

1. INTRODUÇÃO

O processo de Engenharia de Requisitos tem como principal objetivo a aquisição de conhecimentos das regras de negócios e verificação das necessidades do cliente, de forma a obter uma especificação não ambígua e completa dos requisitos de *software*, com o intuito de minimizar os erros, inadequações e falhas no produto final a ser entregue ao cliente. Uma análise e especificação de requisitos com baixa qualidade podem levar a danos e prejuízos, que inviabilizam o projeto e que podem até mesmo colocar em risco vidas humanas, como por exemplo, em sistemas hospitalares, de tráfego aéreo, de controle de trânsito, etc.

O Gerenciamento de Requisitos é o processo de compreender e controlar as mudanças que ocorrem nos requisitos, por força da evolução dos mesmos, refletindo as alterações que ocorrem ao longo do tempo, no ambiente do sistema e nos objetivos da organização [SOM03]. Além da Análise e Especificação, o Gerenciamento de Requisitos é de fundamental importância no processo da Engenharia de Requisitos, pois organiza o controle das mudanças, permitindo subsídios para a análise de impacto e custos em tempo e dinheiro, que estas mudanças trarão para a organização.

Este trabalho tem por objetivo desenvolver uma ferramenta automatizada para documentação e gerenciamento de requisitos, durante todo o ciclo de vida do *software*. A ferramenta deve coletar, armazenar e manter os requisitos acordados entre os *stakeholders*¹, gerenciar as mudanças ocorridas nos requisitos, em razão de sua natural evolução, rastrear os relacionamentos entre requisitos e entre documentos de requisitos, e outros documentos produzidos no processo de desenvolvimento do sistema.

¹ “O termo *stakeholder* é utilizado para se referir a qualquer pessoa que terá alguma influência direta ou indireta sobre os requisitos do sistema” [SOM03].

A motivação para a escolha do tema advém da escassez de ferramentas brasileiras disponíveis no mercado para gerenciamento de requisitos e das dificuldades que uma ferramenta internacional pode trazer, como:

- Custos de aquisição;
- Custos de treinamento;
- Dificuldades dos *stakeholders* com línguas estrangeiras;
- Falta de representantes no Brasil;
- Falta de especialistas do produto no mercado.

A contribuição deste trabalho é oferecer à comunidade de Engenharia de *Software*, uma ferramenta de uso livre para documentação e gerenciamento de requisitos durante todo o ciclo de vida do *software*. Espera-se que esta ferramenta traga benefícios às organizações que venham adotá-la, pois além de não haver custos com aquisição da ferramenta e do banco de dados, irá proporcionar controle sobre as mudanças ocorridas nos requisitos e análise do risco, impacto e custos destas mudanças.

O restante deste trabalho está organizado da seguinte forma: O segundo capítulo expõe a revisão bibliográfica baseada na literatura sobre Engenharia de Requisitos; o terceiro capítulo discorre sobre Gerenciamento de Requisitos que é a base do trabalho desenvolvido. No quarto capítulo é registrado o desenvolvimento da ferramenta, composto dos processos de elicitação de requisitos da ferramenta, de especificação dos requisitos da ferramenta e da modelagem de classes da ferramenta. No quinto capítulo é apresentada a aplicabilidade da ferramenta, através do estudo de caso. O sexto capítulo é fechado com Conclusões do trabalho e propostas de trabalhos futuros.

2. A ENGENHARIA DE REQUISITOS

O sucesso do desenvolvimento de um *software* depende principalmente da clara definição do produto que se pretende obter. A Engenharia de Requisitos, através de métodos, técnicas e ferramentas, propõe-se a organizar a base para a construção do *software* de forma a atender o processo de definição e análise de requisitos [MAR01]. Neste capítulo apresentam-se definições de Requisitos, de Engenharia de Requisitos e as fases ou atividades que a compõe.

2.1. DEFINIÇÕES DE REQUISITOS

A definição do termo requisito é muito ampla, mas convencionalmente pode-se definir como sendo uma declaração de um serviço ou restrição de um sistema a ser desenvolvido [KOT98]. Um conjunto de requisitos pode ser definido como uma condição ou capacidade necessária que o *software* deve possuir para que o usuário possa resolver um problema ou atingir um objetivo, para atender as necessidades e restrições da organização ou dos outros componentes do sistema.

Segundo Kotonya e Sommerville [KOT98], um requisito pode descrever:

- a. Uma facilidade em nível do usuário (por exemplo, um processador de texto com corretor ortográfico);
- b. Uma propriedade geral do sistema (por exemplo, o sistema deve garantir que informações pessoais nunca estarão disponíveis sem autorização);
- c. Restrição específica no sistema (por exemplo, o leitor de cartões magnéticos deve ler a trilha 2);

- d. Restrição no desenvolvimento do sistema (por exemplo, o sistema deve ser desenvolvido usando a linguagem JAVA).

Segundo a IEEE [STD830] requisito pode ser:

- a. Uma condição ou capacidade necessária de um usuário resolver um problema ou alcançar um objetivo;
- b. Uma condição ou capacidade que deve ser alcançada ou possuída por um sistema ou componente de um sistema, para satisfazer um contrato, um padrão, uma especificação ou outros documentos formalmente impostos;
- c. Uma representação documentada de uma condição ou capacidade, conforme os itens (a) e (b).

Para Sommerville [SOM03] o termo requisito não é utilizado pela indústria de *software* de modo consistente, podendo ser visto como uma declaração abstrata, de alto nível, de uma função que o sistema deva fornecer ou de uma restrição do sistema, ou, no outro extremo, é uma definição detalhada, matematicamente formal de uma função do sistema. Define o termo requisitos do usuário para designar os requisitos abstratos de alto nível, o termo requisitos de sistema, para indicar a descrição detalhada do que o sistema deverá fazer e o termo especificação de projeto de *software*, para associar a Engenharia de Requisitos e as atividades de projeto. Abaixo, seguem definições dos três termos:

- a. Requisitos do usuário são declarações em linguagem natural e também em diagramas sobre as funções que o sistema deve fornecer e as restrições sob as quais deve operar.
- b. Requisitos de sistema estabelecem detalhadamente as funções e restrições de sistema. O documento de requisitos de sistema, algumas vezes chamado de especificação funcional, deve ser preciso. Ele pode

servir como um contrato entre o comprador do sistema e o desenvolvedor do *software*.

- c. Especificação de projeto de *software* é uma descrição abstrata do projeto de *software*, que é uma base para o projeto e a implementação. Essa especificação acrescenta mais detalhes à especificação de requisitos do sistema.

2.2. CLASSIFICAÇÃO DOS REQUISITOS DE SISTEMA

Na maior parte da literatura atual encontra-se os requisitos de sistema de *software* classificados em Requisitos Funcionais e Requisitos Não-Funcionais. Sommerville [SOM03] discute uma nova classe: Requisitos de Domínio.

2.2.1. REQUISITOS FUNCIONAIS

Os requisitos funcionais são descrições das diversas funções que clientes e usuários desejam ou necessitam que o *software* ofereça. Determinam o que o *software* deve fazer, e são identificados a partir do ponto de vista do usuário. A especificação de um requisito funcional deve determinar o que se espera que o *software* faça, sem se preocupar como ele irá fazê-lo.

Segundo Sommerville [SOM03] “são declarações de funções que o sistema deve fornecer, como o sistema deve reagir a entradas específicas e como deve se comportar em determinadas situações. Em alguns casos, os requisitos funcionais podem também, explicitamente declarar o que o sistema não deve fazer”.

São exemplos de requisitos funcionais:

- As vendas somente poderão ser efetuadas aos associados do Sindicato (empresas, funcionários e sócios em geral), que no ato da compra devem apresentar a carteira de associado.
- A venda somente poderá ser efetuada mediante receituário que ficará retido na drogaria.

2.2.2. REQUISITOS NÃO-FUNCIONAIS

Requisitos Não-Funcionais são as qualidades globais de um *software*. Determinam características desejáveis do *software*, quanto ao desempenho, confiabilidade, segurança, portabilidade, entre outros. Alguns desses requisitos, posteriormente, serão traduzidos em funções ou operacionalizados [CHU99].

Mylopoulos e outros [MYL99] definem Requisitos Não-Funcionais como atributos de qualidade normalmente descritos de maneira informal, geram alguma controvérsia (por exemplo, o gerente quer segurança, mas os usuários querem flexibilidade e facilidade de uso) e são difíceis de validar.

Segundo Sommerville [SOM03] “Requisitos Não-Funcionais são restrições sobre os serviços ou as funções oferecidos pelo sistema. Entre eles destacam-se restrições de tempo, restrições sobre o processo de desenvolvimento, padrões, entre outros”. A Figura 1 representa uma classificação dos diferentes tipos de Requisitos Não-Funcionais que podem surgir.

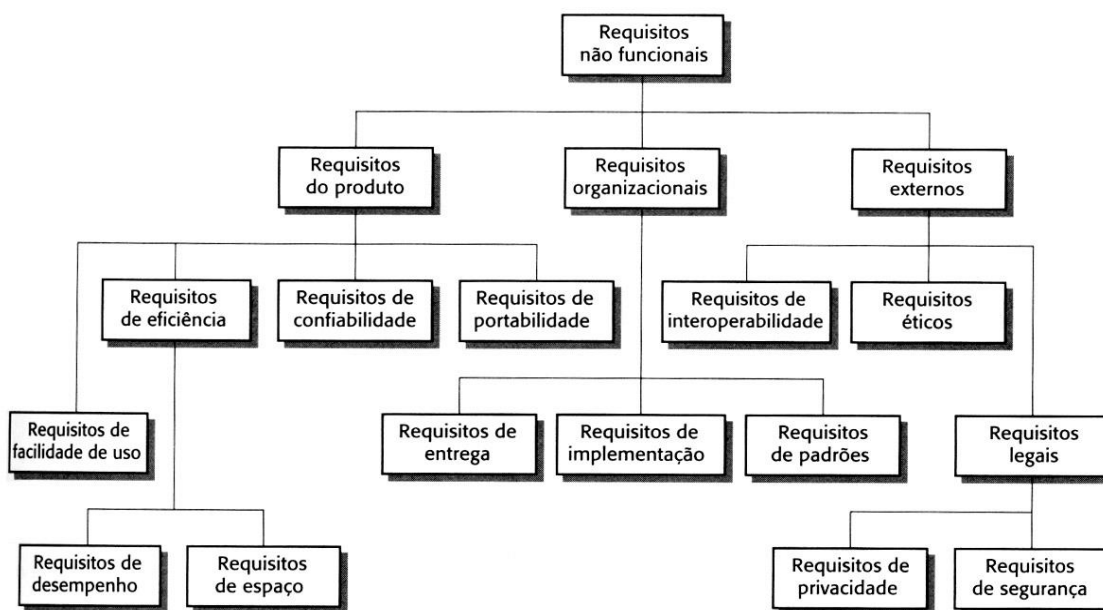


Figura 1 – Tipos de Requisitos Não-Funcionais [SOM03]

Sommerville [SOM03] classifica os diferentes tipos de Requisitos Não-Funcionais conforme sua procedência:

- a. Requisitos de Produtos - são os requisitos que especificam o comportamento do produto, como por exemplo: com que rapidez o sistema deve operar, quanto de memória o sistema necessita, requisitos de confiabilidade, que estabelecem a taxa aceitável de falhas, de portabilidade e de usabilidade.
- b. Requisitos Organizacionais - são oriundos de políticas e procedimentos das organizações do cliente e do desenvolvedor, como por exemplo: padrões de processos que devem ser utilizados, os requisitos de implementação, como a linguagem de programação ou o método de projeto utilizado, e os requisitos de confiabilidade, que especificam quando o produto e seus documentos devem ser entregues.
- c. Requisitos Externos - abrange todos os requisitos procedentes de fatores externos ao sistema e a seu processo de desenvolvimento. Dentre eles destacam-se os requisitos de interoperabilidade, que definem como o sistema interage com sistemas em outras organizações e os requisitos legais, que devem ser seguidos para assegurar que o sistema opera em acordo com a lei.

2.2.3. REQUISITOS DE DOMÍNIO

Segundo Sommerville [SOM03], Requisitos de Domínio são requisitos que se originam do domínio de aplicação do sistema e que refletem características desse domínio, podendo ser requisitos funcionais ou não funcionais. São oriundos do domínio da aplicação do sistema, em vez de serem obtidos a partir das necessidades específicas dos usuários do sistema. Estes requisitos podem ser novos requisitos funcionais em si, podem restringir os requisitos funcionais existentes ou estabelecer como devem ser realizados procedimentos específicos. São importantes, porque, muitas vezes, refletem fundamentos do domínio da aplicação, e se não forem satisfeitos, poderão comprometer a operação do sistema. Os especialistas em um domínio podem deixar de fornecer informações em um requisito, simplesmente porque para eles essas informações são naturais, mas elas podem não ser óbvias para os desenvolvedores de sistemas, que, como resultado disso, podem não implementar o requisito de maneira satisfatória.

2.3. DEFINIÇÕES DE ENGENHARIA DE REQUISITOS

A principal função da Engenharia de Requisitos é gerar especificações que descrevam de forma não ambígua, consistente e completa, o comportamento do universo do domínio de um problema. A Engenharia de Requisitos, como parte da Engenharia de *Software*, busca sistematicamente, a definição do que se quer produzir.

Ridao e outros [RID00] entendem que a Engenharia de Requisitos estabelece o processo de definição dos requisitos como um processo no qual se elicit, modela e analisa o que deve ser feito. Leite [LEI94] afirma que este processo deve lidar com diferentes pontos de vista, e usar uma combinação de métodos, ferramentas e pessoas, dentro de um contexto previamente definido, chamado de Universo de Informações (UdI)².

O IEEE [STD830] define que Engenharia de Requisitos corresponde ao processo de aquisição, refinamento e verificação das necessidades do cliente para um sistema de *software*, objetivando uma especificação completa e correta dos requisitos de *software*.

Sommerville [SOM03] descreve Engenharia de Requisitos como o processo de descobrir, analisar, documentar e verificar os requisitos de um sistema. Já Lamsweerde [LAMS00] define Engenharia de Requisitos como a identificação dos objetivos a serem atingidos pelo futuro sistema, a operacionalização de tais objetivos em serviços e restrições e a atribuição de responsabilidades pelos requisitos resultantes a agentes humanos, dispositivos e *software*.

Para Kotonya e Sommerville [KOT98] Engenharia de Requisitos é um termo relativamente novo para cobrir todas as atividades envolvidas na descoberta, documentação e manutenção de um conjunto de requisitos, em um

² “Universo de Informações é o contexto geral no qual o software deverá ser desenvolvido e operado. O UdI inclui todas as fontes de informação e todas as pessoas relacionadas ao software. Essas pessoas são também conhecidas como os atores desse universo. O UdI é a realidade circunstanciada pelo conjunto de objetos definidos pelos que demandam o software” Leite (1994).

sistema baseado em computador, e implica que técnicas sistemáticas e repetitivas sejam usadas para assegurar que os requisitos do sistema sejam completos, consistentes e relevantes.

Zave [ZAV97] afirma que Engenharia de Requisitos é o “ramo da engenharia de *software* que está preocupado com os objetivos do mundo real para as funções e restrições de sistemas de *software*. Está também preocupado com o relacionamento destes fatores para especificações precisas do comportamento do *software* e com sua evolução no tempo e através de famílias de *software*”.

Nuseibeh [NUS00] apresenta a Engenharia de Requisitos dentro de um contexto que normalmente é um sistema humano de atividades e os donos dos problemas são as pessoas. A Engenharia de Requisitos utiliza ciências sociais e cognitivas para prover fundamentos teóricos e técnicas práticas para extrair e modelar requisitos:

- Psicologia Cognitiva fornece uma compreensão das dificuldades que as pessoas podem ter ao descrever suas necessidades;
- Antropologia provê uma abordagem metodológica observando atividades humanas que ajudam a desenvolver uma compreensão mais rica de como sistemas de computador podem ajudar ou podem impedir essas atividades;
- Sociologia auxilia na compreensão das mudanças políticas e culturais causadas por informatização. A introdução de um sistema de computador causa mudanças na natureza do trabalho dentro de uma organização e pode afetar a estrutura e caminhos de comunicação dentro desta;
- Lingüística é importante porque a Engenharia de Requisitos é em grande parte sobre comunicação. Análises lingüísticas mudaram o modo como o idioma inglês é usado em especificações para, por exemplo, evitar ambigüidade e melhorar o entendimento. Também podem ser usadas ferramentas de lingüística em eliciações de requisitos, por exemplo, para padrões de comunicação de análise dentro de uma organização.

As visões acima explanadas sugerem que a Engenharia de Requisitos está dentro de um amplo contexto, que envolve aspectos organizacionais, gerenciais, técnicos, econômicos, sociais e cognitivos.

2.4. PROCESSO DE ENGENHARIA DE REQUISITOS

Kotonya e Sommerville [KOT98] afirmam não haver um único processo de Engenharia de Requisitos que atenda a todas as organizações. Cada organização deve buscar seu próprio processo, apropriado para os tipos de sistema que desenvolve, sua cultura organizacional, habilidades e experiências dos envolvidos no processo de Engenharia de Requisitos. Apresentamos a seguir, vários modelos encontrados na literatura.

Kotonya e Sommerville [KOT98] definem Processo de Engenharia de Requisitos como um conjunto estruturado de atividades que são seguidas para derivar, validar e manter um documento de requisitos de sistema. A Figura 2 mostra este processo.

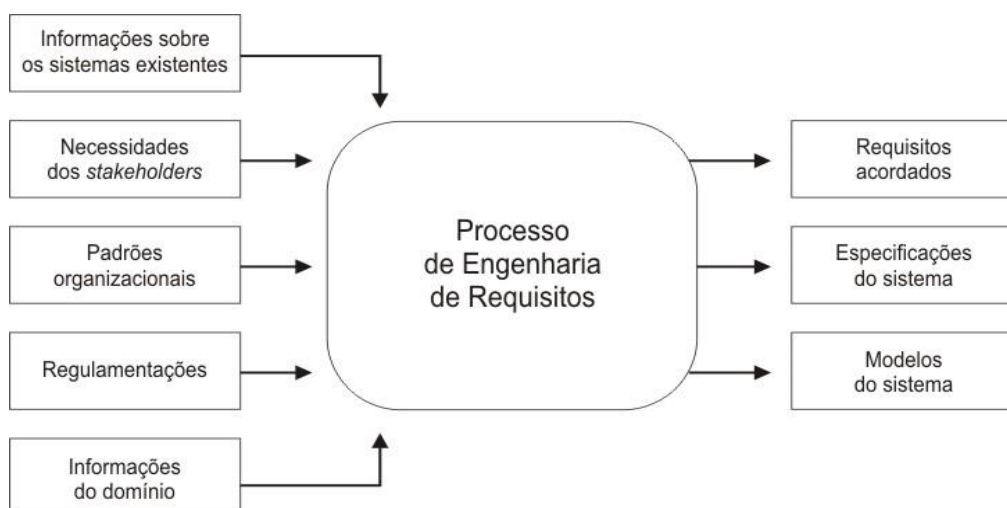


Figura 2 – Processo de Engenharia de Requisitos [KOT98]

As entradas deste processo são:

- a. Informações sobre os sistemas existentes, sobre a funcionalidade do sistema que está sendo substituído ou sobre outros sistemas que interage com o sistema em especificação.
- b. Necessidades dos *stakeholders* – descrição do que os *stakeholders* necessitam do sistema, para apoiar seus trabalhos.
- c. Padrões organizacionais – padrões usados na organização relativos ao desenvolvimento de sistemas e ao gerenciamento da qualidade.
- d. Regulamentações – regulamentações externas como saúde e segurança, que se aplicam ao sistema.
- e. Informações do domínio – informações gerais sobre o domínio da aplicação.

Temos as seguintes saídas deste processo:

- a. Requisitos acordados – uma descrição dos requisitos do sistema que está entendido pelos *stakeholders* e que foi aceito por eles.
- b. Especificação do sistema – a mais detalhada especificação das funcionalidades do sistema, que pode ser produzida em alguns casos.
- c. Modelos do sistema – um conjunto de modelos como de fluxo de dados, de objetos, de processo, etc. que descrevem o sistema de diferentes perspectivas.

Pohl [POH93] propõe três dimensões para o processo de Engenharia de Requisitos:

- a. Representacional – são assuntos que envolvem o conjunto de métodos de representação de linguagem informal, especificações naturais para especificação formal.
- b. Domínio Social – são assuntos que cercam o processo social envolvido na iteratividade, elicitación cooperativa de informações para construir requisitos.
- c. Domínio Cognitivo – são assuntos concernentes à compreensão do próprio problema dentro do domínio de problema que usa vários modelos conceituais.

Loucopoulos e Karakostas sugerem que o processo de Engenharia de Requisitos pode ser dividido em três subprocessos, quais sejam: elicitación, especificação e validação, que transacionam com duas entidades externas, o usuário e o domínio do problema, correspondendo a três importantes preocupações nesta área: compreensão, descrição e concordância de um problema [LOU95]. Dawson e Swatman [DAW99] revisaram este modelo, com notações específicas para interações sociais e cognitivas. O próximo capítulo detalha o modelo proposto por Kotonya e Sommerville [KOT98].

2.5. FASES DA ENGENHARIA DE REQUISITOS

A importância da Engenharia de Requisitos no contexto de desenvolvimento de um produto de *software* está no fato de que a correta identificação e documentação dos requisitos, é primordial para o sucesso do *software*. Para Kotonya e Sommerville, a Engenharia de Requisitos ocorre de forma intensa, nas primeiras etapas do ciclo de vida do *software*, abrangendo a engenharia de sistemas, análise e projeto, embora possa se estender para as demais etapas [KOT98]. Nuseibeh em [NUS00] concorda com esta afirmação, complementando que Engenharia de Requisitos é considerada freqüentemente como uma atividade inicial no desenvolvimento de um sistema, embora normalmente os requisitos mudam durante o desenvolvimento e evoluem depois do sistema estar em operação durante algum tempo.

A Figura 3 mostra o ciclo de vida clássico ou em cascata [PRE95] (o mais tradicional na engenharia de *software*) e demonstra as fases onde ocorre mais intensamente a Engenharia de Requisitos.

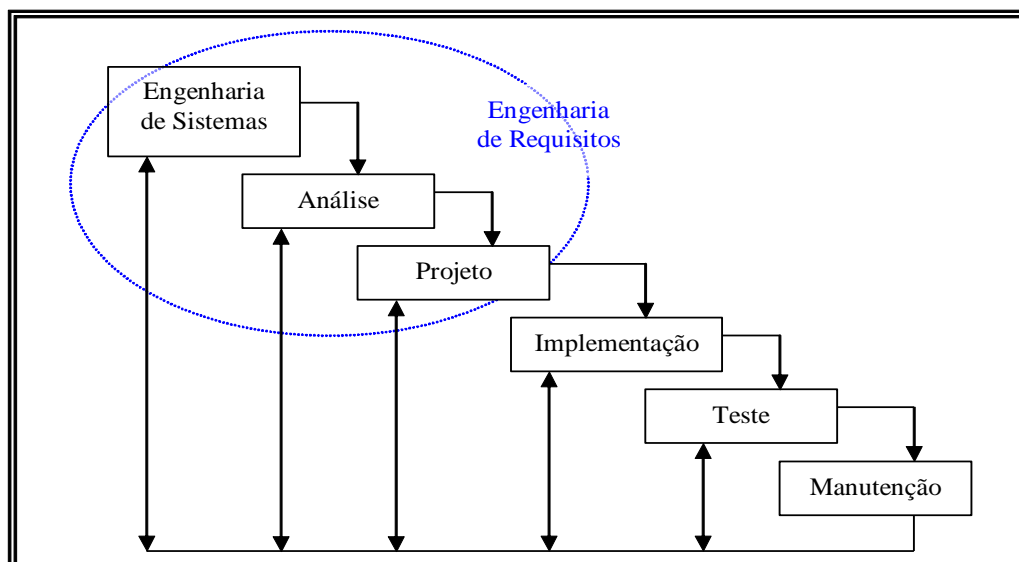


Figura 3 – Contexto do processo de Engenharia de Requisitos [MAR01]

Kotonya e Sommerville [KOT98] apresentam o processo de Engenharia de Requisitos ocorrendo de forma interativa em quatro atividades ou fases: Elicitação, Análise, Especificação e Validação de Requisitos. A Figura 4 mostra um modelo espiral, em que as diferentes atividades em Engenharia de Requisitos são repetidas até que uma decisão seja tomada e o documento de requisito seja aceito.

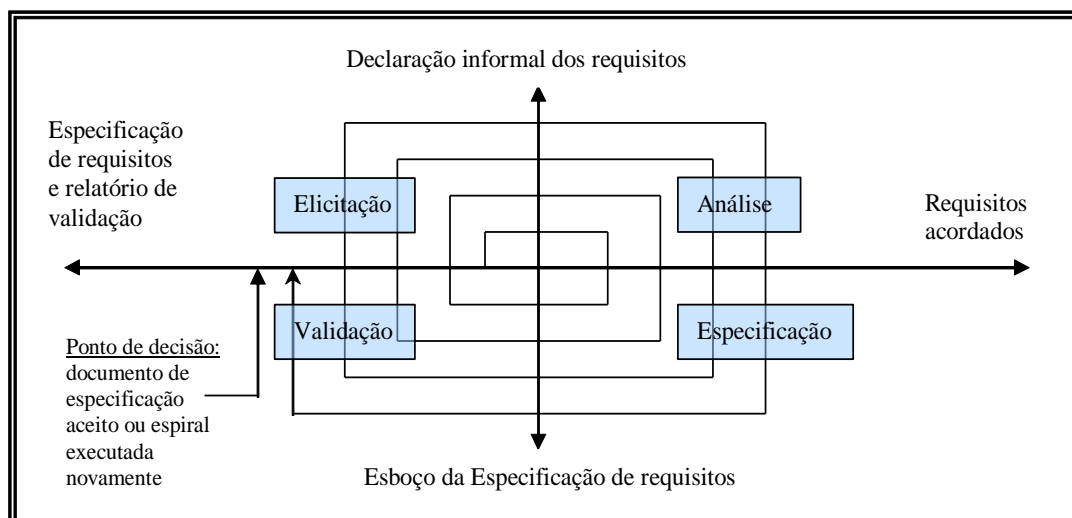


Figura 4 – Modelo espiral do processo de Engenharia de Requisitos [KOT98]

Há ainda uma quinta atividade, Gerenciamento de Requisitos, que ocorre durante todo o ciclo, sendo realizado em conjunto com as outras atividades de Engenharia de Requisitos, que será descrita no Capítulo 3.

2.5.1. ELICITAÇÃO DE REQUISITOS

Elicitação de Requisitos é o nome usualmente dado para a atividade da Engenharia de Requisitos que envolvem o descobrimento ou obtenção de requisitos. Goguen considera que é preferível usar o termo “elicitación” do que “captura”, para evitar a sugestão de que requisitos são simplesmente coletados através de perguntas diretas [GOG97]. Já Kotonya e Sommerville [KOT98] entendem que o termo elicitación sugere que o processo é uma simples transferência de conhecimento, quando o engenheiro de requisitos elicita e documenta os conhecimentos dos clientes. Entretanto, afirmam que na realidade o processo é muito mais complexo, pois raramente o cliente tem uma clara visão dos requisitos, diferentes pessoas na organização têm requisitos conflitantes, há limitações tecnológicas para atender os requisitos, etc..

A elicitación de requisitos é a primeira atividade no processo de Engenharia de Requisitos onde se busca entender quais são as necessidades do usuário, devendo ser atendidas pelo *software* que será desenvolvido [THA97]. As informações coletadas durante a elicitación de requisitos têm que ser interpretadas, analisadas, modeladas e validadas [NUS00]. Como nesta fase se busca descobrir os requisitos do sistema, normalmente pouco claros e confusos no início do desenvolvimento de um sistema de *software*, os *stakeholders* devem trabalhar em conjunto para entender os problemas a serem solucionados, focando principalmente os serviços que o sistema deve oferecer [CAS00]. Este trabalho requer uma análise cuidadosa da organização onde o *software* será implantado, uma análise do domínio da aplicação e dos processos de negócio onde o *software* será utilizado [KOT98].

Seguindo essa linha de pensamento, propõem quatro dimensões para Engenharia de Requisitos, como mostra a Figura 5.



Figura 5 – Componentes da Elicitação de Requisitos [KOT98]

Estes componentes tem a seguinte definição:

- a. Domínio da Aplicação – é o conhecimento geral da área onde o sistema está sendo aplicado. Por exemplo, para entender os requisitos de um sistema hospitalar, você deve conhecer como funciona um hospital;
- b. Problema a ser solucionado – os detalhes dos problemas específicos do cliente onde o sistema será implantado, devem ser minuciosamente entendidos. O entendimento dos problemas aumenta o conhecimento do domínio da aplicação;
- c. Contexto do negócio – sistemas são geralmente implantados para contribuir de algum modo para o desenvolvimento do negócio ou da organização. Deve ser entendido como o sistema irá interagir e afetar as diferentes áreas do negócio e como pode contribuir para atingir plenamente os objetivos da organização;
- d. Necessidades e restrições dos envolvidos – stakeholders são as pessoas afetadas diretamente ou indiretamente pelo sistema, como usuários finais ou gerentes de departamentos. Suas necessidades

específicas devem ser entendidas, para que o sistema possa apoiá-los em seu trabalho.

