**UNIVERSIDADE REGIONAL DE BLUMENAU**

**CENTRO DE CIÊNCIAS EXATAS E NATURAIS**

**CURSO DE CIÊNCIAS DA COMPUTAÇÃO – BACHARELADO**

# FERRAMENTA DE GERÊNCIA DE REQUISITOS DE SOFTWARE INTEGRADA COM ENTERPRISE ARCHITECT

**RAPHAEL MARCOS BATISTA**

**BLUMENAU**
**2007**

**2007/1-39**

**RAPHAEL MARCOS BATISTA**

# FERRAMENTA DE GERÊNCIA DE REQUISITOS DE SOFTWARE INTEGRA COM ENTERPRISE ARCHITECT

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciências da Computação — Bacharelado.

Profa. Fabiane Barreto Vavassori Benitti, Dr[a] Eng. - Orientadora

**BLUMENAU**
**2007**

**2007/1-39**

# FERRAMENTA DE GERÊNCIA DE REQUISITOS DE SOFTWARE INTEGRADA COM ENTERPRISE ARCHITECT

Por

**RAPHAEL MARCOS BATISTA**

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

| | |
|---|---|
| Presidente: FURB | Profa. Fabiane Barreto Vavassori Benitti, Dr[a] Eng. – Orientadora, |
| Membro: | Prof. Everaldo Artur Grahl, Ms. Eng. – FURB |
| Membro: | Prof. Mauro Marcelo Mattos, Dr. Eng. – FURB |

Blumenau, 9 de Julho de 2007

A minha família, que considero o alicerce da minha vida.

## AGRADECIMENTOS

A Deus, pela minha vida.

A minha família, pelos incentivos.

A minha orientadora, Prof.ª Dr.ª Fabiane Barreto Vavassori Benitti, pelos ensinamentos e conhecimentos transmitidos.

Aos meus amigos, pela força.

# RESUMO

Este trabalho apresenta uma ferramenta para gerência de requisitos de software integrada com o software Enterprise Architect. A ferramenta gera um documento de Especificação de Requisitos de Software no padrão IEEE-830-1998. A integração com o software Enterprise Architect vai ao encontro do conceito de ferramenta *Integrated CASE* (I-CASE). Outra finalidade desta ferramenta é oferecer o rastreamento dos requisitos e casos de usos através de uma matriz de rastreabilidade, sinalizando as alterações realizadas nos elementos relacionados da matriz, para poder visualizar o impacto da alteração nos demais requisitos e casos de uso do projeto de software. A ferramenta proporciona uma interface específica para o gerenciamento dos requisitos de software e composição do documento de especificação de requisitos de software, buscando a excelência no levantamento dos requisitos, sendo esta etapa primordial para o sucesso do desenvolvimento e concepção de um projeto de software.

Palavras-chave: Gerência de requisitos. I-CASE. Enterprise Architect. IEEE-830-1998.

# ABSTRACT

This work presents a tool for managing the requirements of software integrated with the Enterprise Architect software. The tool generates a document for Software Requirements Specification in the IEEE-830-1998 standard. Integration with Enterprise Architect software follows the concept of the Integrated CASE (I-CASE) tool. Another end purpose of this tool is to offer tracking of requirements and use cases by means of a tracking matrix, indicating the alterations made in the elements related to the matrix, in order to he able to visualize the impact of alteration on the other requirements and use cases of the software project. The tool enables a specific interface for the management of software requirements and the composition of a document with software requirements specifications, seeking excellence in the gathering of requirements, this being the primordial stage in the successful conception and development of a software project.



Key-words: Requirement management. I-CASE. Enterprise Architect. IEEE-830-1998.

# LISTA DE ILUSTRAÇÕES

# LISTA DE SIGLAS

CASE – *Computer-Aided Software Engineering*

COM – *Component Object Model*

CVS – *Concurrent Version System*

EA – Enterprise Architect

ERS – Especificação de Requisitos de Software

DTD – *Document Type Definition*

I-CASE – *Integrated CASE*

JPEG – *Joint Photographic Experts Group*

IEEE - *Institute of Electrical and Electronics Engineers*

PDF - *Portable Document Format*

PHP - Hypertext Preprocessor

RTF - *Rich Text Format*

RUP - *Rational Unified Process*

SRS - *Software Requirements Specification*

UML - *Unified Modeling Language*

XML - *eXtensible Markup Language*

XSD - *Xml Schema Defination*

# LISTA DE SÍMBOLOS

$ - cifrão

* - estrela

# SUMÁRIO

# 1 INTRODUÇÃO

Obter o êxito dos produtos desenvolvidos é uma busca constante da área de engenharia de software e ferramentas que automatizem os processos de produção. Entre as atividades previstas estão as relacionadas à área de requisitos.

> O termo *requisito* não é utilizado pela indústria de software de modo consistente. Em alguns casos, um requisito é visto como uma declaração abstrata, de alto nível de uma função que o sistema deve fornecer ou de uma restrição do sistema. No outro extremo, ele é uma definição detalhada, matematicamente formal, de uma função do sistema. (SOMMERVILLE, 2003, p. 82).

Segundo Castro, Gimenes e Maldonado (2000, p. 252), engenharia de requisitos integra o ciclo de vida de desenvolvimento de software, responsável pelo processo de levantamento e compreensão dos requisitos. Este processo compreende atividades como:

a) elicitação de requisitos: trata do levantamento dos requisitos;
b) análise de requisitos: rotinas para perceber e solucionar conflitos entre requisitos;
c) validação de requisitos: verifica duplicidade, inconsistências e inconcluso;
d) gerenciamento de requisitos: gerencia as alterações dos requisitos.

Para a realização destas atividades são utilizados métodos, técnicas e ferramentas, como por exemplo, a ferramenta Enterprise Architect (EA), que é uma ferramenta *Computer-Aided Software Engineering* (CASE) baseada na *Unified Modeling Language* (UML). Apesar do EA possuir algumas funcionalidades que auxiliam na especificação de requisitos, esta ferramenta não tem nesta etapa do processo de desenvolvimento seu foco de atuação, possuindo pouca flexibilidade e apresentando problema de usabilidade (conforme é apresentado na seção 2.4).

Ciente das limitações do EA, foi desenvolvida pela SparxSystems Japan uma ferramenta de gerenciamento de requisitos integrada com o EA chamada RaQuest. No entanto, esta ferramenta é proprietária, possuindo custo elevado[1] para fins acadêmicos. Além disso, tanto o EA quanto a ferramenta RaQuest, não viabilizam a produção de um documento de especificação de requisitos completo observando um modelo, como por exemplo o modelo IEEE-830-1998. Segundo Institute of Electrical and Electronics Engineers (1998, p. 1), o modelo IEEE-830-1998 descreve as características necessárias para elaborar um documento de Especificação de Requisitos de Software (ERS).

