2 FUNDAMENTOS

2.1 Requisitos

Apresentaremos neste capítulo alguns conceitos fundamentais para esta monografia que estão relacionados com o conceito de requisitos e engenharia de requisitos. Dentre os conceitos estão os de: requisitos, análise de negócio, léxico ampliado da linguagem, RNF - Framework, SQFD etc. Segundo o Swebok (2004), um requisito de software é uma propriedade que deve ser exibida, a fim de resolver algum problema no mundo real. O Guia refere-se a requisitos de "software", porque é preocupado com os problemas a serem tratados pelo software. A seguir veremos os tipos e alguns exemplos de cada tipo.

2.1.1 Tipos de Requisitos

Os tipos de requisitos que são considerados na literatura de engenharia de software são os requisitos: funcionais, não-funcionais, e de domínio.

Os requisitos funcionais são aqueles que representam o que o software deve fazer. São exemplos de requisitos funcionais, (ÁVILA; SPÍNOLA, 2007):

O software deve permitir o cadastro de clientes;

O software deve permitir a geração de relatórios sobre o desempenho de vendas no semestre;

O software deve permitir o pagamento das compras através de cartão de crédito;

Os requisitos não-funcionais dizem respeito à qualidade com que o software desempenha suas funcionalidades. Esses requisitos colocam restrições no sistema, ou seja expressam condições que o software deve atender. Spínola e Ávila citam exemplos:

“O software deve ser compatível com os browsers IE (versão 5.0 ou superior) e firefox(1.0 ou superior)”;

“o software deve garantir que o tempo de retorno das consultas não seja maior que 5 segundos”;

Já os de domínio têm origem no domínio da aplicação. Descrevem as características da aplicação e qualidades que expressão o domínio. Podem, segundo Ávila e Spínola, ser requisitos funcionais novos, restrições sobre requisitos, ou computações especificas. Exemplos:

- "O cálculo da média final de cada aluno é dado pela formula: (Nota1 *2 + Nota2 * 3) / 5";
- "Um aluno pode se matricular em uma disciplina desde que ele tenha sido aprovado nas disciplinas consideradas pré-requisitos";

O projeto que não atende aos requisitos dos clientes e aos requisitos de processo[1] muito provavelmente irá fracassar. Existem muitos desafios quando se trata de requisito, portanto é necessário que haja uma abordagem sistemática dos requisitos para que se diminua os riscos que os requisitos representam.

2.2 Engenharia de Requisitos

Segundo Soares, o processo de engenharia de requisitos é composto por quatro atividades de alto nível (Soares apud Wikipédia, 2010):

- Identificação.
- Análise e negociação.
- Especificação e documentação.
- Validação.

Outra atividade que pode ser adicionada a esse processo é a gestão dos requisitos (manutenção ou change manegement).

Segundo Ávila e Spínola (2007, p. 49), o termo Engenharia de requisitos pode ser definida como: ”termo usado para descrever as atividades relacionadas à produção (levantamento, registro, validação e verificação) e gerência (controle de mudanças, gerência de configuração, rastreabilidade, gerência de qualidade dos requisitos) de requisitos”.

A Figura 2.1 representa essa definição.

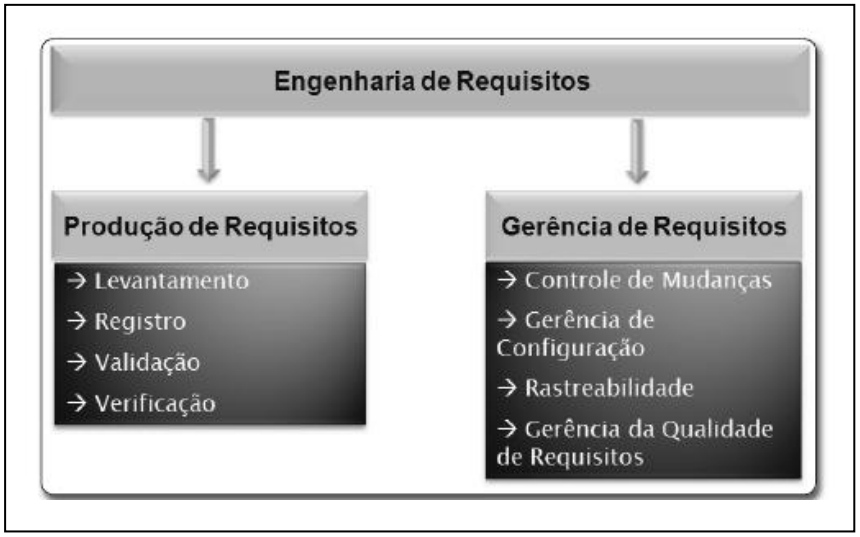


Figura 2.1: Engenharia de Requisitos

Fonte: Ávila e Spínola (2007, p. 48)

2.2.1 Produção de Requisitos

Quatro atividades são consideradas atividades base relacionadas com a Produção de Requisitos: levantamento, registro, verificação e validação.

O *levantamento* está relacionado com a obtenção dos requisitos do software. Nesta atividades os “engenheiros e analistas trabalham com clientes e usuários finais para descobrir o problemas a ser resolvido, os serviços do sistema, o desempenho necessário, restrições de hardware e outras informações” (ÁVILA; SPÍNOLA, 2007, p. 49).

É na atividade de *registro* que ocorre a documentação dos requisitos. Essa

documentação servirá de base para as outras fases do processo de desenvolvimento. Geralmente produz-se um documento de especificação de requisitos.

Na atividade de *verificação* ocorre o exame da especificação de requisitos do software, o objetivo é encontrar e corrigir ambiguidades, inconsistências ou omissões antes que a fase de definição de requisitos acabe.

Por fim, segundo Ávila e Spínola, na atividade de validação o objetivo é conseguir o aceite do cliente sob determinado artefato. Na engenharia de requisitos validar é aprovar requisitos de software.

2.2.2 Gerência de Requisitos

A "atividade de administrar os requisitos ao longo do tempo" (SPÍNOLA; ÁVILA, 2007, p. 57) é denominada gerencia de requisitos.

Segundo Sayão e Leite (2005) existe uma confusão quando se trata de conceitos de Gerencia "dos" requisitos e Gerência "por" requisitos. Na visão de Sayão e Leite, Gerência dos Requisitos está associada ao acompanhamento da evolução dos requisitos ao longo do processo de desenvolvimento (manter registro do status do requisito em relação ao desenvolvimento, modificações aceitas e *rationale* associado). E a Gerência por requisitos (geralmente chamada de Gerência de Requisitos) está associado ao controle de todo o processo de desenvolvimento tendo como referência a *baseline* de requisitos.

Ávila e Spínola (2007) citam atividade de gerência de requisitos que devem ser levadas em consideração: *Controle de Mudanças*, *Gerência de Configuração*, *Rastreabilidade* e *Gerência de Qualidade de Requisitos*.

Controle de Mudanças

As mudanças nos requisitos ocorrem e é "praticamente inevitável" que elas ocorram. Portanto, deve-se sempre trabalhar com a idéia de que mudanças nos requisitos irão ocorrer e que isto é natural. Obviamente que limites devem ser estabelecidos e que deve haver controle sobre tais mudanças, sob o risco de que, se não houver, o projeto possa fracassar.

A “baseline” (uma prática bastante usada) permite que se possa diferenciar o que era o requisito original, o que foi introduzido e o que foi descartado. O uso de sistemas para fazer este controle provê mais exatidão à baseline e rapidez aos manipuladores das baselines.

O estabelecimento de processo de controle de mudanças também é importante. Uma atividade importante é a analise das solicitações de mudanças para que se possa aceitar ou rejeitar a mudança.

(SPÍNOLA; ÁVILA, 2007) sugerem passos a serem seguidos para um processo de controle de mudança:

- “Checar validade da solicitação de mudança”;
- “Identificar os requisitos diretamente afetados com a mudança”;
- “Identificar dependências entre requisitos para buscar os requisitos afetados indiretamente”;
- “Assegurar com solicitante a mudança a ser realizada”;
- “Estimar custos da mudança”;
- “Obter acordo com usuário sobre o custo da mudança”.

Esta atividade esta muito ligada à próxima atividade, gerência de configuração e se não funcionarem corretamente tanto inviabilizarão as negociações como poderão fazer com que o projeto fracasse.