2.5.2. PROBLEMAS NA ELICITAÇÃO DE REQUISITOS

Devido à natureza multidimensional dos processos de Elicitação de Requisitos, muitos problemas são encontrados pelos Engenheiros de Requisitos, quando tentam entender os requisitos dos sistemas. Estes problemas, elencados em uma lista de 10 problemas segundo McDermid em [McD89] podem ser agrupados em três categorias: problemas de escopo, problemas de entendimento dentro dos grupos e entre os grupos (usuários e desenvolvedores) e problemas de volatilidade [CHR92].

a. Problemas de escopo:

- Os limites do sistema não estão bem definidos;
- Há informações desnecessárias ao sistema.

b. Problemas de entendimento:

- Usuários têm entendimento incompleto de suas necessidades;
- Usuários conhecem pouco sobre a capacidade e limitações de computadores;
- Analistas têm pouco conhecimento do domínio do problema;
- Usuários e analistas falam “línguas” diferentes;
- Omissão de informações “óbvias”;
- Visões conflitantes entre os usuários;
- Requisitos freqüentemente vagos e não testáveis.

c. Problemas de volatilidade:

- Requisitos evoluem no tempo, devido a fatores como:
 - ✓ Maior clareza do usuário em relação ao sistema;
 - ✓ Mudanças no negócio;
 - ✓ Mudanças legais e tecnológicas.

Nos Problemas de escopo, os requisitos podem ter pouca ou muita informação e as técnicas de elicitação precisam ser suficientes para estabelecer os limites e objetivos do sistema projetado, evitando contextos que conduzam a requisitos incompletos, não verificáveis, desnecessários e sem uso.

Problemas de entendimento durante a elicitação, podem conduzir a requisitos que são ambíguos, incompletos, incompatíveis e até mesmo incorretos, porque eles não endereçam as verdadeiras necessidades dos *stakeholders*. Podemos classificá-los em três assuntos:

- a. As comunidades envolvidas em elicitação possuem uma variedade de bases e níveis de experiências, de forma que o conhecimento comum a um grupo pode ser completamente diferente de outro;
- b. O idioma que expressa os requisitos dos *stakeholders*, podem ser muito formais ou informais para satisfazer as necessidades de cada grupo, novamente por causa da diversidade das comunidades;
- c. A grande quantia de informações colecionadas durante a elicitação necessita ser estruturada de algum modo. A compreensão desta estrutura depende das características das comunidades dos *stakeholders*.

Problemas de volatilidade estão relacionados à mudança dos requisitos durante o período que leva para desenvolver um sistema. As necessidades dos usuários podem amadurecer, devido ao aumento do conhecimento ganho nas atividades de desenvolvimento, ou os requisitos podem mudar devido a novas necessidades por causa de pressões organizacionais ou ambientais imprevistas. Caso estas mudanças não sejam contempladas, os requisitos originais ficarão incompletos, incompatíveis com a nova situação, e potencialmente sem uso, porque ele detém a informação que ficou obsoleta desde então.

Outra causa da volatilidade de requisitos é que os requisitos são produtos das contribuições de muitos indivíduos e estes podem ter necessidades e

objetivos contraditórios. A complexidade organizacional também causa volatilidade nos requisitos, pois objetivos, políticas, estruturas, *stakeholders* e desenvolvedores podem mudar durante o desenvolvimento do sistema.

2.5.3. TÉCNICAS DE ELICITAÇÃO DE REQUISITOS

Um dos objetivos da Engenharia de Requisitos é ultrapassar as barreiras de comunicação entre clientes, usuários e engenheiros de requisitos, para que os requisitos possam ser capturados e modelados corretamente. Em grande parte a técnica de elicitação usada é dirigida pela escolha do esquema de modelagem e vice-versa: muitos esquemas de modelagem insinuam o uso de tipos particulares de técnicas de elicitação [NUS00]. Abordaremos a seguir algumas das técnicas mais conhecidas e utilizadas, como entrevistas, JAD, pontos de vista (*viewpoints*), cenários, casos de uso, etnografia, prototipação, baseada na Teoria da Atividade, entre outras.

Entrevistas

É provavelmente a técnica mais comum usada para colher informações durante a elicitação de requisitos. Engenheiros de Requisitos entrevistam clientes para definir os objetivos gerais e restrições que o *software* deverá ter. A entrevista deve ser feita de forma objetiva, visando obter o máximo de informações do cliente. Diversas sessões de entrevistas podem ser necessárias para elicitação dos requisitos.

Segundo Kotonya e Sommerville [KOT98] pode haver dois tipos de entrevistas:

- a. Entrevista fechada - os engenheiros de requisitos trabalham com um conjunto de perguntas pré-definidas;
- b. Entrevista aberta - os engenheiros de requisitos trabalham sem agenda pré-definida e discutem de forma aberta, o que os usuários pretendem com o sistema.

Este método não é apropriado para o entendimento do domínio da aplicação, porque os usuários têm dificuldades em explicar suas necessidades usando sua própria terminologia e omitem requisitos que em sua visão são óbvios e que deveriam ser do conhecimento do engenheiro de requisito. Também não é apropriado para o entendimento de questões organizacionais, devido à interferência de fatores políticos e sociais.

JAD

Joint Application Design é uma técnica desenvolvida pela IBM (Canadá) no final da década de 1970. Trata-se de uma técnica orientada ao trabalho em grupo, que oferece um ambiente para clientes e desenvolvedores trabalharem como uma equipe, compartilhando informações e idéias sobre determinado tema ou assunto. Faz uso de sessões de *brainstorm* mediada por um facilitador, composta das etapas de orientação inicial, familiarização com a área de negócios e seu sistema de aplicação relacionado, preparação do material para a sessão JAD e condução da sessão. O processo JAD tem as seguintes etapas:

- a. Orientação inicial – trata das questões de definição de aspectos globais do projeto, especificando itens como: finalidade do projeto, escopo do projeto e áreas funcionais envolvidas, objetivos a serem alcançados pela sessão JAD, suposições técnicas e de negócio que afetam o projeto, fatores críticos de sucesso, entre outros;
- b. Familiarização com a área de negócio e seu sistema de aplicação relacionado – é a etapa onde se realiza uma análise dos processos atuais de negócios e de seus fluxos de informações. As informações obtidas durante esta fase, auxiliarão o facilitador da sessão JAD, que é o líder do processo, a entender melhor o contexto da aplicação do projeto;
- c. Preparação do material para a sessão JAD – consiste na elaboração de um modelo simplificado do sistema a ser desenvolvido. Este modelo será desenvolvido a partir da lista de tarefas de negócio identificadas na etapa anterior. Dependendo da natureza do projeto, o modelo construído

pode consistir de esboços de telas e relatórios que serão revisados durante a sessão JAD;

- d. Condução da sessão JAD – será iniciada com a apresentação dos objetivos e da estrutura básica dentro da qual os participantes da equipe deverão operar. O facilitador assumirá um papel chave neste processo, que irá conduzir as discussões e mediar conflitos durante a sessão. À medida que as discussões progredirem, os requisitos levantados deverão ser transcritos pelo facilitador ou por um auxiliar.

Como vantagens desta técnica estão a promoção de cooperação, compreensão e trabalho de equipe entre clientes, usuários e desenvolvedores. Desenvolvedores ajudam os usuários a formularem problemas e exploram soluções, enquanto os usuários compartilham propriedades dos requisitos e documentos associados. Nessas reuniões acontece a facilitação e ajuda visual, aumentando a geração de idéias, avaliações, comunicações e consensos. Um problema reconhecido é que o sucesso depende fortemente das habilidades do facilitador em conduzir e integrar as informações recebidas.

Pontos de Vista

Independente do tamanho do sistema, de sua complexidade e do tipo organização onde este será desenvolvido, normalmente há diferentes tipos de *stakeholders*, com diferentes interesses e visões. Segundo Sommerville [SOM03] os diferentes pontos de vista a respeito de um problema, ‘vêm’ o problema de modos diferentes, contudo, suas perspectivas não são inteiramente independentes, mas em geral apresentam alguma duplicidade, de modo que apresentam requisitos comuns. As abordagens orientadas a pontos de vista, na Engenharia de Requisitos, reconhecem estes diferentes pontos de vista e os utilizam para estruturar e organizar o processo de levantamento e os próprios requisitos.

Há vários métodos que tratam Pontos de Vista:

- a. Os métodos SADT (*Structured Analysis and Design Technique*) [ROS77] e o CORE (*Controlled Requirements Expression*) [MUL79] tratam pontos de vista para produção ou consumo de dados onde a análise envolve identificar todos os pontos de vista, quais dados serão produzidos ou consumidos e quais processamentos serão realizados. O CORE foi o primeiro método a tratar Pontos de Vistas explícitos.
- b. O método VOSE (*Viewpoint-Oriented System Engineering*) [FIN92] é considerado um tipo particular de modelo de sistema, onde diferentes engenheiros devem desenvolver um modelo de relacionamento de entidades, um modelo de máquinas de estados, entre outros, de forma que cada abordagem de análise descobre diferentes aspectos sobre o sistema que está sendo analisado.
- c. O método VORD (*Viewpoint-Oriented Requirements Definition – Definição de Requisitos Orientado a Pontos de Vista*) de Kotonya e Sommerville [KOT96] foi projetado como um *framework* orientado a serviços, para a eliciação e análise de requisitos.

Cada um dos modelos de ponto de vista apresentam vantagens e desvantagens. Os pontos de vista como o SADT e CORE são particularmente interessantes para a descoberta de conflitos detalhados entre os requisitos [EAS96], mas são menos adequados para estruturar o processo de análise de requisitos, uma vez que não há relação simples entre os pontos de vista e os *stakeholders* do sistema [SOM03].

Para melhor entendimento da técnica de Pontos de Vista, explanamos abaixo os principais estágios do método VORD:

- a. Identificação de ponto de vista, que envolve descobrir os pontos de vista que utilizam serviços do sistema e identificar os serviços específicos fornecidos para cada ponto de vista;

- b. Estruturação de pontos de vista, que envolve agrupar pontos de vista relacionados, segundo uma hierarquia. Serviços comuns estão localizados nos níveis mais altos da hierarquia e herdados por pontos de vista de nível inferior;
- c. Documentação do ponto de vista, que envolve refinar a descrição dos pontos de vista e serviços identificados;
- d. Mapeamento de sistema conforme pontos de vista, que envolve identificar objetos em um projeto orientado a objetos, utilizando as informações de serviço que estão encapsuladas nos pontos de vista.

As informações sobre os pontos de vista e serviços no VORD são coletadas utilizando-se formulários-padrão. Em uma reunião JAD, deve-se tentar identificar os pontos de vista, os serviços do sistema, as entradas de dados, os Requisitos Não-Funcionais, os eventos de controle e as exceções. Identificam-se os pontos de vista (bolhas pretas) e os serviços (bolhas cinza) conforme mostra a Figura 6. Os serviços devem ser alocados aos pontos de vista e serviços não alocados, podem sugerir pontos de vista que não foram identificados e vários pontos de vista podem alocar o mesmo serviço.



Figura 6 – Reunião JAD para a identificação de Pontos de Vista [SOM03]

Na próxima fase, requisitos são coletados em consultas aos *stakeholders* associados com cada ponto de vista e discutidos com os clientes. *Templates* de pontos de vista e de serviços são desenvolvidos para todos os pontos de vista e serviços identificados e as informações podem sofrer uma verificação cruzada, para descobrir erros na análise e conflitos de requisitos. Normalmente devido à geração de grande número de informações, o VORD é usado com o apoio de ferramentas CASE.

Etnometodologia

A etnometodologia é uma técnica de elicitação de requisitos que pode ser utilizada para entendimento dos requisitos de sistema inseridos em um contexto social organizacional. Consiste de se inserir um engenheiro de requisitos no ambiente de trabalho em que o sistema será utilizado, onde o mesmo observa o trabalho diário sendo executado e anota as tarefas em que os participantes estão envolvidos [KOT98].

As pessoas geralmente têm dificuldade de detalhar seu próprio trabalho, porque o trabalho para elas é sua segunda natureza e elas normalmente não tem facilidade de compreender a relação dele com outras atividades na organização. Os fatores sociais e organizacionais que afetam o trabalho podem ficar claros quando examinados por um observador imparcial.

Segundo Sommerville [SOM03] a etnografia é particularmente eficaz na descoberta de dois tipos de requisitos:

- a. Os requisitos derivados da maneira como as pessoas realmente trabalham, em vez da maneira pela qual as definições de processo dizem como elas deveriam trabalhar.
- b. Os requisitos derivados da cooperação e conscientização das atividades de outras pessoas.

Hughes apresenta a etnografia a partir de três pontos de vista [HUG95]:

- a. Ambiente de trabalho - descreve o contexto e a localização física onde o trabalho se realiza e como as pessoas utilizam objetos para executar suas tarefas.
- b. Perspectivas organizacionais e sociais - descrevem a experiência de trabalho do dia a dia conforme vista pelos diferentes agentes que estão envolvidos no trabalho.
- c. Fluxo de trabalho - descreve o trabalho realizado a partir de uma série de atividades com as informações fluindo de uma atividade para a outra.

Sommerville [SOM03] diz que a etnografia pode ser combinada com outras técnicas de elicitação de requisitos como a prototipação. A etnografia informa o desenvolvimento do protótipo, de modo que um número menor de ciclos de refinamento do protótipo seja necessário. Os estudos de etnografia podem revelar importantes detalhes de processos omitidos por outras técnicas de elicitação de requisitos. Contudo, devido ao seu enfoque no usuário final, essa abordagem não é apropriada para descobrir requisitos organizacionais e de domínio. A etnografia não é, portanto, uma abordagem completa para a obtenção de requisitos, devendo ser utilizada com outras abordagens.

Cenários

O uso de cenários para elicitar requisitos tem sido recomendado por muitos autores [ZOR95] [POT94] [BRE99]. Cenários descrevem situações tendo em conta aspectos de uso, permitindo: conhecer o problema, unificar critérios, ganhar o comprometimento com os clientes e usuários, organizar o conhecimento e treinar novos participantes [RID00]. Segundo Breitman [BRE00], nos últimos anos é crescente o interesse pela técnica de Cenários por parte da comunidade de ER, em razão da introdução dos casos de uso proposto por Jacobson [JAC92]. A autora descreve Cenários como situações do macro-sistema e suas relações com

o sistema a ser construído, podendo ser utilizado para descrever as interações entre os diversos objetos presentes no sistema e apresenta evolução durante o processo de desenvolvimento do *software*.

Rolland e outros afirmam que Cenários podem ser desde simples descrições em linguagem natural até modelos mais complexos, contendo informação comportamental (ações, eventos e atividades) e até objetos (entidades, dados e atributos) [ROL98]. Segundo Zorman Cenários descrevem as situações do mundo real, que envolvem agentes interagindo dentro de um determinado contexto [ZOR95]. Breitman e Leite [BRE99] acreditam que a noção de Cenários poderia ser estendida de modo a incluir a descoberta e validação de informações além do escopo dos requisitos, i.e., decisões de desenho, seqüências para testes e partes do código.

A seguir apresentamos alguns métodos existentes na literatura para Cenários:

a. Métodos para Análise de Requisitos Baseados em Cenários (SCRAM) [SUT98]:

- É uma combinação de técnicas de entrevistas, prototipagem, cenários e *rationale*;
- São utilizadas entrevistas e técnicas para descobrimento dos fatos, de modo a eliciar dados suficiente para a construção de um protótipo;
- São construídos protótipos utilizando ferramentas comerciais como *Macromedia Director* ou VB (*Visual Basic*);
- A validação com clientes são feitas através dos protótipos. Este estágio é gravado em vídeo para análise futura;
- A análise é feita através de uma reunião JAD onde os cenários são utilizados para descrever como os usuários conduzem suas tarefas.

b. Ciclos de Questionamento (Inquiry Cycle) [POT94]:

- A estratégia proposta por Potts é um modelo para a descrição e suporte de discussões sobre os requisitos do sistema, baseada em ciclos consecutivos de questionamento e refinamento destes requisitos;
- Baseado em objetivos, onde os cenários são derivados a partir da identificação e decomposição destes;
- O ciclo é iniciado a partir da documentação dos cenários e de uma lista de requisitos, que são discutidas através de um processo de questionamento onde respostas e justificativas (*rationale*) são registradas e o processo só termina quando uma decisão é tomada. Se esta for negativa, a documentação é modificada e pode resultar em um novo ciclo.

c. Cenários como apoio à visualização de requisitos [ZOR95]:

- Discute a dificuldade na comunicação entre clientes e desenvolvedores, apontando ocorrências de mistura de terminologia e erros de interpretação;
- Usuários especialistas e desenvolvedores colaboram com uma representação e vocabulário comum para a elaboração de cenários;
- Linguagens mais formais devem ser utilizadas por pessoas da mesma área;
- Cenários são definidos como descrições parciais do sistema e do comportamento do macrosistema;
- A autora sugere uma ferramenta gráfica para a captura e representação dos cenários (REBUS). Cada cenário é identificado pelo nome, categoria e descrição.

Casos de Uso

Casos de Uso são técnicas baseadas em cenários para obtenção de requisitos. Um caso de uso é uma unidade de funcionalidade apoiada pelo

sistema ou por uma classe do sistema, manifestada por uma seqüência de troca de mensagens entre o sistema e os atores externos ao sistema [JAC92].

Segundo Rumbaugh [RUM94] um caso de uso descreve as possíveis seqüências de interações entre o sistema e um ou mais atores, respondendo a algum estímulo inicial de um dos atores. Afirma ainda, que caso de uso não pode ser considerado um simples cenário, mas sim a descrição de um conjunto de potenciais cenários, cada qual iniciando com algum evento inicial de um ator para o sistema e seguindo uma seqüência de transações para sua conclusão lógica.

Caso de uso, utiliza agentes ou atores, representados por bonecos, interagindo em um ambiente, onde cada classe de interação é representada por uma elipse com um nome [SOM03].

A Figura 7 exemplifica um caso de uso, onde o ator “Cliente” solicita a uma instituição financeira um cartão de crédito e opcionalmente também pode solicitar um seguro de vida.

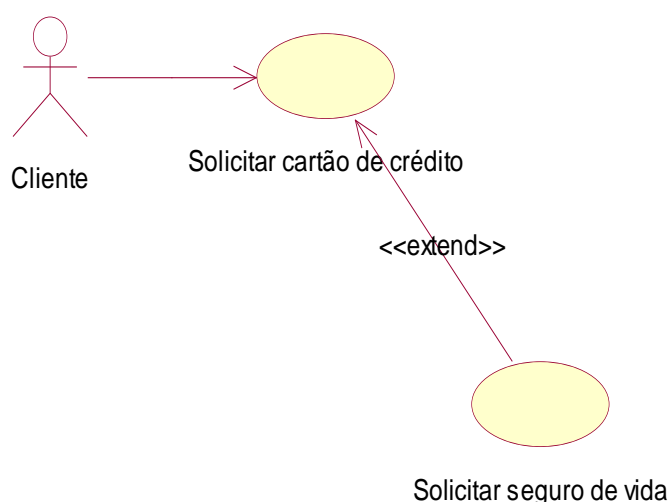


Figura 7 – Diagrama de Caso de Uso para cartão de crédito

Baseada na Teoria da Atividade

A evolução natural dos processos de Engenharia de Requisitos tem levado alguns pesquisadores a se preocuparem com o contexto social na qual essa ocorre, para resolução de problemas encontrados na elicitação de requisitos [GOG97] [KOL02] [MAR01] [MYL99] [NUS00]. Seguindo esta linha de pensamento, Martins [MAR01] entendendo que problemas essenciais inerentes a elicitação de requisitos, como o usuário não saber exatamente o que deseja, dificuldades na comunicação entre desenvolvedores e usuários e a natureza mutante dos requisitos, não podem ser superados somente em uma abordagem puramente tecnológica, propõe uma nova técnica de elicitação baseada na Teoria da Atividade: Metodologia de Elicitação de Requisitos de *Software* Baseada na Teoria da Atividade (META). Em continuidade a esta abordagem, Franceto [FRA04] [FRA05] desenvolveu uma ferramenta automatizada para apoiar a técnica META, que tem como objetivo levantar, armazenar e consultar dados inseridos de forma gráfica (diagrama de Engeström).

O termo “Teoria da Atividade” surgiu durante as décadas de 1920 e 1930, dentro da escola histórico-cultural Soviética de psicologia [KAP97] e é oriunda da filosofia clássica alemã dos séculos XVIII e XIX e dos escritos de Marx e Engels, que elaboraram o conceito de atividade. Os princípios básicos da Teoria da Atividade, segundo Nardi e Kaptelinin [NAR96] [KAP97] podem ser resumidos em:

- a. Princípio da unidade entre consciência e atividade - é considerado o princípio fundamental da Teoria da Atividade onde consciência e atividade são concebidas de forma integrada. A consciência significa a mente humana como um todo, e a atividade a interação humana com sua realidade objetiva.
- b. Princípio da orientação a objetos - este princípio enfoca a abordagem da Teoria da Atividade para o ambiente no qual seres humanos interagem.
- c. Princípio da estrutura hierárquica da atividade - a Teoria da Atividade diferencia os procedimentos humanos em vários níveis (atividade, ação

e operação), levando em conta os objetivos para os quais estes procedimentos são orientados.

- d. Princípio da internalização-externalização - descreve os mecanismos básicos da origem dos processos mentais. Ele declara que processos mentais são derivados das ações externas através do curso da internalização. Internalização é o processo de absorção de informações (nas suas diversas formas) realizado pela mente humana, que ocorre a partir do contato com o ambiente em que a pessoa está inserida. A externalização é o processo inverso da internalização, manifestado através de atos, de tal forma que eles possam ser verificados e corrigidos se necessário.
- e. Princípio da mediação - a atividade humana é mediada por um número de ferramentas, tanto externas como internas. As ferramentas são “veículos” da experiência social e do conhecimento cultural.
- f. Princípio do desenvolvimento - de acordo com a Teoria da Atividade, entender um fenômeno significa conhecer como ele se desenvolveu até sua forma atual, pois ao longo do tempo ele sofre alterações.

Atividade é uma unidade de eliciação de requisitos que oferece um contexto mínimo para o entendimento de um conjunto de ações cooperantes que agem sobre um ou mais objetos, transformando-os num resultado [MAR01]. No nível individual, uma atividade possui três elementos: *sujeito*, *objeto* e *ferramenta de mediação*. O *sujeito* é o agente que atua sobre o objeto da atividade. O *objeto* é o elemento para o qual as ações da atividade estão direcionadas e a *ferramenta de mediação*, faz o papel de mediação, usadas na transformação do objeto [MAR99].

A Figura 8 representa a estrutura de relacionamento no nível individual, entre o sujeito e o objeto no contexto de uma atividade.

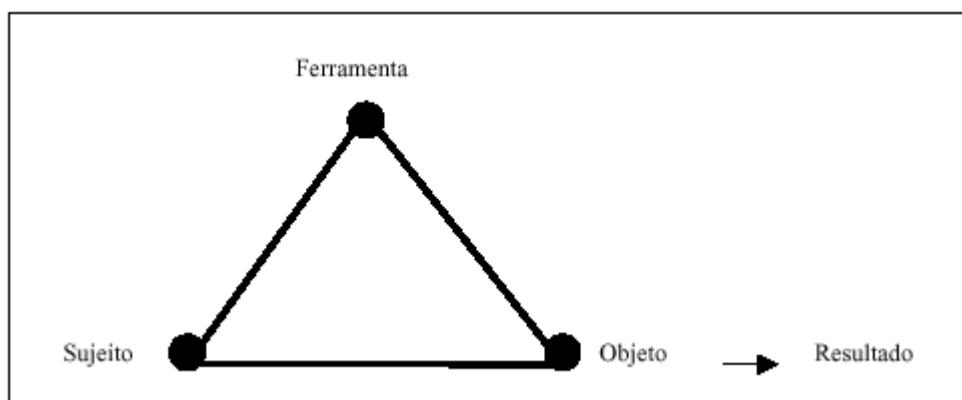


Figura 8 – Relacionamento entre sujeito e objeto no nível individual [KUU96]

A estrutura ao nível individual, é simples demais para representar as considerações de relações sistêmicas existentes entre o sujeito e o seu ambiente, uma vez que essas relações são encontradas em muitas atividades. Assim, novos elementos devem ser adicionados:

- a. Comunidade - uma comunidade é formada por todos os sujeitos que compartilham um mesmo objeto.
- b. Regras - enquanto uma forma de mediação entre o sujeito e a comunidade, são normas implícitas ou explícitas estabelecidas por convenções e relações sociais dentro da comunidade.
- c. Divisão de trabalho - enquanto forma de mediação entre a comunidade e o objeto, se refere a forma de organização de uma comunidade, relacionada ao processo de transformação de um objeto para um resultado.

Todas as formas de mediação (ferramentas, regras e divisão de trabalho) possuem um desenvolvimento histórico próprio, com características particulares relacionadas ao contexto em que foram desenvolvidas. A Figura 9 mostra um modelo sistêmico da atividade.

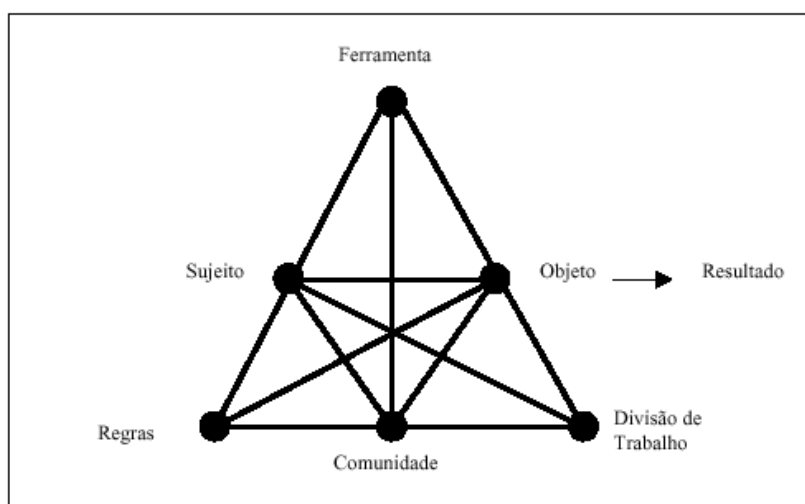


Figura 9 – Modelo sistêmico da atividade [ENG87]

Finalmente, uma atividade é decomposta em *ações*, e cada ação é decomposta em *operações*, conforme mostra a Figura 10 abaixo.

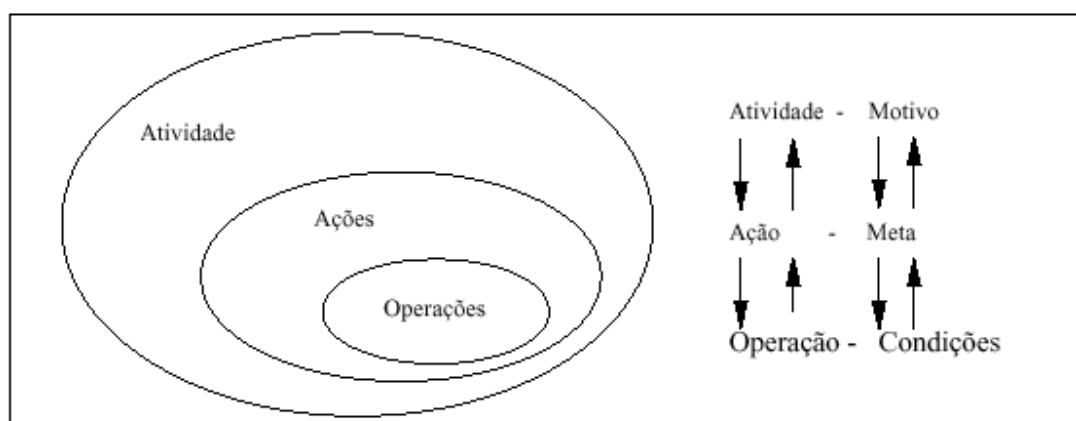


Figura 10 – Níveis hierárquicos de uma atividade [MAR01]

Uma atividade concebida como tal num dado momento, passou por um processo de evolução, onde ações e operações podem ter sido criadas, eliminadas e transformadas para que a atividade chegasse ao seu “formato” atual [NAR96]. Enquanto uma atividade é orientada por um motivo, as ações são orientadas a metas, e as operações orientadas a condições. Uma atividade é realizada através de ações cooperativas ou individuais, podendo se estabelecer cadeias ou redes de ações que estão relacionadas umas com as outras para atingirem a mesma meta.

A ação é planejada antes de sua execução efetiva, diferentemente de uma operação, que é executada de forma automática, sem um planejamento prévio,

bastando apenas uma análise das condições atuais para a sua execução. O planejamento de uma ação é feito de forma consciente, usando-se algum modelo mental para isso, quanto melhor o modelo mais sucesso terá a ação. Este planejamento para a execução de uma ação é chamado de orientação. Quando uma ação é realizada várias vezes e alcança um nível de maturidade suficiente para que ela possa ser executada automaticamente, ou seja, sem um planejamento prévio, então ela passa para o nível de operação. Dessa forma, uma operação foi uma ação que se tornou comum no contexto de uma atividade, pois é executada com um alto grau de repetição dentro deste contexto.

Segundo a Teoria da Atividade, um contexto mínimo é dado quando a ação humana é analisada dentro de uma atividade. Dessa forma, a atividade passa a ser vista como a unidade básica de análise de situações. Para se obter adequadamente os requisitos do *software*, é preciso entender, entre outras coisas, as atividades realizadas pelos agentes envolvidos no sistema em que o futuro *software* será implementado.

