de *software*	é uma análise da alteração de requisito em relação ao projeto de *software* sob o ponto de vista dos participantes de projeto de *software* relevantes. (fonte: CMMI)

Apenas uma nova regra de negócio traduz para o Português Estruturado a 'subprática' REQM 1.3.3, na forma de estrutura de sentença de "Restrição I":

Código	Sentença	Estrutura
REQM-1.3.3-a	A avaliação do impacto da alteração de requisito de projeto de *software* (REQM-T05) **deve obrigatoriamente** *ser realizada* pelo gerente de projeto (REQM-T12).	Restrição I

A regra traduzida acima está indicada no diagrama de processo como comentário ligado às atividades "C8 – Avaliar Impacto Alteração Requisito" e "C9 – Alteração Requisito OK?". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.5.4. A 'Subprática' Específica REQM SP 1.3.4

A quarta e última 'subprática' da prática REQM SP 1.3 estabelece que os requisitos e alterações estejam disponíveis ao projeto na figura de seus participantes.

'Subprática'	Descrição
REQM 1.3.4	Tornar os requisitos e as alterações de requisitos disponíveis ao projeto.

O registro formal e o uso de mecanismos de armazenamento, manutenção e controle das informações dos requisitos e das alterações de requisitos devem ocorrer de forma tal que seja permitida a sua consulta pelos participantes do projeto. Níveis de acesso para consultas e/ou atualizações dos requisitos e das alterações de requisitos devem ser estabelecidos como meio de garantir a segurança e 'confidencialidade' das informações. Essa 'subprática' reforça a ligação com a prática genérica **REQM GP 2.6** de "gerenciar as configurações", e por conseqüência, com a área de processo **CM** – Gerência de Configuração. Está representada em nosso modelo de processo pelas atividades **"C7 – Registrar Requisito"** e **"C10 – Registrar Alteração Requisito"**. Esse registro é sugerido em nosso modelo no objeto "**BD REQUISITOS**", ou seja, o registro numa Base de Dados

informatizada para armazenar as informações formalizadas dos requisitos. Uma BD assim garante a aderência das atividades do processo proposto ao viabilizar os controles necessários para a Gerência de Configuração, bem como pela recomendação desta prática específica de geração de produtos típicos de trabalho. No Anexo 1 esses produtos típicos de trabalho estão apresentados na transcrição parcial das práticas específicas da área de processo **REQM** - Gerência de Requisitos. Além disso, no Anexo 5 pode ser encontrado um mapeamento de todos os produtos típicos de trabalho recomendados para os itens de documentos sugeridos por nossa proposta.

Com isso, definimos os novos termos, apresentados no Anexo 3 – Glossário de Termos:

Identificação	**Termo**	**Definição**
REQM-T28	registro do requisito de projeto de *software*	é a documentação formal da descrição do requisito de projeto de *software*.
REQM-T27	registro da alteração do requisito de projeto de *software*	é a documentação formal da descrição da alteração do requisito de projeto de *software*.

Assim, as seguintes seis novas regras de negócio traduzem para o Português Estruturado a 'subprática' REQM 1.3.4, na forma das estruturas de sentença "Fato IX", "Restrição I" e "Fato I":

Código	**Sentença**	**Estrutura**
REQM-1.3.4-a	Um projeto de *software* (REQM-T23) **está relacionado ao** registro do requisito de projeto de *software* (REQM-T28) **por '***possui'* **com grau** 1**:**1..N.	Fato IX
REQM-1.3.4-b	O registro do requisito de projeto de *software* (REQM-T28) **deve obrigatoriamente** *ser realizado* pelo gerente de projeto (REQM-T12).	Restrição I
REQM-1.3.4-c	Os participantes do projeto (REQM-T19) **têm permissão para** *consultar* o registro do requisito de projeto de *software* (REQM-T28).	Fato I

Código	Sentença	Estrutura
REQM-1.3.4-d	Um projeto de *software* (REQM-T23) **está relacionado a** uma alteração do requisito de projeto de *software* (REQM-T27) **por '***possui'* **com grau** 1**:**0..N.	Fato IX
REQM-1.3.4-e	O registro da alteração do requisito de projeto de *software* (REQM-T27) **deve obrigatoriamente** *ser realizado* pelo gerente de projeto (REQM-T12).	Restrição I
REQM-1.3.4-f	Os participantes do projeto (REQM-T19) **têm permissão para** *consultar* o registro da alteração do requisito de projeto de *software* (REQM-T27).	Fato I

As regras traduzidas acima estão indicadas no diagrama como comentários, sendo REQM-1.3.4-a, REQM-1.3.4-b e REQM-1.3.4-c ligados à atividade "C7 – Registrar Requisito" e REQM-1.3.4-d, REQM-1.3.4-e e REQM-1.3.4-f ligados à atividade "C10 – Registrar Alteração Requisito". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.6. A Prática Específica REQM SP 1.4

A quarta prática específica de Gerência de Requisitos apresentada pelo CMMI é:

Prática	Descrição
REQM 1.4	**Manter um rastreamento bidirecional entre os requisitos e os planos e produtos de trabalho do projeto.**

O processo sugerido para a implementação desta prática específica é apresentado no modelo elaborado segundo o diagrama de atividades da UML, na figura 14:

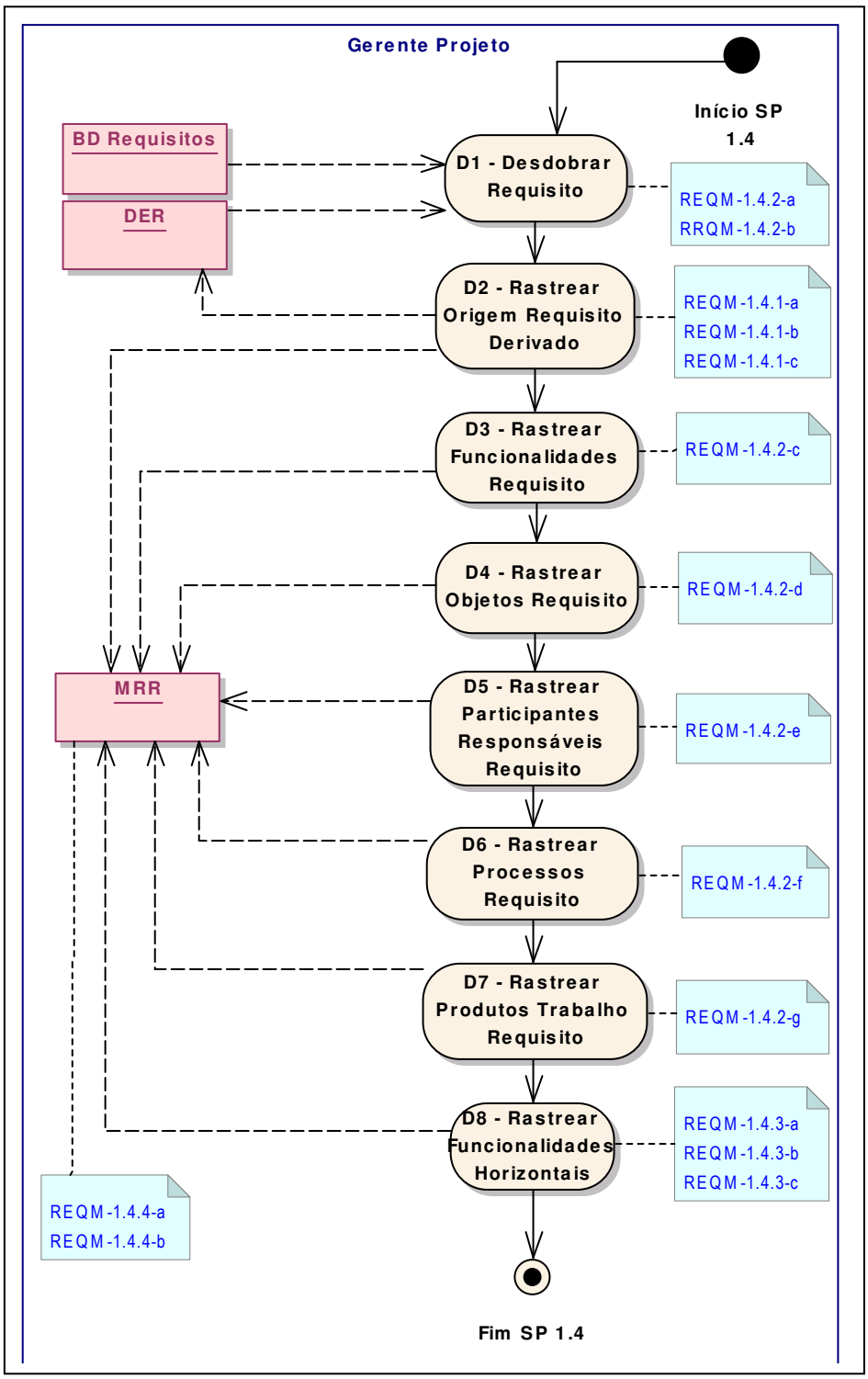

Figura 14 – Modelo de Processo da Prática Específica **REQM SP 1.4**

Para verificar o mapeamento das regras traduzidas em Português Estruturado às atividades do modelo de processo acima vide o Anexo 4.

A intenção desta prática é manter o rastreamento de um requisito desde sua origem, permear sua decomposição de produtos até o nível mais baixo de detalhamento, ou seja, componentes e artefatos de *software* de sua implementação e de volta destes até o requisito original. No caso de requisitos que não são implementados em *software* (p.ex., requisito de desempenho, de capacidade de armazenamento ou de tempo de resposta), o rastreamento deve chegar até os mecanismos ou controles resultantes. Esse rastreamento bidirecional permite que seja determinado se o requisito foi totalmente implementado e se os componentes e artefatos no nível mais baixo de detalhamento podem ser validados quanto à sua origem. Outra importante finalidade do rastreamento é facilitar e tornar mais confiável a avaliação de impacto no projeto pela introdução de alterações de requisitos.

O CMMI desdobra esta prática específica nas quatro 'subpráticas' analisadas a seguir.

5.6.1. A 'Subprática' Específica REQM SP 1.4.1

Na primeira 'subprática' da prática REQM SP 1.4 podemos identificar a recomendação de rastreamento de um requisito em relação aos seus requisitos derivados.

'Subprática'	**Descrição**
REQM 1.4.1	Manter o rastreamento dos requisitos para garantir que a fonte dos requisitos derivados está documentada.