Gerência de Configuração

A atividade de Gerência de Configuração (GC), segundo Presman:

is a set of activities designed to control change by identifying the work products that are likely to change, establishing relationships among them, defining mechanisms for managing different versions of these work products, controlling the changes imposed, and auditing and reporting on the changes made. (PRESMAN, 2001, p. 225).

Dentre os benefícios da Gerência de Configuração podemos citar: Redução de defeitos; Diminuição de custos para manutenção; Melhora no gerenciamento do projeto (rastreabilidade, reproducibilidade; confiabilidade; visibilidade das mudanças); Melhora na produtividade do desenvolvimento; Melhor retorno do investimento etc. Podemos citar como objetivos da Gerência de Configuração:

- Definir políticas para controle de versões, garantindo a consistência dos artefatos produzidos;
- Definir procedimentos para solicitações de mudanças;
- Administrar e auditar o ambiente de GC;
- Facilitar e automatizar a geração de build do sistema;
- Facilitar a geração de Release.

Percebe-se aqui o quão importante é a gerência de configuração para a gerência de projetos de software e engenharia de requisitos. Oportunamente, ela é muito importante para as negociações no processo de desenvolvimento do software, pois provê agilidade no trato dos itens de configuração ao mesmo tempo em que trás formalidade. E isto fortalece a comunicação e, consequentemente, o relacionamento entre os stakeholders do projeto.

Rastreabilidade

A rastreabilidade, tida como centro das atividades de gerenciamento de requisitos, pode ser entendida como "habilidade de se acompanhar a vida de um requisito em ambas as direções (por exemplo: partindo de requisitos e chegando ao projeto ou, partindo do projeto e chegando aos requisitos) do processo de software durante todo o seu ciclo de vida"(ÁVILA ; SPÍNOLA, 2007, p. 52).

Dependendo do sentido das ligações ou "rastros", pode-se fazer uso dos termos "pré-rastreabilidade" ou "pós-rastreabilidade". A pré-rastreabilidade associa os requisitos ao contexto a partir do qual eles surgiram. A pós-rastreabilidade vincula os requisitos ao desenho do sistema e sua implementação (SAYÃO; LEITE, 2007) (DAVIS, 1993).

A figura 2.2 abaixo, extraída de Sayão e Leite (2007), demonstra como as ligações possibilitam acompanhar a vida de um requisito:

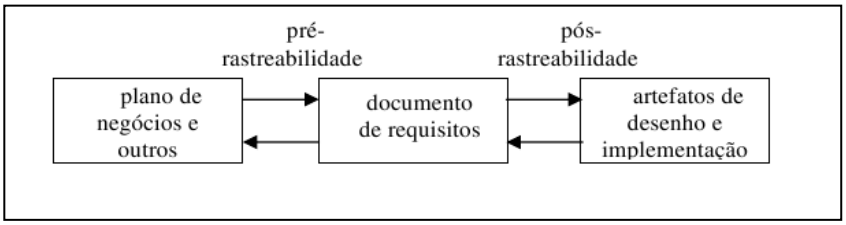


Figura 2.2: Rastreabilidade de Requisitos

Fonte: Sayão e Leite (2005, p.6).

Sayão e Leite comentam a figura:

No primeiro tipo temos a rastreabilidade forward-to (para frente), que liga documentos obtidos no processo de elicitação a requisitos relevantes, e a rastreabilidade backward-from (para trás), que liga requisitos às suas fontes. No segundo tipo temos rastreabilidade forward-from, que liga requisitos a artefatos de desenho e implementação e rastreabilidade backward-to, que liga artefatos de desenho e implementação de volta a requisitos (SAYÃO; LEITE, 2007, p. 7).

Vista como um ponto crítico e difícil de ser realizado, a rastreabilidade geralmente é feita com a ajuda de ferramentas adequadas para que se torne viável. Algumas ferramentas são citadas em Sayão e Leite como:

- C&L;
- RequisitPro® e;
- DOORS;

Sayão e Leite também mencionam algumas técnicas e classificações.

Uma possível classificação para as técnicas de rastreabilidade mais comuns está relacionada

ao foco principal: referências cruzadas ou documentos. Técnicas centradas em referências cruzadas são simples de entender e utilizar e podem ser suportadas pelo uso de hipertexto, esquemas de numeração, indexação ou uso de tags ou ainda por matrizes de rastreabilidade. Técnicas centradas em documentos utilizam modelos (templates) para documentos e artefatos utilizados no processo de desenvolvimento e incorporam mecanismos para transformação e integração de documentos (SAYÃO; LEITE, 2007, p. 17).

Não iremos abordar mais detalhes sobre os tipos de técnicas e classificações que podem ser visto em (SAYÃO; LEITE , 2007).

Gerência de Qualidade dos Requisitos

A gerência de requisitos provê uma infra-estrutura necessária às atividades de verificação, o que possibilita a investigação da qualidade dos requisitos.

O padrão IEEE 830 citam alguns critérios que devemos considerar como: correção, não-ambiguidade, completude, consistência, verificabilidade, modificabilidade (SPÍNOLA; ÁVILA, 2007).

(COUTO; MARTINS, 2008) apresentam um processo para a validação de requisitos não funcionais baseados no framework RNF-framework, que será comentado mais adiante. Este processo pode ser visto como uma forma de gerência de qualidade para requisitos não funcionais, pois visa atender padrões de qualidade baseados, evitando problemas como incompletude, conflitos e ambigüidades, entre outros problemas.

Falhas nestas atividades podem prejudicar bastante as negociações. Pois negociadores estariam utilizando requisitos problemáticos (incorretos, ou desatualizados, ou ambíguos, ou incompletos, ou inconsistentes ou inviáveis – em termos de custo e tempo) para manter suas posições, tomar suas decisões e embasar seus argumentos.

Como os negociadores precisam tratar de assuntos substantivos, assuntos relacionados aos requisitos, se houver algum problema no tocante aos critérios acima citados, os negociadores perderiam tempo em negociações improdutivas e baseadas em dados sem qualidade.

2.3 O léxico ampliado da linguagem

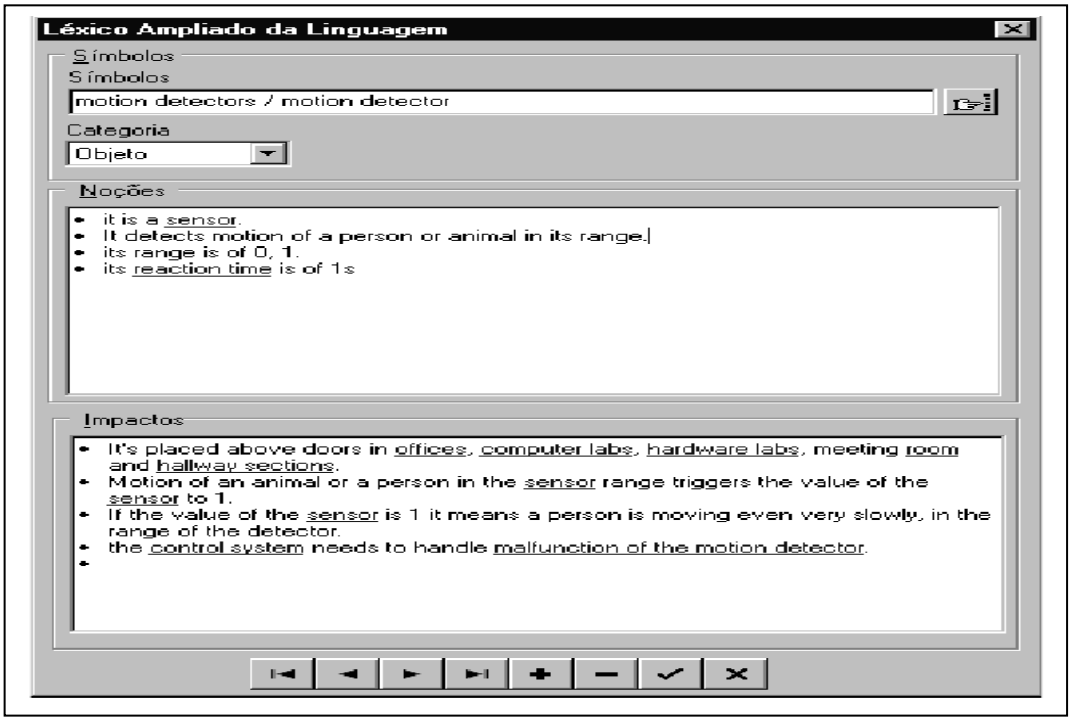
O conceito de *léxico*, de uma forma geral, é o acervo de palavras de um determinado idioma. No contexto do ciclo de desenvolvimento do software e organização para a qual o software será ou esta sendo desenvolvido, o "léxico" pode ser entendido como sendo o acervo de palavras do "idioma de uma organização". O Léxico Ampliado da Linguagem tem como objetivo registrar os termos e não se preocupar com a funcionalidade do Universo De Informações (UDI). De forma resumida o Léxico Ampliado da Linguagem (LAL) do UDI é o conjunto de termos praticados pelos atores do UDI.