Se compararmos o conceito de Cenário e Atividade, veremos que há muitos elementos equivalentes. Entretanto, a atividade tem mais elementos dentro de sua estrutura do que um cenário, por exemplo: comunidade, ferramenta de mediação, divisão de trabalho dentro da comunidade e operações.

2.5.4. ANÁLISE DE REQUISITOS

O objetivo da análise de requisitos é encontrar problemas nos requisitos preliminares obtidos com a elicitación de requisitos. Os problemas encontrados devem ser negociados entre os envolvidos, de tal forma que se obtenha um “acordo” que possa atender a todos os envolvidos (clientes, usuários, gerentes, desenvolvedores etc.), cujo resultado são os requisitos acordados.

Os principais problemas encontrados nos requisitos preliminares são:

- Requisitos conflitantes entre si;
- Requisitos incompletos;
- Requisitos ambíguos;

- Requisitos com informação sobre projeto e implementação;
- Requisitos desnecessários;
- Requisitos irreais.

Segundo Kotonya e Sommerville [KOT98] há uma forte iteração envolvendo elicitação e análise de requisitos, conforme mostra a Figura 11.

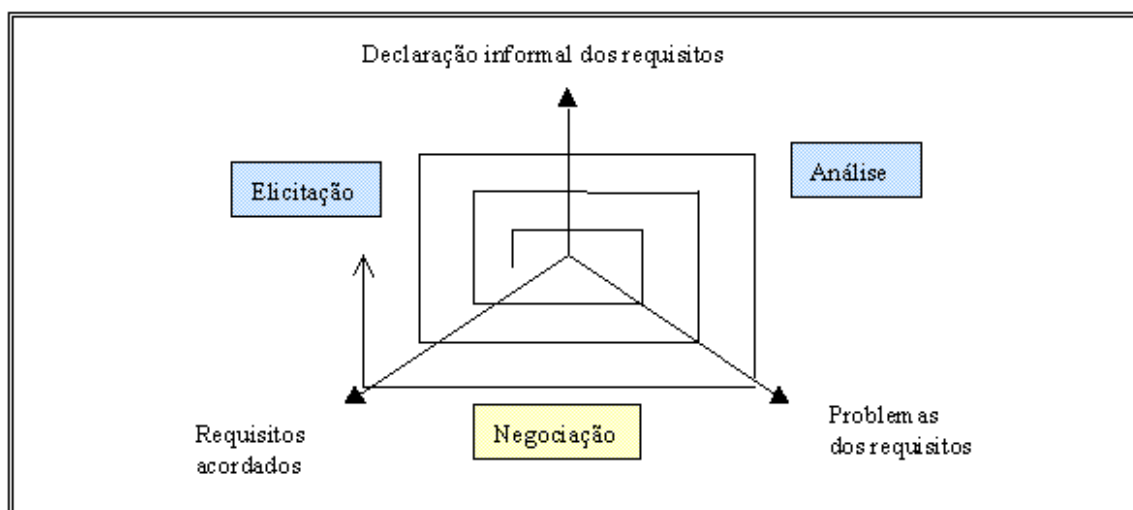
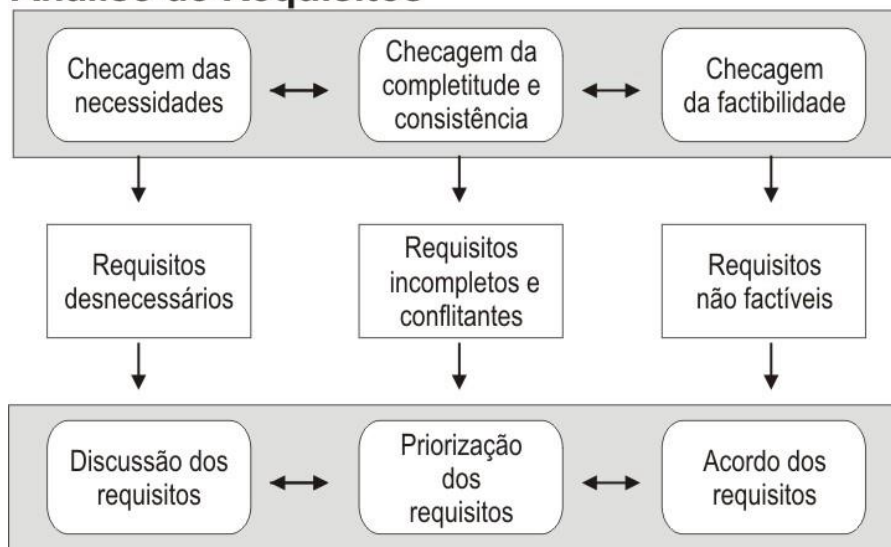


Figura 11 – A espiral de elicitação, análise e negociação [KOT98]

Elicitação e análise são atividades ligadas de tal forma, que alguma análise acaba acontecendo enquanto os requisitos vão sendo elicitados. Os problemas podem ser imediatamente levantados, discutidos e acordados, e nesse caso, elicitação e análise são atividades intercaladas. No entanto, uma análise de requisitos mais efetiva e detalhada ocorrerá quando uma certa quantidade de requisitos tiver sido elicitada e documentada.

Como resultado do processo de elicitação de requisitos, deveremos ter um documento informal que descreve os requisitos do sistema, o qual será analisado para descobrimento dos problemas e conflitos. Os *stakeholders* devem reunir-se para uma negociação, a fim de resolver os problemas e conflitos, chegando a um conjunto de requisitos acordados. O processo de análise de requisitos segundo Kotonya e Sommerville [KOT98] é mostrado na Figura 12.

Análise de Requisitos



Negociação de Requisitos

Figura 12 – O Processo de Análise e Negociação de Requisitos [KOT98]

As atividades típicas da análise de requisitos:

- a. Checagem das necessidades – em alguns casos podem ser propostos requisitos desnecessários, que não contribuirão para os objetivos da organização.
- b. Checagem da consistência e completitude – é feita uma checagem cruzada para consistência e completitude. Consistência significa que não há requisitos contraditórios e completitude significa que nenhum serviço ou restrição foi esquecido.
- c. Checagem da factibilidade – os requisitos são checados para garantir que são factíveis no contexto do orçamento e tempo de desenvolvimento do sistema.

As atividades resultam na identificação de um conjunto de requisitos que serão discutidos no processo de negociação, tendo os seguintes passos:

- a. Discussão dos requisitos – requisitos que são destacadamente problemáticos são discutidos e envolvidos com os *stakeholders*.

- b. Priorização dos requisitos – requisitos disputados são priorizados para identificar requisitos críticos na ajuda ao processo de tomada de decisão.
- c. Acordo dos requisitos – soluções para os problemas de requisitos são identificados e um conjunto de requisitos é acordado. Geralmente isto envolve troca de alguns requisitos.

A análise de requisitos implica em uma série de validações que, no entanto, possui uma natureza distinta das que ocorrerão na etapa de validação de requisitos, sendo feitas sobre um conjunto incompleto de requisitos, diferentemente da validação de requisitos que ocorre sobre um conjunto completo de requisitos acordados e especificados.

Checklist é uma lista de questões que o engenheiro de requisitos usa para julgar cada requisito. Como oferece uma lista do que deve ser verificado, reduz as chances de não validação de características importantes dos requisitos. A Tabela 1 mostra um quadro contendo *checklist* para a análise de requisitos.

Tabela 1 - *Checklist* para análise de requisitos [KOT98]

Item a ser verificado	Descrição
Projeto prematuro	O requisito inclui informação de projeto ou implementação?
Requisitos combinados	A descrição do requisito descreve um único requisito ou pode ser subdividida em vários requisitos diferentes?
Requisitos desnecessários	O requisito é supérfluo? É realmente necessário?
Conformidade com as metas do negócio	O requisito é consistente com as metas do negócio definidas na introdução do documento de requisitos?
Ambigüidade	O requisito pode ser lido de maneiras diferentes por pessoas diferentes? A descrição tem mais de uma interpretação possível?

Tabela 1 - Checklist para análise de requisitos [KOT98] (cont.)

Item a ser verificado	Descrição
Factibilidade	O requisito é realístico dado a tecnologia que será usada para implementar o sistema?
Testabilidade	O requisito é declarado de tal forma que testes podem ser derivados para verificar se o sistema poderá atendê-lo?

Matrizes de interação, também podem ser utilizadas para análise de requisitos. A Tabela 2, mostra a interação que existe entre os requisitos em termos de:

- Conflitos
- Sobreposição
- Independência

Tabela 2 - Matriz de Interação de Requisitos [KOT98]

Requisitos	R1	R2	R3	R4	R5	R6
R1	0	0	2	0	1	1
R2	0	0	0	0	0	0
R3	2	0	0	2	0	2
R4	0	0	2	0	1	1
R5	1	0	0	1	0	0
R6	1	0	2	1	0	0

Conteúdo da tabela:

- 0 para requisitos independentes
- 1 para requisitos em conflito
- 2 para requisitos em sobreposição

A medida em que os problemas forem encontrados, o processo de negociação deve ser iniciado. O processo deve discutir os problemas encontrados nos requisitos, buscando chegar a um acordo com os envolvidos. O processo de negociação deve ser direto e objetivo, assim decisões sobre os requisitos devem ser baseadas em necessidades técnicas e organizacionais.

Grande parte do tempo usado no processo de negociação são investidos nas resoluções de conflitos, porque muitas vezes os critérios para as decisões dos requisitos são baseados em argumentos políticos e pessoais. Reuniões entre os envolvidos é a melhor forma para negociar requisitos e resolver conflitos.

2.5.5. ESPECIFICAÇÃO DE REQUISITOS

O objetivo da atividade de Especificação de Requisitos é documentar os resultados da análise de requisitos. Assim, os requisitos acordados durante a análise de requisitos deverão ser especificados de forma clara e precisa em um conjunto de documentos, que recebe o nome de especificação de requisitos [KOT98].

A IEEE [STD830] denomina estes documentos de ERS (Especificação de Requisitos de *Software*) e define um conjunto de características que uma boa especificação de requisitos de *software* deve ter:

- Correta – o requisito deve realmente ser implementado no Sistema;
- Inambígua – o requisito deve ter apenas uma interpretação;
- Completa – todos os requisitos devem ser significantes, conhecidos e tratados; devem incluir definição das respostas do *software* a todos os tipos de dados de entrada em todas as situações possíveis, válidas ou inválidas, e todas as figuras, tabelas, diagramas e unidades de medida; devem ser referenciadas e rotuladas;
- Consistente – o requisito não deve ter conflito com nenhum outro requisito do Sistema;
- Priorizável por importância e/ou estabilidade – o requisito deve ter um indicador de importância ou estabilidade. Alguns podem ser essenciais, enquanto outros podem ser desejáveis;
- Verificável – para cada requisito contido no ERS deve existir um processo finito e com custo factível, através do qual uma pessoa ou máquina possa assegurar que o produto de *software* atenda ao requisito;
- Modificável – as modificações devem ser agregadas ao documento de forma fácil, completa e consistente, com relação a sua estrutura e estilo;

- Rastreável – o requisito deve ter uma fonte e deve ser documentado de forma a ser relacionado a outros requisitos.

Além das características acima, podemos acrescentar que é interessante que ela seja de fácil leitura. As especificações podem ser compostas de simples descrições em linguagem natural (especificações informais), modelos diagramáticos semi-precisos (especificações semi-formais) e modelos precisos, com toda uma linguagem matemática fundamentada (especificações formais).

Kotonya e Sommerville [KOT98] afirmam que a linguagem natural é a única capaz de ser compreendida por todos os participantes do processo de requisitos, embora reconheçam que se necessário, diagramas, tabelas e gráficos podem ser incluídos, para melhorar o entendimento do documento de requisitos.

Modelos semi-formais oferecem uma notação gráfica que auxilia no entendimento da informação e das funções de um sistema. Para Mylopoulos [MYL92] é uma atividade que expõe formalmente com precisão aspectos do mundo físico e social, com o propósito de melhorar o entendimento e a comunicação. A mais conhecida técnica deste modelo é a UML (*Unified Modeling Language*) [BOO99] criada pela unificação das técnicas OMT (*Object Modeling Technique*) de Rumbaugh [RUM91] a OOSE (*Object-Oriented Software Engineering*) de Jacobson [JAC92] e a OOAD (*Object-Oriented Analysis and Design*) de Booch [BOO94] que é uma linguagem para especificação, visualização, construção e documentação de artefatos de sistemas de *software*. Técnicas formais, como exemplo a Linguagem Z, são pouco utilizadas, mas já encontramos trabalhos relacionados, onde Martins em [MAR05] empiricamente busca comparações entre especificações utilizando métodos formais (linguagem Z) e métodos semi-formais (UML).

2.5.6. VALIDAÇÃO DE REQUISITOS

A fase de Validação de Requisitos está preocupada com a verificação da documentação de requisitos para consistência, perfeição e precisão [KOT98]. Seu objetivo é validar os requisitos, verificando-os para certificar que eles representam

uma descrição aceitável de como o sistema deve ser implementado. O processo envolve *stakeholders*, engenheiros de requisitos e projetistas de sistemas, que analisam os requisitos para problemas, omissões e ambigüidades.

Exemplos de problemas de requisitos descobertos durante a validação:

- Falta de conformidade com padrões de qualidade;
- Requisitos pobres que são ambíguos;
- Erros nos modelos de sistema ou o problema para ser resolvido;
- Conflitos de requisitos que não são detectados durante o processo de análise.

Estes problemas devem ser resolvidos antes da documentação de requisitos ser aprovada e usada como base para o desenvolvimento do sistema. Alguns problemas detectados podem ser reparados simplesmente corrigindo-se a documentação, porém outros problemas exigirão que as atividades anteriores da Engenharia de Requisitos sejam realizadas novamente. O principal problema de validação de requisitos é a falta de documentos que podem ser a base para o processo de validação. A Figura 13 mostra o processo de validação segundo Kotonya e Sommerville [KOT98].

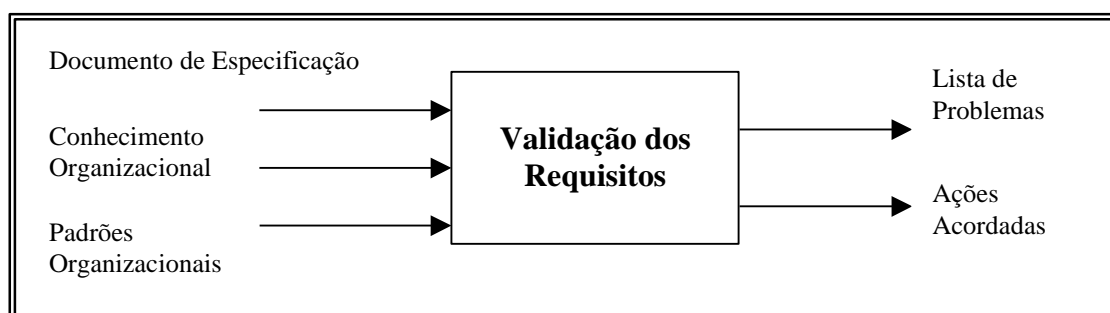


Figura 13 – Entradas e Saídas do Processo de Validação de Requisitos [KOT98]

Entrada de três fluxos de informações:

- Documentação de requisitos – completa versão de documentação formatada e organizada de acordo com padrões organizacionais;
- Padrões organizacionais – O processo de validação de requisitos deve estar em conformidade com o padrão da organização;

- Conhecimentos organizacionais – esta entrada é intangível, mas é importante que pessoas detentoras do conhecimento organizacional estejam envolvidas na validação de requisitos, porque conhecem a estrutura, padrões e a cultura da empresa.

Saída de:

- Lista de problemas – reporta os problemas encontrados e idealmente devem ser classificadas por tipo de problema, ambigüidade, imperfeições, etc.
- Ações acordadas – é a lista de ações em resposta aos problemas dos requisitos que foram acordados pelos envolvidos no processo de validação.

Há uma tendência natural em apressar o processo de validação para iniciar o desenvolvimento, entretanto, se este inicia muito cedo, é grande a probabilidade de aparecer problemas nos requisitos, que causará retrabalho, implicando em maiores custos. Se estes problemas forem descobertos na validação de requisitos, o retrabalho é reduzido.

2.5.7. REVISÃO DE REQUISITOS

A técnica de revisão de requisitos é a mais utilizada para validação de requisitos. Ela envolve um grupo de pessoas que lêem, analisam e criticam os requisitos, observam os problemas, discutem e acordam uma série de ações para solucionar os problemas.

Os principais estágios de revisão de processos, são:

- Revisar o plano – A equipe de revisão é selecionada e as datas e locais das reuniões escolhidas;
- Distribuir documentos – Os documentos dos requisitos são distribuídos para os membros da equipe procederem a revisão;
- Preparar para revisão – Os revisores lêem os documentos de requisitos para identificar conflitos, omissões, inconsistências, desvios de padrões e outros problemas;

- Realizar reunião de revisão – Comentários e problemas são discutidos e as ações são endereçadas para acordo dos problemas;
- *Follow-up* das ações – O coordenador das revisões verifica se as ações acordadas estão sendo levadas a cabo;
- Revisar documentos – Os documentos dos requisitos são revisados para refletir as ações acordadas. Neste estágio ele pode ser aceito ou revisado novamente.

As revisões de requisitos são efetuadas em reuniões formais, coordenadas por alguém que não participou da produção dos requisitos. Durante a reunião, um engenheiro de requisitos apresenta cada requisito por vez para comentários e identificação de problemas, sendo anotados em uma lista.

Ações que podem ser decididas para cada problema:

- Esclarecimento do requisito – O requisito pode ter sido mal expressado ou pode ter havido omissão de informações, durante a fase de elicitação;
- Perda de informação – Alguma informação pode ter desaparecido da documentação de requisitos;
- Conflitos de requisitos – Os *stakeholders* envolvidos devem negociar para resolver estes conflitos;
- Requisitos irreais – Requisitos que não podem ser implementados com a tecnologia disponível ou possuem restrições em relação ao sistema. Os *stakeholders* devem decidir se ele será excluído ou modificado para atender a realidade do processo.

Verificação de Pré-revisão

Este é um meio rápido e barato de verificar se a estrutura de documentos de requisitos está dentro do padrão estabelecido. Os documentos fora do padrão são retornados aos engenheiros de requisitos, para correções necessárias.

Revisão dos Membros da Equipe

Sempre que possível, a revisão dos documentos de requisitos deve ser feita por uma equipe multidisciplinar, composta por pessoas com diferentes experiências, como usuários finais, clientes, especialistas, engenheiros de projetos, engenheiros de implementação, etc. A equipe de revisão deve incluir diferentes *stakeholders* envolvidos nas elicitações de requisitos.

Checklists da Revisão

Checklists devem ser declarados de uma maneira simples e genérica e devem ser entendidos por usuários finais, os quais não são especialistas em informática. Como regra geral, o *checklist* deve ser sintético (menos que 10 itens). Os principais atributos de qualidade são: facilidade de entendimento, redundância, completitude, ambigüidade, consistência, organização, conformidade e rastreabilidade. A Tabela 3 mostra possíveis questões de *checklists* geradas a partir dos atributos de qualidade.

Tabela 3 - Possíveis questões de *checklists* [KOT98]

Questões do Checklist	Atributos de Qualidade
Cada requisito é identificado de forma única?	Rastreabilidade, conformidade
Termos especializados são definidos no dicionário de dados?	Facilidade de entendimento
O requisito é auto-explicativo ou há necessidade de examinar outros requisitos para entendê-lo?	Facilidade de entendimento, completitude
Requisitos individuais usam o mesmo termo de maneiras diferentes?	Ambigüidade
O mesmo serviço é solicitado em requisitos diferentes?	Consistência, redundância
Requisitos relacionados estão agrupados? Se não, eles referenciam uns aos outros?	Organização, rastreabilidade

2.5.8. PROTOTIPAÇÃO

O desenvolvimento de protótipos para demonstrar requisitos é uma maneira fácil de usuários finais descobrirem problemas e sugerirem como podem ser solucionados. Se um protótipo foi desenvolvido para elicitar requisitos, com

certeza ele será usado posteriormente para validar requisitos. Entretanto, se ele não se encontra pronto, não é viável seu desenvolvimento somente para validação. O processo de prototipação é uma atividade que pode estar em paralelo com outras:

- Escolhendo testadores do protótipo – é importante escolher bem as pessoas designadas para testar os protótipos. Os mais indicados são os usuários experientes e de diferentes áreas;
- Desenvolvendo cenários de testes – é importante que o protótipo seja testado de forma sistemática, pois os usuários não dispõem de tempo para efetivar os testes;
- Executando cenários – os usuários que realmente irão trabalhar com o sistema, são os mais indicados para trabalhar com os cenários, pois eles têm mais conhecimentos do assunto e estarão trabalhando de forma mais realística;
- Documentando problemas – os problemas encontrados pelos usuários devem ser documentados manualmente ou em algum meio eletrônico.

É importante que os protótipos de sistema sejam seguros. Se houver problemas, os usuários podem ser desencorajados a abandonarem os sistemas.

2.5.9. VALIDAÇÃO DO MODELO

Partes das especificações dos requisitos para um sistema podem consistir de um ou mais modelos de sistema como modelos de fluxo de dados, de objetos, de entidade-relacionamento, diagramas de UML etc. A validação destes modelos tem três objetivos:

- Demonstrar que cada modelo é autoconsistente;
- Se há vários modelos de sistemas, para demonstrar que estes são consistentes entre si;
- Demonstrar que os modelos refletem os requisitos reais dos usuários do sistema.

2.5.10. TESTE DE REQUISITOS

A execução da implementação do requisito pode ser simulada usando testes de requisitos. Propor testes possíveis é um efetivo modo de revelar problemas nos requisitos como ambigüidade e imperfeições. Se há dificuldade em derivação de casos de testes para um requisito, isto implica em algum tipo de problema no requisito. O objetivo do teste é validar o requisito e não o sistema.

Os resultados dos testes devem ser registrados utilizando-se formulários que contenham as seguintes informações:

- Identificador do requisito – deve ter ao menos um para cada requisito;
- Requisitos relacionados – Devem ser referenciados como um teste e também podem ser relevantes para estes requisitos;
- Descrição do teste – Uma breve descrição de como o teste pode ser aplicado e qual o seu objetivo;
- Problemas nos requisitos – Uma descrição de problemas nos requisitos que pode definir se o teste é difícil ou impossível;
- Comentários e recomendações – São avisos como resolver problemas nos requisitos que foram descobertos.

Há limitações, entretanto nos testes para alguns tipos de requisitos:

- Requisitos de sistemas – Estes são requisitos que aplicam para o sistema como um todo. Em geral estes são os requisitos mais difíceis de validar;
- Requisitos exclusivos – São requisitos que excluem comportamentos específicos. Por exemplo, um requisito pode permitir falha no sistema, mas que nunca corrompa o banco de dados;
- Alguns Requisitos Não-Funcionais – exemplo, um requisito de segurança.

3. GERENCIAMENTO DE REQUISITOS

O Gerenciamento de Requisitos é o processo de compreender e controlar as mudanças nos requisitos de sistemas e ocorre em conjunto com outros processos da Engenharia de Requisitos [SOM03].

Os principais objetivos do Gerenciamento de Requisitos segundo Kotonya e Sommerville [KOT98] são:

- Gerenciar mudanças nos requisitos acordados;
- Gerenciar o relacionamento entre requisitos;
- Gerenciar as dependências entre os documentos de requisitos e outros documentos produzidos no processo de Engenharia de *Software*.

Lam e outros [LAM99] identificam três razões para gerenciamento de requisitos:

- Muitos sistemas são entregues incrementalmente. Entre cada entrega incremental, mudanças nos requisitos são estabelecidas e incorporadas no próximo incremento;
- Tipicamente requisitos mutáveis são os principais causadores de manutenções de *software* e atividades de reengenharia;
- Muitas organizações têm sistemas legados que são críticos e sustentam operações comerciais. Substituir totalmente ou recriar tais sistemas nem sempre é possível e necessitam evoluir para que a empresa sobreviva e permaneça competitiva.

Segundo Sommerville [SOM03] os requisitos devem evoluir a fim de refletir as mudanças que ocorrem ao longo do tempo, no ambiente do sistema e nos objetivos da empresa. Para tanto, a partir da perspectiva de evolução, divide os requisitos em duas classes:

- Requisitos permanentes ou estáveis – São os requisitos relativamente estáveis, que derivam da atividade principal da

organização e se relacionam diretamente com o domínio do problema;

- Requisitos voláteis – São os requisitos mais propensos a modificações durante o desenvolvimento do sistema ou depois que o sistema estiver em operação.

Cysneiros concorda com esta classificação, afirmando que esta auxilia a atividade de gerência de requisitos, uma vez que possibilita antecipar mudanças prováveis de requisitos [CYS01].

Sommerville [SOM03] considera que existem ao menos quatro diferentes tipos de requisitos voláteis:

- a. Requisitos mutáveis – Requisitos que se modificam devido a mudanças do ambiente no qual a organização está operando;
- b. Requisitos emergentes – Requisitos que surgem à medida que a compreensão do cliente do sistema aumenta durante o desenvolvimento do sistema. O processo de projeto pode revelar novos requisitos emergentes;
- c. Requisitos conseqüentes – São requisitos resultantes da introdução do sistema de computação. A introdução do sistema de computação pode modificar os processos da organização e criar novos meios de trabalho que geram novos requisitos de sistema;
- d. Requisitos de compatibilidade – Requisitos dependentes de sistemas ou processos de negócios específicos dentro da organização. À medida que eles se modificam, os requisitos de compatibilidade no sistema encomendado ou entregue podem também ter que evoluir.

Os fatores que mais contribuem para a mudança de requisitos segundo Kotonya e Sommerville [KOT98] são:

- Erros, conflitos e inconsistências nos requisitos – como os requisitos são analisados e implementados, erros e inconsistências emergem e devem ser corrigidos. Estes problemas podem ser descobertos durante a análise e validação de requisitos ou mais tarde durante o processo de desenvolvimento;
- Evolução do conhecimento dos clientes e usuários do sistema – no decorrer do desenvolvimento, os clientes e usuários desenvolvem um entendimento melhor do que eles realmente querem do sistema;
- Problemas técnicos, de prazo e de custos – problemas podem ser detectados quando da implementação de um requisito. Pode ser muito caro ou demorar demais para implementar certos requisitos;
- Mudança nas prioridades dos clientes – a prioridade dos clientes pode mudar durante o desenvolvimento do sistema como resultado de mudanças no ambiente de negócios, surgimento de novos concorrentes, mudanças na equipe, entre outros;
- Mudanças ambientais – o ambiente no qual o sistema será implantado poderá mudar de forma que os requisitos de sistema precisem ser alterados para manter compatibilidade;
- Mudanças organizacionais – a organização que pretende usar o sistema pode precisar mudar sua estrutura e processos, resultando em novos requisitos do sistema.

3.1. GERENCIAMENTO DE MUDANÇAS

O gerenciamento de mudanças segundo Kotonya e Sommerville [KOT98] está relacionado à política de uso de procedimentos, processos e padrões que serão utilizados para gerenciar as mudanças nos requisitos do sistema. Estas políticas incluem:

- O processo de solicitação de mudanças e as informações necessárias para processá-la;
- O processo usado para analisar o impacto e custo da mudança e informações associadas à rastreabilidade;
- Definição dos membros da organização que formalmente consideram as solicitações de mudanças;

- O suporte de *software* necessário para o controle do processo de mudança.

O Processo de Gerenciamento de Mudanças nos Requisitos segundo Kotonya e Sommerville [KOT98], consiste de um conjunto de atividades para documentar, reportar, analisar, definir custos e implementar mudanças de um conjunto de requisitos, conforme ilustra a Figura 14.

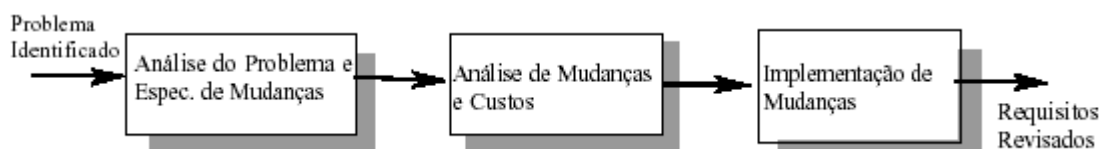


Figura 14 – Processo de Gerenciamento de Mudanças nos Requisitos [KOT98]

Os três estágios representam:

- Algum problema de requisitos é identificado. Isto pode ser oriundo de uma análise do documento de requisitos, de novas necessidades dos *stakeholders*, ou problemas operacionais com o sistema. Os requisitos são analisados usando informações do problema e mudanças nos requisitos são propostas;
- As mudanças propostas são analisadas. Verificam-se quantos requisitos e se necessário componentes do sistema, serão afetados pela mudança e é calculado de forma aproximada o custo em tempo e dinheiro;
- A mudança é implementada. Um conjunto de alterações ou uma nova versão do documento de requisitos é produzido.

Segundo os mesmos autores, o Processo de Análise de Mudança e Custo, é composto de seis estágios conforme apresenta a Figura 15.