[1] Atualmente o preço do RaQuest está estimado entre US $150,00 e US $220,00, dependendo da versão (RAQUEST, 2007).

Esta ferramenta propõe suprir a deficiência do EA, permitindo que se tenha o gerenciamento mais eficiente dos requisitos, focando na produção de um documento de especificação de requisitos completo, primando pela usabilidade e sem custos para os usuários. A ferramenta possui total integração com o EA, possibilitando incorporar os requisitos da ferramenta para o EA, permitindo a convergência com os recursos que o EA disponibiliza, indo de encontro ao conceito de *Integrated CASE* (I-CASE).

Segundo Pressman (2002, p. 815), ferramentas CASE integradas ou simplesmente denominadas I-CASE, são aquelas que têm em sua característica a distribuição da informação da engenharia de software entre as ferramentas, isto significa que a informação deve provir de uma mesma base de dados.

## 1.1 OBJETIVOS DO TRABALHO

O objetivo deste trabalho é o desenvolvimento de uma ferramenta de gerência de requisitos de software integrada ao EA.

Os objetivos específicos do trabalho são:

a) disponibilizar uma extensão da ferramenta CASE EA para especificação e gerenciamento de requisitos;
b) disponibilizar através da ferramenta uma documentação de requisitos baseada no modelo IEEE-830-1998.

## 1.2 ESTRUTURA DO TRABALHO

A parte inicial deste capítulo apresenta uma introdução sobre este trabalho, são citados os objetivos e a estrutura de trabalho, sendo esta última apresentada nesta seção. Os capítulos a seguir descrevem as etapas para a concepção deste trabalho.

O segundo capítulo apresenta a fundamentação teórica essencial para o embasamento deste trabalho onde são abordados os assuntos sobre engenharia de requisitos, modelo de documento de ERS baseado no modelo IEEE-830-1998, ambientes I-CASE, ferramentas similares para gerenciamento de requisitos e o desenvolvimento integrado com a ferramenta

EA.

O terceiro capítulo descreve as etapas do desenvolvimento deste trabalho, apresentando os requisitos funcionais e não funcionais, a especificação, casos de usos, o diagrama de classes e algumas tecnologias e ferramentas utilizadas para a definição da estrutura de dados em *Extensible Markup Language* (XML). Na seção sobre a implementação são apresentadas as principais técnicas utilizadas, os resultados e discussões sobre a ferramenta.

No quarto capítulo é apresentada a conclusão sobre este trabalho e as extensões sugeridas para trabalhos futuros.

# 2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são abordados os assuntos relevantes deste trabalho que estão divididos em cinco seções. A primeira seção aborda sobre engenharia de requisitos, dando uma visão geral sobre o tema. A segunda seção apresenta o modelo IEEE-830-1998. A terceira seção explana sobre os ambientes CASE integrados. A quarta aborda sobre a ferramenta CASE EA, apresentando informações sobre os mecanismos de integração.

## 2.1 ENGENHARIA DE REQUISITOS

Engenharia de Requisitos refere-se a uma área de pesquisa, inserida no contexto da Engenharia de Software, e relacionada com a elicitação, documentação e validação das funcionalidades e limitações que precisam ser respeitadas por um software em sua construção e operação. Para Kotonya e Sommerville (1998, p. 8) engenharia de requisitos é “a forma como escolhemos denominar as atividades desenvolvidas, no contexto do ciclo de vida de software, relacionadas com a definição dos requisitos de um sistema”.

Para Pressman (2002, p. 250), a engenharia de requisitos disponibiliza processos para compreender e analisar as necessidades do cliente, avaliando a possibilidade de execução, realizando a especificação e validação dos requisitos até a transformação em um sistema de software operacional.

### 2.1.1 Processos da Engenharia de Requisitos

Para Pfleeger (2004, p. 112), o processo da engenharia de requisitos é fundamental para o bom desenvolvimento de software. O principal objetivo do Processo da Engenharia de Requisitos é concluir com êxito um acordo entre quem solicita e quem desenvolve, estabelecendo clara e rigorosamente o que deverá ser produzido (FIORINI; STAA; BAPTISTA, 1998, p. 68). A Figura 1 ilustra as atividades que compõem os processos da engenharia de requisitos.

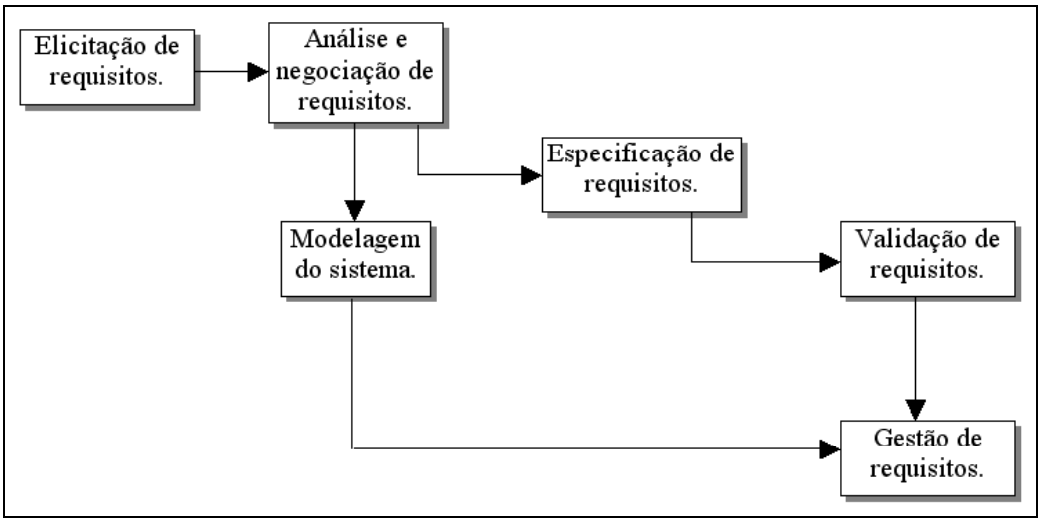



Figura 1 – Processos da engenharia de requisitos

Segundo Pressman (2002, p. 250), o processo da engenharia de requisitos é definido pelas seguintes atividades:

a) elicitação de requisitos: também denominada como levantamento dos requisitos, esta atividade define o escopo e restrições da aplicação. O analista deve compreender o domínio da aplicação. Os clientes/usuários que participam desta atividade acabam omitindo informações que acreditam ser óbvias ou esclarecem detalhes desnecessários que acabam confundindo e não esclarecendo o verdadeiro objetivo da aplicação. Um exemplo de problema que torna a elicitação de requisitos difícil é a mudança que os requisitos sofrem ao longo do tempo;
b) análise e negociação de requisitos: propõem rotinas para perceber e solucionar conflitos entre requisitos uma vez que os requisitos levantados na atividade anterior formam um base para análise de requisitos. A análise classifica os requisitos, relaciona, consiste e verifica se algum requisito foi omitido ou duplicado, sempre observando as necessidades dos clientes/usuários;
c) especificação de requisitos: é o trabalho final desenvolvido pelo engenheiro de requisitos e servirá como fundamento para os engenheiros de hardware, software, base de dados e humana. Delimita os elementos dos sistema alocado e também descreve os dados e controle de entrada e saída da aplicação;
d) modelagem do sistema: avalia os componentes do sistema em relação uns aos outros, para determinar como os requisitos se encaixam nesse quadro e para avaliar a estética do sistema;
e) validação de requisitos: verifica duplicidade, inconsistências e inconcluso dos requisitos especificados. Uma equipe de validação é responsável por esta atividade

que inclui engenheiros de sistemas, clientes e usuários;

f) gestão de requisitos: conjunto de atividades que ajuda a equipe de projeto a identificar, controlar e rastrear requisitos e modificações de requisitos durante o desenvolvimento do sistema de software.