Sua composição e feita de "entradas" relacionadas a "símbolos da linguagem do UDI". Cada entrada é associada a um símbolo do UDI. Os símbolos possuem "sinônimos" e são descritos por "noções" e "impactos". As noções expressam a denotação do símbolo, ou seja, o significado mais as relações fundamentais de existência do símbolo com outros símbolos. E os impactos demonstram a conotação, ou seja expressa os efeitos do uso, ou ocorrência do símbolo ou efeitos de outras ocorrências de símbolos no UDI.

As Entradas do LAL descrevem símbolos do UDI e dependendo do símbolo que elas descrevem elas podem ser classificadas como sujeito, verbo, objeto e estado (predicativo). Ao se descrever entradas do LAL deve-se seguir aos princípios de "vocabulário mínimo" (minimização do uso de símbolos externos ao UDI) e "circularidade" (maximização do uso de símbolos do UDI para descrever símbolos do UDI.

A forma natural de representação do LAL é o **hipertexto.** As entradas são os nós do hipertexto e os símbolos que aparecem nas descrições de símbolos são os "elos" do hipertexto.

Extraímos a figura 2.3 de Cysneiros (2001) que apresenta uma tela da ferramenta OORNF (NETO, 2000) para cadastro de uma entrada do LAL para um sistema de controle automático de luzes. "As palavras sublinhadas são elos para outros símbolos do LAL" (CYSNEIROS, 2001).



A Figura 2.3: Exemplo de Entrada do LAL de um sistema de controle de luzes utilizando a ferramenta OORNF.
Fonte: Cysneiros (2001)

Abaixo temos a figura 2.4 mostrando o modelo SADT do processo de construção do LAL. O processo é formado por três atividades : identificar símbolos, classificar símbolos, descrever símbolos.

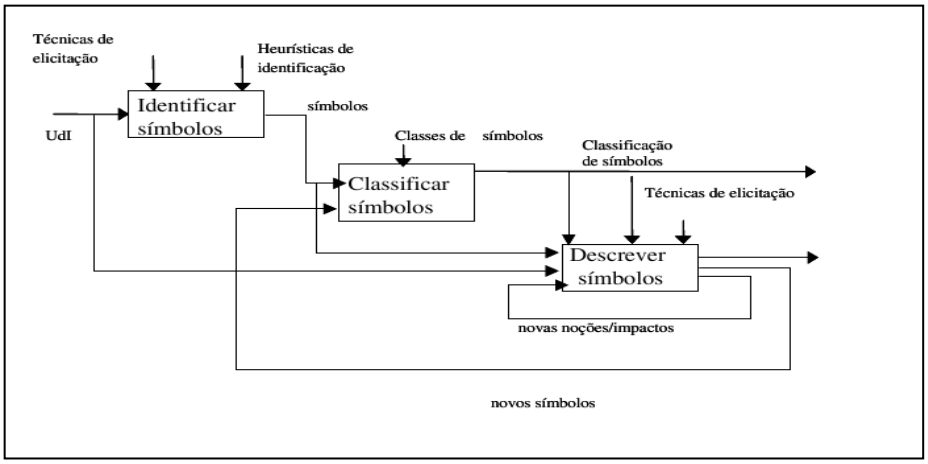


Figura 2.4: Modelo SADT do processo de construção do LAL do UDI.

Fonte: Cysneiros (2001)

2.4 O RNF-Framework

Nesta seção será apresentado o RNF-Framework, feito um esclarecimento sobre conceitos do RNF-Framework e como é feita a geração de grafos em Cysneiros (2001).

2.4.1 O RNF-Framework

No RNF-Framework os requisitos não funcionais são tratados como metas possivelmente conflitantes, devendo ser identificados em sua forma mais geral e refinados até que se chegue a um conjunto de requisitos que satisfaçam ao requisito geral Chichinelli e Cazarini (2001).

Durante o processo de projeto, organiza-se metas formando uma estrutura de grafos, no espírito de árvores E/OU que são usados na resolução de problemas (CHICHINELLI; CAZARINI, 2001). No entanto, essas metas raramente podem ser satisfeitas num sentido bem definido. Decisões de projetos podem influenciar de maneira positiva ou negativa em um determinada meta. A avaliação dessas metas determina o grau que um conjunto de requisitos não funcionais esta sendo apoiado por um projeto particular.

O framework consiste de cinco componentes principais, os quais não estão presentes no SQFD (que será abordado mais a frente) (CHICHINELLI; CAZARINI, 2001, p.

4):

- Um conjunto de metas para representar requisitos não funcionais, decisões de projeto e argumentos a favor ou contra as outras metas;
- Um conjunto de tipos de ligações, para referir-se a metas ou relacionamentos de metas (ligações) com outras metas;
- Um conjunto de métodos genéricos para refinar metas em outras metas;
- Uma coleção de regras de correlação para inferir interações potenciais (positiva ou negativa) entre metas;
- Procedimento de avaliação, o qual determina o grau, através de uma legenda, que determinado requisito não funcional esta sendo conduzido por um conjunto de decisões de projeto;

Para ilustrar o uso do NRF-Framework (ou Framework RNF) retiramos a figura abaixo de Cysneiros (2004, p:

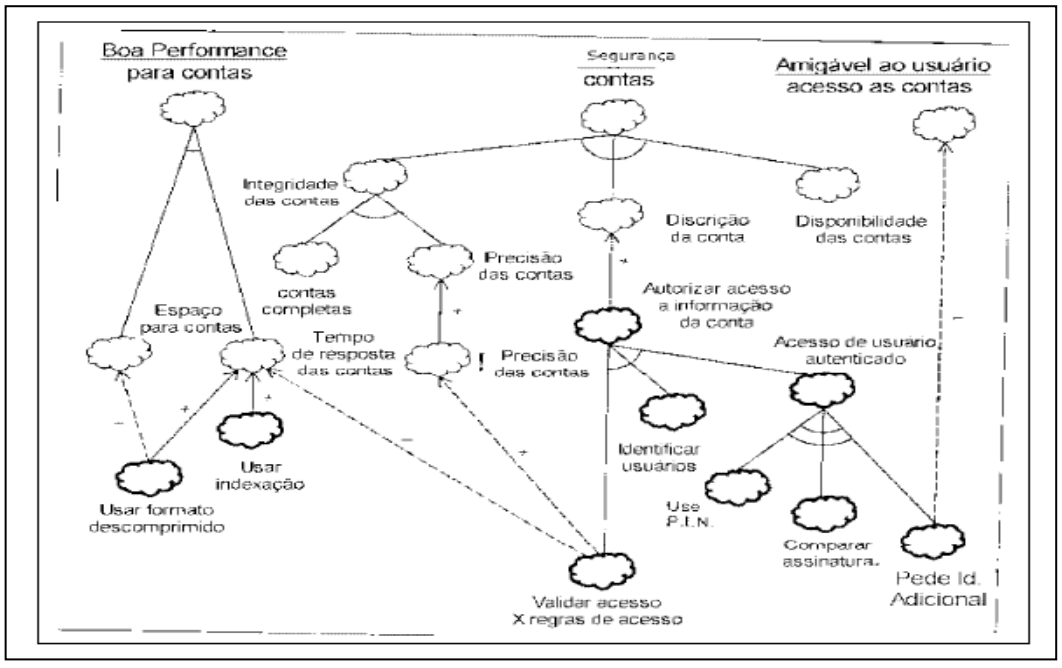


Figura 2.5: Grafo de RNFs decomposto até as Operacionalizações.