Em nosso modelo de processo o rastreamento do requisito para os seus requisitos derivados é viabilizado pelo desdobramento do requisito original. Aqui podemos identificar claramente que os requisitos derivados, sendo detalhados pela equipe do projeto, constituem uma parte da geração de requisitos pelo próprio projeto. Está apresentada pela

atividade **"D1 – Desdobrar Requisito"**. Nesse processo são apresentados logo no início, os objetos "**DER** – DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS" e "**BD REQUISITOS**" criados no processo da prática específica anterior. Esta atividade é seguida por **"D2 – Rastrear Origem Requisito Derivado"**, que, ao recuperar a identificação do requisito origem sendo desdobrado, estabelece o registro desse rastreamento em um novo objeto: "**MRR** – MATRIZ DE RASTREAMENTO DE REQUISITOS" onde são registradas todas as informações de rastreamento de um requisito. Todas as atividades desse processo fazem uso desse objeto para o registro de cada tipo de rastreamento, como será apresentado a seguir. No Anexo 8 é apresentado um modelo deste documento, como parte de nossa proposta.

É importante observar que, apesar do rastreamento de requisitos ser uma atividade da área de processos de Gerência de Requisitos, ela ocorre desde as fases iniciais do projeto e só se finaliza juntamente com o projeto. A principal finalidade do rastreamento é mapear os requisitos à sua implementação em *software* (no caso de requisitos alocados ao *software*) ou à sua implementação aos controles (no caso de requisitos de sistema). Esse mapeamento traz facilidades na fase de verificação interna (fase de testes) e externa (fase de homologação). Permite também a geração imediata de evidências de implementação dos requisitos, garantindo que tudo o que foi solicitado foi efetivamente construído. Por ser um documento de verificação e atualização constante ao longo do ciclo de vida de desenvolvimento do *software*, a **MRR** – Matriz de Rastreamento de Requisitos – se constitui numa ferramenta de monitoração e controle do projeto, estando, portanto, fortemente ligada à outra área de processo de nível 2 de maturidade **– PMC -** Monitoração e Controle de Projeto.

Verificamos, então, a necessidade de definição dos seguintes termos, apresentados no Anexo 3 – Glossário de Termos:

Identificação	Termo	Definição
REQM-T25	rastreamento de um requisito de projeto de *software*	é (1) um mecanismo de acompanhamento do requisito que permite identificar sua origem ou seu desdobramento em outros requisitos, ou em outros produtos. (2) a evidência de uma associação entre um requisito e a fonte do requisito, sua implementação e sua verificação. Identifica a alocação do requisito ao *software*.
REQM-T10	documentação da origem do requisito derivado	é o registro formal da identificação do requisito original do requisito derivado.

Assim, podemos traduzir para o Português Estruturado a 'subprática' REQM 1.4.1 nas seguintes três regras de negócio, na forma da estrutura de sentença "Restrição I":

Código	Sentença	Estrutura
REQM-1.4.1-a	Um rastreamento de requisitos de projeto de *software* (REQM-T25) **deve obrigatoriamente** ser realizado pelo gerente de projeto (REQM-T12).	Restrição I
REQM-1.4.1-b	O rastreamento de requisitos de projeto de *software* (REQM-T25) **deve obrigatoriamente** *ser mantido atualizado* pelo gerente de projeto (REQM-T12).	Restrição I
REQM-1.4.1-c	A documentação da origem de um requisito derivado (REQM-T10) **deve obrigatoriamente** *ser garantida* pelo rastreamento dos requisitos do projeto de *software* (REQM-T25).	Restrição I

As regras traduzidas acima estão indicadas no diagrama de processos na forma de comentário ligado à atividade "D2 – Rastrear Origem Requisito Derivado". Para consultar o mapeamento completo das regras às atividades do processo, vide o Anexo 4.

5.6.2. A ‘Subprática’ Específica REQM SP 1.4.2

A segunda ‘subprática’ da prática REQM SP 1.4 refere-se ao rastreamento do requisito aos outros elementos do projeto:

‘Subprática’	**Descrição**
REQM 1.4.2	Manter o rastreamento de um requisito até os seus requisitos derivados, sua alocação em funções, objetos, pessoas, processos e produtos de trabalho.

Em nosso modelo, estão apresentadas as seguintes atividades que implementam esta ‘subprática’: **“D3 – Rastrear Funcionalidades Requisito”**, **“D4 – Rastrear Objetos Requisito”**, **“D5 – Rastrear Participantes Responsáveis Requisito”**, **“D6 – Rastrear Processos Requisito”**, **“D7 – Rastrear Produtos Trabalho Requisito”**, onde todos os rastreamentos efetuados são registrados no objeto “**MRR** – MATRIZ DE RASTREAMENTO DE REQUISITOS”.

Assim, identificamos a necessidade de definir os seguintes novos termos, apresentados no Anexo 3 – Glossário de Termos:

Identificação	**Termo**	**Definição**
REQM-T30	requisito derivado	é um requisito que não é explicitamente estabelecido nos requisitos do cliente, mas que é obtido por inferência de: (1) requisitos contextuais (p.ex. padrões aplicáveis, leis, políticas, práticas comuns e decisões gerenciais), ou (2) requisitos necessários para especificar um componente de produto. Requisitos derivados também podem surgir durante a análise e projeto de componentes de um produto ou sistema. (fonte: CMMI)
REQM-T11	funcionalidade do projeto de *software*	é uma qualidade ou aplicação funcional a ser implementada num *software* de forma que possibilite uma operação desejada ou necessária ao cliente ou usuário.

Identificação	Termo	Definição
REQM-T04	alocação de um requisito de projeto de *software* às funcionalidades do projeto de *software*	é o mapeamento de um requisito às qualidades operacionais do *software* que atendem o requisito de projeto de *software*.
REQM-T17	objeto do projeto de *software*	é uma instância de uma classe do projeto de *software*.
REQM-T03	alocação de um requisito de projeto de *software* aos objetos do projeto de *software*	é o mapeamento de um requisito aos objetos que instanciam as classes identificadas pelo requisito de projeto de *software*.
REQM-T20	participante relevante de projeto de *software*	é um participante identificado e com envolvimento e responsabilidade em atividades especificadas e incluídas num plano de projeto de *software*.
REQM-T21	processo do projeto de *software*	é uma seqüência de etapas realizadas para atender a um objetivo específico, por exemplo, o processo de desenvolvimento de *software*. (fonte: IEEE-STD-610)
REQM-T22	produto de trabalho do projeto de *software*	é um artefato qualquer criado como parte da definição, manutenção ou uso de um processo de *software*. Pode incluir descrições de processos, planos, procedimentos, programas de computador, documentação associada, as quais podem, ou não, ser entregues ao cliente ou ao usuário final. (fonte: CMM)

A ‘subprática’ REQM 1.4.2 pode ser traduzida para o Português Estruturado por sete novas regras de negócio, na forma da estrutura de sentença “Fato IX”:

Código	Sentença	Estrutura
REQM-1.4.2-a	Um requisito de projeto de *software* (REQM-T29) **está relacionado a** um requisito derivado (REQM-T30) **por ‘***desdobrou-se’* **com grau** 1**:**0..N.	Fato IX
REQM-1.4.2-b	Um requisito de projeto de *software* (REQM-T29) **está relacionado a** seus requisitos derivados (REQM-T30) **por ‘***é implementado por’* **com grau** 1**:**0..N.	Fato IX

Código	Sentença	Estrutura
REQM-1.4.2-c	Um requisito de projeto de *software* (REQM-T29) **está relacionado à** sua alocação às funcionalidades do projeto de *software* (REQM-T04) **por** *'é implementado por'* **com grau** 1**:**1..N.	Fato IX
REQM-1.4.2-d	Um requisito de projeto de *software* (REQM-T29) **está relacionado à** sua alocação aos objetos do projeto de *software* (REQM-T03) **por** *'é implementado por'* **com grau** 1..M**:**1..N.	Fato IX
REQM-1.4.2-e	Um requisito de projeto de *software* (REQM-T29) **está relacionado aos** participantes relevantes do projeto de *software* (REQM-T20) **por** *'é implementado por'* **com grau** 1..M**:**1..N.	Fato IX
REQM-1.4.2-f	Um requisito de projeto de *software* (REQM-T29) **está relacionado aos** processos do projeto de *software* (REQM-T21) **por** *'é implementado por'* **com grau** 1..M**:**1..N.	Fato IX
REQM-1.4.2-g	Um requisito de projeto de *software* (REQM-T29) **está relacionado aos** produtos de trabalho do projeto de *software* (REQM-T22) **por** *'é implementado por'* **com grau** 1..M**:**1..N.	Fato IX

As regras traduzidas acima estão indicadas no diagrama de processos na forma de comentários, sendo REQM-1.4.2-a e REQM-1.4.2-b ligados à atividade "D1 – Desdobrar Requisito", REQM-1.4.2-c ligado à atividade "D3 – Rastrear Funcionalidades Requisito", REQM-1.4.2-d ligado à atividade "D4 – Rastrear Objetos Requisito", REQM-1.4.2-e ligado à atividade "D5 – Rastrear Participantes Responsáveis Requisito", REQM-1.4.2-f ligado à atividade "D6 – Rastrear Processos Requisito" e, finalmente, REQM-1.4.2-g ligado à atividade "D7 – Rastrear Produtos Trabalho Requisito". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

5.6.3. A 'Subprática' Específica REQM SP 1.4.3

Prosseguindo na formalização, a terceira 'subprática' da prática REQM SP 1.4 estabelece a existência do rastreamento horizontal das funções e interfaces do projeto.

‘Subprática’	**Descrição**
REQM 1.4.3	Manter o rastreamento horizontal de função a função e através das interfaces.

Podemos entender o rastreamento horizontal como o rastreamento entre elementos de uma categoria em um mesmo nível de abstração. Em primeiro lugar, devemos esclarecer que nosso entendimento de **funções** é o mesmo de **funcionalidades**. Com o rastreamento das funcionalidades aos requisitos, podemos identificar quais funcionalidades são afetadas por quais requisitos. A associação dos requisitos com suas funcionalidades viabiliza a identificação das funcionalidades que estão relacionadas entre si e, conseqüentemente, as interfaces que possuem em comum. Em nosso modelo, este rastreamento está representado pela atividade **“D8 – Rastrear Funcionalidades Horizontais”**.

Identificamos a necessidade de definir os seguintes novos termos, apresentados no Anexo 3 – Glossário de Termos:

Identificação	**Termo**	**Definição**
REQM-T15	interface de funcionalidade do projeto de *software*	é o conjunto de elementos que possibilitam a conexão entre funcionalidades de projeto de *software* diferentes.
REQM-T26	rastreamento horizontal	é um rastreamento entre elementos de uma categoria em mesmo nível de abstração.