### 2.1.2 Requisitos

Para Peters e Pedrycz (2001, p. 102), requisito de software é uma descrição dos principais recursos de um produto de software, seu fluxo de informações, comportamento e atributos. Já para Sommerville (2003, p. 82), requisito de software é uma declaração abstrata, de alto nível de uma função que o sistema deve fornecer ou de uma restrição do sistema.

Os requisitos de software são necessários para o melhor entendimento de um software a ser construído. Uma definição clássica de requisitos de software expressa-se pelas descrições das funções e das restrições que o sistema deve cumprir e executar (SOMMERVILLE, 2003, p. 119).

Fiorini, Staa e Baptista (1998, p. 65), relatam que determinar requisitos é, em a última análise, entender exatamente o que deve ser feito e o que se espera receber como resultado. Nesta etapa não se argumenta como o trabalho será realizado.

Um requisito de software é uma característica do sistema ou a descrição de algo que o sistema é capaz de realizar, para atingir seus objetivos. A falha no entendimento, documentação e gerenciamento dos requisitos podem acarretar uma série de problemas no desenvolvimento de um sistema de software (PFLEEGER, 2004, p. 111).

Sommerville (2003, p. 82), relata que pode haver alguns problemas na engenharia de requisitos caso não tenha uma separação clara entre os diferentes níveis de descrição dos requisitos. A distinção desses níveis é denominada como requisito de usuário que descreve em linguagem natural as funções e restrições do sistema e requisito de sistema que descreve detalhadamente as funções e restrições do sistema.

Conforme Sommerville (2003, p. 83), os requisitos de sistema geralmente são classificados em requisitos funcionais, não funcionais ou requisitos de domínio.

#### 2.1.2.1 Requisitos funcionais

Para Sommerville (2003, p. 84), requisitos funcionais definem as funcionalidades e serviços que o sistema irá fornecer. Os requisitos funcionais devem ser descritos detalhadamente, todas as funções definidas pelo usuário devem estar descritas e sem conflitos entre si. É comum que em sistemas de grande porte não seja possível descrever os requisitos de forma detalhada, devido a grande complexidade do sistema.

Segundo Fiorini, Staa e Baptista (1998, p. 66), requisitos funcionais especificam as funções que o sistema ou componente do sistema deve ser capaz de realizar, e podem ser definidos como:

a) dinâmico: os resultados do sistema ou componentes são obtidos através da execução sobre circunstâncias determinadas;
b) estático: o desempenho das funções de cada entidade sobre a forma de interação com outras do mesmo ambiente.

Conforme Pfleeger (2004, p. 115), um requisito funcional descreve uma interação entre o sistema e seu ambiente. O requisito funcional também descreve como o sistema deve se comportar.

#### 2.1.2.2 Requisitos não funcionais

Segundo Sommerville (2003, p. 83), requisitos não funcionais definem as restrições sobre as funcionalidades e serviços que o sistema poderá fornecer. Conforme Sommerville (2003, p. 83), enquanto o descumprimento de requisito funcional degrada o sistema, o descumprimento de um requisito não funcional pode tornar o sistema inoperante.

Sommerville (2003, p. 86), destaca três tipos de requisitos não funcionais, os quais são:

a) requisitos de produto: definem o comportamento do produto;
b) requisitos organizacionais: originam de políticas e procedimentos nas organizações do cliente e do desenvolvedor do sistema;
c) requisitos externos: engloba os requisitos de origem externa ao sistema e seu processo de desenvolvimento.

Para Fiorini, Staa e Baptista (1998, p. 66), requisitos não funcionais estão relacionados com:

a) acurácia: exatidão dos dados e resultados, livre de erros;
b) precisão: resultados dentro de margens de tolerância aceitáveis;
c) confiabilidade: resultados acurados, precisos e corretos;
d) segurança: os riscos são compatíveis com a natureza do serviço;
e) desempenho: os recurso são compatíveis com a funcionalidade;
f) rentabilidade: o retorno financeiro está dentro do esperado;
g) manutenibilidade: é de fácil manutenção;
h) disponibilidade: está disponível sempre que solicitado;
i) recuperabilidade: é possível de ser recuperado de caso de falhas e desastres;
j) proteção: é perceptível na ocorrência e interceptação de possíveis problemas;
k) utilizabilidade: está apto a ser utilizado pelos usuários finais.

Pfleeger (2004, p. 115) define um requisito não funcional como uma restrição do sistema que torna limitada a criação de uma solução para o problema. De modo geral, as restrições são limitadas em nível de linguagem, plataforma, técnicas ou ferramentas de implementação.

#### 2.1.2.3 Requisitos de domínio

Conforme Sommerville (2003, p. 88), requisitos de domínio são derivados do domínio da aplicação do sistema, em vez de serem obtidos a partir das necessidades especificas dos usuários do sistema. Os requisitos de domínio são importantes porque, muitas vezes, refletem fundamentos do domínio da aplicação. Se os requisitos não forem satisfeitos, poderá ser impossível fazer o sistema operar satisfatoriamente.

### 2.1.3 Gerenciamento de requisitos

Para manter os requisitos de forma consistente utiliza-se o processo de gerenciamento de requisitos, que segundo Sommerville (2003, p. 119), é um processo de controle de alterações nos requisitos dos sistemas. No decorrer desta fase é necessário decidir sobre alguns aspectos como a identificação de requisitos, processo de gerenciamento de mudanças, políticas de facilidade de rastreamento e suporte de ferramentas CASE. A identificação do requisito estabelece que cada requisito deve ser único para que possa ser facilmente

relacionado com outros requisitos e rastreado. O processo de gerenciamento de mudanças define os impactos e o custo das mudanças. A política de facilidade de rastreamento trata das relações dos requisitos sobre outros requisitos. O suporte de ferramentas CASE abrange sobre a utilização de uma ferramenta que facilite o gerenciamento de um grande número de requisitos.