Fonte: Chung (2000)

Nesta figura as nuvens representam metas, que podem ser classificadas em Metas RNF e Metas Satisfatórias ou Operacionalizações (que são representadas por nuvens mais

espessas, que são decisões de projeto com o objetivo de satisfazer as metas RNF) (CHICHINELLI; CAZARINI, 2001). Os arcos representam ligações entre metas. Temos dois tipos de ligações a E (que são representadas por um único arco) e as OU (que são as de dois arcos). As ligações E representam que a meta pai é satisfeita, se todas as metas filho forem satisfeitas. Já a ligação OU, implica que a meta pai é satisfeita, se qualquer uma das suas metas filho é satisfeita. Existem as sinergias positivas e negativas. A positiva, que é representada pela seta pontilhada mais o sinal de (+), implica que a operacionalização tem um impacto positivo na Meta RNF. A Negativa, que é representada pela seta pontilhada mais o sinal (-), demonstra que a operacionalização tem um impacto negativo na meta RNF. Estes impactos negativos evidenciam a necessidade de negociação para priorização das metas RNF.

Removemos de Cysneiros (2001) a mesma figura após a análise e negociação para satistifação das metas:

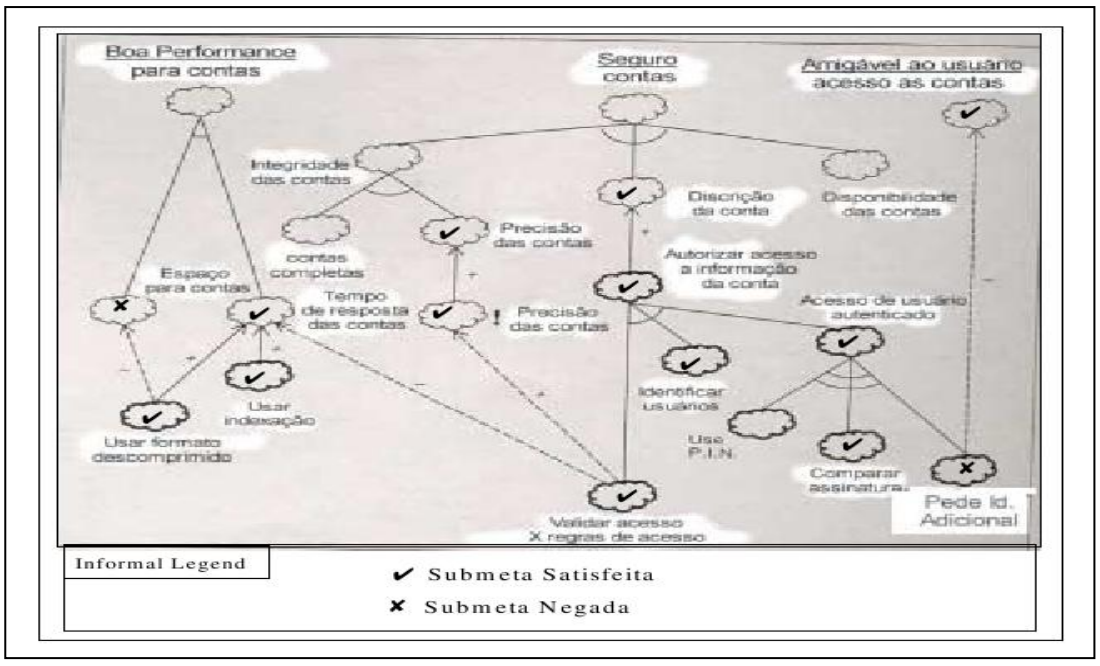


Figura 2.6: Grafo de RNFs com interdependências analisadas.

Fonte: Cysneiros (2001)

A figura mostra quais metas devem ser satisfeitas e quais delas devem ser negadas.

2.4.2 Esclarecendo Diferenças entre, RNF, meta, Operacionalização, tópico e tipo

Julga-se necessário fazer uma diferenciação entre os conceitos de RNF, meta, operacionalização, tópico e tipo no contexto de grafos de RNF para que se consiga mais fluência para prosseguir com o assunto.

O primeiro ponto a ser esclarecido é o conceito de Requisito Não funcional. Os requisitos não funcionais representam as restrições ou atributos de qualidade para um software ou para o processo de desenvolvimento do software. Podemos citar segurança, precisão, usabilidade, performance, portabilidade, manutenibilidade como exemplos de classes ou sub-classes (tipo ou tópico, de acordo com o framework RNF) dentro de uma taxonomia nas quais um determinado requisito não funcional poderia ser enquadrado. Dois exemplos de RNFs que se enquadram nas classificações portabilidade e performance, respectivamente, são citados a abaixo:

"O software deve ser compatível com os browsers IE (versão 5.0 ou superior) e Firefox (1.0 ou superior)";

a) "o software deve garantir que o tempo de retorno das consultas não seja maior que 5 segundos";

Alguns autores defendem que os requisitos devem ser apresentados seguindo uma qualificação ou quantificação (GILB apud RAMIRES, 2004). Pode ser observado que isso se verifica nos exemplos supracitados. O primeiro e o segundo exemplos (itens a e b) expressam, de forma não vaga ou não ambígua, qualidades que se deseja que o software tenha.

Um outro ponto, é que na representação de Chung os requisitos não funcionais são encapsulados na forma de **metas** a serem satisfeitas, o que da um aspecto de ambiguidade. Isto contrasta com idéias de alguns autores como Gilb e Ramires que defendem a idéia que falsos requisitos devem ser evitados. Falsos requisitos, segundo (RAMIRES, 2004, p. 26) "ocorrem quando uma especificação contém elementos desnecessários". Ramires faz referência a Gilb que explana situações onde ocorrem os falsos requisitos:

> é comum encontrar requisitos como *custos reduzidos*, *promover a gestão de controle*, *elevada usabilidade* e *aumentar a satisfação do cliente*, onde: não existe uma clara definição dos objectivos que se pretendem; não existe recurso a um

> conjunto de termos de base que reduzam ambiguidades; não existe quantificação do nível de qualidade corrente nem do nível exigido de futuro (RAMIRES, 2004, p. 26).

Na proposta de grafos de RNFs (CHUNG, 2000)(CYSNEIROS, 2001), *custos reduzidos, promover a gestão de controle, elevada usabilidade, e aumentar a satisfação do cliente* já seriam considerados necessários. Seriam classificados como metas a serem atingidas e seriam decompostas até se chegar às operacioanalizações. "Uma operacionalização corresponde a ações ou atributos que claramente identifiquem o que é necessário para satisfazer a meta principal (CYSNEIROS, 2001, p. 37).

Outro conceito a esclarecer é sobre as formas ou critérios de decomposições para os RNFs no Framework RNF. São elas: a por tipos e a por tópicos. Cysneiros discorre sobre as formas de decomposição de RNFs, conceitua a decomposição por tipos e por tópicos e da exemplos dessas decomposições. Na decomposição por tipo os métodos de decomposição existentes no framework são usados como fundamentos e guiam os engenheiros para as decomposições possíveis de uma determinada meta. "No caso do RNF Segurança, por exemplo, integridade, confidencialidade e disponibilidade podem ser possíveis decomposições desta meta" (CYSNEIROS, 2001, p.).

Percebe-se que Segurança expressa acima funciona como uma classe dentro de uma taxonomia, tal como foi comentado anteriormente no início dessa seção, e que integridade, confidencialidade e disponibilidade podem ser percebidas como sub-classes (ou tipos, de acordo com a terminologia utilizada por Chung (2000), Cysneiros (2001)) de Segurança.

Outra forma de decomposição é a por tópicos: "A decomposição por tópicos trata da decomposição estrutural do problema. Por exemplo, no exemplo acima, o RNF Seguro aplicado a contas poderia ter sido decomposto em contas pessoa física e jurídica" (CYSNEIROS, 2001, p.38).

Assim, resumindo, pode-se ver que tópico (assim como tipo) é uma forma ou critério de decomposição de RNF; meta é a forma de concepção para RNF que deve ser vista como um objetivo a ser alcançado; operacionalização é uma ação ou atributo, indica o que é necessário para atingir a meta principal ou parte dela (ou seja, é a sub-meta imediatamente

superior à operacionalização em questão) e; RNF é uma restrição ou atributo de qualidade.

Analisando as relações entre os conceitos verifica-se que quando atributos de qualidade são decompostos, seguindo o critérios de tópicos, chegamos a alternativas de solução (operacionalizações) que podem ou não impor uma restrição à satisfação de outras metas. Pode-se, então, identificar interdependências entre os RNFs. Estas interdependências são identificadas com maior facilidade quando se chega às operacionalizações. E podem ser do tipo positivas ou negativas. As positivas ajudam na satisfação das outras metas. As negativas impõem restrições (uma restrição pode ser total ou parcial) na satisfações de outras metas. Tais metas têm origem nos seus respectivos atores do UDI. Portanto quando uma operacionalização tem interdependência negativa com metas cuja a origem é outro ator do UDI temos os conflitos de interesses, ocasionando, assim, uma necessidade de negociação para a priorização das metas.