Assim, três novas regras de negócio traduzem para o Português Estruturado a ‘subprática’ REQM 1.4.3, na forma das estruturas de sentença “Fato IX” e “Restrição I”:

Código	**Sentença**	**Estrutura**
REQM-1.4.3-a	Uma funcionalidade do projeto de *software* (REQM-T11) **está relacionada às** funcionalidades do projeto de *software* relacionadas (REQM-T11) **por** *‘é implementada por’* **com grau** 1.M**:**0..N.	Fato IX

Código	Sentença	Estrutura
REQM-1.4.3-b	Uma funcionalidade do projeto de *software* (REQM-T11) **está relacionada às** suas interfaces (REQM-T15) **por** *'é implementada por'* **com grau** 1.M**:**0..N.	Fato IX
REQM-1.4.3-c	Um rastreamento horizontal de funcionalidades de projeto de *software* (REQM-T26) **deve obrigatoriamente** *ser realizada* pelo gerente de projeto (REQM-T12).	Restrição I

As regras traduzidas acima estão indicadas no diagrama de processo como comentário ligado à atividade "D8 – Rastrear Funcionalidades Horizontais". Para consultar o mapeamento completo das regras às atividades do processo, vide o Anexo 4.

5.6.4. A 'Subprática' Específica REQM SP 1.4.4

Por fim, a última 'subprática' específica da prática REQM SP 1.4 estabelece a geração de uma matriz de rastreamento de requisitos.

'Subprática'	Descrição
REQM 1.4.4	Gerar a matriz de rastreamento dos requisitos.

A **MRR** – Matriz de Rastreamento de Requisitos é o documento resultante das atividades de todas as 'subpráticas' anteriores da prática REQM SP 1.4 e já foi indicado no modelo de processo por todas as atividades mencionadas.

Não obstante não havermos indicado nenhuma nova atividade em nosso modelo de processo, identificamos a necessidade de definição de um novo termo:

Identificação	Termo	Definição
REQM-T16	matriz de rastreamento de requisitos de projeto de *software*	é uma matriz que apresenta todos os mapeamentos, ou associação de um requisito para seus requisitos derivado, objetos, processos, funcionalidades, participantes e produtos de trabalho do processo de *software*.

As seguintes duas novas regras de negócio traduzem para o Português Estruturado a 'subprática' REQM 1.4.4, na forma das estruturas de sentença "Fato IX" e "Restrição I":

Código	Sentença	Estrutura
REQM-1.4.4-a	Um projeto de *software* (REQM-T23) **está relacionado a** uma matriz de rastreamento de requisitos de projeto de *software* (REQM-T16) **por** ***'possui'*** **com grau** 1**:**1.	Fato IX
REQM-1.4.4-b	Uma matriz de rastreamento de requisitos de projeto de *software* (REQM-T16) **deve obrigatoriamente** *ser gerada* pelo gerente de projeto (REQM-T12).	Restrição I

As regras traduzidas acima estão indicadas no diagrama de processos na forma do comentário ligada ao objeto "**MRR** – MATRIZ DE RASTREAMENTO DE REQUISITOS". Para consultar o mapeamento completo das regras às atividades do processo proposto, vide o Anexo 4.

6. Discussão

Com esse trabalho foram identificadas as regras de negócio correspondentes a um processo de *software*, definidas a partir das práticas específicas da área de processo REQM – Gerência de Requisitos. Essas práticas constituem parte das recomendações a serem atendidas por um processo de *software* de forma a estar em conformidade com o modelo de capacidade de *software*, em seu nível 2 de maturidade, definido pelo CMMI [CHRISSIS *et al*., 2003].

6.1. Análise das Práticas Específicas de Gerência de Requisitos

Verificamos que, ao elaborar a primeira fase de formalização dessas práticas específicas, obtém-se uma definição mais detalhada das mesmas, o que pode facilitar a modelagem do processo a ser adotado. No processo de análise e detalhamento das 'subpráticas' encontramos oportunidades de tomada de decisão sobre a adequação das recomendações do CMMI ao processo de uma organização específica, respeitando sua cultura organizacional e suas características especiais. A adequação das recomendações a partir do detalhamento das 'subpráticas' e a tomada de decisões quanto à formalização em termos e regras de negócio constitui-se numa otimização do esforço de adaptação do CMMI para os objetivos de uma organização que deseje aprimorar o seu processo de desenvolvimento de *software* e também de formalizar seus processos e regras de negócio.

Nessa formalização foram identificadas, ao longo da exposição da aplicação do método proposto, algumas oportunidades de melhoria das práticas recomendadas pelo CMMI, como por exemplo, a sugestão de esclarecimento sobre papéis e responsabilidades dos participantes na formulação de algumas práticas e/ou 'subpráticas' (p. ex. o estabelecimento de critérios de escolha de provedores de requisitos e de aceite de

requisitos). Esclarecemos que a prática genérica REQM GP 2.7 de "identificar e envolver os *stakeholders* relevantes do projeto" atende essa sugestão de esclarecimento de papéis e responsabilidades, porém como prática genérica, ela pode ter sua importância diminuída. Dessa forma, como fator de aprimoramento das recomendações de práticas, sugerimos a reclassificação da mesma de prática genérica para prática específica.

Foi também explicitada a existência de ambigüidade na formulação da 'subprática' que recomenda o estabelecimento de um acordo informal pela utilização do termo "reach an understanding...", que, se substituído por "informal agreement", poderia ter tornado a 'subprática' mais clara, objetiva e direta. Além disso, identificamos e esclarecemos algumas omissões que devem ser tratadas em cada adaptação pela interpretação dos adotantes das recomendações: por exemplo, não existem recomendações sobre a aplicação do critério de escolha dos provedores de requisitos aos participantes de provisão de requisitos fornecidos pelo cliente.

Adicionalmente foi observada a inexistência de uma prática ou 'subprática' que estabeleça a formalização dos compromissos da equipe do projeto obtidos pelo acordo informal com o cliente e/ou usuário. No caso do documento MRR – Matriz de Rastreamento de Requisitos, a recomendação de sua criação é explícita ('subprática' REQM SP 1.4.4), o que não ocorre para a criação de uma especificação documentada dos requisitos propriamente ditos (no nosso trabalho, o documento DER – Documento de Especificação de Requisitos).

No caso da documentação de requisitos também foi observada a inexistência de sugestão ou recomendação de classificação dos mesmos (requisitos funcionais, técnicos, não-técnicos, de desempenho, de segurança etc), que poderia agregar valor aos mesmos. Esta classificação é recomendada pelo IEEE-STD-830 [1998], e por meio dela pode se tornar mais clara a atribuição de responsabilidades de participação dos envolvidos na implementação e/ou verificação dos requisitos. Por exemplo, os requisitos classificados como funcionais devem ser de responsabilidade de implementação da equipe de desenvolvimento de *software*; requisitos de desempenho devem ser verificados conjuntamente pelas equipes de desenvolvimento de *software* e de sistemas; requisitos de

seguranças devem ser verificados conjuntamente pelas equipes de desenvolvimento de *software* e de segurança de informação etc.

6.2. Análise dos Termos e Regras Formalizados

Considerando o conjunto de termos e regras gerados como resultado da tradução das práticas e 'subpráticas' específicas dessa área de processo para o Português Estruturado podemos constatar que:

- foram consideradas no trabalho: 4 (quatro) práticas e 14 (quatorze) 'subpráticas' específicas da área de processo de Gerência de Requisitos do CMMI;
- foram identificados 30 (trinta) termos (relacionados no anexo 3 – Glossário de Termos);
- as 14 (quatorze) 'subpráticas' foram traduzidas para 43 (quarenta e três) regras em Português Estruturado (apresentadas no anexo 4 – Mapeamento das Regras).

Com base nessas informações, podemos então avaliar os termos e regras sob alguns pontos de vista, relacionados a seguir.

6.2.1. Quanto à Freqüência dos Termos nas Regras

Os termos mais freqüentes nas regras foram:

- REQM-T12 – "gerente de projeto";
- REQM-T29 – "requisito de projeto de *software*".

Cada um possui 16 (dezesseis) apresentações nas 43 (quarenta e três) regras. Em segundo lugar o termo REQM-T23 – "projeto de *software*", pois possui 10 (dez) apresentações. Destacamos, ainda que o termo REQM-T02 – "alteração de requisito de projeto de *software*" é o terceiro mais freqüente, com 4 (quatro) apresentações.

Outra constatação que podemos extrair dessa análise é a de que grande parte (40%) dos termos tem uma freqüência bem menor: 17 (dezessete) termos estão presentes apenas 1 (uma) ou 2 (duas) vezes no conjunto total de regras.

6.2.2. Quanto à Importância dos Termos

Com a freqüência apresentada acima, constatamos a importância dos termos "gerente de projeto", "requisito de projeto de *software*" e "projeto de *software*", e a sua coerência com o domínio de aplicação das regras formalizadas. Essas características podem se constituir em evidências positivas sobre a adequação da formalização adotada. Reforçando essa análise, destacamos o fato de que o termo seguinte no grau de freqüência é "alteração de requisito de projeto de *software*", também totalmente em conformidade com o domínio de aplicação do negócio em questão.

Adicionalmente, a freqüência de apresentação do termo "gerente de projeto" realça o seu papel ao atribuir-lhe responsabilidades de confecção, análise e verificação ao longo do processo de Gerência de Requisitos. Isso apóia a nossa sugestão de que a prática genérica **REQM GP 2.7** de "identificar e envolver os *stakeholders* relevantes do projeto" poderia sofrer uma reclassificação para prática específica, dando-lhe assim maior relevância.

6.2.3. Quanto à Ordem, Precedência e Dependência entre as Regras

Durante o trabalho, na análise e interpretação das práticas e 'subpráticas' objeto de nosso estudo, a ordem de apresentação das mesmas reforçou a idéia de que ela deve ser respeitada por induzir o encadeamento lógico das atividades dos processos propostos. Essa ordem respeita a precedência necessária entre as 'subpráticas', e em alguns casos, reforça a dependência existente entre algumas delas. Por exemplo, na primeira prática específica (SP 1.1) estão as 'subpráticas' que indicam a necessidade de estabelecimento do critério de escolha de participantes de provisão de requisitos e do critério de aceite de requisitos. Somente após o estabelecimento e negociação desses critérios com o usuário é que são apresentadas as 'subpráticas' de fornecimento de requisitos. Por este motivo, em nosso esforço de formalização e na elaboração das regras identificadas e traduzidas para o Português Estruturado, procuramos manter essa mesma ordenação.

No Anexo 6 é apresentada a relação completa das atividades dos processos apresentados, acompanhadas de suas respectivas atividades antecessoras e sucessoras e a relação de seus insumos de entrada e de saída.

6.2.4. Quanto à Consistência da Regras Formalizadas

Neste trabalho, considerando o método de formalização adotado, não identificamos inconsistências nas regras e termos elaborados em relação às práticas e 'subpráticas' escolhidas como objeto de estudo.