Segundo Sommerville (2003, p. 117), os requisitos nos grandes sistemas estão em constante modificação. O gerenciamento de requisitos vem ao encontro de controlar as alterações dos requisitos de sistemas. Para Pressman (2002, p. 254), o gerenciamento de requisitos deve controlar as alterações dos requisitos auxiliando a equipe de projeto na identificação, controle e rastreamento de requisitos modificados. E essas alterações são melhores visualizadas em uma matriz de rastreabilidade (abordada na seção 2.1.3.2). Conforme Fiorini, Staa e Baptista (1998, p. 69), gerência de requisitos tem a finalidade de propor um acordo com o cliente com relação aos requisitos a serem considerados no sistema de software. É na gerência de requisitos que há o controle e evolução dos requisitos de um sistema de software, seja pelo surgimento de novos requisitos ou constatação de falhas nos requisitos já existentes.

#### 2.1.3.1 Gerenciamento de mudanças

Conforme Sommerville (2003, p. 121), o gerenciamento de mudanças de requisitos deve ser aplicado sobre quaisquer mudanças propostas para os requisitos. Os principais estágios do processo de gerenciamento são:

a) análise do problema e especificação da mudança: identifica as mudanças propostas para os requisitos;
b) análise e custo da mudança: é feito o rastreamento dos requisitos relacionados e realizado a estimativa de custo da mudança;
c) implementação de mudança: o documento, projeto de sistema ou a implementação são modificados.

#### 2.1.3.2 Rastreabilidade

Para Sommerville (2003, p. 120), é fundamental que se verifique o impacto das

mudanças dos requisitos sobre os outros requisitos relacionados. Para facilitar este trabalho é recomendável a utilização de uma matriz de rastreabilidade (Quadro 1).

| | Requisito 1 | Requisito 2 | Requisito 3 | Requisito 4 | Requisito 5 |
|---|---|---|---|---|---|
| Requisito 1 | | | * | * | |
| Requisito 2 | | | | | * |
| Requisito 3 | | | | * | * |
| Requisito 4 | | * | | | |
| Requisito 5 | | | | | |

Quadro 1 – Exemplo de uma Matriz de Rastreabilidade

Segundo Paula Filho (2003, p. 91), uma especificação dos requisitos é rastreável desde que permita uma fácil determinação dos antecedentes e conseqüências de todos os requisitos, que são denominadas como:

a) rastreabilidade para trás: possibilidade de localizar a origem do requisito;
b) rastreabilidade para frente: possibilidade de localizar quais os resultados do desenvolvimento que serão afetados por cada requisito.

Peters e Pedrycz (2001, p. 151) definem que a rastreabilidade é um atributo para uma ERS de boa qualidade. Pressman (2002, p. 255), relata que a tabela de rastreamento relaciona os requisitos com um ou mais aspectos do sistema ou do seu ambiente. Geralmente, essas tabelas de rastreamento fazem parte da base de dados dos requisitos, para que possam ser acessadas rapidamente a fim de entender onde a alteração ocorrida num requisito vai afetar diferentes aspectos do sistema de software.

Para Kotonya e Sommerville (1998, p. 135), rastreabilidade é parte crítica do gerenciamento de mudanças de requisitos, esse processo é responsável por mostrar o impacto das alterações sobre todo o sistema. As tabelas de rastreamento mostram o relacionamento entre requisitos ou entre requisitos e componentes do sistema, facilitando a visualização das alterações no sistema. Conforme Sommerville (2003, p. 120), a facilidade de rastreamento deve estar ao nível de identificar o impacto das mudanças do requisito sobre outros requisitos. Estas mudanças geralmente são representadas em uma matriz de rastreabilidade. A matriz de rastreabilidade representa o requisito relacionado a outro requisito através de algum símbolo indicador de relacionamento, apresentado no Quadro 1.

## 2.2 IEEE 830-1998

Um resultado importante gerado pelo gerenciamento de requisitos é um documento de

requisitos que abrange todos os requisitos necessários para o entendimento do desenvolvimento de uma aplicação. Segundo Sommerville (2003, p. 95), o documento de requisitos de software também conhecido como *Software Requirements Specification* (SRS) ou Especificação de Requisitos de Software (ERS) é a manifestação oficial exigida pelos desenvolvedores de software. Este deve conter os requisitos de usuário para um sistema e uma especificação minuciosa dos requisitos de sistema. Vários organismos reguladores propuseram padrões para a produção do documento de requisitos, entre eles o Institute of Electrical Engineers (IEEE) com a proposta do modelo IEEE-830-1998.

Para Davis (1993, p. 16), a especificação de requisitos de software é um documento que contém uma completa descrição do que o software deve fazer e como fazer. Para Kotonya e Sommerville (1998, p. 15), o documento é utilizado para apresentar os requisitos aos clientes, engenheiros de software e de sistema e gerenciadores de processos da engenharia de sistemas.

Segundo Institute of Electrical and Electronics Engineers (1998, p. 3), o modelo destaca as considerações essenciais para desenvolver um bom documento de ERS, as quais são:

a) natureza do ERS: especificar as particularidades do sistema;
b) ambiente do ERS: considerar a parte que o ERS representa no plano total do projeto;
c) características do ERS: correto, não ambíguo, completo, consistente, classificado por importância e/ou estabilidade, verificação, modificável, rastreável;
d) junção preparatória do ERS: o processo de desenvolvimento do software iniciará com o acordo entre o fornecedor e o cliente sobre o que o software irá fazer quando concluído;
e) evolução do ERS: a ERS poderá evoluir durante o progresso de desenvolvimento do software;
f) protótipos: é usado freqüentemente durante uma fase dos requisitos do projeto;
g) incluindo interfaces externas no ERS: um requisito especifica uma função visível externamente ou atributo do sistema;
h) requisitos de projeto gerados no ERS: o ERS deve estar focada no produto de software, e não ao processo de produzir esse produto de software.

Segundo Institute of Electrical and Electronics Engineers (1998, p. 11), o modelo IEEE-830-1998 é uma prática recomendada para especificação dos requisitos de software. As partes essenciais de um documento de ERS definidas no modelo IEEE-830-1998 são:

a) introdução: compreende o propósito, escopo, definições, referências e visão geral;
b) descrição geral: descreve a perspectiva do produto, funções, características do usuário, restrições, opções e dependência;
c) requisitos específicos: requisitos funcionais e não funcionais;
d) apêndices;
e) índice.

## 2.3 AMBIENTES CASE INTEGRADOS

Segundo Pressman (2002, p. 823), ferramentas CASE são ferramentas de engenharia de software apoiada por computador, que auxiliam na automatização do processo de desenvolvimento de um projeto de sistema de software. É utilizada principalmente por gerentes e profissionais da engenharia de software. Para Sommerville (2003, p. 120), o suporte de ferramentas CASE no gerenciamento de requisitos envolve processar um grande volume de informações sobre os requisitos. Vários tipos de ferramentas podem ser utilizadas, como por exemplo, sistemas especializados de gerenciamento de requisitos.