2.4.3 Geração de Grafos de RNfs

Cysneiros (2001) propós a utilização do grafo de RNFs apresentado em Chung (CHUNG, 1993) e (CHUNG, 2000) com algumas alterações definidas para auxiliar a rastreabilidade de origem e a integração com o modelo funcional.

As alterações propostas para o grafo de RNFs visam introduzir algum mecanismo de rastreabilidade do RNF à sua origem e a facilitar a integração deste à visão funcional.

Alteração para Facilitar a Rastreabilidade Reversa

[2]

Para auxiliar a rastreabilidade Reversa, Cysneiros passou a *representar acima do grafo de RNFs qual o ator do UdI* de onde se originou o conhecimento do domínio que levou-nos a estes RNFs e operacionalizações. Abaixo uma figura extraída de (CYSNEIROS, 2001) demonstrando as origens Administrador do Prédio e CIPA para os RNFs Custos Operacionais e Seguro, respectivamente.

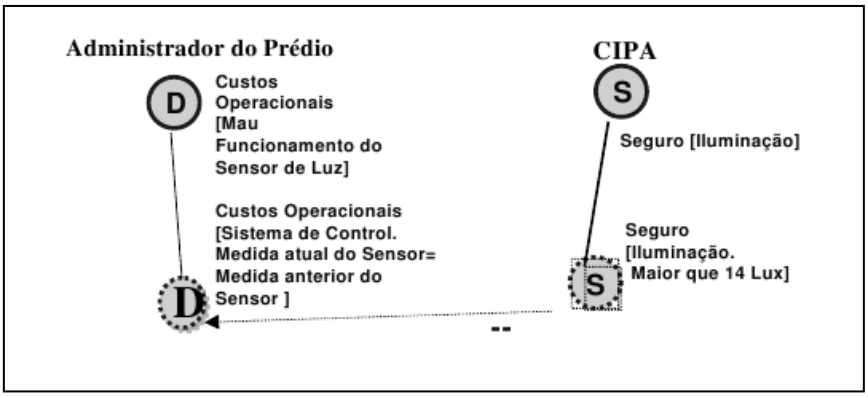


Figura 2.7: Exemplo de Grafo de RNFs com Identificador de Origem do RNF
Fonte: Cysneiros (2001).

Alterações para Facilitar a Integração das Visões Funcional e Não - Funcional

Para facilitar a integração entre as duas visões Cysneiros propôs uma regra, uma classificação para as operacionalizações, representação escrita para as metas e adaptações na representação gráfica das operacionalizações.

Quanto a regra criada, esta diz que "*Ambas as visões deveriam ser construídas tendo o LAL como âncora"*. Cysneiros faz algumas resalvas quanto a essa regra que são exibidas abaixo**:**

> No caso do Grafo de RNFs, significa dizer que todo tópico de um RNF será, obrigatoriamente um símbolo do LAL. Quando estivermos refinando as metas em submetas esta restrição não mais se aplica. Contudo, sempre que um símbolo do LAL puder ser usado isto deverá ser feito já que isso resultará numa diferente abordagem a ser utilizada durante a integração das visões (CYSNEIROS, 2001).

Quanto a classificação para as operacionalizações (Uma operacionalização corresponde a ações ou atributos que claramente identifiquem o que é necessário para satisfazer a meta principal)**,** estas são classificadas em "dinâmicas" e "estáticas". As operacionalizações dinâmicas são representadas por um círculo pontilhado e podem aparecer em qualquer nível do grafo e podem, ainda, serem decompostas em estática. Já as estáticas somente aparecerão no nível folha. A baixo a ilustração e comentário tirados de Cysneiros:

> A Figura 2.8 mostra um exemplo onde aparecem ambas as operacionalizações. Nesta

figura podemos ver duas operacionalizações estáticas que refinam a submeta de custos operacionais relacionada a sensores de movimento. Uma delas determina que as luzes devem ser apagadas nas salas depois de T3 minutos vazia e a outra determina que as luzes devem ser apagadas no corredor depois de T2 minutos vazio. Observamos também que na decomposição de custos operacionais relacionados com os sensores de luz, teremos uma operacionalização dinâmica enfatizando a necessidade do sistema usar a luz externa,captada pelos sensores de luz, para manter o ambiente iluminado. Isto nos leva à operacionalização estática que mostra que o esquema de luzes deve ser armazenado também sob forma de quantidade de iluminação, e não apenas na forma de percentual de atenuação das luzes, ou seja, denota a necessidade de um atributo que armazene esta informação (CYSNEIROS, 2001).

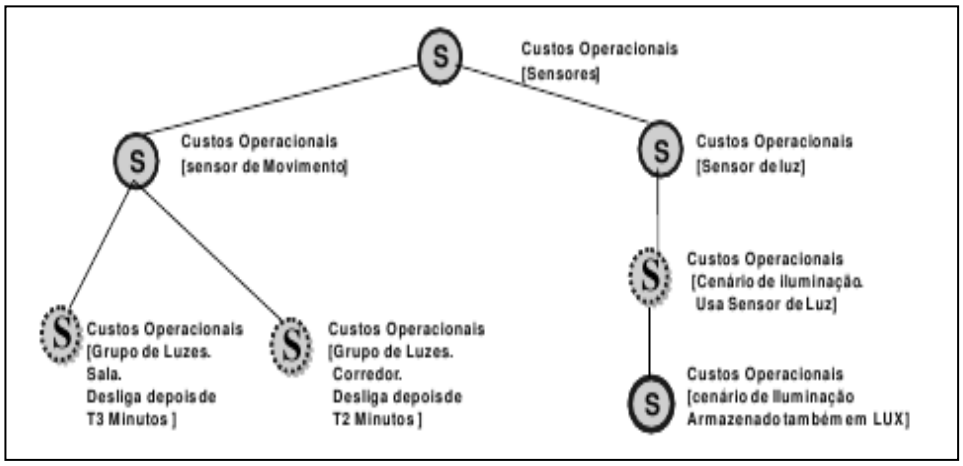


Figura 2.8: Exemplo de Operacionalizações Estáticas e Dinâmicas
Fonte: Cysneiros(2001).

Cysneiros comenta alteração na notação de grafos:

> A notação do grafo por nós utilizada é bem próxima da proposta por Chung [Chung 93] e baseia-se na representação do RNF seguido das metas/submetas entre colchetes, ver seção 2.7, onde as decomposições estão separadas por ponto, podendo esta submetas serem omitidas. Da mesma forma que proposto por Chung e explicado na seção 2.7, podemos também decompor metas sob ótica de tipo de RNF (Seguro em integridade e confiabilidade) ou do tópico ao qual ela se aplica (Sensores em sensor de movimento e sensor de luz) (CYSNEIROS, 2001).

A seção 2.7 da tese de Cysneiros trata sobre conceitos básicos relativos aos grafos de RNF os quais já foram introduzidos anteriormente neste texto.

Abaixo fizemos um quadro resumitivo sobre a classificação das

operacionalizações proposta por Cysneiros:

Classificação de Operacioanalização	Associação		Exemplo	Observações	Representação grafica
	necessidade	Diagrama de classes			
estática	a necessidade de um dado específico	atributos	iluminação deve ser armazenada em lux	sempre no nível folha	cículo com borda mais espessa que a normal
estática	nem dados /nem ação	n/a	num sistema de informações laboratorial, poderemos encontrar um RNF Usabilidade aplicado a Listas cuja operacionalização estabelece a necessidade de a interface de requisição ser a mesma para os diversos tipos de mapas.		
dinâmica	ações a serem realizadas	métodos	desligar a luz depois de T2 minutos	podem aparecer em qualquer nível,inclusive sendo decomposta em uma estática.	círculo pontilhado

Figura 2.9: Classificação das metas.

Quanto a *representação escrita para as metas*, a Figura 2.10 ilustra as diversas formas de representação que podemos utilizar. Cysneiros comenta que na figura podemos ver que o **RNF Custos Operacionais** já decomposto para S*ensor de Movimento* é representado sem a sua meta anterior (S*ensores*), sendo representada como Custos Operacionais [Sensor de Movimento]. Existe uma outra forma possível de representarmos esse mesmo nó do grafo que é Custos Operacionais [Sensores. Sensor de Movimento].