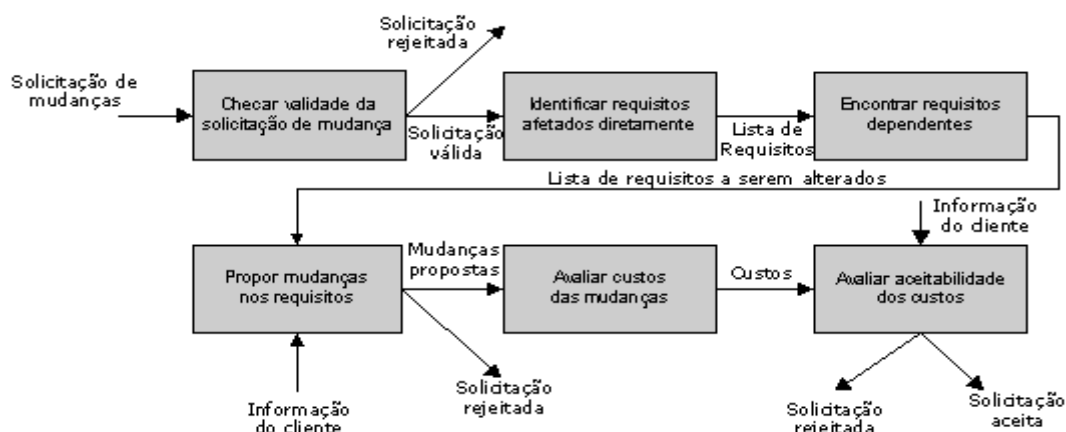


Figura 15 – Processo de Análise de Mudança e Custo [KOT98]

Os seis estágios deste processo contemplam:

- a. A requisição de mudança é verificada quanto à sua validade, pois os *stakeholders* podem não entender os requisitos e sugerem mudanças desnecessárias;
- b. Os requisitos afetados diretamente pelas mudanças são descobertos;
- c. Informações de rastreabilidade são usadas para encontrar os requisitos dependentes que podem ser afetados pela mudança;
- d. As mudanças que podem ser feitas para os requisitos são propostas;
- e. Os custos da realização da mudança são estimados;
- f. São efetuadas negociações com os clientes para verificar se os custos das mudanças propostas são aceitáveis.

As requisições de mudanças podem ser rejeitadas:

- Se a solicitação de mudança for inválida. Isto normalmente acontece se o cliente não entendeu algo sobre um requisito e propôs uma mudança que não é necessária;

- Se a solicitação de mudança resultar em conseqüências que não são aceitáveis ao usuário;
- Se o custo da implementação for muito alto ou se demorar demais.

As requisições de mudanças são normalmente armazenadas em formulários próprios, que são repassados aos envolvidos na análise da mudança e devem incluir no mínimo, as seguintes informações:

- Documentação da análise da mudança;
- Data da solicitação;
- Responsável;
- Prioridade;
- *Status*;
- Comentários.

Segundo Espíndola e outros em [ESP04], deve-se garantir que informações similares sejam coletadas para cada proposta de mudança de requisitos e que o julgamento seja feito com base em uma análise de custo e benefício de cada proposta, de forma que estas propostas possam suportar os objetivos de negócio fundamentais do sistema.

3.2. RASTREABILIDADE

Segundo Gotel e Finkelstein [GOT94] “rastreabilidade de requisitos é a habilidade de descrever e acompanhar a vida de um requisito em ambas as direções do processo de *software* (do planejamento do negócio à especificação do projeto), idealmente durante todo o seu ciclo de vida”.

Espíndola e outros em [ESP04], afirmam “que um requisito pode ser rastreado se é possível determinar quem sugeriu o requisito, porque o requisito existe, a quais outros requisitos ele está relacionado e como ele está relacionado com outras informações como artefatos de projeto, implementação e documentação de usuário. O rastreamento também permite encontrar outros requisitos que podem ser afetados quando uma mudança é solicitada”.

Pinheiro [PIN00] entende rastreabilidade de requisitos como a habilidade de definir, capturar e acompanhar os rastros deixados pelos requisitos nos outros elementos do ambiente de desenvolvimento de *software* e os rastros deixados por esses elementos nos requisitos.

Ramesh [RAM95] diz que a rastreabilidade de requisitos é usada para capturar o relacionamento entre requisitos, projeto e implementação de um sistema. Assim, todos os componentes do sistema (hardware, *software*, pessoas, manuais, políticas e procedimentos) criados nos vários estágios do processo de desenvolvimento, são ligados aos requisitos.

Davis [DAV90] classifica o processo de rastreabilidade em quatro tipos:

- a. A partir dos requisitos – Relaciona requisitos ao projeto e componentes de implementação. A responsabilidade pela realização dos requisitos deve ser atribuída aos componentes do sistema, como considerar e estabelecer avaliações dos impactos de mudanças dos requisitos.
- b. De volta aos requisitos – Relaciona o projeto e componentes de implementação aos requisitos. A aderência do sistema aos requisitos deve ser verificada e devem-se evitar projetos para os quais não existem requisitos.
- c. Em direção a novos requisitos – Relaciona outros documentos que possam ter precedido os documentos de requisitos, aos requisitos relevantes. Mudanças das necessidades dos *stakeholders* bem como suposições técnicas podem exigir reavaliação dos requisitos relevantes.
- d. De volta a partir dos requisitos – Relaciona requisitos a suas fontes em outros documentos e pessoas. Associa os requisitos às estruturas que contribuíram para originá-los, imprescindíveis na validação dos requisitos, especialmente em cenários altamente políticos.

Jarke [JAR98] entende que os dois primeiros tipos são rotulados como pós-rastreabilidade ligando os requisitos ao projeto e implementação, distribuindo responsabilidade da documentação, verificando aderência ou análise de impacto de um requisito, enquanto os dois últimos tipos são habilitados a pré-rastreabilidade, o qual documenta o *rational* e o contexto político social.

Kotonya e Sommerville [KOT98] estendem a definição de Davis de rastreabilidade *backward-from* e *forward-to* para permitir ligações para o mesmo documento de requisitos, bem como para documentos externos. A Figura 16 mostra como uma declaração de requisitos pode incluir ligações de rastreabilidade para e de uma especificação de projetos.

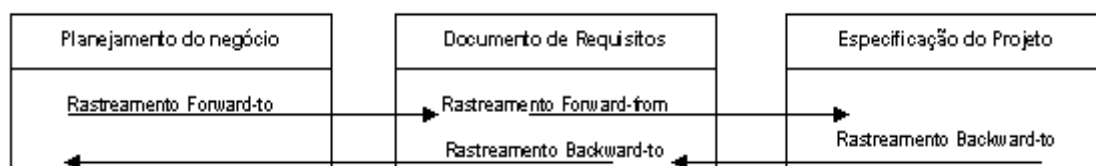


Figura 16 – Rastreabilidade Backwards / Forwards [KOT98]

Kotonya e Sommerville [KOT98] complementam os tipos de rastreabilidade definidos por Davis e definem como estes diferentes tipos podem ser instanciados mais concretamente por ligações entre informações específicas no documento de requisitos e em outros documentos do sistema:

- Rastreamento das fontes de requisitos – Relaciona o requisito, pessoas e documentos que especificaram os requisitos;
- Rastreamento da razão dos requisitos – Relaciona o requisito com a descrição do porque o requisito foi especificado;
- Rastreamento requisitos-requisitos – Relaciona requisitos com outros requisitos que são de alguma forma dependentes deles. Deve ser um relacionamento em duas direções (dependentes e dependentes-de);
- Rastreamento requisitos-arquitetura – Relaciona os requisitos com os subsistemas onde estes requisitos estão implementados.
- Rastreamento requisitos-projeto – Relaciona os requisitos com o hardware específico ou com componentes de *software* que serão usados para implementar requisitos;

- Rastreamento requisitos-interface – Relaciona os requisitos com a interface externa do sistema que será usada para prover os requisitos.

Tabelas de rastreabilidade podem ser usadas para demonstrar os relacionamentos entre requisitos ou entre requisitos e componentes de projetos. Os requisitos são listados ao longo dos eixos horizontais e verticais e os relacionamentos são marcados nas células da tabela. As tabelas de rastreamento que mostram as dependências devem ser definidas com os números dos requisitos que são usados para rotular as linhas e colunas da tabela. Abaixo, a Tabela 4 mostra a rastreabilidade entre requisitos.

Tabela 4 – Rastreabilidade entre requisitos

	R1	R2	R3	R4	R5	R6
R1			X	X		
R2					X	X
R3				X	X	
R4		X				
R5						X
R6	X					

Segundo Kotonya e Sommerville [KOT98] se o número de requisitos a ser gerenciado for pequeno (digamos até 250 requisitos), as tabelas podem ser implementadas usando planilhas. Se o número de requisitos for grande (centenas ou milhares), as tabelas ficarão esparsamente populadas e neste caso, indica-se a utilização de lista de rastreabilidade que podem ser implementadas como texto ou tabelas simples, conforme a Tabela 5 abaixo.

Tabela 5 – Lista de Requisitos

Requisito	Depende de
R1	R3, R4
R2	R5, R6
R3	R4, R5
R4	R2
R5	R6

3.3. FERRAMENTAS DE APOIO PARA GERENCIAMENTO DE REQUISITOS

Como o gerenciamento de mudanças de requisitos pode envolver grandes volumes de informações, o suporte pode ser provido através de ferramentas específicas de gerenciamento de requisitos ou de ferramentas CASE, as quais podem incluir as seguintes facilidades:

- Formulários eletrônicos de solicitação de mudanças, os quais serão preenchidos pelos diferentes participantes do processo;
- Um banco de dados para armazenar e gerenciar os formulários de mudança;
- Um modelo de mudança poderá ser instanciado de forma que a pessoa responsável por um estágio do processo saberá quem é responsável pela próxima atividade do processo;
- Transferência eletrônica de formulários entre as pessoas com diferentes responsabilidades e notificação, quando as atividades estiverem concluídas;
- Em alguns casos, *links* diretos para um banco de dados de requisitos.

Uma ferramenta de Gerenciamento de Requisitos não resolve problemas causados por requisitos de baixa qualidade. Requisitos pobres incrementam custos e estendem o cronograma. No Apêndice I, relacionamos uma série de ferramentas de Gerenciamento de Requisitos, encontradas na *web*. Alguns destes sites provêem *download* gratuito para demonstração. Estaremos dedicando maior atenção a algumas ferramentas mais utilizadas no mercado, como DOORS da Telelogic, Requisite-Pro da Rational e Caliber RM da Borland.

DOORS

DOORS (*Dynamic Object Oriented Requirements Systems*) é uma ferramenta desenhada especificamente para capturar, relacionar, rastrear, analisar e gerenciar requisitos. Suas principais características são:

- Trabalha em ambiente *Windows* e *Unix*;

- Pode apoiar usuários em uma rede local ou na Internet. Possui uma estrutura voltada para ambiente *web* que permite usuários lerem, revisarem e comentarem dados via *browser*;
- Provê rastreabilidade de requisitos – permite rastrear requisitos para o documento fonte, em todo o ciclo de vida do programa;
- Pode produzir aplicações sob medida, usando a linguagem própria (similar ao C++);
- Provê um simples banco de dados proprietário;
- Há limitações na importação e exportação de dados. Caracteres limitados nas células de tabelas *Excel* (255 caracteres), podem causar perda de dados e exportação de grandes arquivos de dados do DOORS para o *Word*, é lento;
- Equipes trabalhando em locais diferentes usando uma WAN (*Wide Area Network*) causam problemas de performance.

REQUISITE-PRO

É uma ferramenta de gestão de requisitos da *Rational*, usada para gerenciamento e documentação de requisitos, e suas principais características são:

- Os requisitos podem ser criados, modificados, gerenciados e contém relações de rastreabilidade e histórico de revisões;
- O *Microsoft Word* pode ser usado para editar os requisitos;
- Os requisitos podem ser organizados por tipo. Cada tipo disponibiliza um conjunto de atributos de requisitos que podem ser modificados;
- É possível relacionar os requisitos de modo a ajudar a compreender o impacto das alterações;
- Os projetos podem conter *packages* pré-definidos e *packages* definidos pelos usuários. Os *packages* pré-definidos mais importantes são: *Software Requirements*, *Use Cases*, *Features* e *Stakeholder Requests*;
- Os projetos podem ser armazenados em banco de dados *Microsoft Access*, *SQL Server* e *Oracle*;
- O acesso à base de dados é feito via ODBC (*Open Database Connectivity*);

- As informações de segurança, permissões e estrutura, são armazenadas em arquivos XML;
- Possui gestão integrada com outra ferramenta da *Rational*, o *Rational Rose*, permitindo que casos de uso sejam criados no *Rose* e detalhados e gerenciados no *Requisite-Pro*;
- Contém também um assistente de integração com o *Microsoft Project* que permite sua utilização a partir dos requisitos criados no *Requisite-Pro*.

CALIBER RM

CaliberRM é uma ferramenta da *Borland* para Gerenciamento de Requisitos, e suas principais características são:

- Controle de acesso e permissões;
- Suporte a históricos;
- Geração de relatórios e exportação de documentos de requisitos;
- Fórum de discussões;
- Notificações a usuários;
- Glossário;
- Integração com outras ferramentas, por exemplo, o *Test Director* e *Rational Rose*;

4. FERRAMENTA AUTOMATIZADA PARA GERENCIAMENTO DE REQUISITOS

4.1. CONTEXTO DE DESENVOLVIMENTO DA FERRAMENTA

A ferramenta automatizada para gerenciamento de requisitos foi desenvolvida com o objetivo de coletar, armazenar e manter os requisitos acordados, durante todo o ciclo de vida do *software*, gerenciando as mudanças ocorridas nos requisitos, permitindo rastrear os relacionamentos entre os requisitos e entre os requisitos e documentos produzidos no processo de desenvolvimento do sistema. A ferramenta organiza o controle das mudanças, permitindo subsídios para a análise de impacto e custos em tempo e dinheiro que estas mudanças trarão para a organização.

A elicitação, análise e validação dos requisitos para o desenvolvimento da ferramenta foram efetuadas junto a uma equipe formada por analistas de sistemas, coordenadores e gerentes de projetos, oriundos de empresa prestadora de serviços médicos e por fornecedora de *software* de gestão ligada à área de saúde, que prestaram apoio integral ao desenvolvimento da ferramenta.

Como um dos principais benefícios da ferramenta é sua livre distribuição e uso, seu desenvolvimento foi direcionado e orientado a ferramentas da mesma abordagem, de forma a propiciar mais facilmente melhorias futuras. Desta forma, a ferramenta é operada via interface *Web* e foi desenvolvida em linguagem *Java* com *JSP (Java Server Pages)* e com o SGBD (Sistema Gerenciador de Banco de Dados) *Firebird 1.5*.

Java é uma tecnologia contemporânea e de ponta, referência em portabilidade. As vantagens de utilização como tecnologia são muitas: a portabilidade, a flexibilidade, a sua rica biblioteca de classes, a sintaxe limpa, simples e bem definida, a orientação a objetos e a extensa comunidade de desenvolvedores, responsáveis pela criação de vários projetos de sistemas e

plataformas *open-source*, como por exemplo o *Jakarta Tomcat*, uma solução utilizada atualmente para suporte a *JSP* e *Servlets*.

A interface *Web* tem muitas vantagens. Assim, qualquer máquina que contiver um *browser* poderá ser um cliente da aplicação. Os *Servlets* são a solução do Java para comunicação com um servidor *Web*, por exemplo, *Apache* e *IIS (Internet Information Services)*, através de um *Servlet Runner (Resin, Orion, Tomcat)*.

O *JSP* é uma página *HTML (HyperText Markup Language)* mesclada com código *Java* que internamente é transformada em um *Servlet*, compilado e executado pelo *Servlet Runner*. Sua interface é transparente e similar a outras como *ASP (Active Server Pages)*, *PHP (Hypertext PreProcessor)* e *CFM (Cold Fusion)*, mas com a vantagem de ser *Java*.

O SGBD *Firebird* foi desenvolvido por um grupo independente de programadores voluntários, a partir do código fonte do SGBD *InterBase™*, disponibilizado pela *Borland* ao abrigo da “*InterBase Public License v.1.0*” em 25 de Julho de 2000. Trata-se de um produto *open-source* que oferece todos os benefícios de um banco de dados relacional. Maiores informações sobre o produto pode ser obtido em <http://www.comunidade-firebird.org>.

Para o desenvolvimento da ferramenta foi utilizada a IDE *Eclipse* (www.eclipse.org), projeto de código aberto alavancado pela IBM, que oferece amplos e inovativos recursos de produtividade (*refactoring*, geração de código etc.) e é baseado em *plug-ins*. Especificamente para esta aplicação foi utilizado o *plugin* *Lomboz* (www.objectlearn.com), próprio para desenvolvimento de aplicações *Web*, trazendo facilidades como configuração automática do *Tomcat*, geração automática da aplicação *Web* e reinício da aplicação, quando ocorrer uma alteração no código fonte que necessitar disso, aumentando em muito a produtividade.

A Figura 17 abaixo representa a estrutura para aplicações *Web* em 3 camadas onde a ferramenta será operada:

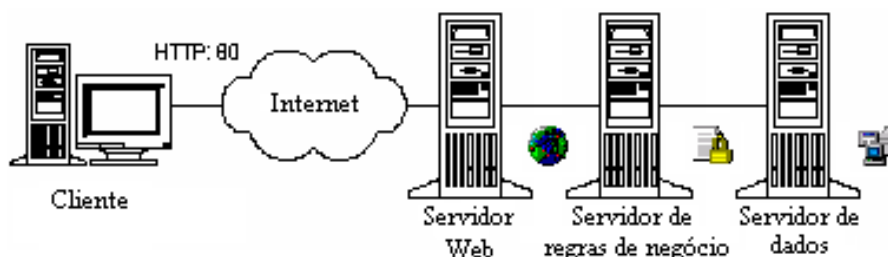


Figura 17 – Ambiente de operação da ferramenta em 3 camadas

O ambiente acima é composto de 3 camadas, descritas a seguir:

- a. 1ª camada – composta por Servidor *Web* (*Apache Server*) que gerencia as requisições vindas da Internet.
- b. 2ª camada – composta pelo Servidor de regras de negócio (*Jakarta Tomcat*) que gerencia o acesso às informações. As aplicações são componentizadas em classes *JAVA* que realizam todas as rotinas da ferramenta.
- c. 3ª camada – composta pelo Servidor de Dados (SGBD *Firebird*), que abriga a base de dados que alimenta todo o sistema.

4.2. ELICITAÇÃO DOS REQUISITOS DA FERRAMENTA

Para elicitação dos requisitos da ferramenta, foram efetuadas reuniões *JAD* (*Joint Application Design*) e entrevistas com a equipe de analistas, coordenadores e gerentes, já especificados na seção 4.1, estabelecendo os limites do sistema, os Requisitos Funcionais e os Requisitos Não-Funcionais.

4.2.1. LISTA DOS REQUISITOS FUNCIONAIS

R1 – Deverá permitir a manutenção dos projetos de sistemas de informação da organização, que serão a base para a manutenção de requisitos e de suas versões.

R1.1 - A exclusão do projeto somente poderá acontecer caso esteja em sua fase inicial, ou seja, desde que não haja nenhum requisito ou versão de requisito ativa para o projeto.

R1.2 – Poderá haver ainda a desativação do projeto em quaisquer fases em que este se encontrar. Esta desativação pode ser entendida como um “congelamento” do projeto, assim, o sistema não deverá permitir quaisquer manutenções no projeto ou em seus requisitos.

R2 – Deverá permitir a manutenção de usuários que estarão envolvidos com os projetos da organização, mantendo atualizado atributos como: departamento a que está alocado, cargo ocupado, telefone, e-mail e nível de conhecimento que possa expressar a experiência específica, como por exemplo: Administração Hospitalar, Área Farmacêutica, Programação Web, Sistemas e Negócios, etc.

R2.1 - A exclusão do usuário somente poderá acontecer se o mesmo não estiver alocado em projeto ativo.

R3 – Deverá permitir a manutenção de glossários que conterão os termos mais usuais utilizados na organização, tanto no âmbito geral como em áreas específicas. Exemplo: poderá haver um glossário geral para a empresa de saúde e outro glossário específico para a área hospitalar.

R3.1 - Não há restrição quanto à exclusão ou desativação de glossários ou de quaisquer de seus termos, pois o projeto não depende diretamente deste, ou seja, a função do glossário é apenas melhorar o entendimento dos envolvidos no projeto.

R4 – Deverá permitir a manutenção dos departamentos que compõem a organização aos quais os usuários se vinculam.

R4.1 - A exclusão do departamento somente será possível se não houver nenhum usuário pertencente a este departamento, alocado em algum projeto ativo.

R5 – Deverá tratar a manutenção dos tipos de requisitos considerados na organização.

R5.1 - A exclusão do tipo de requisito somente será possível se não houver nenhum requisito ativo classificado com este tipo.

R5.2 – A desativação ou ativação do tipo de requisito poderá ser feita a qualquer momento.

R6 – Deverá tratar a manutenção dos motivos da criação da versão de requisitos.

R6.1 - A exclusão do motivo somente será possível se não houver nenhuma versão de requisito ativa que o contenha.

R7 – Deverá tratar a manutenção da volatilidade de requisitos. Este atributo terá a função de medir o grau de instabilidade de um requisito no decorrer do tempo, de forma a auxiliar o analista responsável, a atribuir riscos e impactos na análise da alteração do requisito. Para tal, deverá ser prevista a seguinte estrutura:

- Descrição – texto que represente o atributo de acordo com cada organização.
- Quantidade mínima – quantidade mínima de novas versões do requisito no período.
- Quantidade máxima – quantidade máxima de novas versões do requisito no período.
- Número de meses – é a quantidade de meses a ser considerada para aplicação do índice de volatilidade (período).
- Peso – não será tratado na versão inicial da ferramenta, mas foi definido de forma a deixar espaço para trabalhos futuros. A ideia inicial é atribuir valores em uma determinada escala (0 a 10 por exemplo), que representem o grau de significância do mnemônico acima.
- A Tabela 6 exemplifica dados de volatilidade a serem incluídos para uma organização:

Tabela 6 – Volatilidade de Requisitos

Descrição	Quantidade Mínima	Quantidade Máxima	Número de Meses	Peso
Volatilidade baixa	0	1	6	1
Volatilidade média	2	5	6	5
Volatilidade alta	6	99	6	10

No exemplo acima, decorrido o período de seis meses, o sistema verificará a quantidade de versões criadas para cada requisito e atualizará o atributo Volatilidade de acordo com o resultado obtido.

R7.1 - A exclusão da volatilidade somente será possível se não houver nenhuma versão de requisito ativa que o contenha.

R8 – O sistema deverá manter cadastros auxiliares (Impacto, Risco e Importância), cujos dados visam subsidiar o Gerente do Projeto nas análises das propostas de alterações formuladas na ferramenta. Estes cadastros deverão conter:

- Descrição – texto que represente o atributo, de acordo com cada organização.
- Peso – não será tratado na versão inicial da ferramenta, mas foi definido de forma a deixar espaço para trabalhos futuros. A idéia inicial é atribuir valores em uma determinada escala (0 a 10, por exemplo), que representem o grau de significância do mnemônico acima.
- A Tabela 7 exemplifica dados dos cadastros (Impacto, Riscos e Importância) a serem incluídos para uma organização:

Tabela 7 – Parâmetros para Atributo Risco

Descrição	Peso
Baixo	1
Médio	5
Alto	10

R9 – O sistema deverá permitir ao Gerente do Projeto proceder a manutenção dos usuários alocados no projeto (Analistas e Usuários do sistema), definindo a alçada dos mesmos. Entende-se por alçada a atribuição dada ao usuário alocado, de propor solicitação de alteração de versão, de ser responsável pelo requisito, de listar requisitos, de participar de fórum de discussões, etc.

R10 – O sistema deverá permitir ao Gerente do Projeto proceder a manutenção dos requisitos no sistema, definindo para cada requisito, o tipo, a volatilidade inicial e o analista responsável.

R11 – Deverá haver a possibilidade de cadastrar sub-requisitos, ou seja, permitir a subdivisão dos requisitos com o objetivo de melhorar o entendimento do requisito.

R12 – O sistema deverá prever o registro das dependências entre os requisitos de forma a possibilitar a rastreabilidade dos mesmos. O Gerente do Projeto será o responsável por esta atribuição.

R13 – A necessidade de alteração de um requisito no sistema deverá partir de algum envolvido no projeto, com alçada suficiente, que irá gerar uma proposta de versão de requisitos. A proposta será incluída na ferramenta pelo Gerente do Projeto ou pelo Analista Responsável e deverá conter dados básicos como data da proposta, descrição, impacto, risco, importância, motivo, prioridade e custo.

R14 – Utilizando os recursos de dependência anteriormente definidos, o sistema deverá prover a rastreabilidade dos requisitos dependentes e relacionar estes requisitos, de forma que os responsáveis possam analisá-los e atribuir os valores de forma idêntica à descrita em R13.

R15 – Na geração de uma nova versão de requisitos, o sistema deverá através de “correio interno”, notificar todos os envolvidos nos requisitos (requisitos origem / requisitos dependentes), de forma que possam tomar conhecimento das versões propostas.

R16 – As alterações nos requisitos propostos deverão atender os seguintes níveis de maturidade:

- Na geração inicial a alteração deve figurar como “proposta”;
- Após verificação de viabilidade pelo Gerente do Projeto, este atribuirá a situação “em análise”;
- O Gerente do Projeto promoverá após análise geral de custos, impacto, riscos e importância, para alteração “aprovada”;
- Assim que o analista responsável iniciar a implementação, o requisito estará “em desenvolvimento”;
- Após implementação e testes, a alteração será finalizada como “implementada”;
- Importante: a passagem de uma fase para a outra, do requisito origem (“pai”), somente será possível, quando todos os seus requisitos dependentes (“filhos”) forem devidamente tratados. Exemplo: a alteração do requisito somente será “aprovada”, quando todas as alterações dos seus requisitos dependentes encontrarem-se na condição de “aprovada”.

4.2.2. LISTA DOS REQUISITOS NÃO-FUNCIONAIS

R1 – O sistema deverá prever mecanismos para controlar o acesso ao sistema (login e senha).

R2 – O sistema deverá ter a figura de um Administrador que será responsável por todo controle e gerência do sistema, promovendo a manutenção de todos cadastros básicos.

R3 – Deverá haver controle interno no sistema para que o mesmo seja abandonado automaticamente caso o usuário deixe de usá-lo por dez minutos. A tela inicial de *login* deverá ser apresentada para continuidade do trabalho suspenso.

R4 – O sistema deverá ter portabilidade. Assim, deverá ser desenvolvido em linguagem e plataforma adequada (ex: Java).

R5 - O sistema deverá prever interoperabilidade, de forma a promover intercâmbio com outros sistemas ou *software* de apoio.

4.3. ESPECIFICAÇÃO DOS REQUISITOS DA FERRAMENTA

Para especificar os requisitos da ferramenta foram utilizados vários diagramas da UML (*Unified Modeling Language*), descritos nas próximas subseções.

4.3.1. MODELAGEM DE CASOS DE USO

Módulo Administração

O Módulo Administração expressa a fase inicial da ferramenta onde são criados e geridos os cadastros básicos de Projetos, Usuários e Glossários. Nesse cenário também são contemplados os cadastros de Volatilidade, Risco, Importância, Impacto, Motivo, Tipo e Departamento, que serão básicos para todos os projetos.

O administrador da ferramenta será o preposto direto da organização, responsável pela formatação e parametrização dos cadastros acima mencionados e entre outras funções, este executa a alocação dos Glossários aos Projetos e a atribuição de Gerentes aos Projetos. Ao finalizar esta última ação, o administrador da ferramenta, capacita o Gerente do Projeto ao total domínio sobre os projetos de sua alçada, de forma que este possa dar andamento às fases subseqüentes.

A Figura 18 abaixo apresenta os casos de uso que representam o Módulo Administração.