As ferramentas CASE também podem ser integradas, no entanto, existem alguns desafios, entre estes, se forem de fabricantes diferentes. Embora muitos fabricantes disponibilizem meios de integração com outras ferramentas CASE, a maioria ainda mantém sua ferramenta fechada (PRESSMAN, 2002, p. 823).

Conforme Pressman (2002, p. 815), os ambientes *Integrated* CASE (I-CASE) abrangem um conjunto de ferramentas com interface padronizada e operando sobre um único repositório de dados. A interface padronizada garante que os membros do projeto visualizem a informação com o mesmo aspecto nas diferentes ferramentas integradas. O repositório de dados disponibiliza o acesso à informação para todas as ferramentas integradas. O I-CASE também possibilita que a informação alterada no repositório de dados seja visualizada pelas outras ferramentas integradas, em tempo real. Essa rastreabilidade garante a integridade da informação.

Sommerville (2003, p. 186), destaca as vantagens e desvantagens do compartilhamento do repositório, são elas:

a) não existe a necessidade de transmitir dados explicitamente de um subsistema para outro;

b) os subsistemas devem concordar com o modelo de dados, novos subsistemas podem entrar em conflito com o modelo proposto;
c) os dados gerados pelos subsistemas são utilizados por outros subsistemas;
d) devido ao grande volume de informação, a transferência do modelo de dados para outro pode ser inviável ou impossível de ser realizado;
e) a manipulação dos dados é centralizada, facilitando as atividades do gerente de repositório;
f) pode ocorrer redundância e inconsistência dos dados.

## 2.4 FERRAMENTAS SIMILARES DE GERENCIAMENTO DE REQUISITOS

No mercado existem inúmeras ferramentas para o gerenciamento de requisitos, outras ferramentas similares também foram propostas como trabalho de conclusão de curso. Nesta seção são apresentadas algumas ferramentas destacando suas principais características e um comparativo entre elas.

### 2.4.1 Documentação de requisitos

Em José (2002) é apresentada uma ferramenta de apoio à documentação de requisitos de software que abrange os principais aspectos da engenharia de requisitos para a documentação e gerenciamento dos documentos de requisitos. Foram utilizados modelos de documentos de requisitos para constituir o trabalho, tais como: formulário com questionários para aplicar nas entrevistas com o cliente e *stakeholders*[2]; cadastro para manter informações sobre clientes e os envolvidos no projeto; formulário para elicitação dos requisitos junto aos *stakeholders*; telas com diversas funções como, por exemplo, cadastro e consulta; emissão de um documento constando informações, tais como requisitos e projetos. No Quadro 2 é apresentado a ficha de requisitos.

[2] O termo *stakeholders* é utilizado para se referir a qualquer pessoa que terá alguma influência direta ou indireta sobre os requisitos do sistema (SOMMERVILLE, 2003, p. 104).

Ficha de Requisitos

Contexto do Requisito

1 - Código do Requisito:_____ 2 - Data de Cadastro:___/___/_____ 3 - Data de Alteração:___/___/_____

4 - Código do Requisito "Pai" (se ele é dependente hierárquicamente a algum outro requisito):_____

5 - Stakeholder responsável pelo fornecimento da informação:_____

6 - Área do Responsável (Depto., função):_____

7 - Domínio de uso do requisito (foco e abrangência):_____

8 - Qualificação funcional do requisito: ( ) Operacional ( ) Gerencial ( ) Estratégico

9 - Área de origem: ( ) Interna ( ) Externa ( ) Ordem Legal 10 - Prioridade: ( ) Baixa ( ) Média ( ) Alta

11 - Situação do Requisito: ( ) Proposto ( ) Aprovado ( ) Recusado 12 - Custo: ( ) Baixo ( ) Médio ( ) Alto

13 - Dificuldade: ( ) Baixa ( ) Média ( ) Alta

Domínio do Requisito

14 - Descrição do Requisito [descrever na forma (sujeito + verbo + objeto), (funcional)]:

15 - Problema Identificado ( O que é e o porque do problema):

16 - Produto (Qual a aplicação?):

17 - Aplicação ( para quem?):

Complementares:

18 - Atributos:

19 - Restrições

20 - Preferências

21 - Expectativas

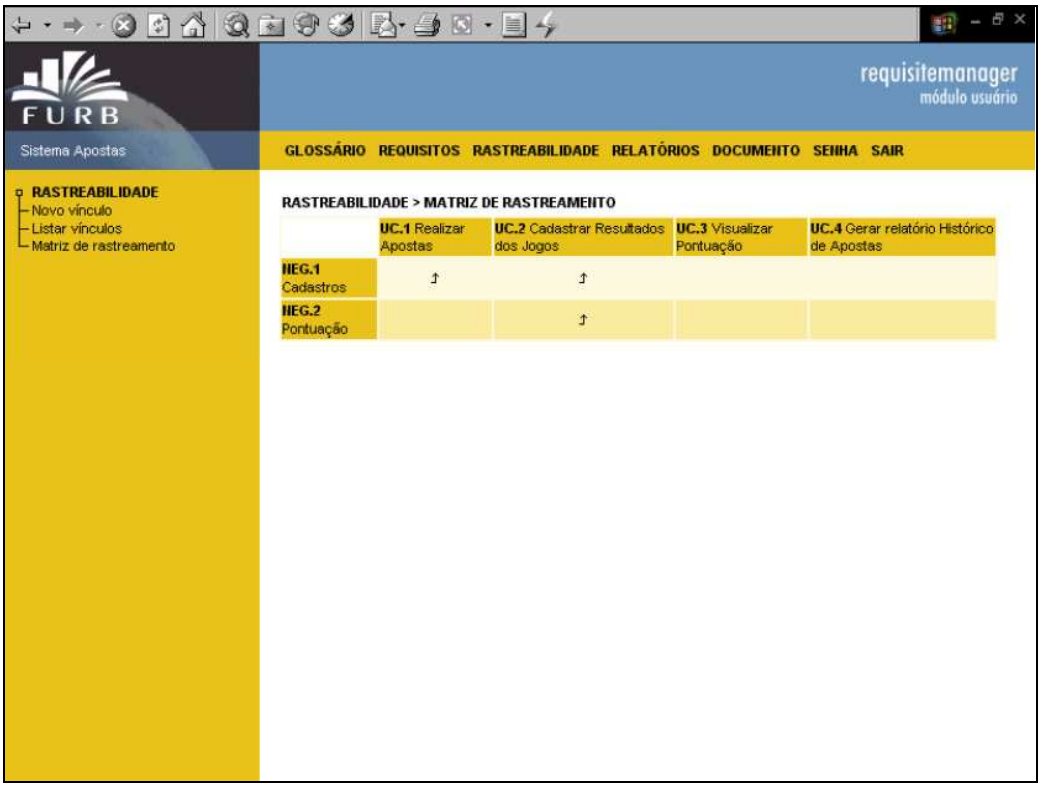
Fonte: José (2002, p. 46).