O quadro abaixo mostra a decomposição de metas e as opções para a representação das metas.

META	*TÓPICO*	*REPRESENTAÇÃO 1 (COMPLETA)*	*REPRESENTAÇÃO 2 (SIMPLIFICADA)*
Custos Operacionais	*Sensores*	*Custos Operacionais [sensores]*	*igual a primeira*
	Sensor de Movimento	*Custos Operacionais [sensores. Sensor de Movimento]*	*Custo Operacional[Sensor de Movimento]*
	Sensor de luz	*Custos Operacionais [sensores.Sensor de luz]*	*Custo Operacionais [Sensor de luz*

Figura 2.10: Quadro de representação escritas das metas.

Fonte: do Autor

Pode-se escolher qualquer uma dentre as opções de representação das metas, cada uma com suas vantagens e desvantagens.

> De uma maneira geral a inclusão das metas/submetas anteriores pode e deve ser evitada para efeito de simplicidade da representação do grafo. Entretanto, muitas vezes a representação das mesmas pode exercer um papel de ressaltar ao que se aplica a submeta sendo representada, o que por vezes auxilia na confecção do grafo (CYSNEIROS,2001).

Construção do Grafo de RNFs

Um outra contribuição de (CYSNEIROS, 2001) foi o desenvolvimento de heurísticas para auxiliar na contrução sistemática do grafo de RNFs que são apresentadas a seguir:

- Percorrer os símbolos do LAL já com RNFs representados, verificando quais possuem RNFs;
- Para cada símbolo que possuir um RNF faça:

- Definir a raiz do RNF utilizando o RNF seguido do símbolo do LAL como tópico.
- Decompor esta meta em submetas seja pela ótica de decomposição em tipos de RNFs (primários em secundários) ou em tópico (sensor em tipos de sensores). Representar o resultado obtido no grafo.
- Continuar a decomposição até que se possa visualizar o que é necessário para satisfazer este RNF, ou seja, suas operacionalizações. Representar as operacionalizações no grafo de RNF classificando-as em estáticas ou dinâmicas conforme o caso.
- Nas decomposiçoes anteriores utilizar o conhecimento próprio do domínio e sempre que possível envolver o cliente na definição destas decomposições, avaliando em conjunto com este se as decomposições estão corretas e são suficientes.
- Verificar no LAL se o RNF não têm conseqüências (operacionalizações) já definidas seja neste mesmo símbolo , seja em outros símbolos através do uso de opção da ferramenta OORNF de Navegar RNF -> Conseqüências (ver capítulo 4).
- Confrontar operacionalizações existentes no LAL com as que foram achadas via passos 2.2 a 2.4. As operacionaliações presentes no LAL que não estiverem no grafo devem ser representadas no grafo, o que eventualmento pode demandar novas decomposições. As operacionalizações existentes no grafo que não se encontrem no LAL evem ser incluídas no LAL. De uma maneira geral, operacionalizações estáticas se traduzem em noções de síbolos e dinâmicas em impactos.
- No caso de dificuldades em se decompor o RNF (passos 2.2 a 2.4) a identificação de possíveis operacionalizações já descritas no LAL (passo 2.5) pode servir como uma ajuda num processo que alterne abordagens bottom-up e top-down de decomposição deste RNF.

A construção do grafo pode dar-se aos poucos, ou seja, de maneira evolutiva fazendo-se grafos separadamente e depois fazendo-se a convergência posteriormente, se for o caso.

> A construção do grafo de RNFs é feita caminhado-se por todas as entradas definidas no LAL que já contenha os RNFs representados. Todo símbolo que possuir RNFs irá gerar um grafo de RNF, sendo que este RNF junto com o símbolo formarão a raiz de um grafo. Em um momento futuro dois ou mais grafos poderão ser agregados em um só por ter o engenheiro de software verificado que estes grafos são decomposições de um grafo maior. Um exemplo disto pode ser visto na Figura 5.3, onde originalmente tínhamos dois grafos distintos, um para o sensor de movimento e outro para o sensor de luz. Posteriormente avaliamos que na verdade ambos seriam a decomposição do RNF Custo Operacional aplicado a sensores.(CYSNEIROS, 2001).

Na decomposição por tipos, como a finalidade de nos orientar, devemos utilizar algo que sirva como checklist. Em sua proposta, Cysneiros, (2001) utilizou a base de conhecimento provida pela ferramenta OORNF, apresentada por Neto (2000) e adaptada em Cysneiros, (2001). A ferramenta OORNF mostra quais são os RNFs secundários que decompõe um dado RNF primário.

Extraimos de Cysneiros (2001) o seguite exemplo (adaptado):

> Tomando como exemplo o RNF Custo Operacional relacionado a sensor de movimento, representado por Custo Operacional [Sensor de Movimento], poderíamos pensar em decompor esta meta nas submetas de sala e corredor, ou seja, teremos que ter operacionalizações diferentes dependendo da localização do sensor de movimento. É necessário ressaltar que, neste caso, as decomposições também são símbolos do LAL, mas isto não é mandatório.
>
> Uma vez decomposta em sala e corredor, temos de verificar se alguma outra decomposição é necessária. Notamos então que temos ainda que especificar como a necessidade de satisfazer a baixos custos operacionais pode ser satisfeita quando observada da ótica do uso de sensores de movimento em salas e corredores. Verificamos, junto ao cliente ou por conhecimento do domínio, que para tal é necessário existirem as ações de desligar as luzes das salas após T3 minutos destas estarem vazias e desligar as luzes do corredor após T2 minutos deste estar vazio. A representação deste processo pode ser vista na Figura 7(CYSNEIROS, 2001).

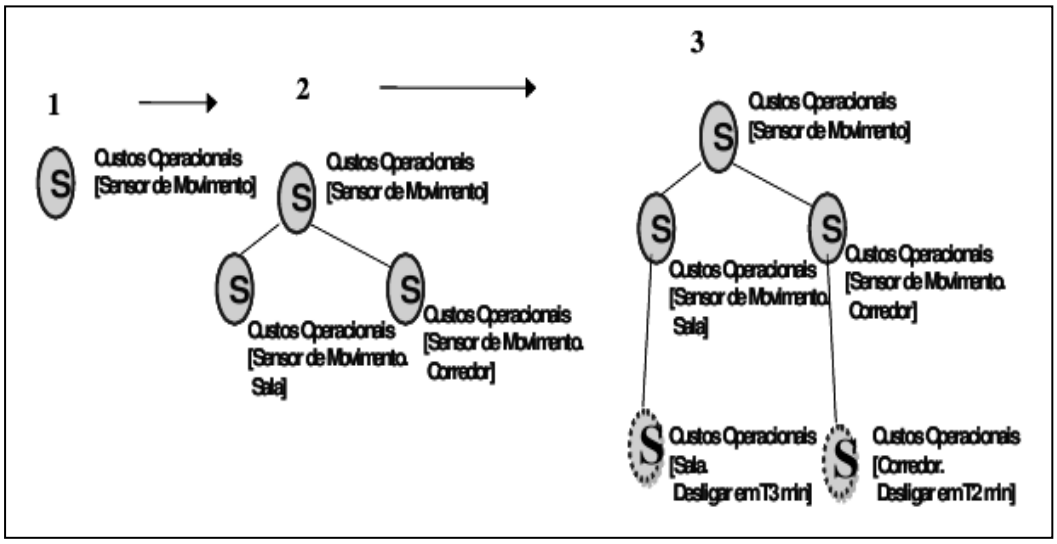


Figura 2.11 – Exemplo da Construção do Grafo de RNFs

Fonte: Cysneiros,(2001)

Por fim, devemos verificar no "LAL" quais as "noções" e "impactos" existentes que são conseqüência da necessidade de satisfazer o RNF Custo Operacional para o símbolo Sensor de Movimento.

É necessário que o engenheiro de software tenha atenção para verificar ao incluir uma operacionalização se esta é refletida pelo LAL. Se isto não acontecer o mesmo deverá ser atualizado para refletir essa operacionalização.