Por outro lado, consideramos que durante todo o processo de formalização é altamente recomendável que, para cada 'subprática' traduzida em termos e regras, seja

efetuado um ciclo completo de análise sobre a sua consistência em relação aos demais termos e regras identificados e formalizados. Dessa forma, por meio de refinamentos sucessivos, foi possível identificarmos ajustes necessários que foram sendo efetuados gradativamente, ao longo do processo de análise e formalização. Como conseqüência, sugerimos que sua implementação posterior em ferramentas de automação deve ser efetuada após a etapa de formalização em Português Estruturado. Adicionalmente, destacamos a vantagem de que tal ferramenta possibilite um rigor de cadastramento quanto aos modelos de sentença, de forma a facilitar a identificação de inconsistências remanescentes.

7. Conclusões

A gestão organizacional pela gestão de processos de negócio traz vantagens competitivas no contexto atual de constante transformação e dos desafios diários da competição de mercado. Os processos de negócio estão passando para o centro das atenções na arquitetura dos negócios e dos sistemas que os implementam e apóiam. Algumas empresas vêm organizando seus empregados em equipes colaborativas responsáveis por processos de negócio completos, do início ao fim.

Sob a perspectiva do negócio, as regras de negócio podem ser entendidas como as diretrizes de condução, ação, prática ou procedimento de uma atividade em particular. Sob a perspectiva dos sistemas de informação, as regras de negócio estabelecem as definições ou restrições de algum aspecto do negócio. Seu objetivo é definir a estrutura do negócio, ou controlar e influenciar o comportamento do mesmo. É imperativo que elas sejam usadas na definição dos processos de negócio associados. Surge então a necessidade de uma garantia mínima de que as regras de negócio, após serem estabelecidas, sejam corretamente interpretadas e implementadas nos respectivos processos de negócio.

A definição formal de regras de negócio apresenta vantagens de explicitação das mesmas em linguagem não-ambígua e documentada. Além disso, possibilita maior grau de detalhamento para a definição do processo de negócio em questão.

O processo de negócio escolhido neste trabalho é o processo de desenvolvimento de *software*. Na definição das regras de negócio correspondentes a tal processo, consideramos adequada a adoção de um modelo de maturidade para o desenvolvimento e manutenção de *software* – o **CMMI** – *Capability Maturity Model Integration*. Este modelo apresenta

recomendações de melhores práticas específicas e genéricas, por área de processo, a serem adotadas por organizações envolvidas com a construção e manutenção de *software*.

Uma proposta de formalização considerada adequada compõe-se de três etapas. A primeira consiste na estruturação das regras de negócio em sentenças escritas na linguagem Português Estruturado. A segunda é a identificação e modelagem do processo de negócio correspondente. Finalmente, a terceira constitui-se da transformação das sentenças em Português Estruturado para Lógica de Primeira Ordem e posterior tradução para uma linguagem de programação em lógica, como PROLOG. Com esta proposta de formalização é viabilizada, ao menos parcialmente, uma verificação automatizada das regras de negócio.

Neste trabalho, detalhamos a primeira etapa do processo de formalização. Prosseguindo nesta formalização e nos direcionando para a automação da formalização de regras de negócio, propomos a adoção posterior da ferramenta *REGULA,* que automatiza as duas etapas finais e permite facilidades de verificação de integridade no cadastramento das regras em Português Estruturado.

Em resumo, nosso trabalho consistiu da formalização em Português Estruturado das práticas e ’subpráticas’ específicas da área de processo REQM – Gerência de Requisitos do CMMI e da elaboração de um conjunto de modelos de processo aderente a estas práticas. Tais processos foram modelados como diagramas de atividades da UML.

7.1. Trabalhos Futuros

Como trabalho futuro, em primeiro lugar, sugerimos que os termos e regras correspondentes à área de processo escolhida, e que foram identificados e formalizados, sejam efetivamente cadastrados na ferramenta *REGULA* para que sejam verificados por meio de seu módulo de consultas às permissões e proibições. Dessa forma, a verificação automatizada pela ferramenta pode se constituir numa validação mais efetiva dos termos e regras formalizadas e, conseqüentemente, também do processo proposto. O cadastramento dos termos e regras identificados neste trabalho na ferramenta *REGULA* não foi possível ser realizado por estar a mesma ainda em fase de construção.

Prosseguindo nesta linha de pesquisa, outras áreas de processo devem ser submetidas a essa formalização para que todo um nível de maturidade (por exemplo, o nível 2 de maturidade) seja formalizado e seus processos modelados. Isso pode viabilizar que uma adoção completa de recomendações do CMMI seja efetuada na representação nivelada. Essa formalização se dará pela tradução das recomendações em regras de negócio e pela definição do(s) processo(s) de *software* correspondente(s). A formalização gradual das recomendações de cada área de processo e a validação de cada regra formalizada em relação às demais regras cadastradas pode trazer benefícios como evitar inconsistências das práticas adotadas e garantir maior aderência das regras adotadas em relação às recomendações originais do CMMI. Além disso, o processo de elaboração das regras deverá levar em consideração a adaptação das recomendações aos objetivos e à cultura organizacional da empresa adotante do modelo de maturidade. Por isso, sugerimos que nesta adoção, uma organização real seja utilizada no estudo de caso, de forma a aprimorar o trabalho com os desafios reais enfrentados nas empresas, assim como obter as vantagens do alinhamento da pesquisa acadêmica aos rigores da realidade empresarial, trazendo enriquecimento de experiências aos dois ambientes.

Outra abordagem que pode aprimorar o presente trabalho é a conjugação das duas áreas de processo referentes a requisitos apresentadas no CMMI [CHRISSIS *et al*., 2003]: complementar a formalização da área de REQM – *Requirementes Management* com formalização das práticas da área de processo de RD – *Requirements Development*. As práticas e 'subpráticas' das duas áreas de processo conjugadas e formalizadas poderão formar um conjunto rico e mais completo de regras de negócio e ampliar a identificação de um processo de negócio que englobe todos os aspectos referentes a requisitos. Além disso, isso constituir-se-á numa abordagem contínua, de nível de maturidade 3, apta a enfrentar os rigores de um verificação formal (*assessment)* por entidade regulamentada pelo SEI.

Por fim, acreditamos que essa formalização pode auxiliar o processo de auto-avaliação da organização em relação à adoção das recomendações sugeridas pelo CMMI, funcionando como uma autoverificação (*self-assessment)* a ser efetuado pela própria organização com o objetivo de promover melhorias contínuas no(s) processo(s) de *software* estabelecido(s), ou para permitir à organização se preparar para um processo de verificação formal aplicado por entidade regulamentada pelo SEI.

Referências Bibliográficas

ARITY, Arity Corporation, **The Arity Prolog Compiler and Interpreter – Version 6.1**, http://www.arity.com - obtido da web em 19/01/2002.

AURÉLIO, Aurélio Buarque H. F., **Aurélio Século XXI – O Dicionário da Língua Portuguesa**, Editora Nova fronteira, Rio de Janeiro, Brasil, 1999.

BASTOS, Alexandre R. Gualter, **Um estudo sobre adoção de um modelo que visa promover melhorias no processo de desenvolvimento de *software*: O caso CMM em uma empresa do setor financeiro no Brasil**, Dissertação (Mestrado em Ciências e Engenharia de Sistemas e Computação), COPPE / Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil, 1996.

BOOCH, G., Rumbaugh, J., Jacobson, I., **Unified Modeling Language User Guide**, Addison-Wesley, EUA, 1994.

BOOCH, G., Rumbaugh, J., Jacobson, I., **The Unified Software Development Process**, Addison-Wesley, EUA, 1999.

BRATKO, Ivan, **PROLOG Programming for Artificial Intelligence**, 3ª ed., Addison-Wesley, EUA, 2001.

BRG, Business Rules Group, **Business Rules Group - Final Report**, revisão 1.3, julho de 2000, http://www.businessrulesgroup.org - obtido em 04/06/2001.

BRG, Business Rules Group**, What is a Business Rule?**, http://www.businessrulesgroup.org - obtido em 13/06/2004.

CAMBRIDGE, **Cambridge International Dictionary of English**, Procter, Paul, Cambridge University Press, Bath, Great Britain, 1995.

CASANOVA, Marco A., Giorno, Fernando A. C., Furtado, Antonio L., **Programação em Lógica e a Linguagem PROLOG**, Ed. Edgard Blücher Ltda., São Paulo, Brasil, 1987.

CHRISSIS, Mary Beth, Konrad, Mike, Shrum, Sandy, **CMMI: Guidelines for Process Integration and Product Improvement**, SEI Software Engineering Institute Addison-Wesley, Pensilvânia, EUA, 2003.

CAPPELLI, Cláudia, **Melhoria de Qualidade na Manutenção de *Software***, Dissertação (Mestrado em Sistemas de Informação), Instituto de Matemática – Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil, 2000.

COPI, Irving M., **Introdução à Lógica**, 1ª. ed, Editora Mestre Jou, São Paulo, Brasil, 1961.

CORRÊA, Sérgio M., Siqueira, José R. B., Schmitz, Éber A, **Representação de Regras de Negócio e seu Mapeamento em Lógica de Primeira Ordem**, IV Simpósio de Pesquisa Operacional da Marinha **–** SPOLM 2001, Rio de Janeiro, Brasil, 2001.

CORRÊA, Sérgio M., **RuleCheck – Uma Ferramenta para catálogo e Administração de Regras de Negócio**, Projeto Final de Curso de Graduação, Bacharelado em Informática, Instituto de Matemática – Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil, 2002.

CROSBY, Philip B., **Quality is Free – The Art of Making Quality Certain**, McGraw-Hill Publishing Co., EUA, 1979.

CRUZ, Pedro Oscar S., **Heurísticas para identificação de Requisitos de Sistemas de Informações a partir de Modelos de Processos**, Dissertação (Mestrado em Informática), Instituto de Matemática – Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil, 2004.

CRUZ, Tadeu, **Sistemas, Organização e Métodos**, 3ª ed., Ed. Atlas, São Paulo, Brasil, 2002.

DAVENPORT, Thomas, **Reengenharia de Processos**, 5ª ed., Ed. Campus, Rio de Janeiro, Brasil, 1994.

DIAS, Felipe G., Morgado, Gisele P., Martins, Alissandra E., *et al.*, **Um ambiente para Modelagem Organizacional Baseado em Regras de Negócio**, artigo submetido em 19/08/2004 ao 1º Simpósio Brasileiro de Sistemas de Informação, Rio Grande do Sul, Brasil, (a se realizar em) Outubro de 2004.

DRUCKER, Peter F., **Desafios Gerenciais para o Século XXI**, 1ª ed., Ed. Guazzelli Ltda., São Paulo, Brasil, 1999.

GHEZZI, Carlo, Jazayeri, Mehdi, Mandrioli, Dino, **Fundamentals of Software Engineering**, Prentice-Hall Inc., New Jersey, EUA, 1991.

HUMPHREY, Watts S., **Preliminary Report on Conducting SEI Assisted Assessments of Software Engineering Capability**, SEI Technical Report, EUA, 1987.

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

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

ITSMF, Information Technology Service Management Forum, **IT Service Management, an introduction**, 1ª ed., Van Haren Publishing, Holanda, 2002.