Quadro 2 – Ficha para elicitação dos requisitos

### 2.4.2 Requisitemanager

Em Marquardt (2004) é apresentada uma ferramenta de gerenciamento de requisitos de software via Web, codificada com PHP e banco de dados MySQL. As principais funcionalidades incluem a definição de tipos de requisitos, tipos de atributos, controle de requisitos, rastreabilidade e geração de relatórios. A ferramenta permite emitir o documento de requisitos de software que obedece ao padrão IEEE/ANSI 830-1993. A proposta deste

trabalho visa o auxílio na disciplina de requisitos de software. Na Figura 2 é ilustrada a matriz de rastreabilidade.

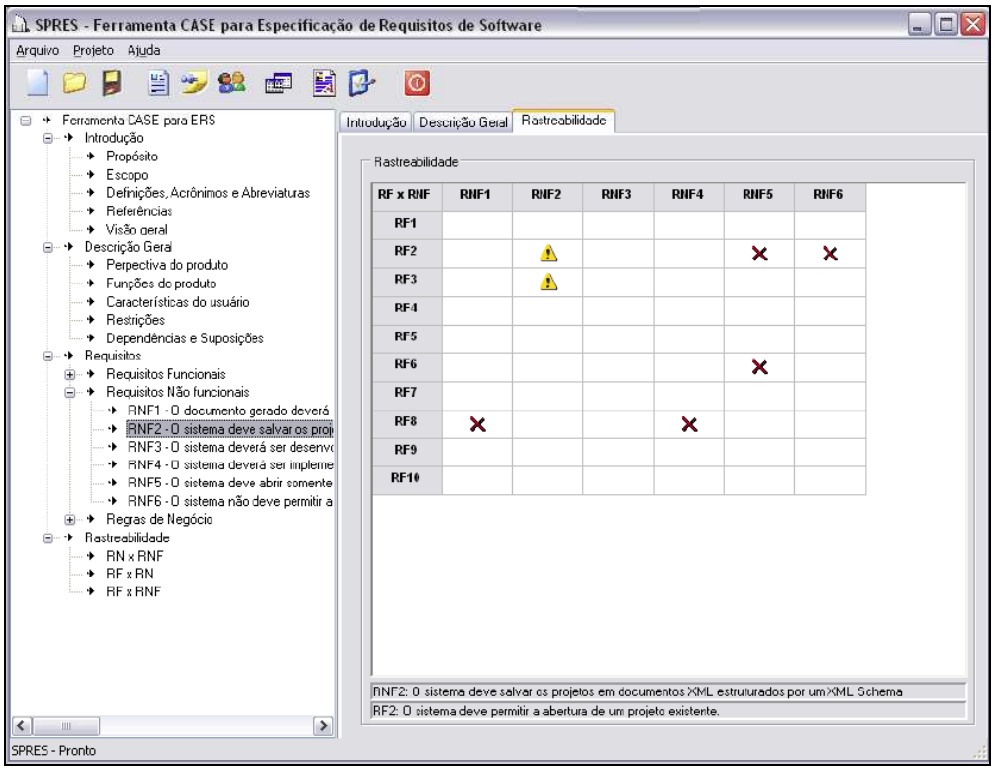



Fonte: Marquardt (2004, p. 71).

Figura 2 – Matriz de rastreabilidade

### 2.4.3 SPRES

Em Quirino (2004) é apresentada uma ferramenta CASE para especificação de requisitos de software observando o modelo de documento IEEE-830-1998 e do *Rational Unified Process* (RUP). A ferramenta permite criar atributos para identificar os requisitos e classificá-los por categoria, versões para o documento de ERS, matriz de rastreabilidade entre os elementos do ERS, cadastro do conteúdo das seções do documento de ERS e gerar o documento de ERS. Na Figura 3 é ilustrada a matriz de rastreabilidade.

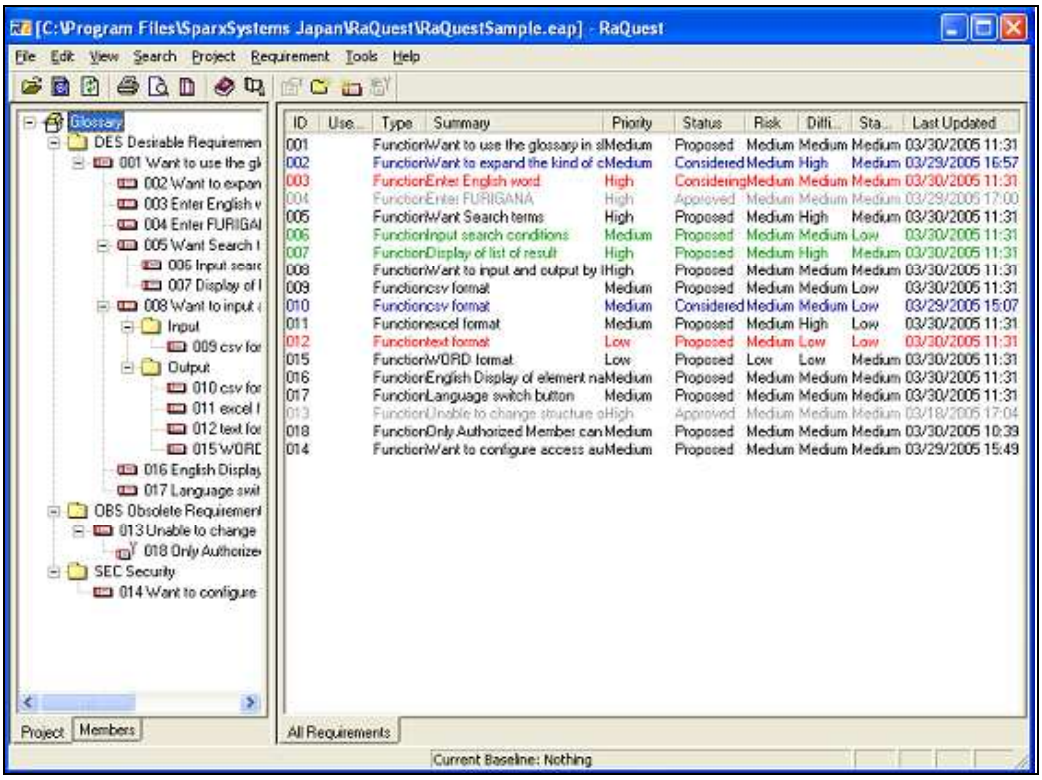



Fonte: Quirino (2004, p. 62).