2.5 Análise de Negócio: Integração dos RNFs aos Modelos de Processos de Negócio

Neste tópico serão abordados o conceito de modelo de negócio e modelo de processo de negócio. Também será comentado sobre a importância do uso de modelagem de processos de negócio como atividade integrante da fase inicial da Engenharia de Requisitos e como ela ajuda à "comunicação" entre engenheiros de software e solicitantes de software, e consequentemente, contribui para uma boa negociação da qualidade do software e do processo inicial de criação do software. Também será apresentado o resumo (conceito, contribuições e o processo) da abordagem BPMNRNF.

2.5.1 Modelos de negócio

Laís Xavier (XAVIER, 2009) cita Davenport e Stoddard, Eriksson e Penker , Berio e Vernadat, Paim e Bittencourt ao apresentar o conceito modelo de negócio. Segundo Xavier (2009) modelos de negócio são "conjuntos de modelos que representam um abstração da realidade de uma organização sob o ponto de vista do negócio e representam o ponto de vista do negócio, e representam o ambiente no qual a organização esta inserida" (XAVIER, 2009, p 7).

Xavier também fala sobre alguns benefícios, em relação à tecnologia da informação, do uso dos modelos de negócio. Entre as contribuições estão: permitem compreender as interfaces organizacionais e o fluxo de informações através das unidades de negócio; contribuem para evitar a redundância em sistemas; contribuem na integração de base dados; e permitem descrever o processo de desenvolvimento de um produto. Em sua proposta foi importante a aplicação dos modelos de negócio no desenvolvimento de sistemas, pois permitem, segundo Knight (apud Xavier (2009)), unificar o conhecimento dos envolvidos e entender as necessidades que o sistema tem (2004 apud XAVIER, 2009, p. 8). Em particular, Xavier concentra-se nos os modelos de processos de negócio que têm sido vistos como uma forma de integrar as linguagens dos profissionais (o que ajuda a resaltar a importância da análise de negócio como uma das atividades iniciais da Engenharia de Requisitos) que têm visão de negócios e dos profissionais com visão tecnológica (XAVIER, 2009; Davenport and Stoddard, 1994; Paim, 2007).

2.5.2 Modelos de Processos de negócio

Em sua dissertação (XAVIER, 2009, p. 9) faz uma definição e explora o conceito de processo constituído por um conjunto de atividades iniciadas em resposta a um evento, ordenadas no tempo e no espaço, com entradas e saídas claramente identificados, gerando produtos, serviços ou informação. Um processo cumpre um objetivo, seja da organização ou da aplicação que será gerada, e deve ser executado levando-se em consideração o escopo em que está inserido e suas particularidades (DAVENPORT, 1993 apud XAVIER, 2009, p. 9).

As *atividades, atores, eventos, regras, recursos e artefatos de entrada e saída* são elementos utilizados no modelo de processo de negócio que mais se destacam. Xavier faz referência a Bittencurt ao explicar que as informações utilizadas para especificação de um sistema de informação vêm dos elementos de negócio:

> as atividades se traduzem em funcionalidades; artefatos que são transformados pelas atividades e se transformam em entidades de informação manipuladas pelo sistema; e atores que representam os elementos (cargos, organizações, sistemas equipamentos) que interagem com o sistema (BITTENCOURT, 2009 apud XAVIER, 2009, p.9).

2.5.3 A importância do uso de modelagem de processos de negócio para a Engenharia de Requisitos

Percebemos que os modelos de processos de negócio ajudam a viabilizar uma melhor percepção dos fluxos de negócios, permite o alinhamento dos pontos de vistas dos leitores (na medida em que em que os leitores se guiem pelos mesmos modelos), ou melhor permitem que diferentes leitores tenham o mesmo entendimento ao ler os modelos.

Tais modelos são o ponto de comunicação entre profissionais que têm visão de negócio e os profissionais de visão tecnológica. Uma boa comunicação entre tais profissionais é vital para o sucesso das negociações no decorrer do processo de produção do software. Particularmente, no caso de negociação de qualidade do software, na fase de Engenharia de Requisitos a integração de requisitos não funcionais aos modelos de processo de negócio ajuda, de certa forma, a formar uma linguagem em comum entre os profissionais que precisam que haja eficiência e eficácia da comunicação.

2.5.4 A Abordagem BPMNRNF

Em sua tese, intitulada de “Integração de Requisitos Não-Funcionais a Processos de Negócios: Integrando BPMN e RNF, Laís Xavier propõem uma abordagem chamada de BPMNRNF, em que integra os requisitos Não-Funcionais à notação BPMN. Esta abordagem proporciona uma solução à deficiência de expressividade dos RNFs, isto é, define um

processo que viabiliza a inserção de operacioanalizações dos requisitos não-funcionais nos diagramas de processos de negócio (Xavier, 2009). Essa integração proporciona uma melhor comunição entre os stakeholders, principalmente entre os engenheiros e solicitantes do software. Entre as contribuições da proposta de Xavier (2009), é o fato de que a proposta evidencia a importância do uso de modelagem de processos de negócio como um passo inicial no processo de Engenharia de Requisitos.

2.6 O SQFD

O SQFD é uma metodologia de desenvolvimento de software que fornece métodos formais para transformar os requisitos de qualidade em especificações de desenho. O SQFD (Software Quality Function Deployment – SQFD) é uma extensão do Desdobramento da Função Qualidade (Quality Function Deployment – QFD) aplicada ao processo de desenvolvimento de software.

O QFD tem como objetivo a identificação dos stakeholders que pretendem o sistema, a identificação do que desejam e a identificação de como satisfazer esses desejos.

A figura a baixo exibe uma matriz simplificada do QFD:

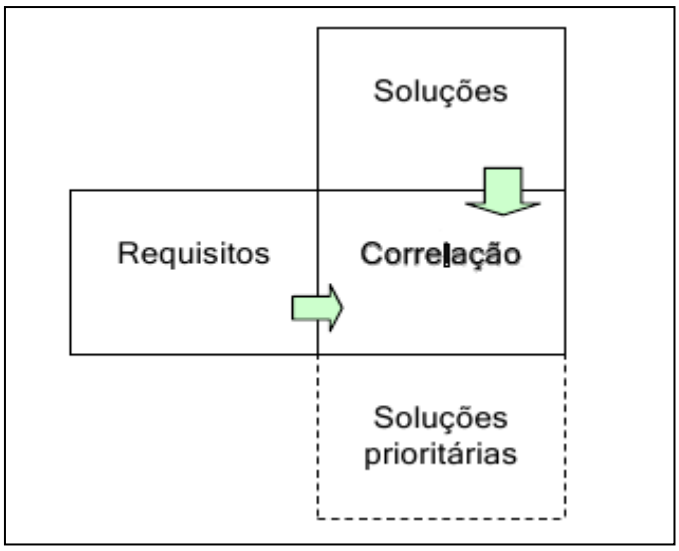


Figura 2.12: Matriz QFD simplificada.

Fonte: Ramires (2004).

No Capitulo 3 será comentada mais sobre o QFD e SQFD.

2.7 Integração visão funcional com visão não-funcional

Luiz Márcio Cysneiros (Cysneiros, 2001), em sua tese intitulada de "Requisitos Não-Funcionais: Da elicitação ao modelo Conceitual", apresenta alguns objetivos a serem atingidos em seu trabalho:

- Desenvolver processo para se lidar com requisitos não-funcionais desde as etapas iniciais do processo de desenvolvimento de software.
- Obter uma melhoria da qualidade final do produto sob a ótica, tanto do cliente quanto do desenvolvedor, bem como em um menor tempo de entrega de produtos e menores custos de manutenção. Para isso buscou atingir um outro objetivo que é:
- Obter modelos conceituais mais completos e com os impactos da satisfação dos requisitos não funcionais já analisados. Para este ser atendido, por sua fez, verificou que era necessário atingir outro objetivo:
  - Obter um modelo conceitual que enfoque os requisitos funcionais e não funcionais ao mesmo tempo.

Para que esses objetivos fossem alcançados Cysneiros propoem uma estratégia que se baseia no uso Léxico Ampliado da Linguagem (LAL) para que sirva como âncora da construção dos modelos funcionais e não funcionais.

Elaborou-se uma extensão ao LAL para que ele pudesse sustentar a natureza evolutiva e as negociações da elicitação dos RNFs. A ferramenta OORNF (Neto 00) foi customizada para suportar as novas características dadas ao LAL. Assim o engenheiro de software poderia administrar melhor tal aspecto evolutivo dos RNFs.

Outro aspecto dos RNFs, o do tratamento de interdependências entre RNFs, teve bastante atenção em seu trabalho. Estes por terem a possibilidade de originar importantes decisões de desenho e que "implicam em grande montante de retrabalho se não forem prévia e convenientemente identificadas" (Cysneiros, 2001, p.13.).