MARSHALL, Isnard, *et al*., **Gestão da Qualidade**, FGV Management – série Gestão Empresarial, 1ª ed., Editora FGV, Rio de Janeiro, Brasil, 2003.

MOORE, Connie, **Process Knowledge**, http://e-workflow.org/downloads/gue-pro.pdf, obtido em 21/05/2004.

MORGAN, Tony, **Business Rules and Information Systems**, Addison-Wesley – Pearson Education, Inc, 1ª ed., Massachusetts, EUA, 2002.

PAULK, Mark C., *et al*., **The Capability Maturity Model: Guidelines for Improving the Software Process**, v1.1, Carnegie Mellon University / Software Engineering Institute, Addison-Wesley Longman Inc., Massachusetts, EUA, 1994.

PMI, Project Management Institute, **History**, Pensilvânia, EUA, http://www.pmi.org/info/AP_IntroOverview.asp, obtido em 26/06/2004.

SMITH, Howard, Fingar, Peter, **Business Process Management: The Third Wave**, 1ª ed., Meghan-Kiffer Press, Florida, EUA, 2003.

SCHMITZ, E. A., Lima, P. M. V., Silveira, D. S. *et al*., **Uma Infra-estrutura Virtual para Projetos de Empresas**, Relatório Técnico, Instituto de Matemática, Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro, Rio de Janeiro, Brasil, Agosto de 2004.

SPICE, Software Process Improvement and Capability Determination, **What is SPICE?**, http://www.sqi.gu.edu.au/spice, obtido em 26/06/2004.

VERNADAT, François. B., **Enterprise Modeling and Integration – Principles, Modeling and Applications**, 1ª ed., Chapman & Hall, EUA, 1996.

Anexo 1

Práticas Específicas da Área de Processo de

Gerência de Requisitos do CMMI

REQM – Requirements Management – An Engineering Process Area at Maturity Level 2

The purpose of Requirements Management (REQM) is to manage he requirements of the project's products and product components and to identify inconsistencies between those requirements and the project's plans and work products.

Specific Practices by Goal

- **Specific Goal 1 - Manage Requirements**

Requirements are managed and inconsistencies with project plans and work products are identified.

The project maintains a current and approved set of requirements over the life of the project by doing the following:

- ✓ Managing all changes to the requirements;
- ✓ Maintaining the relationship among the requirements, the project plans, and the work products;
- ✓ Identifying inconsistencies among the requirements, the project plans, and the work products;
- ✓ Taking corrective action.

- **Specific Practice 1.1 - Obtain an Understanding of Requirements**

  Develop an understanding with the requirements providers on the meaning of the requirements.

  As the project matures and requirements are derived, all activities or disciplines will receive requirements. To avoid requirements creep, criteria are established to designate appropriate channels, or official sources, from which to receive

requirements. The receiving activities conduct analyses of the requirements with the requirements provider to ensure that a compatible, shared understanding is reached on the meaning of the requirements. The result of this analysis and dialog is an agreed-to set of requirements.

Typical Work Products:

a. Lists of criteria for distinguishing appropriate requirements providers
b. Criteria for evaluation and acceptance of requirements
c. Results of analysis against criteria
d. An agreed-to set of requirements

Subpractices:

1. Establish criteria for distinguishing appropriate requirements providers.
2. Establish objective criteria for the acceptance of requirements.

   Lack of acceptance criteria often results in inadequate verification, costly rework, or customer rejection.

   Examples of acceptance criteria include the following:

   - Clearly and properly stated
   - Complete
   - Consistent with each other
   - Uniquely identified
   - Appropriate to implement
   - Verifiable (testable)
   - Traceable
3. Analyze requirements to ensure that established criteria are met.
4. Reach an understanding of the requirements with the requirements provider so that the project participants can commit to them.

- **Specific Practice 1.2 - Obtain Commitment to Requirements**

  Obtain commitment to the requirements from the project participants.

  Whereas the previous specific practice dealt with reaching an understanding with the requirements providers, this specific practice deals with agreements and commitments

among those who have to carry out the activities necessary to implement the requirements. Requirements evolve throughout the project, especially as described by the specific practices of the Requirements Development process area and Technical Solution process area. As the requirements evolve, this specific practice ensures that project participants commit to the current, approved requirements and the resulting changes in project plans, activities, and work products.

Typical Work Products:

a. Requirements impact assessment

b. Documented commitments to requirements and requirements changes

Subpractices:

1. Assess the impact of requirements on existing commitments.

   The impact on the project participants should be evaluated when the requirements change or at the start of a new requirement.

2. Negotiate and record commitments.

   Changes to existing commitments should be negotiated before project participants commit to the requirement or requirement change.

- **Specific Practice 1.3 - Manage Requirements Change**

  Manage changes to the requirements as they evolve during the project.

During the project, requirements change for a variety of reasons. As needs change and as work proceeds, additional requirements are derived and changes may have to be made to the existing requirements. It is essential to manage this additions and changes efficiently and effectively. To effectively analyze the impact of the changes, it is necessary that the source of each requirement is known and the rationale for any change is documented. The project manager may, however, want to track appropriate measures of requirements volatility to judge whether new or revised controls are necessary.

Typical Work Products:

a. Requirements status

b. Requirements database

c. Requirements decision database

Subpractices:

1. Capture all requirements and requirements changes that are given to or generated by the project.
2. Maintain the requirements change history with the rationale for the changes.
3. Evaluate the impact of requirements changes from the standpoint of relevant stakeholders.
4. Make the requirements and change data available to the project.

- **Specific Practice 1.4 - Maintain Bidirectional Traceability of Requirements**

  Maintain bi-directional traceability among the requirements and the project plans and work products.

  The intent of this specific practice is to maintain the bidirectional traceability of the requirements for each level of product decomposition. When the requirements are managed well, traceability can be established from the source requirements to its lower level requirements and from the lower level requirements back to their source. Such bidirectional traceability helps determine that all source requirements have been completely addressed and that all lower level requirements can be traced to a valid source. Requirements traceability can also cover the relationships to other entities such s intermediate and final work products, changes in design documentation, test plans, and work tasks. The traceability should cover both the horizontal and vertical relationships, such as across interfaces. Traceability is particularly needed in conducting the impact assessment of requirements changes on the project plans, activities, and work products.

  Typical Work Products:

  a. Requirements traceability matrix
  b. Requirements tracking system

  Subpractices:

  1. Maintain requirements traceability to ensure that the source of lower level (derived) requirements is documented.
  2. Maintain requirements traceability from a requirement to its derived requirements and allocation to functions, objects, people, processes, and work products.
  3. Maintain horizontal traceability from function to function and across interfaces.

4. Generate the requirements traceability matrix.

- **Specific Practice 1.5 - Identify Inconsistencies between Project Work and Requirements**

Identify inconsistencies between the project plans and work products and the requirements.

This specific practice finds the inconsistencies between the requirements and the project plans and work products and initiates the corrective action to fix them.

Typical Work Products:

a. Documentation of inconsistencies including sources, conditions, and rationale
b. Corrective actions

Subpractices:

1. Review the project's plans, activities, and work products for consistency with the requirements and the changes made to them.
2. Identify the source of the inconsistency and the rationale.
3. Identify changes that need to be made to the plans and work products resulting from changes to the requirements baseline.
4. Initiate corrective actions.

Anexo 2

Práticas Genéricas da Área de Processo de

Gerência de Requisitos do CMMI

Generic Practices by Goal

- **Generic Goal 1 – Achieve Specific Goals**

The process supports and enables achievement of the specific goals of the process area by transforming identifiable input work product to produce identifiable output work products.

- **Generic Practice 1.1 – Perform Base Practices**

  Perform the base practices of the requirements management process to develop work products and provide services to achieve the specific goals of the process.

- **Generic Goal 2 – Institutionalize a Managed Process**

The process is institutionalized as a managed process.

COMMITMENT TO PERFORM

- **Generic Practice 2.1 – Establish an Organizational Policy**

  Establish and maintain an organizational policy for planning and performing the requirements management process.

  Elaboration

  This policy establishes organizational expectations for managing requirements and identifying inconsistencies between the requirements and the project plans and work products.

ABILITY TO PERFORM

- **Generic Practice 2.2 – Plan the Process**

  Establish and maintain the plan for performing the requirements management.

  Elaboration

  Typically, this plan for performing the requirements management process is a part of the project plan as described in the Project planning process area.

- **Generic Practice 2.3 – Provide Resources**

  Provide adequate resources for performing the requirements management process, developing the work products, and providing the services of the process.

  Elaboration

  Examples of resources provided include the following:

  - *Requirements tracking tools*
  - *Traceability tools*

- **Generic Practice 2.4 – Assign Responsibility**

  Assign responsibility and authority for performing the process, developing the work products, and providing the services of the requirements management process.

- **Generic Practice 2.5 – Train People**

  Train people performing or supporting the requirements management process as needed.

  Elaboration

  Examples of resources provided include the following:

  - *Application domain*
  - *Requirements definition, analysis, review, and management*

- *Requirements management tools*
- *Configuration management*
- *Negotiation and conflict resolution*

DIRECTING IMPLEMENTATION

- **Generic Practice 2.6 – Manage Configurations**

  Place designated work products of the requirements management process under appropriate levels of configuration management.

  Elaboration

  Examples of resources provided include the following:

  - *Requirements*
  - *Requirements traceability matrix*

- **Generic Practice 2.7 – Identify and Involve Stakeholders**

  Identify and involve the relevant stakeholders of the requirements management process as planned.

  Elaboration

  Select relevant stakeholders from customers, and users, developers, producers, testers, suppliers, marketers, maintainers, disposal personnel, and others who may be affected by, or may affect, the products as well as the process.

  Examples of activities for stakeholder involvement include:

  - *Resolving issues on the understanding of the requirements*
  - *Assessing the impact of requirements changes*
  - *Communicating the bidirectional traceability*
  - *Identifying inconsistencies among project plans, work products, and requirements*

- **Generic Practice 2.8 – Monitor and Control the Process**

  Monitor and control the requirements management process against the plan for performing the process and take appropriate corrective action.

  Elaboration

  Examples of measures used in monitoring and controlling include the following:

  - *Requirements volatility (percentage of requirements changed)*

VERIFYING IMPLEMENTATION

- **Generic Practice 2.9 – Objectively Evaluate Adherence**

  Objectively evaluate adherence of the requirements management process against its process description, standards, and procedures, and address noncompliance.

  Elaboration

  Examples of activities reviewed include the following:

  - *Managing requirements*
  - *Identifying inconsistencies among project plans, work products, and requirements*

  Examples of work products reviewed include the following:

  - *Requirements*
  - *Requirements traceability matrix*

- **Generic Practice 2.10 – Review Status with Higher level Management**

  Review the activities, status, and results of the requirements management process with higher level management and resolve issues.