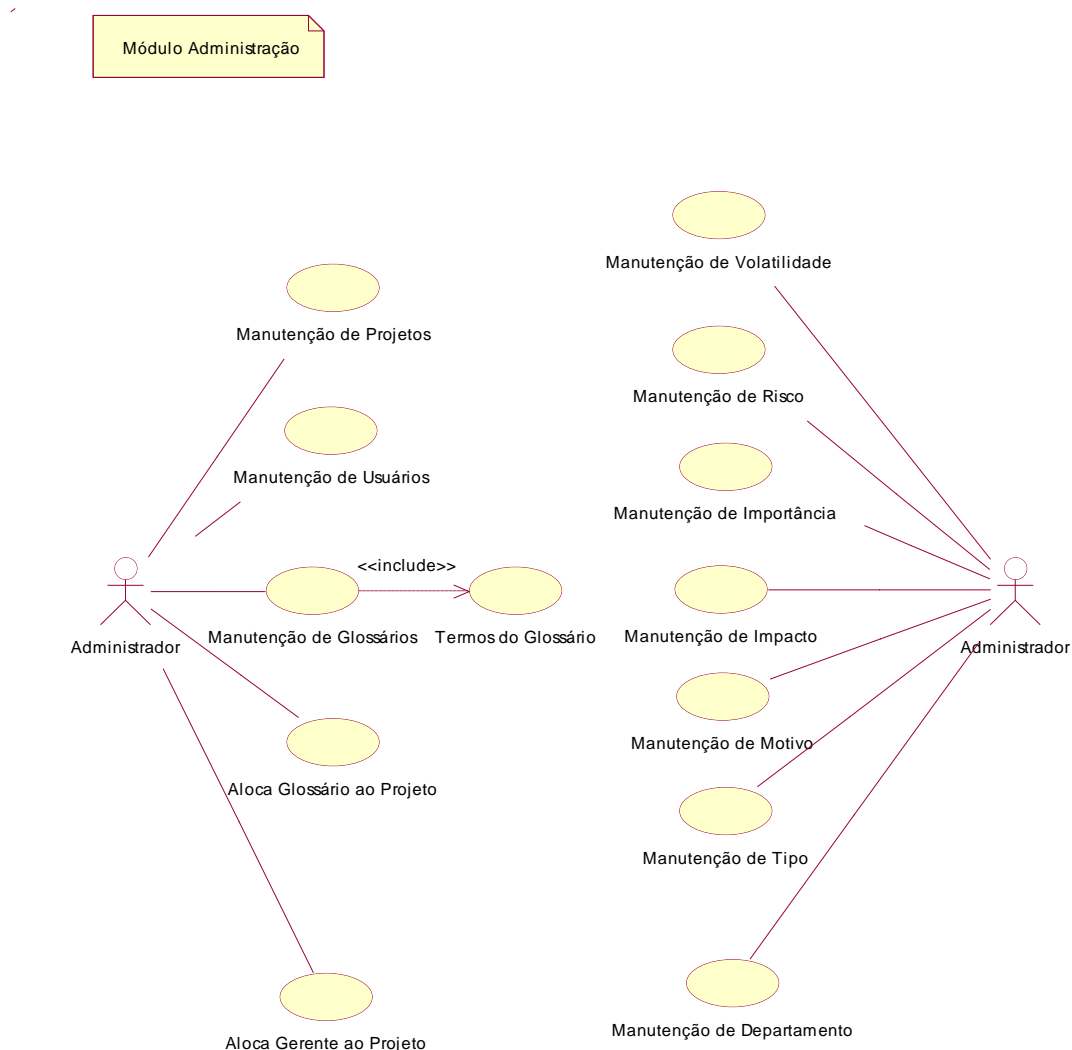


Figura 18 – Módulo Administração

Módulo Requisitos

O Módulo Requisitos contempla ações e tarefas específicas do Gerente do Projeto que estão abaixo descritas e especificadas em seguida na Figura 19:

a. Alocações de usuários

A primeira tarefa do Gerente do Projeto é alocar os usuários que devem fazer parte do projeto sob sua responsabilidade, nos níveis de alçada competentes. Os níveis de alçada indicam que o usuário é investido de determinadas funções para aquele projeto específico. Assim, por exemplo, um

determinado usuário poderá ter funções mais abrangentes no Projeto A e outras de menor responsabilidade no Projeto B.

b. Cadastro de requisitos

O Gerente do Projeto é o responsável pelo cadastro de todos os requisitos do Sistema, definindo:

- Tipo do requisito - identifica a classificação do requisito no sistema. Depende de cada organização, mas na literatura atual há predominância de classificá-los como Funcionais ou Não-Funcionais.
- Volatilidade - a volatilidade do requisito representa o grau de instabilidade do requisito no decorrer do tempo. Há necessidade do Gerente do Projeto estimar o comportamento deste requisito e ao longo do tempo, o próprio sistema, através de mecanismos de refinamento, irá depurando esta propriedade.
- Analista responsável – é o recurso humano que responderá por atributos importantes na análise, desenvolvimento e implantação das alterações propostas aos requisitos.
- Subdivisão de requisitos - a ferramenta permite ainda o desmembramento de um requisito em sub-requisitos, por exemplo:

R3 – As vendas poderão ser:

R3.1 – À vista em dinheiro

R3.2 – À Prazo (cheque ou faturado)

R3.3 – Com cartão de crédito

Neste caso cria-se uma dependência implícita que obrigatoriamente vincula os requisitos “filhos” ao requisito “pai”, de forma que um requisito “pai” só poderá ser excluído ou desativado, após todos os seus “filhos” terem sofrido processo semelhante. Deve-se ressaltar, entretanto, que os requisitos filhos tem sua própria “autonomia” conforme abaixo descrito.

c. Dependência de Requisitos

Outra importante função desempenhada pelo Gerente do Projeto é definir as dependências entre os requisitos, de forma a permitir rastreabilidade, que será fundamental no processo de análise do impacto e custos que a alteração de determinado requisito trará à organização. A dependência de requisitos não alcança a relação de dependência acima especificada, ou seja, um requisito “filho” pode ter relação de dependência com quaisquer outros requisitos e seu “pai” poderá não ter obrigatoriamente a mesma dependência.

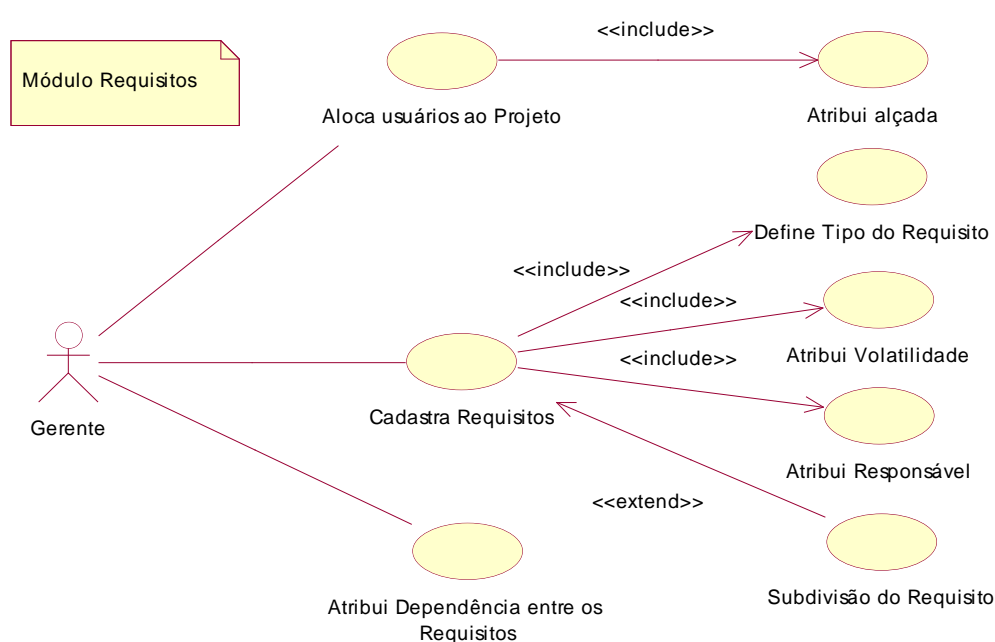


Figura 19 – Módulo Requisitos

Módulo Versão de Requisitos

Ao ser cadastrado, o requisito não possui versão, portanto para efeito prático lhe é atribuída a versão 0 (zero). A geração de uma alteração no requisito do sistema implica em criação de uma nova versão desse requisito, que somente pode ser executada pelo Gerente do Projeto ou pelo analista responsável do requisito.

Ações a serem tomadas quando é criada uma nova versão do requisito:

- a. Atribui impacto - mensura o impacto que a alteração trará ao sistema.

- b. Atribui risco - deve ser mensurado pelo analista responsável, qual o risco que a implementação da nova versão do requisito trará para o projeto.
- c. Atribui importância - identifica o grau de importância desta alteração para o projeto.
- d. Atribui prioridade - identifica a prioridade da implementação da versão, relativamente às demais implementações em andamento.
- e. Atribui motivo - informa a causa que motivou a geração de nova versão do requisito.
- f. Estima custo de requisito - é o número estimado de horas a ser consumida na implementação da versão do requisito que está sendo tratado. O sistema faz o tratamento do custo total da alteração proposta, que nada mais é que o somatório dos custos dos requisitos dependentes mais o custo do requisito origem.
- g. Relacionamento das dependências do requisito – o sistema usa a rastreabilidade do requisito especificada no Módulo Requisitos, para relacionar todos os requisitos dependentes (de forma recursiva), que deverão ser analisados independentemente pelos respectivos analistas responsáveis, com objetivo de gerar ou não alteração de versão do requisito. Estes analistas receberão comunicação do sistema em sua caixa de mensagens, informando-os do ocorrido (item i abaixo especificado). A cada requisito dependente que gera nova versão de requisito, o analista responsável deve atribuir valores conforme acima descrito, e assim sucessivamente até que todos os requisitos dependentes sejam analisados. A atribuição da situação das versões de requisitos é controlada internamente pelo sistema ou pelo Gerente do Projeto de acordo com o estágio em que a versão do requisito origem e dependentes se encontram, e são compostas de:
 - *Proposta* – no momento em que o analista responsável (ou o próprio Gerente do Projeto) propõe alteração de um requisito, que

chamamos “requisito origem”, é gerada uma nova versão deste requisito com situação “Proposta”. É responsabilidade do Gerente do Projeto, que tem a macro-visão do sistema, verificar a viabilidade técnica inicial dando continuidade ao processo atribuindo ao requisito a situação “em análise” ou declinando da alteração, procedendo a exclusão lógica da versão proposta.

- *Em análise* – ao atribuir a situação “em análise”, o Gerente do Projeto estende a todos os envolvidos no projeto oportunidade de tomar conhecimento da alteração proposta e participar através de um Fórum descrito abaixo, onde poderão se expressar a respeito da alteração solicitada e das alterações decorrentes nos requisitos dependentes.
 - *Aprovada* – após análise dos envolvidos, o Gerente do Projeto contempla todo contexto da alteração do requisito original e dos requisitos dependentes, observando custos, impactos, riscos, etc., que a alteração trará à organização e decidirá pela continuidade ou não. Se houver continuidade, todos os requisitos dependentes serão aprovados pelo Gerente na ordem ascendente e finalmente este aprovará a versão do requisito origem. Caso negativo, o Gerente declina da alteração tornando-a sem efeito.
 - *Em desenvolvimento* – em seguida à aprovação da alteração os analistas podem iniciar a implementação propriamente dita da alteração do requisito. Para indicar que este processo teve início, o analista responsável elege a situação “em desenvolvimento” ao requisito em questão.
 - *Implementada* – concluído os testes necessários é feita a implantação do requisito alterado e o analista responsável finaliza o processo de alteração do requisito atribuindo a esta a situação “implementada”.
- h. Fórum de requisitos – o Fórum de requisitos é uma área de livre acesso a todos os envolvidos no projeto e é destinado a registrar críticas,

sugestões, opiniões, alertas, entre outros, a respeito da alteração do requisito em análise. É aberto automaticamente quando o requisito entra no processo de análise e fechado pelo sistema quando o requisito está implementado.

- i. Notifica envolvidos – as notificações aos envolvidos nas alterações dos requisitos são feitas após o sistema identificar e relacionar todos os requisitos dependentes, através da emissão de uma mensagem a cada analista responsável pelo requisito dependente, informando que uma nova versão de requisitos foi gerada e que há necessidade de sua intervenção no processo de análise.
- j. Relacionamento de documentos de especificação – o analista responsável deverá ainda relacionar os documentos produzidos na especificação da nova versão como casos de uso, layouts, documentos textos, etc. A Figura 20 abaixo representa graficamente os casos de uso envolvidos no Módulo Versão de Requisitos.



Figura 20– Módulo Versão de Requisitos

4.3.2. MODELAGEM DE ATIVIDADES

O principal processo da ferramenta é o relacionado à alteração dos requisitos. De forma a melhor entender esse processo foram desenvolvidos diagramas de atividades que demonstram de maneira lógica, os passos percorridos da proposta de alteração do requisito até sua implementação. As atividades desenvolvidas seguem o processo de gerenciamento de mudanças nos requisitos definidos por Kotonya e Sommerville [KOT98], discutidos na subseção 3.1 e ilustrados nas Figuras 14 e 15 presentes na mesma subseção.

Atividade Proposta de alteração de requisito

O primeiro passo para criação de uma nova versão de requisito parte da necessidade identificada pelos *stakeholders* para correção ou melhoria (evolução) de um requisito. A atividade “Proposta de alteração de requisito”, representada pela Figura 21 abaixo demonstra a composição deste processo que contém as seguintes etapas:

- a. O usuário ou analista (que possuem alçada para tanto) solicita proposta para alteração de requisito no sistema. Esta proposta irá gerar uma nova versão do requisito origem.
- b. Nesta versão de requisito, o analista responsável define atribuições iniciais de impacto, risco, importância, prioridade, motivo e documentos ao requisito.
- c. O analista responsável define o custo em horas para implementação do requisito origem (neste momento, os requisitos dependentes ainda não foram analisados, assim, ainda não há possibilidade de obter o custo total da alteração).
- d. O sistema identifica e lista os requisitos dependentes.
- e. O Gerente do Projeto verifica a viabilidade inicial da alteração proposta analisando as atribuições, custo e os requisitos dependentes relacionados.

- f. Se a alteração for viável, o Gerente do Projeto atualiza a situação da versão do requisito de “proposta” para “em análise” e o sistema envia comunicado a todos os envolvidos notificando que há uma nova alteração a ser analisada, e abre o fórum de discussões para que todos possam se manifestar a respeito.

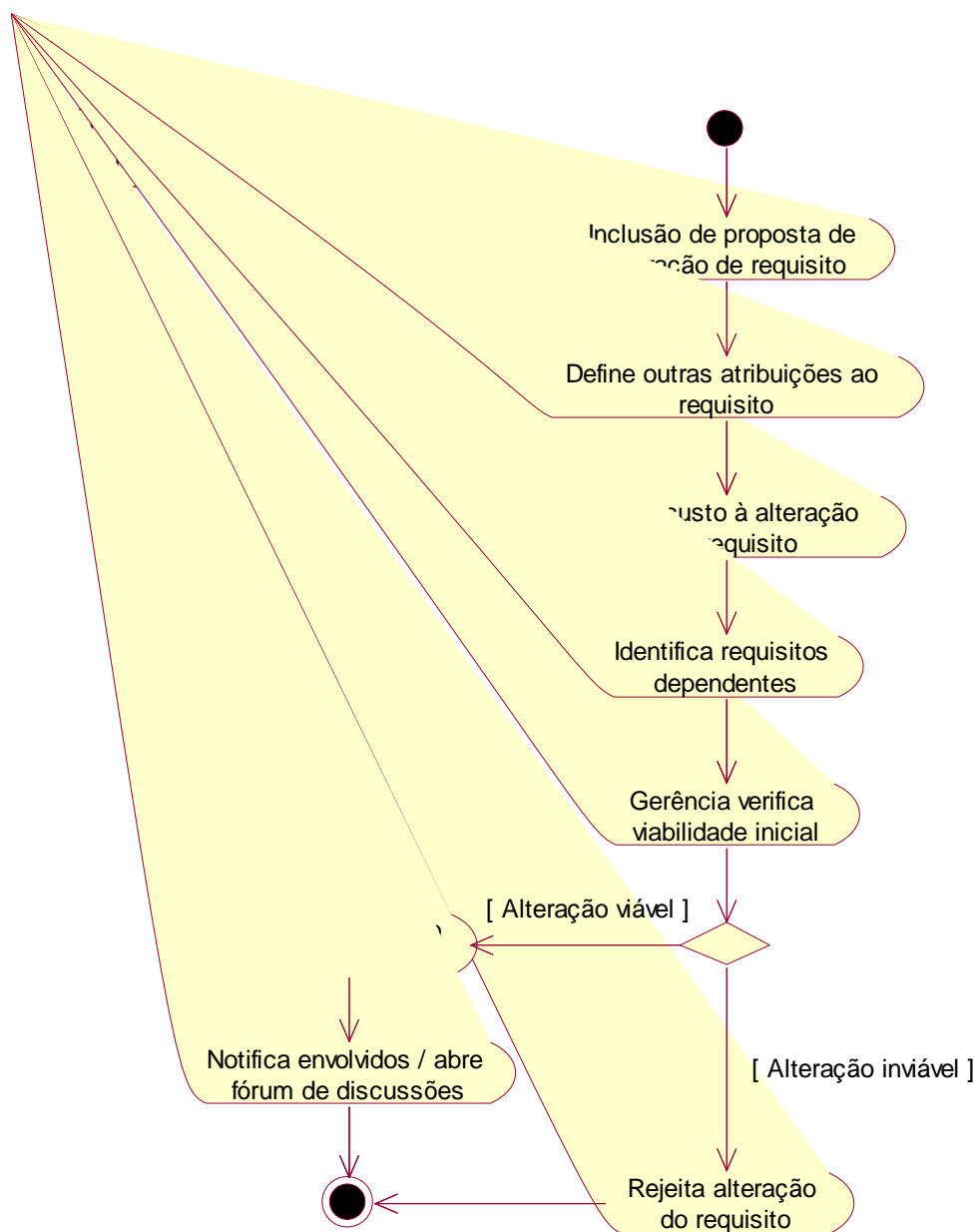


Figura 21 – Atividade Proposta de alteração de requisito

Atividade Análise de alteração de requisito

Após aprovação inicial e notificação dos envolvidos, estes deverão analisar detalhadamente os requisitos sob sua responsabilidade, de forma a subsidiar informações ao Gerente do Projeto, que tomará a decisão sobre a implementação ou não da alteração. Este processo se encontra representado pela atividade “Análise de alteração de requisito” (Figura 22 abaixo), que é composta pelas seguintes etapas:

- a. São identificados e relacionados todos os requisitos dependentes do requisito origem.
- b. Para cada requisito dependente encontrado, o analista responsável, previamente notificado, deverá proceder à análise do mesmo, verificando se haverá necessidade de alteração.
- c. Se for necessária a alteração do requisito dependente, o analista responsável irá definir a alteração, as atribuições iniciais de impacto, risco, importância, prioridade, motivo e documentos ao requisito e o custo em horas para implementação.
- d. Quando todos os requisitos forem tratados, a ferramenta disponibilizará ao Gerente do Projeto, o somatório dos custos dos requisitos dependentes e do requisito origem.
- e. De posse de todas as informações referentes à alteração solicitada (dados do requisito origem + dados dos requisitos dependentes + custo total em horas), o Gerente do Projeto decidirá pela aprovação. Se aprovada, a situação de todos os requisitos envolvidos será alterada para “aprovada” e os analistas envolvidos estarão liberados para a implementação. Caso contrário, o processo será finalizado.

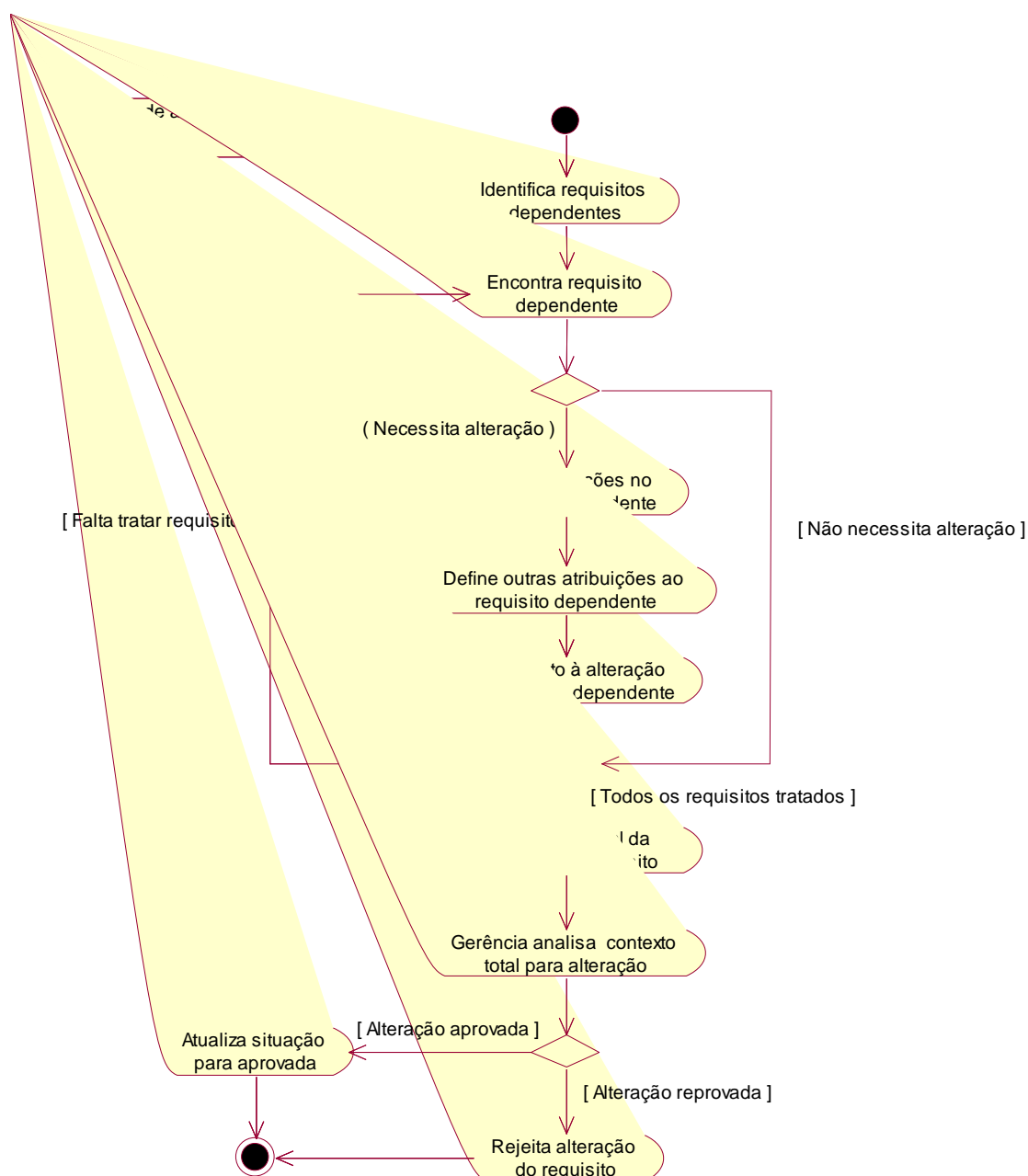


Figura 22 – Atividade Análise de alteração de requisito

Demais fases do processo de alteração do requisito

Após aprovação da alteração dos requisitos, os analistas responsáveis estão liberados para a implementação do mesmo. Assim que iniciar o processo de implementação de cada um dos requisitos (origem ou dependente), o analista responsável de cada requisito atualizará a situação para “em desenvolvimento” e quando este for finalizado, a situação deverá ser atualizada para “implementada”.

4.4. MODELAGEM LÓGICA DA FERRAMENTA

A modelagem lógica da ferramenta foi especificada através do diagrama de classes (UML) representado pela Figura 23 abaixo e permite uma visão geral das classes e relacionamentos que a compõe. A classe Projeto apresenta os dados relativos aos projetos que serão gerenciados na organização. Esta classe se relaciona com as classes Glossário e Termos, que contém os termos mais usuais da organização e da área afim ao projeto e devem ser utilizados nas descrições dos requisitos e de suas versões, evitando ambigüidades e trazendo padronização.

A classe Usuário contém os atributos dos *stakeholders* (analistas e usuários) do sistema, tais como nome, departamento, cargo, telefone, *e-mail*, *login*, senha e nível de conhecimento. A Classe Departamento identifica o vínculo organizacional do usuário. A classe UsuarioProjeto identifica os usuários envolvidos nos projetos, com alçada pré-determinada pelo Gerente de cada Projeto. A classe Alçada determina os níveis de autorização/restrição dos usuários à determinadas funções da ferramenta.

A classe Requisito identifica os requisitos de cada projeto e contém os atributos nome, tipo, volatilidade, responsável e dados estruturais (ordem, codigoCompleto, IdRequisitoPai e sub). Os dados estruturais representam a decomposição dos requisitos demonstrada no diagrama acima e serão base para geração da estrutura de subrequisitos. A classe DependenciaRequisito trata a rastreabilidade do requisito registrando a dependência entre os requisitos.

O processo de gerenciamento está centrado na classe VersãoRequisito, que contém a descrição da alteração solicitada, motivo da alteração, solicitante, responsável, descrição, prioridade, risco, importância, custo, impacto e situação da versão. A situação representa o *status* atual da versão do requisito, que poderá ser 'proposta', 'em análise', 'aprovada', 'em desenvolvimento' e 'implementada'. A classe AlteracaoVersao, apóia e complementa a classe VersaoRequisito, com dados de controle das versões atual, anterior e alteração origem.

A classe documentos permite ao analista relacionar todos os documentos envolvidos no processo de alteração, permitindo a rastreabilidade destes, através do registro do nome do documento, tipo do documento e localização.

As classes Motivo, Risco, Impacto, Prioridade, Importância e Volatilidade, representam elementos de apoio ao Gerenciamento da Versão de Requisitos e contém atributos para mensurar os riscos envolvidos nas alterações dos requisitos e de seus requisitos dependentes.

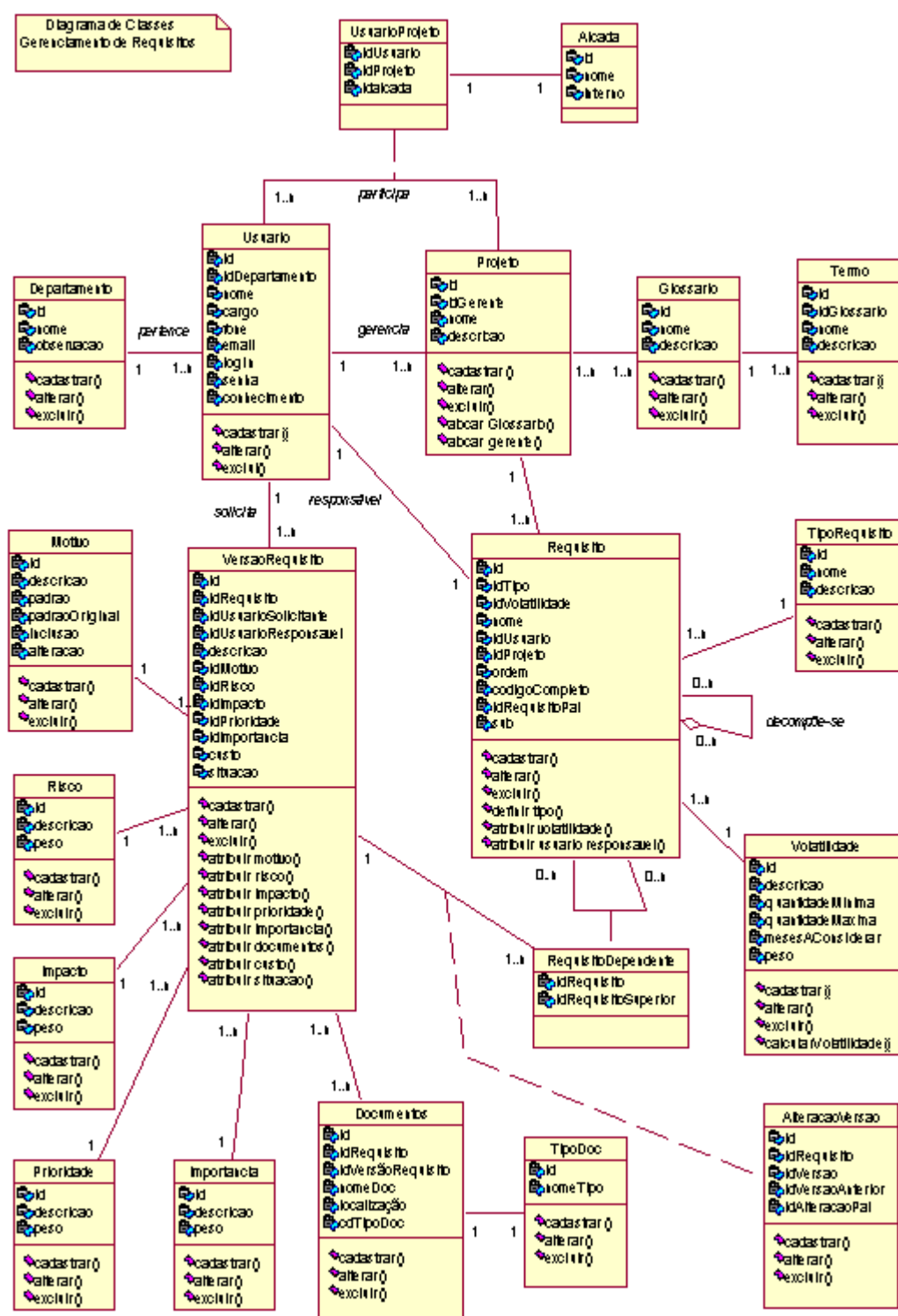


Figura 23 – Modelagem da Classe Gerenciamento de Requisitos

4.5. COMPARATIVO DE FERRAMENTAS DE GERENCIAMENTO DE REQUISITOS

Após o desenvolvimento da ferramenta, buscou-se um comparativo básico das características e principais funções da ferramenta desenvolvida em relação às ferramentas comerciais estudadas na subseção 3.3, para melhor contextualizá-la na comunidade de Engenharia de Software, conforme Tabela 8 abaixo.

Procedendo a uma rápida análise, podemos afirmar que as ferramentas possuem funcionalidades e características parecidas, valendo destacar:

- A ferramenta Sigerar utiliza banco de dados de uso livre, enquanto as demais utilizam banco de dados comerciais ou proprietário;
- Embora na elicitação de requisitos da ferramenta Sigerar tenha o Requisito Não-Funcional de que a mesma deva contemplar interoperabilidade, este requisito não foi implementado, sendo proposto para trabalhos futuros;
- A rastreabilidade da ferramenta Sigerar tem tratamento diferenciado das demais, pois quando está sendo proposta uma alteração de requisito, esta exige que todos os requisitos dependentes sejam analisados e tratados, de forma a subsidiar informações ao Gerente do Projeto na tomada de decisão sobre o aceite da alteração proposta. As demais ferramentas, demonstram os requisitos dependentes, sem contudo fazer o mesmo tratamento.
- O *Help On Line* da ferramenta Sigerar não é estático, podendo ser customizado, de acordo com as características da organização usuária.