Mais uma contribuição proposta pelo autor foi uma adaptação do grafo de RNFs proposto por Chung (Chung 2000). O objetivo primário dessa adaptação era o de obter um certo nível de rastreabildade dos RNFs para auxiliar em seu aspecto evolutivo. Outro objetivo

é prover facilidade de integração entre as visões funcional e não-funcional.

Cysneiros (2001) mostra ainda como organizar os grafos de forma que possibilite que a realização da análise sistemática das interdependências intra-grafos seja facilitada. Uma proposta para a integração das visões funcional e não funcional com o fim de obter um modelo conceitual consolidado refletindo ambas as visões também foi apresentada.

Por fim foi proposto uma extensão a UML nos diagramas de classe, sequencia e colaboração. O objetivo é representar as consequências das operacionalizações dos RNFs e estabelecer uma ligação dos modelos da visão funcional com os RNFs que possibilite a rastreabilidade desses RNFs. Essa rastreabilidade auxilia bastante o tratamento do aspecto evolutivo do software, pois podemos nos reportar às fontes que dão origem a inclusão nos modelos conceituais de uma determinada classe, operação, atributo ou mesmo mensagem vinda da operacionalização de um RNF (CYSNEIROS, 2001).

Seu trabalho se enquadra sob a ótica de processo de desenvolvimento que propõem formas de justificar decisões sobre inclusão ou exclusões de requisitos que refletirão no desenho do software e que também guia, racionalizando em termos de RNFs, o processo de desenvolvimento de software.

Cysneiros ressalta um aspecto dessa ótica que é o fato de tomadas de decisões serem forçadas quando ocorre adição de RNFs a uma especificação de requisitos. Isso poderá provocar um efeito positivo ou negativo em outros RNFs e estes efeitos podem ser visualizados através de *interdependências entre RNFs*. Um RNF pode afetar um outro RNF de forma positiva ou negativa. A positiva ajuda na satisfação do requisito. E a negativa força uma tomada de decisão.

A estratégia formulada por Cysneiros (2001) pode ser visualizada, visão geral, na figura abaixo:

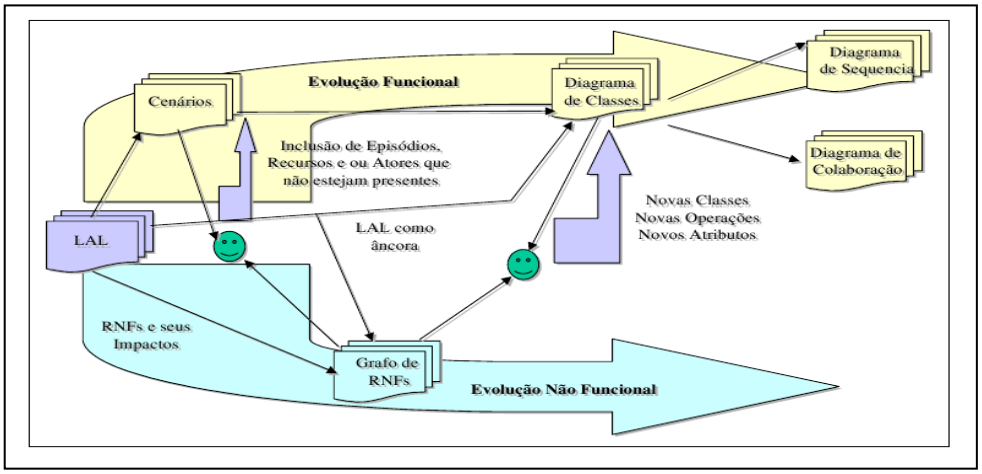


Figura 2.13: Detalhamento da estratégia proposta por Cysneiros.

Fonte: Cysneiros (2001).

A estratégia geral tem como objetivo: demonstrar como identificar e representar RNF de maneira geral e integrar as visões funcional e não funcional. Também propõem o processo de elicitação que busca a sistematização para obtenção, representação e análise de RNF, possibilitando a análise sob duas óticas: interdependências um RNF e outro(s) RNF(s) e interdependências entre RNF(s) e Visão Funcional.

A estratégia impõe algumas restrições como a construção de diagramas de classe e gráfos que devem ser realizadas a partir de símbolos de LAL (os nomes dos grafos e diagramas de classes tem que ser um símbolo do LAL) e o LAL será fonte de definição de **cenários**[1]. O motivo das restrições esta na facilidade de análise de impacto do RNF no modelo conceitual.

Na proposta a evolução dos requisitos é vista sobe dois ciclos distintos que é o funcional e não-funcional. E o LAL não esta classificado em nenhum deles, é a ancora ou base do dois.

O tratamento do aspecto evolutivo dos RNF foi favorecido pela extensão realizada no LAL que foi a adição, no LAL do UDI, de necessidade de RNFs e dos seus impactos sobre os símbolos. A integração das visões foi favorecida pela inclusão de RNF no modelo funcional. A integração é feita ao nível de cenários ou de diagramas de classe, sequência e colaboração. Ao nível de cenários deve-se avaliar o título do cenário e procurar relação com o

[1] O termo Cenário pode ser entendido como descrição de situações que ocorrem no macro-sistema e suas relações com o sistema.

grafo. Ao nível de diagrama de classe deve-se, para cada classe, procurar a ocorrência do símbolo que nomeia a classe no conjunto de grafos. Os dois ciclos evoluem independentemente. Sempre que ocorrer uma alteração em qualquer das visões, o engenheiro deve atualizar a outra visão e fazer a integração da alteração na outra visão.

Quando for detectado um novo RNF ou condição de satisfação, deve-se analisar os impactos na visão não-funcional e integrá-lo à visão funcional. Deve-se analisar os cenários que se relacionam com os RNF alterados e o mesmo deve ser feito com os diagramas de classes.

Quando a mudança ocorre na visão funcional ocorre o processo inverso: faz-se a atualização do LAL, dos cenários, diagramas de classes, de sequência e de colaboração. No caso de uma nova classe for incluída, inicia-se o processo de integração para essa nova classe e verifica-se se as novas funções não demandam novos RNFs (se isso ocorrer realiza-se novamente a integração do RNF na visão funcional.

Um processo de construção para a visão Não-Funcional pode ser visualizado na figura abaixo:

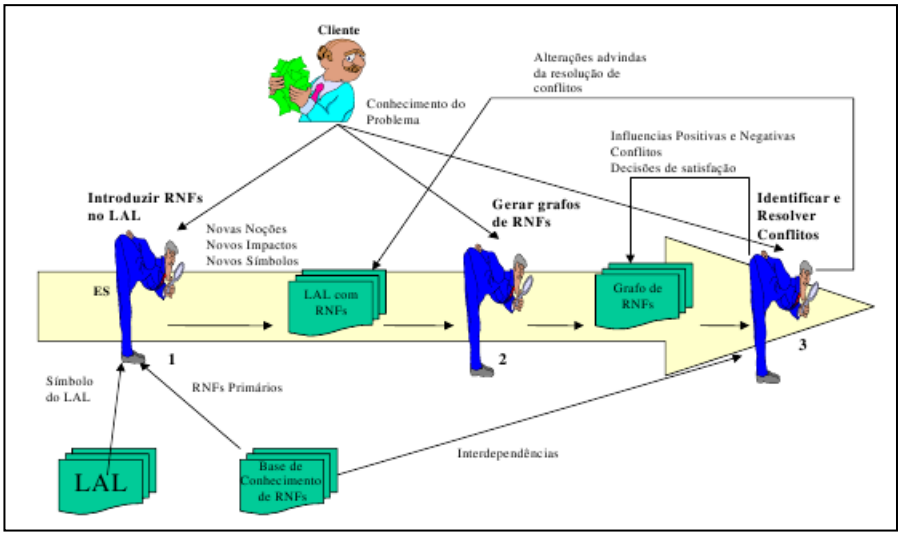


Figura 2.14: Processo de elicitação de Requisitos Não-Funcionais

Fonte: Cysneiros (2001)

2.8 Considerações finais

Neste capítulo foram apresentados conceitos de engenharia de requisitos considerados básicos para o entendimento da análise a ser apresentada no capitulo 4. Mais conceitos básicos serão apresentados no capítulo 3 se referindo especificamente à negociação de forma geral e negociação no processo de desenvolvimento de software.