  Elaboration

  Proposed changes to commitments to be made external to the organization are reviewed with higher level management to ensure that all commitments can be accomplished.

Anexo 3

Glossário de Termos

Identificação	Termo	Definição
REQM-T01	acordo informal sobre os requisitos de projeto de *software*	é (1) o resultado obtido da análise conjunta e objetiva do requisito de projeto de *software*. Esse acordo garante que o requisito pode ser incluído num conjunto compatível e factível de requisitos de projeto de *software*. (2) acordo uniforme e compartilhado por todos os participantes do projeto de *software*.
REQM-T02	alteração de requisito de projeto de *software*	é a alteração de um requisito de projeto de *software* previamente elaborado e acordado para o projeto de *software*. A alteração deve ser analisada para a avaliação de seu impacto no andamento do projeto de *software* e deve ser acordada com o cliente ou usuário antes de ser aceita no projeto de *software*.
REQM-T03	alocação de um requisito de projeto de *software* aos objetos do projeto de *software*	é o mapeamento de um requisito aos objetos que instanciam as classes identificadas pelo requisito de projeto de *software*.
REQM-T04	alocação de um requisito de projeto de *software* às funcionalidades do projeto de *software*	é o mapeamento de um requisito às qualidades operacionais do *software* que atendem o requisito de projeto de *software*.
REQM-T05	avaliação do impacto da alteração de requisito de projeto de *software*	é uma análise da alteração de requisito em relação ao projeto de *software* sob o ponto de vista dos participantes de projeto de *software* relevantes. (fonte: CMMI)
REQM-T06	compromissos de projeto de *software* existentes	são os compromissos assumidos pelos participantes do projeto com os requisitos considerados aceitos, num dado momento do projeto de *software*.
REQM-T07	compromisso dos participantes do projeto	é um acordo livremente assumido, visível e esperado dos participantes envolvidos no projeto de *software*.

Identificação	Termo	Definição
REQM-T08	critério de aceite de requisitos	é uma regra que valida um requisito, de forma a ser aceito pelo usuário, cliente ou outra entidade autorizada. (fonte: CMM)
REQM-T09	critério de escolha de participantes da provisão de requisitos	é uma regra que define os participantes da provisão de requisitos.
REQM-T10	documentação da origem do requisito derivado	é o registro formal da identificação do requisito original do requisito derivado.
REQM-T11	funcionalidade do projeto de *software*	é uma qualidade ou aplicação funcional a ser implementada num *software* de forma que possibilite uma operação desejada ou necessária ao cliente ou usuário.
REQM-T12	gerente de projeto	é: (1) o papel com responsabilidade total do negócio de um projeto inteiro. É o indivíduo que dirige, controla, administra e regula um projeto de construção de um sistema de *hardware*/*software*. O gerente de projeto é o indivíduo responsável, em última análise, perante o cliente. (2) a pessoa responsável pelo planejamento, direção, controle, estruturação e motivação do projeto. O gerente do projeto é responsável pela satisfação do cliente. (fonte: CMMI)
REQM-T13	histórico da alteração de requisito de projeto de *software*	é o registro formal sobre as alterações de requisito de projeto de *software* que foram acordadas para os requisitos de projeto de *software*.
REQM-T14	impacto de um requisito de projeto de *software*	é o efeito que um requisito exerce nos compromissos assumidos, no andamento, nos produtos de trabalho envolvidos e nas tarefas do projeto. Pode significar re-trabalho e provocar re-planejamento do projeto.
REQM-T15	interface de funcionalidade do projeto de *software*	é o conjunto de elementos que possibilitam a conexão entre funcionalidades de projeto de *software* diferentes.

Identificação	**Termo**	**Definição**
REQM-T16	matriz de rastreamento de requisitos de projeto de *software*	é uma matriz que apresenta todos os mapeamentos, ou associação de um requisito para seus requisitos derivado, objetos, processos, funcionalidades, participantes e produtos de trabalho do processo de *software*.
REQM-T17	objeto do projeto de *software*	é uma instância de uma classe do projeto de *software*.
REQM-T18	participante da provisão de requisitos	é um representante do cliente ou usuário na explicitação das necessidades, desejos e expectativas dos mesmos em relação ao projeto.
REQM-T19	participante do projeto de *software*	é um indivíduo que é afetado ou é responsável pelo resultado de um projeto de *software*. Um participante pode incluir membros da organização de desenvolvimento do projeto, fornecedores, clientes, usuários finais e outros.
REQM-T20	participante relevante de projeto de *software*	é um participante identificado e com envolvimento e responsabilidade em atividades especificadas e incluídas num plano de projeto de *software*.
REQM-T21	processo do projeto de *software*	é uma seqüência de etapas realizadas para atender a um objetivo específico, por exemplo, o processo de desenvolvimento de *software*. (fonte: IEEE-STD-610)
REQM-T22	produto de trabalho do projeto de *software*	é um artefato qualquer criado como parte da definição, manutenção ou uso de um processo de *software*. Pode incluir descrições de processos, planos, procedimentos, programas de computador, documentação associada, as quais podem, ou não, ser entregues ao cliente ou ao usuário final. (fonte: CMM)
REQM-T23	projeto de *software*	é o esforço necessário que é focalizado na análise, especificação, design, desenvolvimento, teste e/ou manutenção de componentes de *software* e documentação associada de um sistema. Um projeto de *software* pode ser parte de um projeto de construção de um sistema de *hardware*/*software*. (fonte:CMM)
REQM-T24	racional da alteração de requisito de projeto de *software*	é a explicação ou justificativa formal e documentada sobre a necessidade ou causa da alteração do requisito de projeto de *software*.

Identificação	**Termo**	**Definição**
REQM-T25	rastreamento de um requisito de projeto de *software*	é (1) um mecanismo de acompanhamento do requisito que permite identificar sua origem ou seu desdobramento em outros requisitos, ou em outros produtos. (2) a evidência de uma associação entre um requisito e a fonte do requisito, sua implementação e sua verificação. Identifica a alocação do requisito ao *software*.
REQM-T26	rastreamento horizontal	é um rastreamento entre elementos de uma categoria em mesmo nível de abstração.
REQM-T27	registro da alteração do requisito de projeto de *software*	é a documentação formal da descrição da alteração do requisito de projeto de *software*.
REQM-T28	registro do requisito de projeto de *software*	é a documentação formal da descrição do requisito de projeto de *software*.
REQM-T29	requisito de projeto de *software*	é uma condição ou capacidade que deve ser atendida pelo *software*, necessária a um cliente ou usuário para resolver um problema ou alcançar um objetivo. (fonte: IEEE-STD-610)
REQM-T30	requisito derivado	é um requisito que não é explicitamente estabelecido nos requisitos do cliente, mas que é obtido por inferência de: (1) requisitos contextuais (p.ex. padrões aplicáveis, leis, políticas, práticas comuns e decisões gerenciais), ou (2) requisitos necessários para especificar um componente de produto. Requisitos derivados também podem surgir durante a análise e projeto de componentes de um produto ou sistema. (fonte: CMMI)

Anexo 4

Mapeamento de Regras Traduzidas em Português Estruturado às Atividades dos Processos Propostos

Código	Sentença	Tipo Sentença	Processo / Atividade
REQM-1.1.1-a	Um critério de escolha de participantes da provisão de requisitos (REQM-T09) **está relacionado a** um projeto de *software* (REQM-T23) **por** *'possui'* **com grau** 1**:**1.	Fato IX	**SP 1.1** **A1** – Estabelecer Critério Provedores Requisitos
REQM-1.1.1-b	Um critério de escolha de participantes da provisão de requisitos (REQM-T09) **deve obrigatoriamente** *ser estabelecido* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.1** **A1** – Estabelecer Critério Provedores Requisitos
REQM-1.1.2-a	Um critério de aceite de requisitos (REQM-T08) **está relacionado a** um projeto de *software* (REQM-T23) **por** *'possui'* **com grau** 1**:**1.	Fato IX	**SP 1.1** **A7** – Estabelecer Critério Aceite Requisitos
REQM-1.1.2-b	Um critério de aceite de requisitos (REQM-T08) **deve obrigatoriamente** *ser estabelecido* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.1** **A7** – Estabelecer Critério Aceite Requisitos
REQM-1.1.3-a	O critério de aceite de requisitos (REQM-T08) **deve obrigatoriamente** *ser cumprido por* um requisito do projeto de *software* (REQM-T29).	Restr I	**SP 1.1** **A10** – Requisito OK?
REQM-1.1.3-b	Um requisito de projeto de *software* (REQM-T29) **deve obrigatoriamente** *ser analisado* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.1** **A9** – Analisar Requisito
REQM-1.1.4-a	**Se** for estabelecido um acordo informal sobre os requisitos do projeto de *software* (REQM-T01) **então** o requisito de projeto de *software* (REQM-T29) **é considerado como** aceito.	Deriv I	**SP 1.1** **A11** – Estabelecer Acordo Informal
REQM-1.1.4-b	**Se** um requisito de projeto de *software* (REQM-T29) **é considerado** aceito, **então** o compromisso dos participantes do projeto (REQM-T07) **é considerado como** estabelecido.	Deriv I	**SP 1.1** **A11** – Estabelecer Acordo Informal
REQM-1.2.1-a	O impacto de um requisito de projeto de *software* (REQM-T14) **deve obrigatoriamente** *ser avaliado* em relação aos compromissos de projeto de *software* existentes (REQM-T06).	Restr I	**SP 1.2** **B3** – Avaliar Impacto Compromissos