Funcionalidades	Doors (Telelogic)	RequisitePro (Rational)	Caliber (Borland)	Sigerar
Controle de acesso e permissões	Sim	Sim	Sim	Sim
Banco de Dados	Proprietário	IBM DB2 Oracle SQL Server	Oracle SQL Server	<i>Firebird</i>
Notificações a usuários	Não	Sim	Sim	Sim
Fórum de discussões	Não	Sim	Sim	Sim
Glossários	Não	Sim	Sim	Sim
Controle de versões (Históricos)	Sim	Sim	Sim	Sim
Interoperabilidade	Sim	Sim	Sim	Não
Rastreabilidade	Sim	Sim	Sim	Sim
<i>Help on line</i>	Sim	Sim	Sim	Sim

Tabela 8 – Comparativo de Ferramentas para Gerenciamento de Requisitos

5. APLICABILIDADE DA FERRAMENTA

Após os testes iniciais da ferramenta partiu-se para a fase de aplicabilidade, utilizando-a em um estudo de caso real. Para a escolha do estudo de caso foram levantadas algumas possibilidades entre sistemas em desenvolvimento e outros já consolidados, de empresa fornecedora de *software* de gestão ligada à área de saúde, abaixo discutidos.

5.1. ESCOLHA DO ESTUDO DE CASO

Para se obter os melhores resultados na aplicabilidade da ferramenta foram efetuados estudos e considerações sobre alguns projetos de sistemas em desenvolvimento e outros já implantados da empresa fornecedora de sistemas de gestão na área de saúde, de forma a escolher dentro deste universo, o projeto que mais pudesse contribuir, abrangendo o maior escopo possível. Os sistemas considerados foram o Sistema de *Home Care*, o Sistema de PCMSO - Programa de Controle Médico de Saúde Ocupacional e Sistema de Gestão de Farmácias.

O Sistema de *Home Care* teve início em Junho de 2003, tendo sido implementado em cerca de 50%, mas infelizmente devido a mudanças estratégicas da empresa, o projeto foi temporariamente suspenso e com esta estagnação, seus requisitos não sofreram a evolução esperada e como a evolução dos requisitos e o registro das alterações dos mesmos são imprescindíveis para aplicabilidade da ferramenta, não foi possível aproveitá-lo.

O PCMSO - Programa de Controle Médico de Saúde Ocupacional foi outro sistema que mereceu a atenção para verificação da possibilidade de utilização para a aplicabilidade da ferramenta, pois este foi desenvolvido durante o ano de 2003, pela mesma empresa fornecedora de *software* de gestão na área de saúde, para atendimento de seus clientes. O escopo do sistema era suficiente para o estudo de caso, mas o mesmo foi descartado, pois a documentação encontrada não atendia requisitos mínimos para registrar os requisitos iniciais e a evolução dos mesmos no decorrer dos períodos subseqüentes.

O Sistema de Gestão de Farmácias mostrou-se mais adequado dentre os três apresentados para o estudo de caso, pois foi desenvolvido e implantado em 2002, tendo seu escopo, abrangência e documentação suficientes para registrar os requisitos iniciais e a evolução dos mesmos no decorrer dos períodos subseqüentes de forma a aplicar a ferramenta.

5.2. ESTUDO DE CASO SISTEMA DE GESTÃO DE FARMÁCIAS

Conforme acima descrito, o estudo de caso escolhido para a aplicabilidade da ferramenta foi o Sistema de Gestão de Farmácias. Inicialmente foram levantados todos os documentos envolvidos no desenvolvimento e manutenção do sistema e estes dados foram tabulados de forma a melhor organizar o material a ser usado no processo de aplicabilidade. Para melhor entendimento deste sistema, segue breve contexto.

5.2.1. CONTEXTO DO SISTEMA DE GESTÃO DE FARMÁCIAS

O desenvolvimento de um sistema de gestão de farmácias nasceu da necessidade de inúmeros clientes (empresas prestadoras de serviços médicos), oferecerem produtos farmacêuticos a preço de custo, de forma a atender as necessidades de seus usuários (pessoas físicas, pessoas jurídicas e seus funcionários, etc.), agregando com isto valor ao produto que operam (planos de saúde).

As compras de medicamentos para suprir a demanda são efetuadas com base nas necessidades apresentadas pela rotatividade dos produtos, utilizando mecanismos indicadores de “estoque mínimo” e “ponto de pedido”, pela apresentação de novos medicamentos indicados pelos médicos conveniados, etc., respeitando o *budget* disponibilizado para o negócio.

As vendas são efetuadas por pontos de vendas nos balcões (PDVs) e por pontos de vendas em retaguarda, através de atendimento ao cliente, via telefone (Disk-Medicamentos), com entrega domiciliar do produto adquirido, inicialmente sem taxa de entrega.

O controle de estoque faz o recebimento das mercadorias solicitadas aos fornecedores (fabricantes, distribuidores, laboratórios, etc.) por meio de pedidos de compras, efetua a devolução das mercadorias em desacordo com o pedido, repõe os produtos nas gôndolas dos balcões de atendimentos, despacha os produtos a serem entregues em domicílio, executa inventário rotativo e normal, entre outras atividades.

A Controladoria é responsável pela condução do negócio como um todo, fazendo os registros legais necessários junto aos órgãos públicos (Vigilância Sanitária, Prefeitura Municipal, etc.), gerenciando e auditando todos os processos internos e externos envolvidos. Também é responsável pela área fiscal, a qual deve emitir os livros fiscais de entradas e saídas de mercadorias, livros de apuração de entradas e saídas de ICMS e apurar os valores a crédito/débito de ICMS.

O Sistema foi desenvolvido para gerir os principais processos, tais como: compras, recebimentos, vendas, estoque, livros fiscais, entre outros e conta com apoio de ferramentas EDI e de Internet (*e-commerce*, *e-procurement*, *web*, *e-mail*, etc.), de forma a facilitar a comunicação das unidades de negócio, com seus fornecedores, laboratórios, associados, etc.

No desenvolvimento do sistema, os requisitos foram elicitados com a utilização de questionários e entrevistas com os principais *stakeholders* do sistema, uma vez que esta técnica já fora utilizada anteriormente com sucesso, e documentados em arquivos texto e planilhas.

5.2.2. RELAÇÃO DE REQUISITOS DO SISTEMA DE GESTÃO DE FARMÁCIAS

Abaixo, apresenta-se a Tabela 9, contendo os Requisitos Funcionais (RF) e a Tabela 10, contendo os Requisitos Não-Funcionais (RNF), inicialmente acordados entre os *stakeholders* do Sistema de Gestão de Farmácias. Deve-se ressaltar que os dados foram colhidos de maneira fidedigna da documentação original, e poderá haver imprecisões ou inconsistências, que não são objetos e tratativas deste trabalho. É apresentada também a Tabela 11, contendo a lista de

dependência dos requisitos, que é o ponto de partida para a rastreabilidade entre os mesmos.

Tabela 9 – Lista dos RF do Sistema de Gestão de Farmácias

Identificação do Requisito	Descrição do Requisito
R1	As vendas só poderão ser efetuadas aos clientes (Pessoas Físicas / Jurídicas), que devem apresentar o cartão no ato da compra
R2	A venda de medicamento controlado somente poderá ser efetuada mediante receituário que ficará retido na drogaria
R3	No balcão, os tipos de vendas poderão ser à vista ou à prazo
R3.1	À vista (dinheiro)
R3.2	À prazo (Cheque), que deverá ser consultado
R3.3	À prazo para as empresas conveniadas e seus respectivos funcionários
R3.4	À prazo para seus próprios funcionários
R4	Vendas através de telefone (Disk-Medicamentos) somente para clientes
R5	Deverá limitar e/ou bloquear venda à prazo aos convênios – por cliente
R6	Deverá possibilitar emissão de cupons fiscais de vendas, notas fiscais de vendas e devolução
R7	O Sistema deverá prever o estorno e/ou cancelamento da venda (total/parcial)
R8	Deverá tratar devolução de medicamentos vendidos
R9	As Operações de vendas, devoluções e cancelamentos deverão ser integradas com estoque
R10	Deverá promover o fechamento diário e mensal das vendas, para fechamentos dos caixas e integração com as áreas financeira e contábil
R11	Deverá emitir informações diárias e mensais dos movimentos de entrada e saída de medicamentos
R12	Deverá emitir relatórios estatísticos de vendas no período solicitado (Valores e quantidades – por tipo de venda)
R13	Deverá possibilitar o registro das vendas de medicamentos controlados (psicotrópicos, alucinógenos, anti-depressivos, etc.)
R14	Os preços dos medicamentos deverão ser atualizados conforme publicações padrões aceitas pelo mercado (ABCfarma, Brasíndice)
R15	A Controladoria deverá ter indicadores de atendimento e desempenho dos atendentes
R16	Deverá permitir recebimento dos produtos comprados, de acordo com os pedidos colocados. O recebimento deverá estar integrado com estoques e compras

Tabela 9 – Lista dos RF do Sistema de Gestão de Farmácias (cont.)

Identificação do Requisito	Descrição do Requisito
R17	No recebimento dos produtos, renegociar ou devolver os itens em desacordo com o pedido
R18	No recebimento dos produtos, deverá ser registrado o lote e data de vencimento, imprimindo etiquetas com código de barras, que deverão ser aplicadas nas embalagens individualizadas
R19	Os usuários deverão ter indicadores de estoque mínimo, máximo e ponto de pedido para orientação no processo de reposição de estoques
R20	Deverá permitir registrar ajustes de inventário com motivo
R21	Deverá haver rotina para inventário rotativo de estoque
R22	Deverá haver rotina para inventário periódico
R23	Deverá haver emissão de posição dos produtos em estoques vencidos e a vencer
R24	Deverá haver listas de produtos em ponto de pedido
R25	Deverá emitir histórico das últimas compras por produto / fornecedor
R26	Deverá ser listado o <i>Ranking</i> dos produtos mais vendidos
R27	Permitir cotações por meios convencionais e eletrônicos (<i>e-mail, Web, EDI</i>)
R28	Deverá registrar o retorno das cotações efetuadas
R29	Permitir escolha da melhor cotação levando em conta parâmetros como valor, condição de pagamento, data de entrega, etc
R30	Colocar os pedidos de compras, com base nas cotações vencedoras, utilizando meios convencionais e eletrônicos (<i>e-mail, Web, EDI</i>)
R31	Permitir o <i>follow-up</i> dos pedidos colocados
R32	Criar página na Internet, para divulgação dos produtos e preços

Tabela 10 – Lista dos RNF do Sistema de Gestão de Farmácias

Identificação do Requisito	Descrição do Requisito
R33	Os usuários deverão ter senhas de identificação, que possibilite estabelecer níveis de operação do sistema
R34	O controle de acesso deve ser vinculado ao horário de trabalho do funcionário
R35	Permitir que o sistema opere “ <i>off-line</i> ” em caso de contingência
R36	Controle de entrada e saída dos produtos com leitura do código de barra

Tabela 11 – Lista de Dependência dos Requisitos do Sistema de Gestão de Farmácias

Requisito Origem	Depende de
R1	R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R15
R2	R13
R3	R6, R7, R8, R10, R11, R12, R15
R6	R7, R8, R9, R10, R11, R12, R15
R7	R9, R10, R11, R12, R15
R8	R9, R10, R11, R12, R15
R16	R31
R17	R31
R18	R31
R21	R20
R29	R27, R28
R30	R27, R28, R29
R31	R30

5.2.3. EVOLUÇÃO DOS REQUISITOS DO SISTEMA DE GESTÃO DE FARMÁCIAS

A evolução dos requisitos do Sistema de Gestão de Farmácias foi recuperada de dados registrados em planilha Excel, e após revisão conjunta com o analista responsável pelo sistema, chegou-se às alterações representadas pela Tabela 12 abaixo, em ordem ascendente de data da solicitação. A tabela apresenta os seguintes dados:

- Nº - numeração seqüencial das alterações solicitadas.
- Alterações de Requisitos – descrição do novo requisito ou de alteração de requisito existente.
- Requisito Origem – é o código do requisito que sofrerá a alteração, oriundo das tabelas 9 ou 10 acima. Quando este dado tem a informação “Novo”, trata-se de nova funcionalidade requerida.
- Motivo Alteração – é o motivo que ocasionou a alteração e foi convencionado:
 - Evolução: quando se tratar de alteração referente à melhoria que indica evolução do sistema;
 - Correção: a alteração se faz necessária para corrigir problemas encontrados;

- Legal: a alteração é necessária por motivo de Lei ou determinação hierárquica superior (Unimed Brasil / Federação).
- Data Solicit. – é a data da solicitação da alteração.
- Pri – indica a prioridade que a alteração tem em relação à outras alterações solicitadas na mesma data.
- Custo (hs/h) – representa o custo estimado em horas / homem para implementar a solicitação de alteração ou de novo requisito.
- Início / Término – são as datas reais de início e término da implementação do solicitado.
- Observação – é o estado atual do requisito. Pode-se observar que a maioria das alterações de requisitos está na condição de “implementada”, visto tratar-se de dados históricos. Há ainda de se ressaltar que há casos de solicitações de requisitos que não foram implementadas em razão de impedimento legal, veto do Gerente do Projeto ou ainda estar em desenvolvimento.

Tabela 12 – Evolução dos Requisitos do Sistema de Gestão de Farmácias

Nº	Alterações de Requisitos	Requisito Origem	Motivo Alteração	Data Solicit.	Pri	Custo (hs/h)	Início Término	Observação
1	Criar no Cadastro de Cliente a situação de cliente Inadimplente	R1	Evolução	01/07/02	0	8	01/07/02 03/07/02	Implementada
2	Atualização de Preços de Medicamentos na WEB	R32	Evolução	04/07/02	0	12	05/07/02 11/07/02	Implementada
3	Permitir a exclusão do item serviço de entrega do Disk-Medicamentos	R4	Evolução	11/07/02	0	8	24/07/02 25/07/02	Implementada
4	Relatório de Débitos de Cooperados	Novo	Evolução	23/07/02	0	8	19/08/02 21/08/02	Implementada

Tabela 12 – Evolução dos Requisitos do Sistema de Gestão de Farmácias (cont.)

Nº	Alterações de Requisitos	Requisito Origem	Motivo Alteração	Data Solicit.	Pri	Custo (hs/h)	Início Término	Observação
5	Incluir hora da entrega no relatório do Disk-Medicamentos	R4	Evolução	27/08/02	0	4	09/09/02 09/09/02	Implementada
6	Implementar taxa de entrega para compras de até determinado valor; acima deste valor, as entregas estarão isentas da taxa	R4	Evolução	11/07/02	0	16	12/09/02 16/09/02	Implementada
7	Inventário rotativo - Escolha de n itens aleatórios para inventário	Novo	Evolução	21/08/02	0	40	16/09/02 03/10/02	Implementada
8	Implementar controle de medicação de uso contínuo	Novo	Evolução	21/08/02	1	40	15/10/02 01/11/02	Implementada
9	Alterar programa para atender legislação sobre nova numeração de CFOP	R6	Legal	12/12/02	0	24	13/12/02 16/12/02	Implementada
10	Relatório Mensal de compras separados por percentual de Pis/Cofins	Novo	Legal	17/12/02	0	16	17/12/02 19/12/02	Implementada
11	Emitir relatório de compras médias de 3 meses, com valor médio + Custo + Fabricante + Fornecedor	Novo	Evolução	17/12/02	1	8	20/12/02 20/12/02	Implementada
12	Relatório de Movimentação diária de Psicotrópicos	R13	Legal	20/12/02	0	16	23/12/02 30/12/02	Implementada
13	Limite de venda para convênios de empresas e retenção de receitas	R3.3	Evolução	02/09/02	0	32	20/01/03 30/01/03	Implementada
14	Alteração de mensagem de retenção de receitas de convênios	R3.3	Evolução	03/03/03	0	4	20/03/03 20/03/03	Implementada

Tabela 12 – Evolução dos Requisitos do Sistema de Gestão de Farmácias (cont.)

Nº	Alterações de Requisitos	Requisito Origem	Motivo Alteração	Data Solicit.	Pri	Custo (hs/h)	Início Término	Observação
15	Livro Psicotrópicos: Criar manutenção de Livro, para corrigir erros de digitação de balcão	R13	Legal	29/04/03	0	32	02/05/03 27/05/03	Implementada
16	Livro Psicotrópicos: Incluir páginas dos produtos que não tiveram movimento no período, contendo apenas saldo do estoque	R13	Legal	29/04/03	1	16	28/05/03 03/06/03	Implementada
17	Relatório de Vendas: Separar os medicamentos vendidos, por fornecedor / laboratório e incluir o campo telefone dos mesmos	R12	Evolução	29/04/03	2	16	03/06/03 09/06/03	Implementada
18	Livro Psicotrópicos: Alterar apresentação da página e acrescentar o campo apresentação do produto	R13	Legal	29/04/03	3	24	09/06/03 13/06/03	Implementada
19	Permitir acumular várias vendas para emissão de um único cheque	R3.2	Evolução	20/02/04	0			Não Implementada - Procedimento ilegal
20	Criar relatório do número de entregas de medicamentos vendidos através do serviço Disk-Medicamentos (Sede e região)	R4	Evolução	26/05/04	1	8	26/10/04 27/10/04	Implementada

Tabela 12 – Evolução dos Requisitos do Sistema de Gestão de Farmácias (cont.)

Nº	Alterações de Requisitos	Requisito Origem	Motivo Alteração	Data Solicit.	Pri	Custo (hs/h)	Início Término	Observação
21	No recebimento de medicamentos comprados, a entrada dos produtos deverá ser feita através do código de barras (desde que possível)	R36	Evolução	26/05/04	0	4	28/07/04 29/07/04	Implementada
22	Criar controle específico para bloquear vendas a usuários com cheques devolvidos no balcão e Disk-Medicamentos	R3.2	Evolução	02/08/04	0	8	03/08/04 05/08/04	Implementada
23	Corrigir o programa de vendas através do serviço Disk-Medicamentos para não permitir mais de uma forma de pagamento	R4	Correção	15/10/04	0	8	25/10/04 27/10/04	Implementada
24	Mudar a logomarca para nova padronização segundo orientação da Federação	Novo	Legal	27/10/04	0	16	28/10/04 04/11/04	Implementada
25	No pedido de medicamentos através do serviço de Disk-Medicamentos, informar a data da última compra na consulta de produtos	R4	Evolução	07/12/04	0	4	10/12/04 10/12/04	Implementada
26	Verificar diferenças no relatório movimento de caixa com as reduções Z	R10	Correção	10/12/04	0	4	13/12/04 13/12/04	Implementada
27	Excluir estorno de uma nota de saída feito erroneamente e acertar os saldos dos itens	R7	Correção	17/12/04	0	4	20/12/04 20/12/04	Implementada
28	Promover acertos no relatório de valorização de estoque e reimprimir todo ano de 2004	R11	Correção	05/01/05	0	8	05/01/05 06/01/05	Implementada

Tabela 12 – Evolução dos Requisitos do Sistema de Gestão de Farmácias (cont.)

Nº	Alterações de Requisitos	Requisito Origem	Motivo Alteração	Data Solicit.	Pri	Custo (hs/h)	Início Término	Observação
29	Incluir indicativo de fechamento mensal de estoque, para evitar que um mês já valorizado seja reprocessado	R10	Evolução	07/01/05	0	8	11/01/05 12/01/05	Implementada
30	Criar novo campo no cadastro de produtos (Saldo Zero) para informar se o produto está com saldo zero e sem movimentação há pelo menos um ano	Novo	Evolução	25/01/05	0	16	26/01/05 28/01/05	Implementada
31	Gerar relatório dos medicamentos com saldo zero sem movimentação nos últimos 12 meses	Novo	Evolução	25/01/05	1	8	28/01/05 31/01/05	Implementada
32	Revisar e corrigir falhas nos programas de inventário. Acertar lançamentos que ficaram de fora do inventário	R22	Correção	29/01/05	0	16	31/01/05 02/02/05	Implementada
33	Correções na tela do pedido de compras - inserir estatísticas de últimas compras e últimas vendas	R30	Correção	04/03/05	0	4	07/03/05 07/03/05	Implementada
34	No relatório de vendas por laboratório colocar opção para imprimir ou não logotipo, saldo e valores	R12	Evolução	04/03/05	1	4	08/03/05 08/03/05	Implementada
35	Implementar controle de cheques em custódia	Novo	Evolução	04/03/05	2	16	21/03/05 30/03/05	Implementada
36	Obrigatório venda de medicamentos por código de barras, exceto produtos sem código e vendas pelo Disk-Medicamentos	R1	Evolução	26/05/05	0	8	26/05/05 27/05/05	Implementada
37	Listagem da última compra realizada através do Disk-Medicamentos	R4	Evolução	02/06/05	0	4	07/06/05 07/06/05	Implementada

Tabela 12 – Evolução dos Requisitos do Sistema de Gestão de Farmácias (cont.)

Nº	Alterações de Requisitos	Requisito Origem	Motivo Alteração	Data Solicit.	Pri	Custo (hs/h)	Início Término	Observação
38	Ajustes nas telas do sistema para adequação ao Windows XP	Novo	Evolução	12/08/05	0	8	12/08/05 16/08/05	Implementada
39	Implementação no sistema do módulo de Vendas do Laboratório Novartis	Novo	Evolução	09/09/05	0	320	16/9/2005	Em Desenvolvimento

5.3. APLICAÇÃO DA FERRAMENTA NO ESTUDO DE CASO

Uma vez consolidados os dados acima apresentados partiu-se para a aplicação da ferramenta utilizando o Sistema de Gestão de Farmácias objeto do estudo de caso. Os recursos e funcionalidades da ferramenta serão apresentados de acordo com as atividades desenvolvidas.

5.3.1. INTERFACES DA FERRAMENTA COM O USUÁRIO


No desenvolvimento da ferramenta procurou-se adotar uma padronização amigável de forma a facilitar a interface com o usuário. Abaixo, segue a descrição dos ícones e da padronização adotada que poderão ser observados nas figuras apresentadas a partir da subseção 5.3.2.

Pesquisa – todas as telas da administração tem uma área de pesquisa, semelhante à Figura 24 abaixo, onde é possível localizar rapidamente o item procurado através do filtro por argumento digitado.


Figura 24 – Pesquisa por argumento


✚ **cadastrar novo projeto** – quando acionado este *link*, leva o controle do programa para a tela de cadastro de novo item.

< anterior próximo > – tem como função facilitar a navegação nas telas de cadastros.

 **Ajuda** – ao clicar neste ícone é aberta uma nova tela contendo texto explicativo sobre a tela corrente. Vale ressaltar que o administrador da ferramenta dispõe de cadastro específico, onde o texto da ajuda pode ser editado para alteração personalizada para a organização.

< sair – este ícone presente na parte direita superior das telas, quando acionado, retorna à tela inicial da ferramenta.

 **topo** - este ícone presente na parte inferior esquerda das telas, quando acionado, retorna ao topo da tela.

 - este ícone é um atalho para editar e alterar os dados referentes ao item selecionado.

 - este ícone é um atalho para excluir o item selecionado.

5.3.2. APLICABILIDADE DO MÓDULO ADMINISTRAÇÃO

A primeira ação a ser tomada por quaisquer organizações que utilizar a ferramenta será eleger um administrador que preparará o ambiente onde a mesma será implantada e parametrizada, segundo as normas e características da própria organização. No acesso ao módulo de Administração do Sistema é apresentada a tela de identificação do administrador da ferramenta, conforme Figura 25 abaixo, onde o mesmo deverá digitar seu *login* e a senha correspondente.

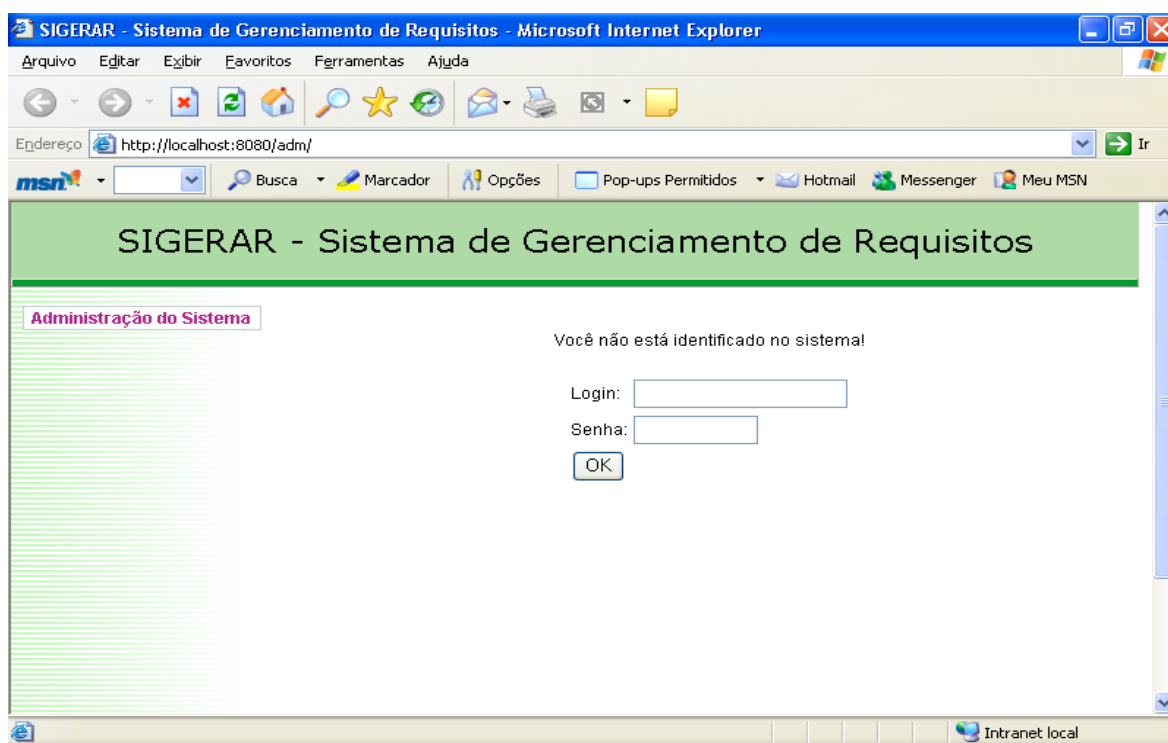


Figura 25 -Tela de Login do Administrador da Ferramenta

Caso o login ou senha não sejam reconhecidos, aparecerá a mensagem “**Login/senha inválidos!**”. Caso contrário, será apresentada a tela da página inicial da administração da ferramenta que contém opções para cadastros e parametrizações. Descreveremos a seguir, as atividades executadas para criação do ambiente de aplicabilidade através do estudo de caso:

Cadastrando Glossários e seus termos

A primeira tarefa do administrador é cadastrar glossários específicos cujos termos serão utilizados nas descrições de requisitos. Conforme demonstram as Figuras 26 e 27 abaixo, foi criado o glossário Farmácias para suportar o estudo de caso. Notar que também há um outro glossário ativo (Geral) que foi criado anteriormente e contém dados genéricos referentes à área de saúde.