Figura 3 – Matriz de rastreabilidade

### 2.4.4 Raquest

RaQuest (2007) é uma ferramenta de gerenciamento de requisitos desenvolvida por uma empresa filial da Spark Systems no Japão. Esta ferramenta é uma extensão do EA, sendo necessária a sua instalação como requisito para utilização. Os principais recursos desta ferramenta são que as informações serão salvas no mesmo repositório do EA. Os requisitos criados no RaQuest poderão ser alterados no EA e os requisitos poderão ser relacionados com os casos de uso criados no EA. Na Figura 4 são ilustrados os requisitos cadastrados.




Fonte: RaQuest (2007).

Figura 4 – Requisitos cadastrados

### 2.4.5 Comparativo entre as ferramentas

No Quadro 3 é apresentado um comparativo entre as ferramentas e suas principais características.

| Ferramenta / Característica | Documentação de Requisitos | Requisitemanager | SPRES | RaQuest |
|---|---|---|---|---|
| **Rastreabilidade dos Requisitos** | Não | Sim | Sim | Sim |
| **Finalidade de Uso** | Acadêmica | Acadêmica | Acadêmica | Comercial |
| **Padrão do documento de ERS** | Próprio | IEEE-830-1993 | IEEE-830-1998 e RUP | Próprio |
| **Plataforma** | Windows | Web | Windows | Windows |
| **Integração com Ferramentas CASE** | Não | Não | Não | Sim |
| **Histórico de Alterações** | Não | Sim | Sim | Sim |

Quadro 3 – Comparativo entre as ferramentas de gerenciamento de requisitos

Vale salientar que a ferramenta RaQuest é a única a possuir a característica de integração com ferramentas CASE, também disponibiliza entre outros recursos o histórico de alterações dos requisitos, entretanto, é necessário a compra da licença de uso. A ferramenta Requisitemanager possui a vantagem de utilização através da Web, também possui uma

funcionalidade que gera o histórico de alterações dos requisitos, indo ao encontro do conceito de gerenciamento de mudanças, porém o padrão do documento de ERS é anterior ao padrão IEEE-830-1998. A ferramenta de Documentação de Requisitos possui inúmeras características fundamentais para uma ERS de boa qualidade, como, por exemplo, emissão de fichas como a de requisitos, ficha de projeto e questões de livre contexto, cadastro de requisitos e diversos relatórios dos quais pode-se destacar o relatório de requisitos, no entanto, não possui uma matriz de rastreabilidade de requisitos. Já a ferramenta SPRES disponibiliza uma matriz de rastreabilidade de requisitos, a geração do documento de ERS no padrão IEEE-830-1998 e RUP e permite gerar o histórico de alterações do documento de ERS, porém não permite a integração com outras ferramentas CASE.

## 2.5 DESENVOLVIMENTO INTEGRADO AO ENTERPRISE ARCHITECT

Esta seção apresenta uma visão geral da ferramenta Enterprise Architect, bem como os seus recursos para gerência de requisitos e a interface de automação para promover a integração com outras ferramentas.

### 2.5.1 Visão geral

Segundo Sparx Systems (2005, p. 3), o EA é uma ferramenta CASE baseada na UML utilizada no desenho e construção de projetos de sistema de software. A UML permite descrever de forma visual as necessidades que existe em um projeto de sistema de software através de mapas e modelos. O EA abrange todas as fases do ciclo de desenvolvimento do projeto do sistema de software, desde o levantamento das necessidades, o desenvolvimento até a manutenção. Dentre os recursos mais relevantes, pode-se citar a criação de elementos no modelo da UML, a conectividade entre os elementos e geração de documentos sobre os elementos criados.

Além da UML o EA permite incluir requisitos, neste sentido, pode-se observar na Figura 5 um elemento para representar um requisito e a forma de conectividade entre os elementos, onde se tem a vinculação de requisitos com casos de uso.