Código	**Sentença**	**Tipo Sentença**	**Processo / Atividade**
REQM-1.2.2-a	Os compromissos do projeto de *software* existentes (REQM-T06) **devem obrigatoriamente** *ser registrados* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.2** **B7** – Registrar Compromisso
REQM-1.2.2-b	Os compromissos do projeto de *software* (REQM-T06) **devem obrigatoriamente** *ser negociados* pelos participantes do projeto de *software* (REQM-T19).	Restr I	**SP 1.2** **B4** – Negociar Compromisso
REQM-1.3.1-a	Um requisito de projeto de *software* (REQM-T29) **está relacionado a** um projeto de *software* (REQM-T23) **por** ***'possui'*** **com grau** 1..N**:**1.	Fato IX	**SP 1.3** **C1** – Fornecer Requisito **C3** – Gerar Requisito **C7** – Registrar Requisito
REQM-1.3.1-b	Um projeto de *software* (REQM-T23) **está relacionado a** um requisito de projeto de *software* (REQM-T29) **por** 'gerou' **com grau** 1**:**0..N.	Fato IX	**SP 1.3** **C3** – Gerar Requisito
REQM-1.3.1-c	O requisito de projeto de *software* (REQM-T29) **deve obrigatoriamente** *ser capturado* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.3** **C5** – Capturar Requisito
REQM-1.3.1-d	O requisito de projeto de *software* (REQM-T29) **deve obrigatoriamente** *ser registrado* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.3** **C7** – Registrar Requisito
REQM-1.3.1-e	Um projeto de *software* (REQM-T23) **está relacionado a** uma alteração de requisito de projeto de *software* (REQM-T02) **por** *'recebeu'* **com grau** 1**:**0..N.	Fato IX	**SP 1.3** **C2** – Fornecer Alteração Requisito **C4** – Gerar Alteração Requisito
REQM-1.3.1-f	Um projeto de *software* (REQM-T23) **está relacionado a** uma alteração de requisito de projeto de *software* (REQM-T02) **por** *'gerou'* **com grau** 1**:**0..N.	Fato IX	**SP 1.3** **C4** – Gerar Alteração Requisito
REQM-1.3.1-g	Uma alteração de requisito de projeto de *software* (REQM-T02) **deve obrigatoriamente** *ser capturada* por um projeto de *software* (REQM-T23).	Restr I	**SP 1.3** **C6** – Capturar Alteração Requisito
REQM-1.3.1-h	A alteração de requisito de projeto de *software* (REQM-T02) **deve obrigatoriamente** *ser registrada* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.3** **C10** – Registrar Alteração Requisito
REQM-1.3.2-a	O racional das alterações de requisitos de projeto de *software* (REQM-T24) **deve obrigatoriamente** *ser registrado* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.3** **C11** – Registrar Racional/Histórico Alteração Requisito

Código	**Sentença**	**Tipo Sentença**	**Processo / Atividade**
REQM-1.3.2-b	O histórico das alterações de requisitos de projeto de *software* (REQM-T13) **deve obrigatoriamente** *ser registrada* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.3** **C11** – Registrar Racional/Histórico Alteração Requisito
REQM-1.3.3-a	A análise de impacto das alterações de requisitos de projeto de *software* (REQM-T05) **deve obrigatoriamente** *ser realizada* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.3** **C8** – Avaliar Impacto Alteração Requisito **C9** – Alteração Requisito OK?
REQM-1.3.4-a	Um projeto de *software* (REQM-T23) **está relacionado ao** registro do requisito de projeto de *software* (REQM-T28) **por '***possui'* **com grau** 1**:**1..N.	Fato IX	**SP 1.3** **C7** – Registrar Requisito
REQM-1.3.4-b	O registro do requisito de projeto de *software* (REQM-T28) **deve obrigatoriamente** *ser realizado* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.3** **C7** – Registrar Requisito
REQM-1.3.4-c	Os participantes do projeto (REQM-T19) **têm permissão para** *consultar* o registro do requisito de projeto de *software* (REQM-T28).	Fato I	**SP 1.3** **C7** – Registrar Requisito
REQM-1.3.4-d	Um projeto de *software* (REQM-T23) **está relacionado a** uma alteração do requisito de projeto de *software* (REQM-T27) **por '***possui'* **com grau** 1**:**0..N.	Fato IX	**SP 1.3** **C10 –** Registrar Alteração Requisito
REQM-1.3.4-e	O registro da alteração do requisito de projeto de *software* (REQM-T27) **deve obrigatoriamente** *ser realizado* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.3** **C10 –** Registrar Alteração Requisito
REQM-1.3.4-f	Os participantes do projeto (REQM-T19) **têm permissão para** *consultar* o registro da alteração do requisito de projeto de *software* (REQM-T27).	Fato I	**SP 1.3** **C10** – Registrar Alteração Requisito
REQM-1.4.1-a	Um rastreamento de requisitos de projeto de *software* (REQM-T25) **deve obrigatoriamente** ser realizado pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.4** **D2** – Rastrear Origem Requisito Derivado
REQM-1.4.1-b	O rastreamento de requisitos de projeto de *software* (REQM-T25) **deve obrigatoriamente** *ser mantido atualizado* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.4** **D2** – Rastrear Origem Requisito Derivado
REQM-1.4.1-c	A documentação da origem de um requisito derivado (REQM-T10)) **deve obrigatoriamente** *ser garantida* pelo rastreamento dos requisitos do projeto de *software* (REQM-T25).	Restr I	**SP 1.4** **D2** – Rastrear Origem Requisito Derivado
REQM-1.4.2-a	Um requisito de projeto de *software* (REQM-T29) **está relacionado a** um requisito derivado (REQM-T30) **por '***desdobrou-se'* **com grau** 1:0..N.	Fato IX	**SP 1.4** **D1** – Desdobrar Requisito

Código	**Sentença**	**Tipo Sentença**	**Processo / Atividade**
REQM-1.4.2-b	Um requisito de projeto de *software* (REQM-T29) **está relacionado a** seus requisitos derivados (REQM-T30) **por** *'é implementado por'* **com grau** 1**:**0..N.	Fato IX	**SP 1.4** **D1** – Desdobrar Requisito
REQM-1.4.2-c	Um requisito de projeto de *software* (REQM-T29) **está relacionado a** sua alocação às funcionalidades do projeto de *software* (REQM-T04) **por** *'é implementado por'* **com grau** 1**:**1..N.	Fato IX	**SP 1.4** **D3** – Rastrear Funcionalidades Requisito
REQM-1.4.2-d	Um requisito de projeto de *software* (REQM-T29) **está relacionado à** sua alocação aos objetos do projeto de *software* (REQM-T03) **por** *'é implementado por'* **com grau** 1..M**:**1..N.	Fato IX	**SP 1.4** **D4** – Rastrear Objetos Requisito
REQM-1.4.2-e	Um requisito de projeto de *software* (REQM-T29) **está relacionado aos** participantes relevantes do projeto de *software* (REQM-T20) **por** *'é implementado por'* **com grau** 1..M**:**1..N.	Fato IX	**SP 1.4** **D5** – Rastrear Participantes Responsáveis Requisito
REQM-1.4.2-f	Um requisito de projeto de *software* (REQM-T29) **está relacionado aos** processos do projeto de *software* (REQM-T21) **por** *'é implementado por'* **com grau** 1..M**:**1..N.	Fato IX	**SP 1.4** **D6** – Rastrear Processos Requisito
REQM-1.4.2-g	Um requisito de projeto de *software* (REQM-T29) **está relacionado aos** produtos de trabalho do projeto de *software* (REQM-T22) **por** *'é implementado por'* **com grau** 1..M**:**1..N.	Fato IX	**SP 1.4** **D7** – Rastrear Produtos Trabalho Requisito
REQM-1.4.3-a	Uma funcionalidade do projeto de *software* (REQM-T11) **está relacionada às** funcionalidades do projeto de *software* relacionadas (REQM-T11) **por** *'é implementada por'* **com grau** 1.M**:**0..N.	Fato IX	**SP 1.4** **D8** – Rastrear Funcionalidades Horizontais
REQM-1.4.3-b	Uma funcionalidade do projeto de *software* (REQM-T11) **está relacionada às** suas interfaces (REQM-T15) **por** *'é implementada por'* **com grau** 1.M**:**0..N.	Fato IX	**SP 1.4** **D8** – Rastrear Funcionalidades Horizontais
REQM-1.4.3-c	Um rastreamento horizontal de funcionalidades de projeto de *software* (REQM-T26) **deve obrigatoriamente** *ser realizada* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.4** **D8** – Rastrear Funcionalidades Horizontais
REQM-1.4.4-a	Um projeto de *software* (REQM-T23) **está relacionado a** uma matriz de rastreamento de requisitos de projeto de *software* (REQM-T16) **por** *'possui'* **com grau** 1**:**1.	Fato IX	**SP 1.4** **MRR** – Matriz de Rastreamento de Requisitos
REQM-1.4.4-b	Uma matriz de rastreamento de requisitos de projeto de *software* (REQM-T16) **deve obrigatoriamente** *ser gerada* pelo gerente de projeto (REQM-T12).	Restr I	**SP 1.4** **MRR** – Matriz de Rastreamento de Requisitos

Anexo 5

Mapeamento dos Produtos Típicos de Trabalho das Práticas Específicas da Área de Processo de Gerência de Requisitos do CMMI aos Itens dos Documentos Propostos

REQM Prática Específica	**Produto Típico de Trabalho**	**Documento**	**Item do Documento**
SP 1.1	Lista do critério de escolha dos provedores de requisitos adequados	DER	**Item 2.1** Critério de Escolha de Provedores de Requisitos Vide Documento no Anexo 7
	Critério para a avaliação e aceite de requisitos	DER	**Item 3.1** Critério de Aceite de Requisitos Vide Documento no Anexo 7
	Resultados da análise do requisito contra o critério de aceite	DER	**Item 3.2** Relação de Requisitos Aceitos **Item 3.3** Relação de Requisitos Reprovados Vide Documento no Anexo 7
	Conjunto de requisitos acordados	DER	**Item 3.2** Relação de Requisitos Aceitos Vide Documento no Anexo 7
SP 1.2	Verificação de impacto do requisito	Documentos de outras Áreas de Processo (PP - Planejamento de Projeto)	
	Compromissos documentados aos requisitos e às alterações de requisitos		
SP 1.3	Status do Requisito	MRR	**Item 1** Rastreamento de Requisitos – coluna Status
	Base de Dados de Requisitos	-	(não se aplica)
	Base de Dados de Decisão de Requisitos	-	(não se aplica)
SP 1.4	Matriz de Rastreamento de Requisitos	MRR	Vide Documento no Anexo 8
	Sistema de Acompanhamento de Requisitos	-	(não se aplica)