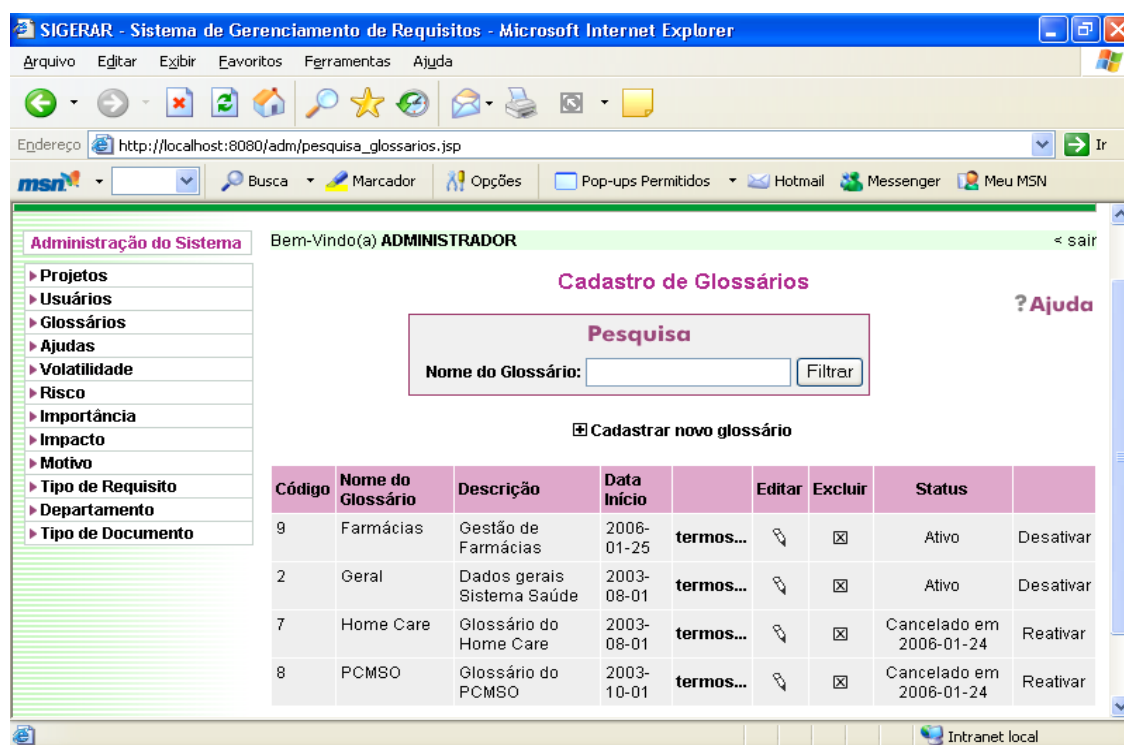


Figura 26– Cadastro de Glossários

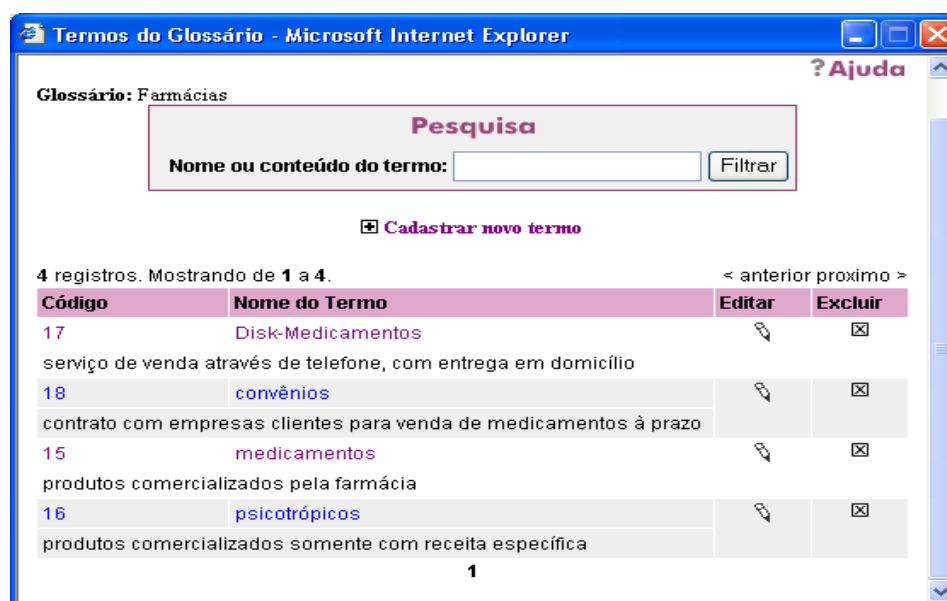


Figura 27 – Cadastro de Termos do Glossário

Cadastrando Departamentos e Usuários – a tarefa seguinte do administrador foi cadastrar todos os Departamentos e Usuários do sistema que serão alocados ao projeto “Farmácias” como mostra a Figura 28 abaixo.

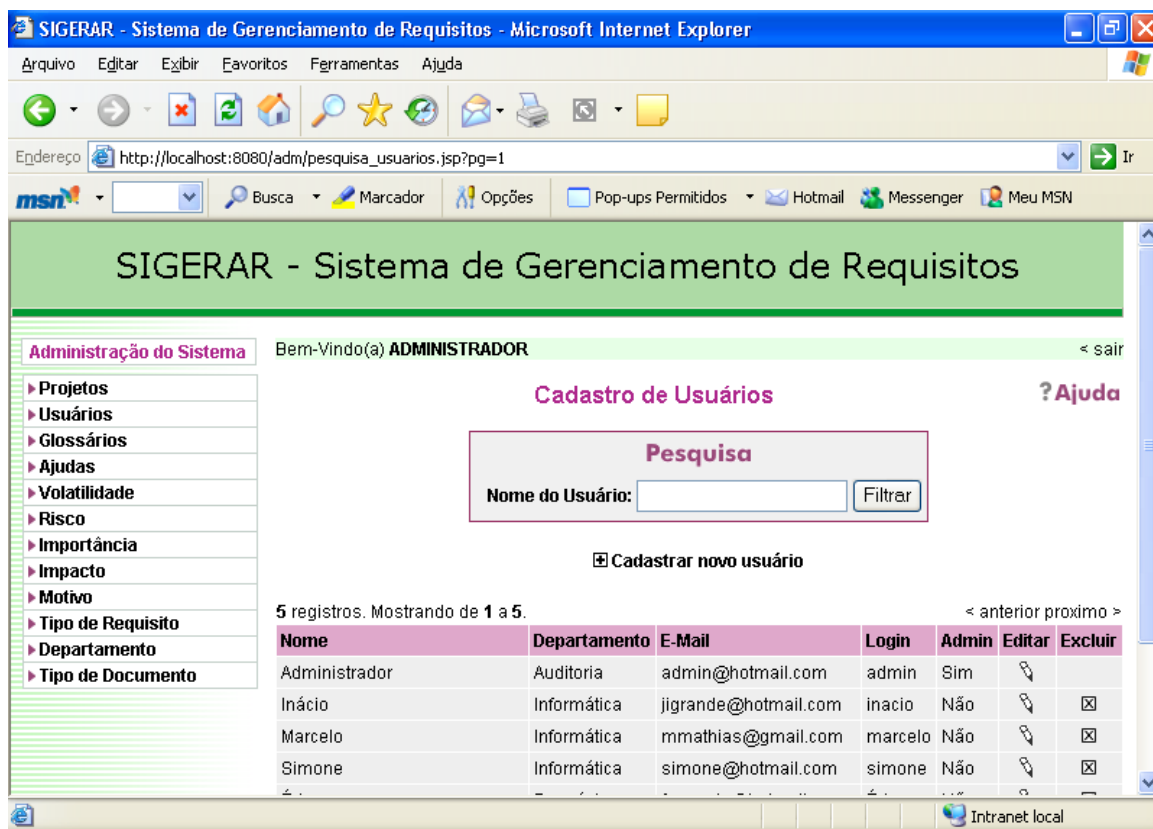


Figura 28 – Cadastro de Usuários

Cadastrando o projeto

A Figura 29 abaixo representa o cadastro do projeto “Farmácias” com a definição de um Gerente do Projeto e a alocação dos glossários “Farmácias” e “Geral”, que serão utilizados para melhorar o entendimento dos envolvidos sobre termos específicos da área farmacêutica.

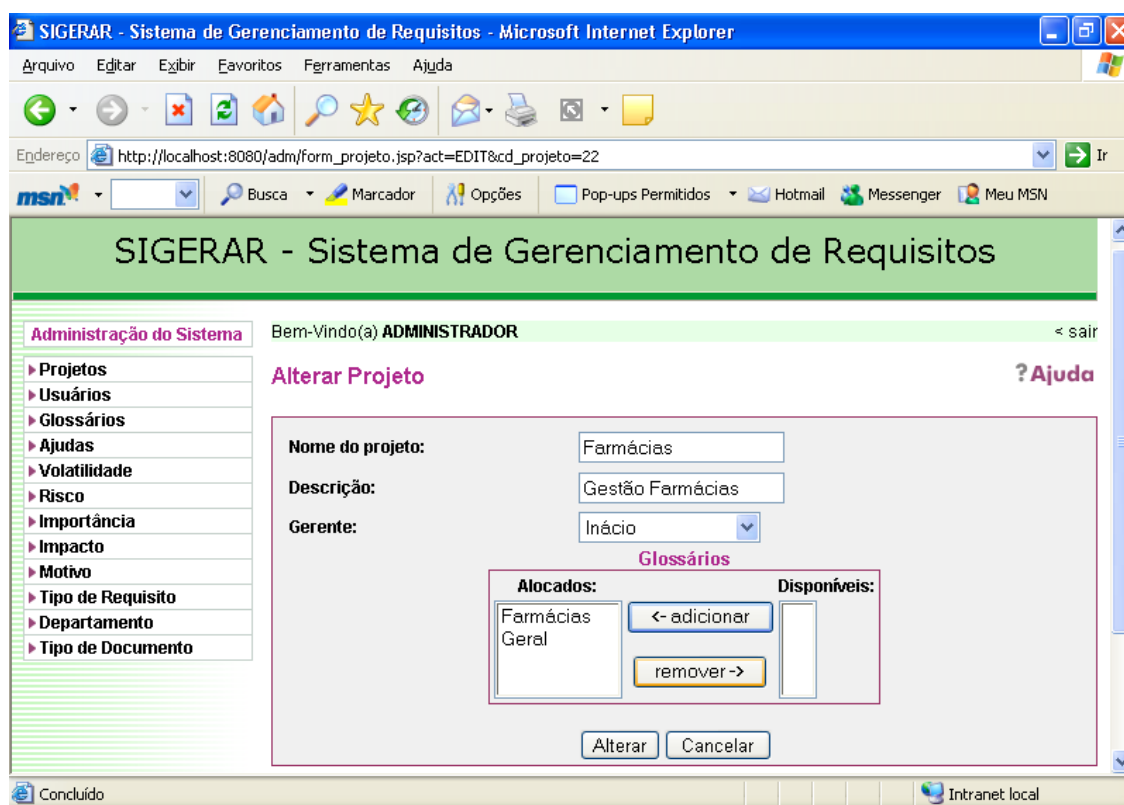


Figura 29 – Cadastro de Projetos

Definição dos demais Cadastros

Em continuidade à configuração do ambiente foram definidos os dados dos cadastros de Volatilidade, Risco, Importância e Impacto, e de comum acordo, os envolvidos decidiram pela uniformidade dos dados que receberam as atribuições de: “Baixo(a) – Peso 1”, “Médio(a) – Peso 5” e “Alto(a) – Peso 10”. No cadastro “Tipo de Requisito” estes foram classificados como Funcionais e Não-Funcionais e no cadastro “Tipo de Documento” foram cadastrados Word, Excel e UML.

O cadastro “Motivo” recebeu os dados “Evolução”, “Legal” e “Correção”, constantes da Tabela 11, ressaltando que os motivos “Inclusão de Requisitos” e “Alteração por Dependência” são tratados internamente na ferramenta e desta forma o sistema não permite exclusão dos mesmos.

5.3.3. APLICABILIDADE DO MÓDULO PRINCIPAL

Após a configuração inicial do ambiente, o Gerente do Projeto e os demais usuários foram liberados para a utilização do “Menu Principal” que é o módulo da ferramenta onde é tratado o gerenciamento dos requisitos. De forma idêntica ao módulo “Administração”, há uma tela de acesso ao módulo “Principal” onde o usuário se identifica através de *login* e senha. Caso o usuário obtenha acesso é apresentada a tela inicial conforme Figura 30, que o orienta a selecionar o projeto desejado.

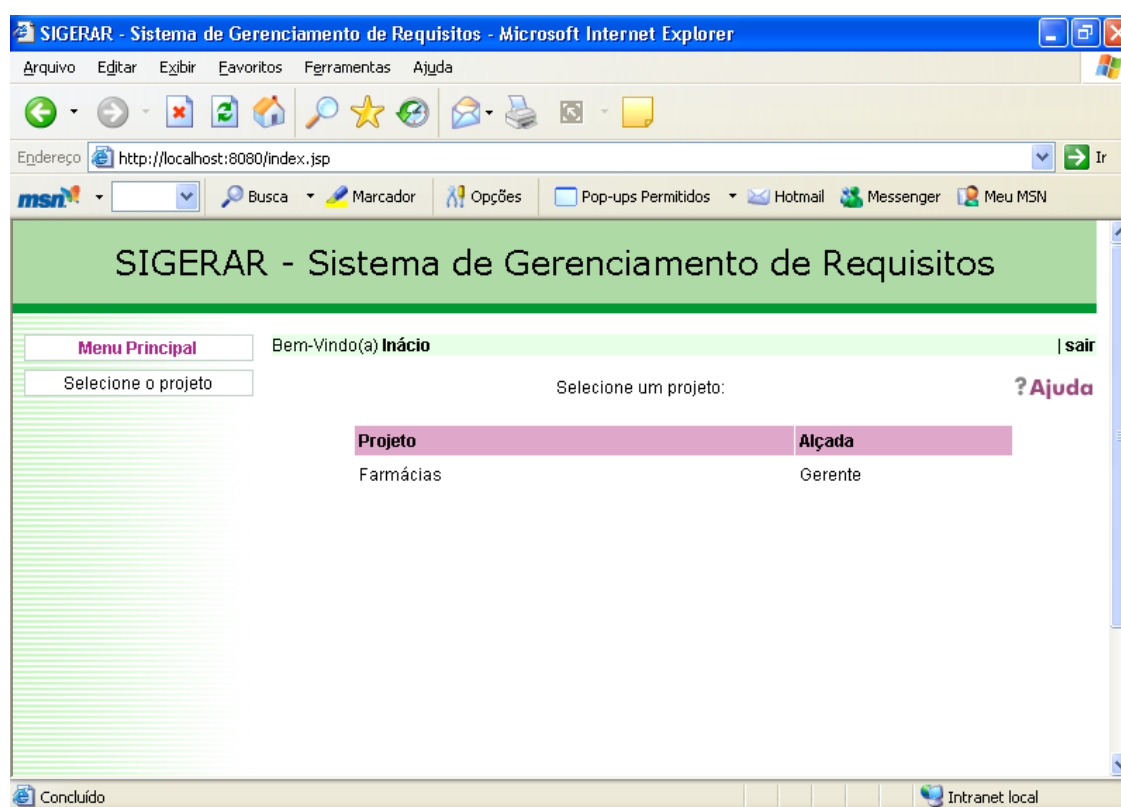


Figura 30 – Seleção de Projeto

No Estudo de Caso, após a escolha do projeto “Farmácias” o sistema apresenta a página inicial do projeto selecionado contendo as opções disponíveis conforme a alçada do usuário. Esta tela apresenta o acesso feito pelo Gerente do Projeto, assim, estão presentes as opções: “Usuários do Projeto”, “Requisitos do Projeto” e “Alterações de Versões”. Descrevemos a seguir as atividades executadas para alocação dos usuários ao projeto e tratamento dos requisitos (inclusão / alteração) conforme dados tabulados anteriormente.

Alocação de Usuários ao Projeto

Conforme Figura 31, o Gerente do Projeto alocou os envolvidos no projeto “Farmácias”, atribuindo Permissão (alçada) específica ao usuário para o projeto em questão.

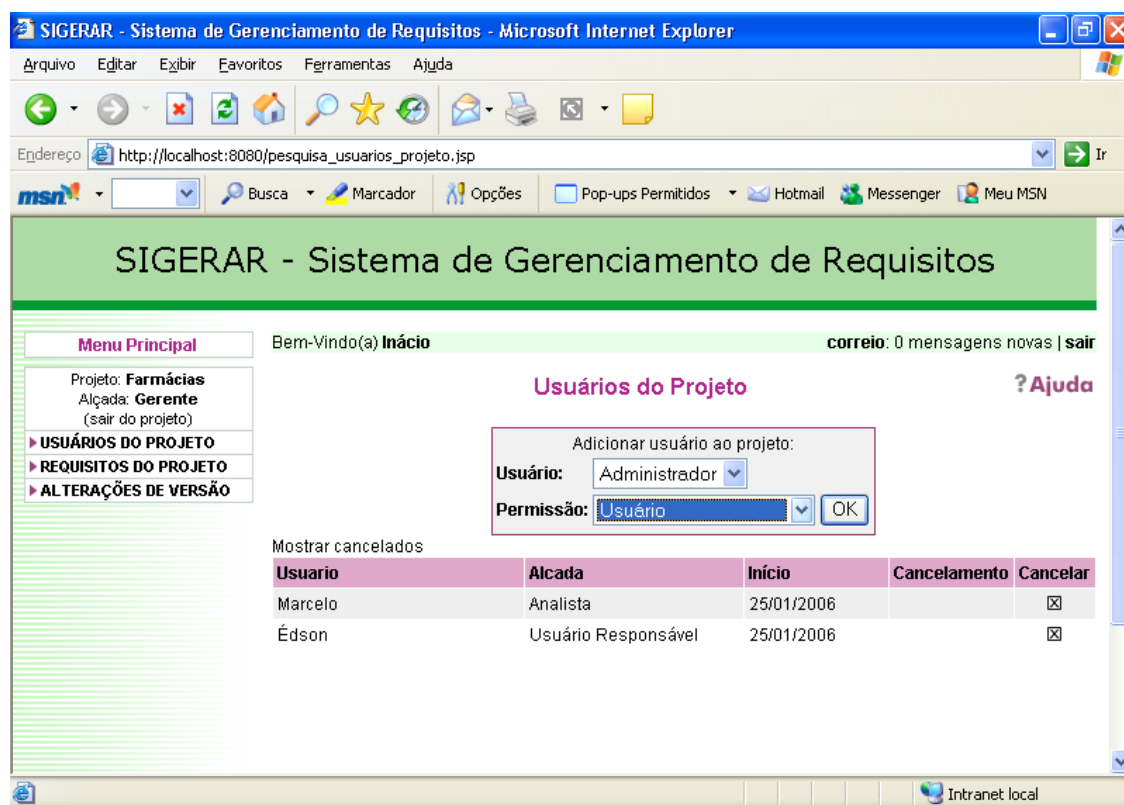
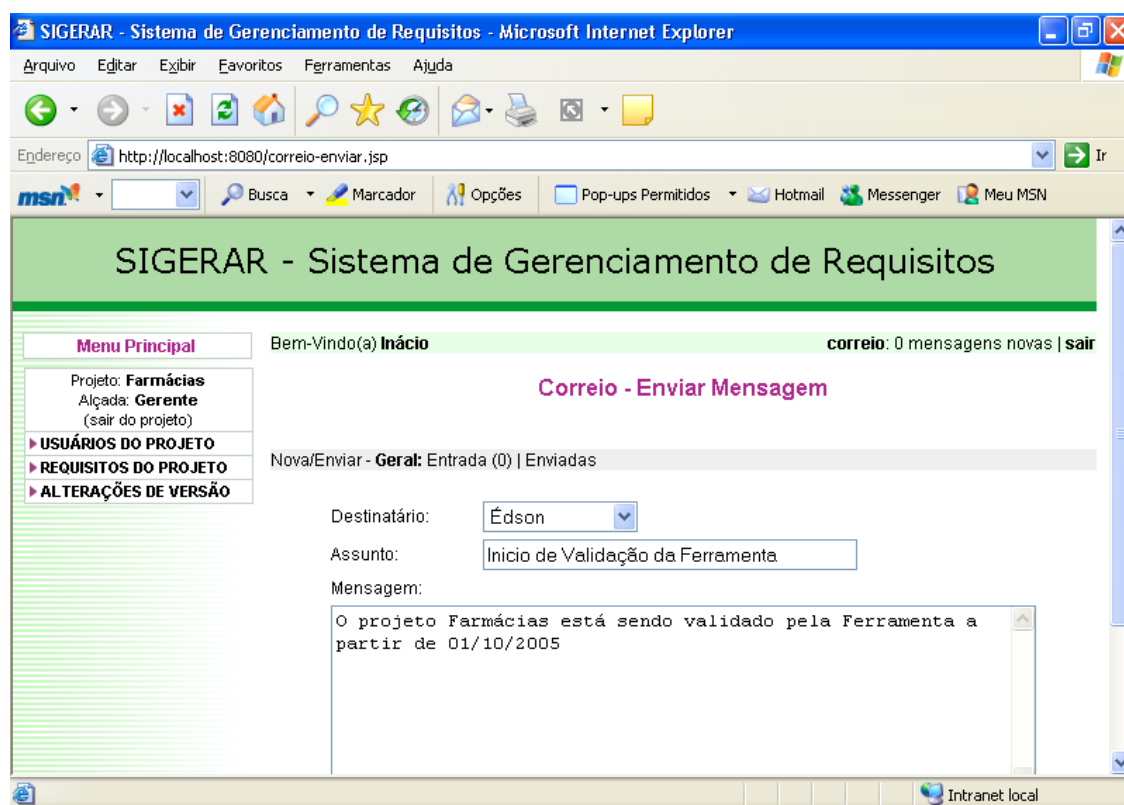


Figura 31 – Alocação de Usuários ao Projeto

Após a alocação dos envolvidos no projeto, estes estão automaticamente aptos a trocarem mensagens através do correio interno. No canto superior direito da tela há o ícone [**correio: 0 mensagens novas**] que é o atalho para o correio do projeto onde é feito o tratamento de envio e recebimento de mensagens aos envolvidos no projeto. No exemplo representado na Figura 32 o Gerente do Projeto envia mensagem a um dos envolvidos no projeto notificando-o que houve o início da validação da ferramenta através do projeto “Farmácias”. Assim, quando o usuário envolvido abrir a sessão de trabalho, irá verificar que tem novas mensagens e acessará o correio.



Figuras 32– Correio interno

Inclusão dos requisitos iniciais do projeto

Nesta atividade todos os requisitos iniciais do projeto listados nas Tabelas 8 e 9 foram incluídos na ferramenta conforme exemplificado na Figura 33 abaixo. Podemos observar que nesta tela, a descrição do requisito foi digitada e os dados Tipo e Volatilidade foram selecionados dos respectivos *combobox* que apresentam os dados cadastrais parametrizados anteriormente no módulo administração. Convencionou-se cadastrar todos os requisitos com volatilidade “baixa”, pois se trata do cadastro inicial e um dos objetivos é o próprio sistema apurar periodicamente a volatilidade dos requisitos conforme descrito na subseção 4.2.1. – Lista dos Requisitos Funcionais da Ferramenta (R7). O Responsável foi selecionado da relação dos usuários alocados conforme item a acima. Notar que no final da descrição, aparece o texto “(1)”, que é o código sequencial atribuído ao requisito pelo sistema.

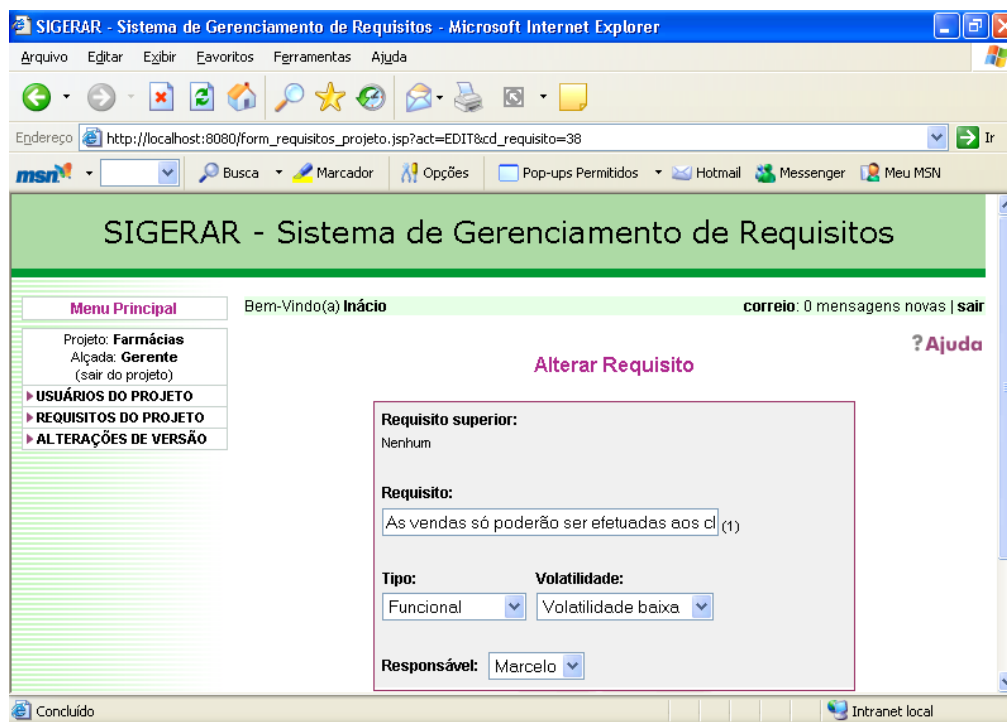


Figura 33 – Manutenção de Requisitos

Na Figura 34 a seguir podemos verificar que todos os requisitos iniciais foram incluídos na ferramenta e apresentam no final da descrição o texto “**(sem versão)**”, significando que é um requisito original sem nenhuma alteração.

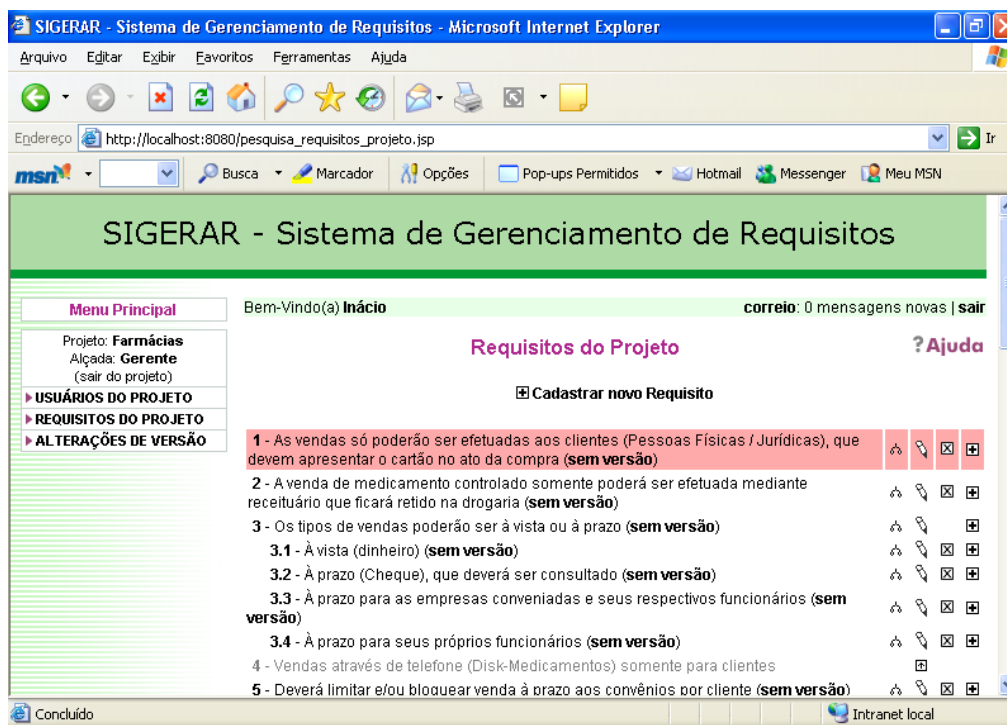


Figura 34 – Requisitos do Projeto

No final da descrição dos Requisitos do Projeto encontramos ícones que exercem as seguintes funções:

🌳 - este ícone representando estrutura de árvore, é utilizado para estabelecer a dependência entre os requisitos.

✎ - é utilizado para editar e alterar os dados referentes ao requisito selecionado.

✖ - é utilizado para excluir o requisito selecionado.

⊕ - é utilizado para incluir um sub-requisito. Notar no exemplo abaixo que após a criação dos sub-requisitos 3.1, 3.2, 3.3 e 3.4, o requisito 3 (que é chamado de requisito “pai”) perde o ícone ✖ que é utilizado para exclusão, ou seja, ele não poderá ser excluído enquanto os requisitos filhos estiverem ativos.

⊞ - Após a exclusão de um requisito a descrição aparece em tonalidade mais clara que os requisitos ativos e o ícone é apresentado. Sua função é recuperar (reativar) o requisito excluído.

O passo seguinte à inclusão dos requisitos iniciais é a inclusão da relação de dependências entre os requisitos conforme a Tabela 10 e é feita na tela de Requisitos do Projeto, clicando o *mouse* no ícone 🌳 localizado à direita do requisito. A Figura 35 abaixo mostra a tela desta atividade onde o requisito tratado é apresentado e na relação dos requisitos fica destacado com a cor verde. Os demais requisitos são apresentados com um *checkbox* ao lado esquerdo que deverá ser clicado com o *mouse* quando este tiver relação de dependência (depende de). Após o clique do *mouse*, o *checkbox* recebe a marcação e o requisito fica destacado com a cor amarela. Se o mesmo for clicado novamente, o *checkbox* é desmarcado e a função é desfeita. Ao término das marcações clica-se no botão “Gravar Alterações” no final da página. A relação de dependência apresentada (depende de) significa que quando o requisito tratado sofrer alguma proposta de alteração, todos os requisitos relacionados como dependentes deverão ser examinados pelos seus respectivos analistas responsáveis, a fim de verificarem se estes também deverão ser modificados.