Anexo 6

Mapeamento das Atividades dos Processos Propostos

Processo	Atividade	Atividade Antecessora	Atividade Sucessora	Entradas	Saídas
SP 1.1	**A1** – Estabelecer Critério Provedores Requisitos	Início SP 1.1	**A2** – Avaliar Critério Provedores Requisitos	-	REQM-**T09**
SP 1.1	**A2** – Avaliar Critério Provedores Requisitos	**A1** – Estabelecer Critério Provedores Requisitos	**A3** – Critério Provedores Requisitos OK?	REQM-**T09**	REQM-**T09**
SP 1.1	**A3** – Critério Provedores Requisitos OK?	**A2** – Avaliar Critério Provedores Requisitos	Sim: **A4** – Fornecer Provedores Requisitos	REQM-**T09**	REQM-**T09** **Aceito**
SP 1.1	**A3** – Critério Provedores Requisitos OK?	**A2** – Avaliar Critério Provedores Requisitos	Não: **A1** – Estabelecer Critério Provedores Requisitos	REQM-**T09**	REQM-**T09** **Rejeitado**
SP 1.1	**A4** – Fornecer Provedores Requisitos	**A3** – Critério Provedores Requisitos OK?	**A5** – Aplicar Critério Provedores Requisitos	REQM-**T09** **Aceito**	REQM-**T18**
SP 1.1	**A5** – Aplicar Critério Provedores Requisitos	**A4** – Fornecer Provedores Requisitos	**A6** – Provedores Requisitos OK?	REQM-**T18**	REQM-**T09** REQM-**T18**
SP 1.1	**A6** – Provedores Requisitos OK?	**A5** – Aplicar Critério Provedores Requisitos	Sim: **A7** – Estabelecer Critério Aceite Requisito	REQM-**T09** REQM-**T18**	REQM-**T18** **Aceito**
SP 1.1	**A6** – Provedores Requisitos OK?	**A5** – Aplicar Critério Provedores Requisitos	Não: **A4** – Fornecer Provedores Requisitos	REQM-**T09** REQM-**T18**	REQM-**T18** **Rejeitado**
SP 1.1	**A7** – Estabelecer Critério Aceite Requisito	**A6** – Provedores Requisitos OK?	**A8** – Fornecer Requisito	REQM-**T18** **Aceito**	REQM-**T08**
SP 1.1	**A8** – Fornecer Requisito	**A7** – Estabelecer Critério Aceite Requisito	**A9** – Analisar Requisito	REQM-**T08**	REQM-**T08** REQM-**T29**
SP 1.1	**A9** – Analisar Requisito	**A8** – Fornecer Requisito	**A10 –** Requisito OK?	REQM-**T08** REQM-**T29**	REQM-**T29** **Analisado**
SP 1.1	**A10 –** Requisito OK?	**A9** – Analisar Requisito	Sim: **A11 –** Estabelecer Acordo Informal	REQM-**T29** **Analisado**	REQM-**T29** **Aceito**

Processo	**Atividade**	**Atividade Antecessora**	**Atividade Sucessora**	**Entradas**	**Saídas**
SP 1.1	**A10 –** Requisito OK?	**A9** – Analisar Requisito	Não: **A8 –** Fornecer Requisito	REQM-**T29 Analisado**	REQM-**T29 Rejeitado**
SP 1.1	**A11 –** Estabelecer Acordo Informal	**A10 –** Requisito OK?	Fim SP 1.1	REQM-**T29 Aceito**	REQM-**T29 Aceito** REQM-**T01**
SP 1.2	**B1 –** Fornecer Novo Requisito	Início SP 1.2	**B3** – Avaliar Impacto Compromissos	-	REQM-**T29 Aceito**
SP 1.2	**B2** – Fornecer Alteração Requisito	Início SP 1.2	**B3** – Avaliar Impacto Compromissos	-	REQM-**T02**
SP 1.2	**B3** – Avaliar Impacto Compromissos	Junção de: **B1 –** Fornecer Novo Requisito **B2** – Fornecer Alteração Requisito	**B4 –** Negociar Compromissos	REQM-**T29 Aceito** REQM-**T02** REQM-**T07** REQM-**T01**	REQM-**T05**
SP 1.2	**B4 –** Negociar Compromissos	**B3** – Avaliar Impacto Compromissos	Junção de: **B5 –** Documentar Alteração Requisito **B6** – Documentar Requisito **B7** – Registrar Compromisso	REQM-**T05**	REQM-**T01** REQM-**T05**
SP 1.2	**B5 –** Documentar Alteração Requisito	**B4 –** Negociar Compromissos	**B8** – Documentar Racional/Histórico Alteração Requisito	REQM-**T02**	**DER** REQM-**T27**
SP 1.2	**B6** – Documentar Requisito	**B4 –** Negociar Compromissos	Fim SP 1.2	REQM-**T29**	**DER** REQM-**T28**
SP 1.2	**B7** – Registrar Compromisso	**B4 –** Negociar Compromissos	Fim SP 1.2	REQM-**T06**	REQM-**T07**
SP 1.2	**B8** – Documentar Racional/Histórico Alteração Requisito	**B5 –** Documentar Alteração Requisito	Fim SP 1.2	REQM-**T02**	**DER** REQM-**T24** REQM-**T13**

Processo	**Atividade**	**Atividade Antecessora**	**Atividade Sucessora**	**Entradas**	**Saídas**
SP 1.3	**C1 –** Fornecer Requisito	Início SP 1.3	Junção de: **C5** – Capturar Requisito **C6** – Capturar Alteração Requisito	-	REQM-**T29**
SP 1.3	**C2 –** Fornecer Alteração Requisito	Início SP 1.3	Junção de: **C5** – Capturar Requisito **C6** – Capturar Alteração Requisito	-	REQM-**T02**
SP 1.3	**C3** – Gerar Requisito	Início SP 1.3	Junção de: **C5** – Capturar Requisito **C6** – Capturar Alteração Requisito	-	REQM-**T29**
SP 1.3	**C4** – Gerar Alteração Requisito	Início SP 1.3	Junção de: **C5** – Capturar Requisito **C6** – Capturar Alteração Requisito	-	REQM-**T02**
SP 1.3	**C5** – Capturar Requisito	Junção de: **C1 –** Fornecer Requisito **C2 –** Fornecer Alteração Requisito **C3** – Gerar Requisito **C4** – Gerar Alteração Requisito	**C7** – Registrar Requisito	**DER** REQM-**T29**	REQM-**T29**
SP 1.3	**C6** – Capturar Alteração Requisito	Junção de: **C1 –** Fornecer Requisito **C2 –** Fornecer Alteração Requisito **C3** – Gerar Requisito **C4** – Gerar Alteração Requisito	**C8** – Avaliar Impacto Alteração Requisito	REQM-**T02**	REQM-**T02**
SP 1.3	**C7** – Registrar Requisito	**C5** – Capturar Requisito	Fim SP 1.3	REQM-**T29**	REQM-**T28**
SP 1.3	**C8** – Avaliar Impacto Alteração Requisito	**C6** – Capturar Alteração Requisito	**C9** – Alteração Requisito OK?	REQM-**T02**	REQM-**T05**

Processo	**Atividade**	**Atividade Antecessora**	**Atividade Sucessora**	**Entradas**	**Saídas**
SP 1.3	**C9** – Alteração Requisito OK?	**C8** – Avaliar Impacto Alteração Requisito	Sim: **C10** – Registrar Alteração Requisito	REQM-**T02**	REQM-**T02** **Aceito**
SP 1.3	**C9** – Alteração Requisito OK?	**C8** – Avaliar Impacto Alteração Requisito	Não: Fim SP 1.3	REQM-**T02**	REQM-**T02** **Rejeitado**
SP 1.3	**C10** – Registrar Alteração Requisito	**C9** – Alteração Requisito OK?	Fim SP 1.3	REQM-**T02**	REQM-**T27**
SP 1.3	**C11** – Registrar Racional/Histórico Alteração Requisito	**C9** – Alteração Requisito OK?	Fim SP 1.3	REQM-**T24** REQM-**T13**	REQM-**T24** REQM-**T13**
SP 1.4	**D1** – Desdobrar Requisito	Início SP 1.4	**D2** – Rastrear Origem Requisito Derivado	**DER** REQM-**T29**	**MRR** REQM-**T29** REQM-**T30**
SP 1.4	**D2** – Rastrear Origem Requisito Derivado	**D1** – Desdobrar Requisito	**D3** – Rastrear Funcionalidades Requisito	**DER** REQM-**T29**	**MRR** REQM-**T29** REQM-**T10**
SP 1.4	**D3** – Rastrear Funcionalidades Requisito	**D2** – Rastrear Origem Requisito Derivado	**D4** – Rastrear Objetos Requisito	**DER** REQM-**T29**	**MRR** REQM-**T29** REQM-**T04**
SP 1.4	**D4** – Rastrear Objetos Requisito	**D3** – Rastrear Funcionalidades Requisito	**D5** – Rastrear Participantes Responsáveis Requisito	**DER** REQM-**T29**	**MRR** REQM-**T29** REQM-**T03**
SP 1.4	**D5** – Rastrear Participantes Responsáveis Requisito	**D4** – Rastrear Objetos Requisito	**D6** – Rastrear Processos Requisito	**DER** REQM-**T29**	**MRR** REQM-**T29** REQM-**T20**
SP 1.4	**D6** – Rastrear Processos Requisito	**D5** – Rastrear Participantes Responsáveis Requisito	**D7** – Rastrear Produtos Trabalho Requisito	**DER** REQM-**T29**	**MRR** REQM-**T29** REQM-**T21**
SP 1.4	**D7** – Rastrear Produtos Trabalho Requisito	**D6** – Rastrear Processos Requisito	**D8** – Rastrear Funcionalidades Horizontais	**DER** REQM-**T29**	**MRR** REQM-**T29** REQM-**T22**

Processo	**Atividade**	**Atividade Antecessora**	**Atividade Sucessora**	**Entradas**	**Saídas**
SP 1.4	**D8** – Rastrear Funcionalidades Horizontais	**D7** – Rastrear Produtos Trabalho Requisito	Fim SP 1.4	**DER** REQM-**T29**	**MRR** REQM-**T26**

Anexo 7

DER – Documento de Especificação de Requisitos

DER - DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS

Projeto:

1. Dados do Projeto

1.1. Descrição

1.2. Objetivos

1.3. Benefícios

1.4. Restrições

2. Participantes

2.1. Critério de Escolha de Provedores de Requisitos

2.2. Relação de Participantes Aprovados X Responsabilidades

Área	Nome	Responsabilidade		
		Prov Req	Testes	Aprov

2.3. Relação de Participantes Reprovados – Racional

Área	Nome	Racional de Reprovação

DER - DOCUMENTO DE ESPECIFICAÇÃO DE REQUISITOS

Projeto:

3. Requisitos

3.1. Critério de Aceite de Requisitos

3.2. Relação de Requisitos Aceitos

Identif	Título e Descrição do Requisito / Racional da Alteração	Origem	Áreas Envolvidas	Datas	
				Inclusão	Acomp.

3.3. Relação de Requisitos Reprovados

Identif	Título e Descrição do Requisito	Origem	Racional de Reprovação	Datas	
				Inclusão	Reprov

4. Controle de Versão

Versão	Data	Descrição	Área	Responsável

5. Aprovações

Área	Nome	Assinatura	Data
		_____	___/___/___
		_____	___/___/____

Anexo 8

MRR – Matriz de Rastreamento de Requisitos

MRR - MATRIZ DE RASTREAMENTO DE REQUISITOS

Projeto:

1. Rastreamento de Requisitos

Identif	Título do Requisito	Status	Datas		Identif Requisito Derivado	Funcionalidade	Processo	Participante Responsável	Objeto	Produto de Trabalho
			Inclusão	Acomp.						

2. Rastreamento Horizontal de Funcionalidades / Interfaces

Funcionalidade A	Funcionalidade B	Interface

MRR - MATRIZ DE RASTREAMENTO DE REQUISITOS

Projeto:

3. Controle de Versão

Versão	Data	Descrição	Área	Responsável

4. Aprovações

Área	Nome	Assinatura	Data
		_____	____/____/____
		_____	____/____/____
		_____	____/____/____