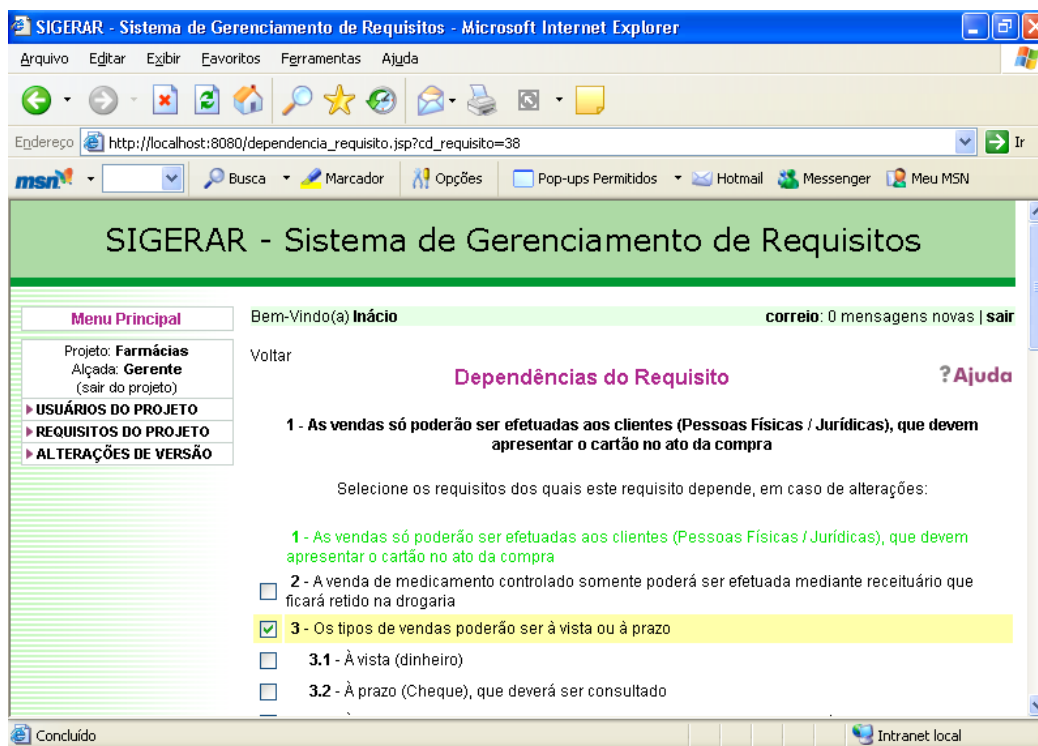


Figura 35 – Dependências do Requisito

Alteração nos requisitos iniciais - Evolução dos Requisitos

Após inclusão dos requisitos iniciais na ferramenta foram efetuados os tratamentos dos requisitos novos e/ou alterados conforme descrito na Tabela 11. A Figura 36 abaixo mostra o resultado da busca dos requisitos a serem alterados.

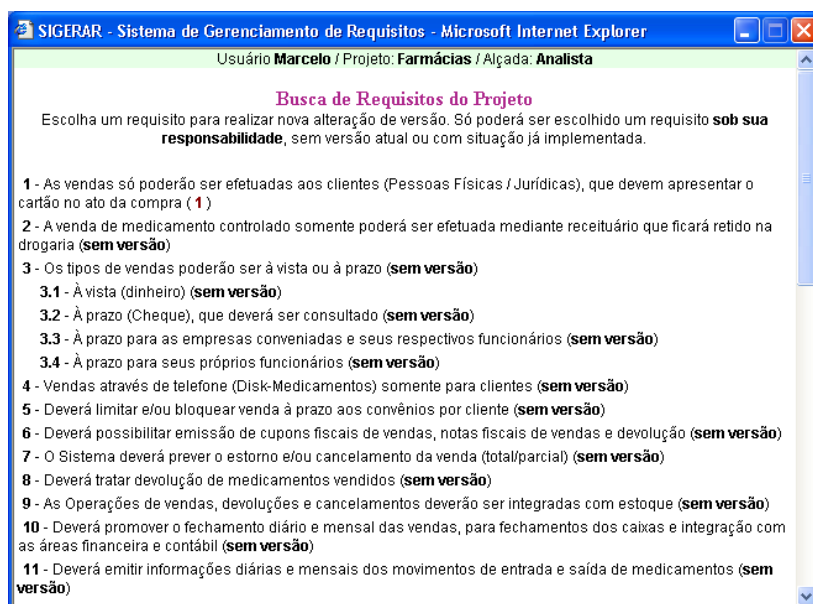


Figura 36 – Busca de Requisitos do Projeto

Ao clicar no requisito que será alterado a ferramenta abre nova tela onde será executada a função de alteração conforme mostra a Figura 37. Nesta tela são preenchidos os dados: motivo, risco, importância, impacto, custo (em horas/homem), prioridade e descrição. A primeira versão do requisito normalmente é a versão originalmente implementada e como não há todos os dados históricos para preenchimento de dados reais, optou-se por incluir a “versão 1” com dados semelhantes em todos os requisitos, para início do processo de evolução dos mesmos. Notar que na parte inferior da tela são registrados os documentos criados na geração da versão do requisito (nome, tipo e localização).

	Versão 2	Versão 1
Solicitante:	Marcelo	Marcelo
Situação:	Proposta	Implementada
Solicitado em:	01/07/2002	01/01/2002
Responsável:	Marcelo	Marcelo
Motivo:	Evolução	Inclusão de Requisito
Risco:	Baixo	Baixo
Importância:	Alta	Baixa
Impacto:	Baixo	Baixo
Custo:	8,00 horas (Total: 8,00 horas)	1.0 horas
Prioridade:	0	0
Descrição:	Criar e tratar no Cadastro de Cliente a situação de "Cliente Inadimplente"	Inclusão

Arquivos:		
Nome	Tipo	Localização
R1v2	Word	c:/projetos/farmácia

Figura 37 – Alteração de Requisitos

Após a inclusão da proposta de alteração do requisito na ferramenta, esta será analisada pelo Gerente do Projeto que verificará a viabilidade inicial, recusando a alteração proposta ou dando prosseguimento ao processo colocando-a em “análise”, conforme Figura 38 abaixo. Neste caso foi automaticamente aberto o fórum de discussões para que os envolvidos se manifestassem.

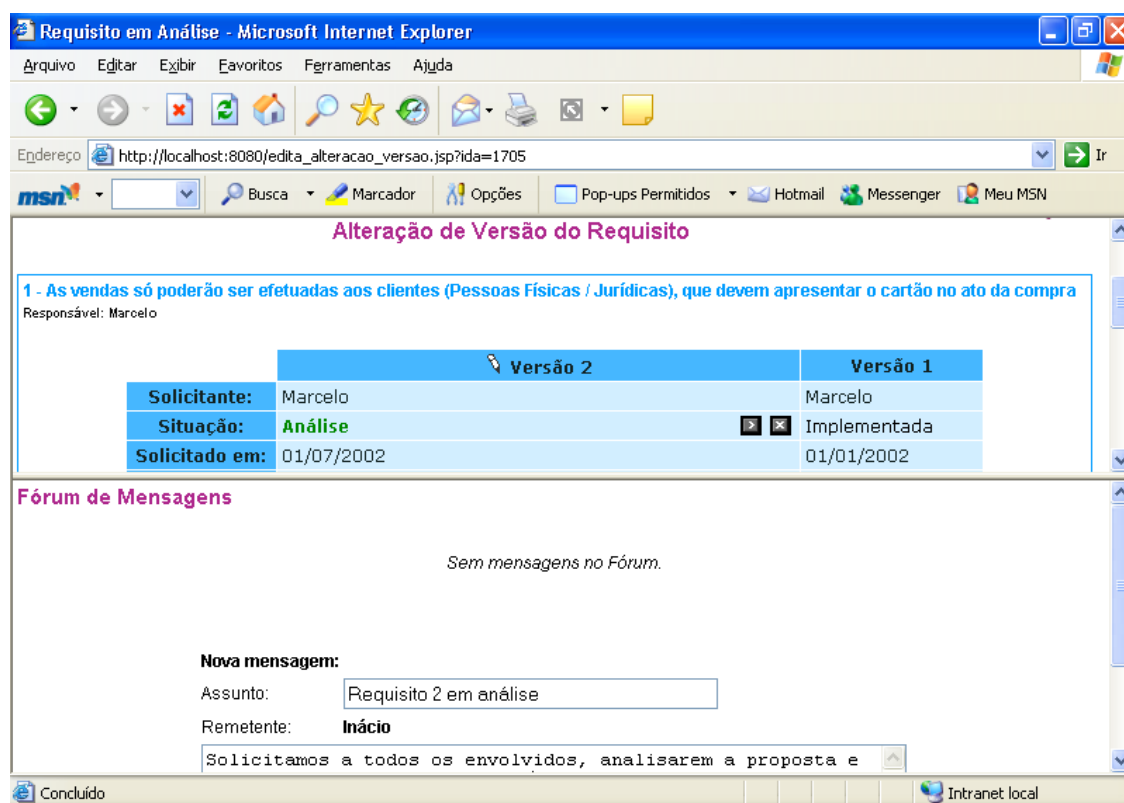


Figura 38 – Alteração de Requisitos em Análise

Após análise do requisito original e dos requisitos dependentes, o Gerente do Projeto tem a visão de todos os riscos, impactos e custos que estes trarão para a organização. Vale destacar a recursividade da ferramenta que trata o requisito original e seus requisitos dependentes, permitindo a análise e atribuição de risco, impacto, prioridade e principalmente o custo em horas/homem a cada um destes requisitos, acumulando o total dos custos no requisito original, fornecendo assim, subsídios ao Gerente do Projeto pela continuidade ou não da implementação da alteração solicitada. Se o Gerente do Projeto decidir pela aprovação, a situação de cada requisito tratado será alterada para “Aprovada”. A partir deste momento, o controle fica integralmente com o analista responsável, que irá promover a situação da alteração para “em desenvolvimento” quando este tiver início e finalmente para “implementada” após testes e implantação.

A Figura 39 mostra parte da tela de alteração de versão de requisitos onde há tratamento da recursividade. Notar que em cada nível que o requisito desce, a cor da tela vai alternando de cor, para melhor visualização dos envolvidos. No exemplo abaixo, o analista responsável pelo requisito dependente (em cor verde),

sendo incluídos e testados na ferramenta, de forma que ao final desta atividade, concluímos o que segue:

- Módulo Administração – todos os cadastros básicos necessários ao estudo de caso foram alimentados na ferramenta e nenhum problema foi constatado. Embora a ferramenta não apresente no momento relatórios específicos dos cadastros, esta necessidade não foi sentida neste módulo, pois a impressão existente no próprio navegador *Web* utilizado foi suficiente, visto que nenhum cadastro ocupa mais de uma página. Ainda assim, acreditamos ser necessário o desenvolvimento de relatórios dos cadastros básicos.
- Módulo Principal
 - A alocação dos usuários no projeto com alçadas específicas foram efetuadas com sucesso;
 - Todas as inclusões dos requisitos iniciais (constantes na Tabela 8 e 9) foram efetuadas com sucesso e não houve nenhum impedimento ou problema constatado;
 - As dependências iniciais listadas na Tabela 10 foram incluídas com sucesso;
 - Nas gerações de versões:
 - Após a inclusão do requisito (com versão 0) o sistema exige a criação da versão 1 que é o registro dos dados da versão de desenvolvimento inicial do requisito. Como não havia todos os dados históricos necessários, o sistema foi alimentado com alguns dados fictícios e semelhantes para podermos iniciar o processo de evolução dos requisitos (conforme item c da subseção 5.3.3.). Por conta disto, análises futuras poderão ser prejudicadas, como por exemplo, um relatório da quantidade de horas consumidas no desenvolvimento e manutenção de requisitos;

- Como na alteração dos requisitos foram tratados dados históricos (de 2002 a 2005), foi necessário a edição do campo data da solicitação (era assumida automaticamente a data do sistema) para aceitar a digitação de datas anteriores. A ferramenta foi alterada e o problema foi solucionado;
- Foram encontrados problemas no tratamento da alteração Nº 38 da tabela 11 - “Ajustes nas telas do sistema para adequação ao Windows XP” que envolve praticamente todos os requisitos e consiste em rever todas as telas do sistema. É necessário rever o conceito de alteração que envolva um grande número de requisitos, pois esta situação não foi contemplada na elicitação da ferramenta. Como em nosso estudo de caso o escopo do sistema é bem definido, não houve muitos problemas em analisar e criar uma versão para cada requisito afetado;
- Outro ponto positivo na aplicabilidade foi o teste da rotina de atualização da volatilidade do requisito, que se mostrou útil para refinamento e atualização do parâmetro, mas fica clara a necessidade de melhorar estes controles com a manutenção de dados históricos para melhor compreender os motivos que levam a aumentar ou diminuir a volatilidade do requisito. Há necessidade também da criação de uma tela amigável no Módulo Administração para executar esta tarefa, que atualmente é executada em *background*;
- Neste módulo sentiu-se ainda a necessidade de relatórios específicos que deverão ser implementados em próximas versões ou trabalhos futuros relacionados a este, como por exemplo, relatório de quantidade de horas consumidas no desenvolvimento e manutenção de requisitos, relatório de requisitos por situação (proposta / em análise / aprovada / em desenvolvimento / implementada), relatórios das dependências (rastreadabilidade), etc.

Posto isto, pode-se afirmar que a ferramenta atende aos requisitos inicialmente traçados, embora careça de pequenas correções e melhorias, ressaltando que esta possui grande potencial para evolução na comunidade de Engenharia de *Software*, buscando atender as organizações que desejam maior controle sobre a questão do Gerenciamento de Requisitos.

6. CONCLUSÕES

Este trabalho contribui no sentido de propor uma ferramenta de uso livre, para coletar, armazenar e manter os requisitos acordados entre os *stakeholders*, durante todo o ciclo de vida do *software*, de forma que equipes de gerentes, analistas e usuários de sistemas das organizações tenham controle sobre duas importantes questões do gerenciamento de requisitos: controle das versões de requisitos e rastreabilidade dos requisitos.

A ferramenta apresenta muitos benefícios às organizações que venham a adotá-la, entre outros, a ausência de custos de aquisição da ferramenta e banco de dados, pois foi desenvolvida em plataforma portátil e opera via *Web* com o SGBD *Firebird* de livre distribuição.

A ferramenta conta com muitos dos recursos disponíveis nas ferramentas comerciais, como controle de acesso e permissões, controle de versões (históricos), glossários, notificações e fórum de discussões, entre outros e apresenta alguns pontos diferenciais. Um ponto diferencial, é que embora a ferramenta contenha os textos de “ajuda” das telas previamente formatados, esta permite que cada organização possa customizá-los, de acordo com sua própria cultura ou características, de forma a obter maior adequação. Outro importante diferencial da ferramenta é o tratamento da rastreabilidade dos requisitos no processo de alteração, onde a ferramenta não somente demonstra os requisitos dependentes da modificação do requisito origem, mas exige que os responsáveis pelos requisitos (origem e dependentes) analisem e atribuam valores de risco, importância, impacto, prioridade e custo de cada um dos requisitos envolvidos. É importante ressaltar que um requisito dependente também tem sua própria matriz de dependência, assim sendo, temos que dar a ele o tratamento como se requisito origem fosse (efeito recursivo), tratando todos os seus dependentes até que o ciclo se feche. Desta forma, a ferramenta garante que todos os requisitos envolvidos sejam rastreados, analisados e tratados, produzindo informações de vital importância ao Gerente do Projeto que poderá analisar todo o contexto do

impacto e custos da alteração e com estes subsídios, tomará a decisão de aprovar ou rejeitar a proposta de alteração do requisito.

Trabalhos Futuros

A consolidação da ferramenta através da aplicação no estudo de caso indica que a mesma necessita de evolução e melhorias. Em trabalhos futuros, a ferramenta poderá ser aprimorada com inclusões de novas funcionalidades, como por exemplo, interoperabilidade com ferramentas eletrônicas utilizadas para documentação, como Word, Excel, PowerPoint, Project, UML entre outras. Também poderão ser desenvolvidos relatórios e consultas com informações dos históricos das alterações dos requisitos, de custos das alterações dos requisitos, de evolução da volatilidade dos requisitos, entre outros.

Um dos grandes problemas a ser resolvido por trabalhos futuros é o tratamento dos atributos prioridade, risco, importância e impacto dos requisitos origem e dependentes, pois cabe ao Gerente do Projeto avaliar estes dados e projetar mentalmente ou com a ajuda de outras ferramentas, os riscos e impactos potenciais da alteração. Como cada dado tem um “peso” associado, a idéia é que a própria ferramenta possa obter o somatório dos “pesos” (de forma análoga ao praticado com o atributo custo), e compará-lo a valores previamente tabulados pelo administrador, que forneçam informações sobre o risco e impacto que a mudança solicitada trará ao projeto, de forma a auxiliar Gerentes de Projetos na tomada de decisão sobre o aceite ou rejeição da alteração proposta.

REFERÊNCIAS BIBLIOGRÁFICAS

- [BOO94] BOOCH, G., "Object-Oriented Analysis and Design with Application", Benjamin/Cummings, 1994.
- [BOO99] BOOCH, G., RUMBAUGH, J. e JACOBSON, I, "The Unified Modeling Language User Guide", Addison Wesley, 1999.
- [BRE00] BREITMAN, K. K., "Evolução de Cenários", PUC-Rio, Maio de 2000. Tese de Doutorado apresentada ao Departamento de Informática da PUC/RJ.
- [BRE98] BREITMAN, K. K. e LEITE, J. C. S. P., "Suporte Automatizado à Gerência da Evolução de Cenários", Workshop de Engenharia de Requisitos - WER98, 1998.
- [BRE99] BREITMAN, K. K. e LEITE, J. C. S. P., "Processo de *Software* Baseado em Cenários", II (Ibero-American) Workshop on Requirements Engineering – Buenos Aires, September, 1999, pp. 95-105.
- [CAS00] CASTRO, J., ALENCAR, F. e CYSNEIROS, G., "Closing the GAP Between Organizational Requirements and Object Oriented Modeling", Journal of the Brazilian Computer Society, Nº 1, Vol. 7, July 2000, pp. 5-16.
- [CHR92] CHRISTEL, G e KANG, K. C., "Issues in Requirements Elicitation", Technical Report CMU/SEI-92-TR-012 ESC-TR-92-012, Software Engineering Institute Carnegie Mellon University Pittsburgh, Pennsylvania, 1992.
- [CHU99] CHUNG, L.; NIXON, B.; YU, E. e MYLOPOULOS, J., "Non-Functional Requirements to Systematically Support Change", 2nd IEEE Symposium on Requirements Engineering, March 1995, York, England.

- [CYS01] CYSNEIROS, L. M., “Requisitos Não-Funcionais: Da Elicitação ao Modelo Conceitual”, PUC-Rio, Fevereiro de 2001. Tese de Doutorado apresentada ao Departamento de Informática da PUC/RJ.
- [DAV90] DAVIS, A. M., “The Analysis and Specification of System and Software Requirements”, Systems and Software Requirements Engineering, IEEE Computer Society Press, 1990, pp. 119-144.
- [DAW99] DAWSON, L. L. e SWATMAN, P. A., “The Use of Object-Oriented Models in Requirements Engineering: A Field Study”, Proceeding of the 20th international conference on Information Systems, ACM, 1999, pp. 260-273.
- [EAS96] EASTERBROOK, S. e NUSEIBEH, B., “Using Viewpoints for Inconsistency Management”, BCS/IEE Software Engineering Journal, pp. 31-43.
- [ENG87] ENGESTROM, Y., "Learning by Expanding", Helsinki: Orienta-Konsultit, 1987.
- [ESP04] ESPÍNDOLA, R. S., MAJDENBAUN e A., AUDY, J.L.N., “Uma Análise Crítica dos Desafios para Engenharia de Requisitos em Manutenção de Software”, Anais do WER04 - Workshop em Engenharia de Requisitos, Tandil, Argentina, Dezembro 9-10, 2004, pp 226-238.
- [FIN92] FINKELSTEIN, A., KRAMER, J., NUSEIBEH, B. e GOEDICKE, M., “Viewpoints: A Framework for Integrating Multiple Perspectives in Systems Development”, International Journal of Software Engineering and Knowledge Engineering, 1992. vol 2, pp. 31-58.
- [FRA04] FRANCETO, S.; MARTINS, L.E.G., “Elicitação de Requisitos para o Desenvolvimento de uma Ferramenta de Apoio à Metodologia de Elicitação de Requisitos de *Software* Baseada na Teoria da Atividade (META1)” – 7º Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes Software. Arequipa, Peru, 3 a 7 de Mayo, 2004. Pages: 169-174. ISBN: 9972-9876-1-2.

- [FRA05] FRANCETO, S “Especificação e Implementação De Uma Ferramenta Para Elicitação de Requisitos De Software Baseada Na Teoria da Atividade”, Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, da Faculdade de Ciências Exatas e da Natureza, da Universidade Metodista de Piracicaba – UNIMEP- Dez 2005.
- [GOG97] GOGUEN, J. A. e LINDE, C., “Techniques for Requirements Elicitation”, Software Requirements Engineering, 2nd . Ed., IEEE CS Press, 1997, pp 110-122.
- [GOT94] GOTEL, O. e FINKELSTEIN, A., “An analysis of the Requirements Traceability Problem,” in Proceedings of the First International Conference on Requirements Engineering, (Colorado springs, CO), pp. 94-101, April 1994.
- [HUG95] HUGHES, J., O'BRIEN, J., RODDEN, T., ROUNCEFIELD, M. e SOMMERVILLE, I. “Presenting Ethnography in the Requirements Process” 2nd IEEE Symposium on Requirements Engineering, march 1995, York, England.
- [JAC92] JACOBSON, I., CHRISTERSON, M., JONSSON, P., e GUNNAR, O., “Object-Oriented Software Engineering: A Use Case Driven Approach”, Addison-Wesley. 1992.
- [JAR98] JARKE, M., “Requirements Tracing”, Communications of the ACM, v. 41, n. 12, dec. 1998.
- [KAP97] KAPTELIN, V. e NARDI, B. A., “Activity Theory: Basic Concepts and Applications”, CHI 97 Electronics Publications: Tutorials, march 1997, <http://turing.acm.org/sigs/sigchi/chi97/proceedings/tutorial/bn.htm>.
- [KOL02] KOLP, M., GIORGINI, P. e MYLOPOULOS, J., “Information Systems Development through Social Structures”, Proceedings of the 14th International Conference on Software Engineering & Knowledge Engineering (SEKE'02), July 15-19, Ischia, Italy.

- [KOT96] KOTONYA G. e SOMMERVILLE, I., "Requirements Engineering: With Viewpoints", BCS/IEEE Software Engineering Journal., pp. 5-18.
- [KOT98] KOTONYA, G.e SOMMERVILLE, I., "Requirements Engineering: Processes and Techniques", John Wiley and Sons, 1998.
- [KUU96] KUUTI, K., "Activity Theory as a Potential Framework for Human-Computer Interaction", Context and Consciousness - Activity Theory and Human-Computer Interaction, MIT Press, 1996, pp. 17-44.
- [LAMS00] LAMSWEERDE, A., "Requirements Engineering in theYear 00: A Research Perspective", International Conference on Software Engineering, 22., jun. 2000, Limerick, Ireland, Proceedings ACM, 2000, pp. 5-19.
- [LAM99] LAM, W., LOOMES, M. e SHANKARARAMAN, V., "Managing Requirements Change Using Metrics and Action Planning", Third European on Software Maintenance, mar. 1999, Amsterdam, Netherlands.
- [LEI94] LEITE J.C.S.P. e OLIVEIRA, A.P.A., "A Client Oriented Requirements Baseline", Proceedings of the Second IEEE International Symposium on Requirements Engineering, York, UK, IEEE Computer Society Press, 1995, pp. 108-115.
- [LOU95] LOUCOPOULOS, P. e KARAKOSTAS, V., "Systems Requirements Engineering", London: McGraw-Hill, 1995.
- [MAR01] MARTINS, L. E. G., "Uma Metodologia de Elicitação de Requisitos de Software Baseada na Teoria da Atividade", Agosto 2001, Tese de Doutorado apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas.
- [MAR05] MARTINS, L. E. G., "An Empirical Study Using Z and UML for the Requirements Specification of an Information System" In: EMPIRICAL

SOFTWARE ENGINEERING LATIN AMERICAN WORKSHOP (ESELAW), 2005, Uberlândia. Proceedings of the 2nd Empirical Software Engineering Latin American Workshop. 2005.

[MAR99] MARTINS, L. E. G. e DALTRINI, B. M., "An Approach to Software Requirements Elicitation Using Precepts from Activity Theory", Proceedings of the 14th IEEE International Conference on Automated Software Engineering, 1999.

[McD89] McDERMID, J. A., "Requirements Analysis: Problems and the STARTS Approach.", IEE Colloquium on 'Requirements Capture and Specification for Critical Systems' (Digest No. 138), 4/1-4/4. Institution of Electrical Engineers, November, 1989.

[MUL79] MULLERY, G. P., "CORE: A Method for Controlled Requirements Expression", Proceedings of the Fourth International Conference on Software Engineering., IEEE Press, 1979.

[MYL92] MYLOPOULOS, J., CHUNG, L. e NIXON, B., "Representing and Using Nonfunctional Requirements: A Process-Oriented Approach", IEEE Trans. on Software. Engineering, Vol. 18 No. 6, June 1992, pp. 483-497.

[MYL99] J. MYLOPOULOS, CHUNG, L. e YU, E., "From Object-Oriented to Goal-Oriented Requirements Analysis", Communications of the ACM, Vol. 42 No. 1, January 1999, pp.31-37.

[NAR96] NARDI, B. A., "Studing Context", Context and Consciousness – Activity Theory and Human-Computer Interaction, MIT Press, 1996, pp. 69-102.

[NUS00] NUSEIBEH, B. e EASTERBROOK, S., "Requirements Engineering: a Roadmap", International Conference on Software Engineering – Proceedings of the conference on the future of Software Engineering 2000, Limerick, Ireland - ACM Computing Surveys. Limerick, Ireland, Jun. 2000.

- [PIN00] PINHEIRO, F. A. C., "Formal and Informal Aspects of Requirements Tracing", III Workshop de Engenharia de Requisitos (WER 2000), 2000, Rio de Janeiro, Brasil.
- [POH93] POHL, K., "The Three Dimensions of Requirements Engineering.", Proceedings of Fifth International Conference on Advanced Informations Systems Engineering (CaiSE'93), Paris, 1993, pp.12-16.
- [POH96] POHL, K., "PRO-ART: Enabling Requirements Pre-Traceability", Proceedings of the Second International Conference on Requirements Engineering (ICRE), IEEE Computer Society Press – Colorado Springs, April, 1996, pp.76-85.
- [POT94] POTTS, C., TAKAHASHI, K. e ANTON, A.I., "Inquiry-Based Requirements Analysis.", IEEE Software, vol. 11, no. 2, February, 1994.
- [PRE95] PRESSMAN, R. S., "Engenharia de Software", Makron Books, 1995.
- [RAM95] RAMESH, B., POWERS, T. e STUBBS, C., "Implementing Requirements Traceability: A Case Study", 2nd IEEE Symposium on Requirements Engineering, March 1995, York, England.
- [RAM98] RAMESH, B., "Factors Influencing Requirements Traceability Practice", Communications of de ACM, December, 1998, vol 41 N° 12.
- [RID00] RIDAO, M., DOORN, J. e LEITE, J.C.S.P., "Uso de Patrones en la Construcción de Escenarios" III Workshop de Engenharia de Requisitos (WER2000), Rio de Janeiro, Brasil.
- [ROL98] ROLLAND, C., SOUVEYET, C. e ACHOUR, C.B., "Guiding Goal Modeling Using Scenarios.", IEEE Transactions on Software Engineering, vol. 24, no. 12, December, 1998.

- [ROS77] ROSS, D., "Structured Analysis (AS): A Language for Communicating Ideas", IEEE Transactions on Software Engineering. 3,1. pp. 16-34. Jan. 1977.
- [RUM91] RUMBAUGH, J. et al., "Object-Oriented Modeling and Design", Prentice Hall, 1991.
- [RUM94] RUMBAUGH, J., "Getting Started: Using Use Cases to Capture Requirements", J. Object-Oriented Programming, Sept. 1994, pp. 8-12.
- [SOM03] SOMMERVILLE, I., "Engenharia de Software", 6ª edição. Pearson Education do Brasil, 2003.
- [STD830] IEEE Std. 830-98, "Recommended Practice for Software Requirements Specification", The Institute of Electrical and Electronics Engineers, New York 1990.
- [SUT98] SUTCLIFFE, A.G., MAIDEN, N. A. M., MINOCHA, S. e MANUEL, D., "Supporting Scenario-Based Requirements Engineering", IEEE Transactions on Software Engineering, vol. 24, no. 12, December, 1998.
- [THA97] THAYER, R. H. e DORFMAN, M., "Introduction to Tutorial Software Requirements Engineering", Software Requirements Engineering, 2nd Ed., IEEE CS Press, 1997, pp. 1-2.
- [TOR99] TORANZO, M. e CASTRO, J., "A Comprehensive Traceability Model to Support the Design of Interactive Systems", International Workshop on Interactive System Development and Object Models - WISDOM 99. Lisboa, Portugal. Jun. 1999.
- [ZAV97] ZAVE P., "Classification of Research Efforts in Requirements Engineering" ACM Computing Surveys, 1997, pp. 315-321.
- [ZOR95] ZORMAN, L., "Requirements Envisaging by Utilizing Scenarios (Rebus)", PhD. Dissertation, University of Southern California (1995).

Apêndice I – Relação de Sites de Ferramentas de Gerenciamento de Requisitos

Ferramenta	URL
Active!Focus	www.xapware.com
AnalystPro	www.analysttool.com/
Caliber-RM	www.borland.com/caliber/index.html
Catalyze	www.chiaracorp.com/requirements.htm
CORE	www.vtcorp.com
Cradle	www.threesl.com
DOORS	www2.telelogic.com/doors
IRqA	www.irqaonline.com
Mac A&D and Win A&D	www.excelsoftware.com
Projectricity	www.projectricity.com/specification_tool.htm
Prosareq Requirements Manager	www.prosa.fi/eng/pr2Req.htm
Qualica QFD	qualica.de/english/requirements_mgmt.htm
RDD-100	www.holagent.com
RDT	igatech.com/rdt/rdtwalkthrough.html
Reconcile	compuware.com/products/reconcile.html
Requirement Tracing System	www.bandwood.com/cms_exec_summary.htm
Requisite Pro	www.rational.com/products/reqpro/index.jsp
RMTrak	www.rmtrak.com
RTM Workshop	www.chipware.com
SpeedReq	www.speedev.com
Team-Trace	www.team-trace.com