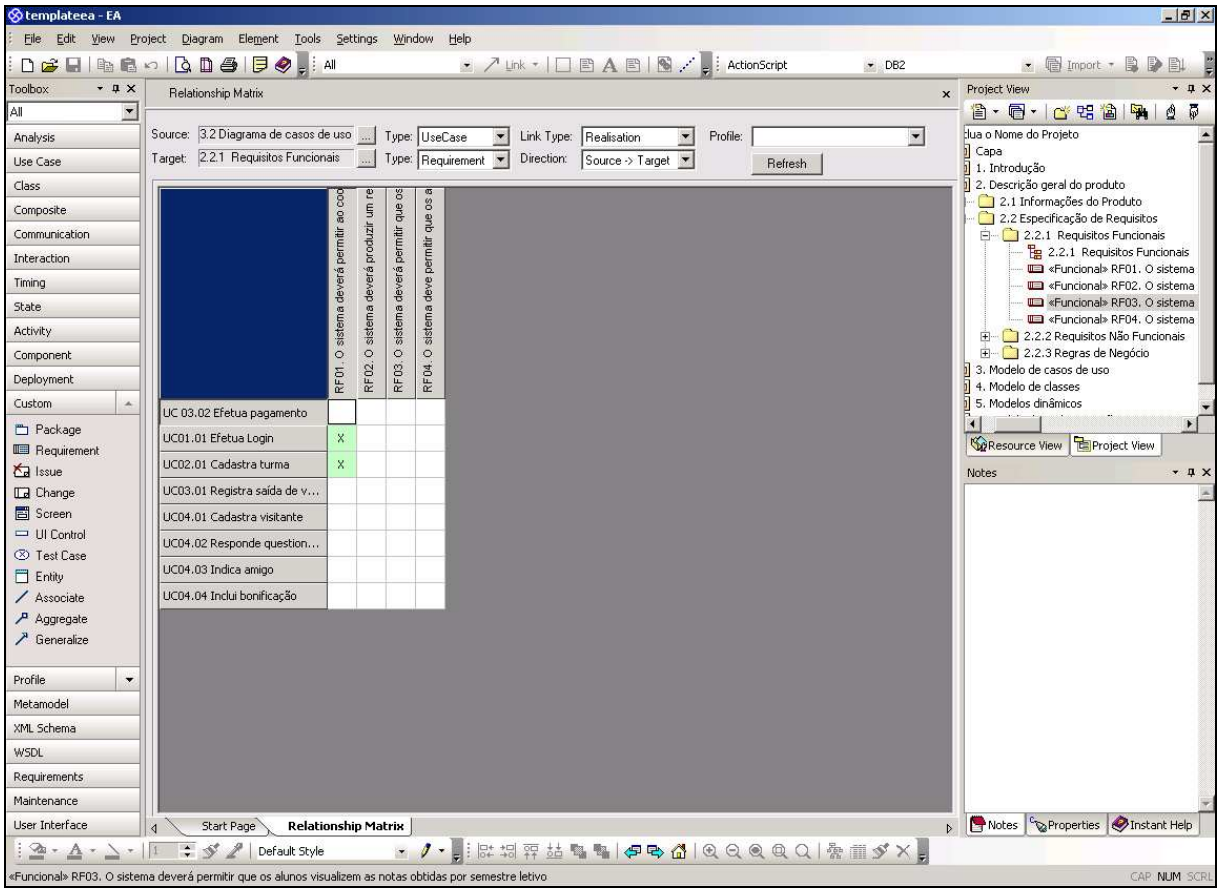



Figura 5 – Matriz de rastreabilidade do EA

Conforme Sparx Systems (2005, p. 1133), o EA disponibiliza dentre outros recursos a automação de interface, permitindo que outros aplicativos acessem os seus dados. Este recurso está disponível apenas para plataforma Windows, através de controles ActiveX. A biblioteca denominada *Enterprise Architect Object Model 2.0*, permite o acesso a sua interface através das linguagens de programação como, por exemplo, o Object Pascal disponível no software Borland Delphi 6.0. Através da biblioteca de automação, outras ferramentas que incorporam os seus recursos, poderão manipular todos os elementos que o EA disponibiliza através do seu repositório, conforme é apresentado na seção 2.5.3.

### 2.5.2 Gerenciamento de requisitos no Enterprise Architect

O EA permite o gerenciamento de requisitos, todavia os recursos disponibilizados apresentam algumas vantagens e desvantagens que são descritas a seguir:

a) não proporciona a confecção do documento de ERS no padrão IEEE-830-1998;
b) permite ao usuário definir os tipos de requisitos e sua estrutura de organização, como por exemplo, uma estrutura contendo requisitos funcionais e outra contendo

os não funcionais;

c) permite a formatação do nome do requisito podendo configurar o prefixo, numeração seqüencial e sufixo;
d) não redefine a numeração seqüencial dos requisitos após a exclusão de algum requisito intermediário aos demais requisitos;
e) permite o relacionamento dos elementos, como por exemplo, casos de uso e requisitos através de uma matriz de relacionamento;
f) a descrição dos elementos relacionados na matriz de relacionamento quando excede o limite definido pela ferramenta aparece truncada, impossibilitando a identificação precisa do elemento, conforme pode ser observado na Figura 5;
g) o documento de ERS é gerado em formato gráfico, não permitindo a alteração ou complementação do conteúdo gerado através de editor de texto.

De modo geral, o EA disponibiliza os vários recursos para uma ERS, contudo a ferramenta não apresenta uma boa flexibilidade quanto a usabilidade. Para complementar a carência da usabilidade do EA pode-se adotar uma ferramenta CASE integrada que proporcione uma extensão do EA para o gerenciamento de requisitos.

### 2.5.3 Estrutura e interface de automação do Enterprise Architect

A interface de automação é um recurso disponibilizado pelo EA que permite a outras ferramentas acessarem e gerenciarem seus elementos, este recurso vai ao encontro do conceito de I-CASE. Na biblioteca para interface de automação são disponíveis em código fonte várias classes com métodos e propriedades para manipular os elementos do EA.

Todos os ambientes de desenvolvimento que suportam ActiveX COM *clients* são capazes de se conectarem ao *Enterprise Architect Automation Interface* (SPARX SYSTEMS, 2006, p. 1162). Conforme Borland Software Corporation (2001), o *Component Object Model* (COM) é baseado em Windows e prove interoperabilidade entre objetos, a partir de rotinas pré-definidas chamadas de interface. A interface propõe separar os objetos da implementação, o que permite a utilização dos objetos em diferentes ambientes de desenvolvimento. Um dos aspectos chave do COM é de permitir a comunicação entre aplicações cliente e servidor através de interfaces bem definidas.

Segundo Borland Software Corporation (2001), um controle ActiveX é um componente de software que integra e estende a funcionalidade de qualquer aplicação

principal que suporta esses controles. Os controles ActiveX implementam um conjunto particular de interfaces que permitem esta integração. De modo geral, um controle ActiveX facilita a integração entre as aplicações. O ambiente de desenvolvimento Borland Delphi 6.0 suporta controles ActiveX.

Para gerar o ActiveX COM *clients* no Delphi basta clicar em *Project* no menu principal e na opção *Import Type Library* e na tela que precede, selecionar o arquivo da biblioteca do EA e como último passo clicar no botão *Create Unit*, conforme ilustra a Figura 6. Após estes procedimentos um arquivo contendo a biblioteca é criado com o nome EA_TLB com a extensão .pas.

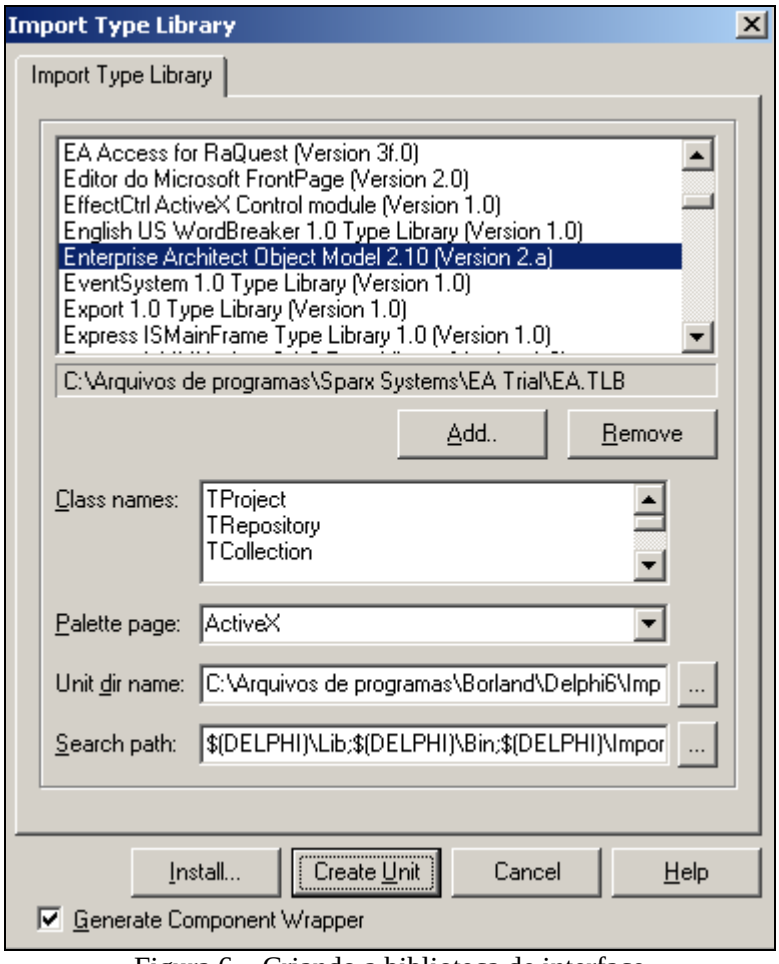



Figura 6 – Criando a biblioteca de interface

Para utilizar a biblioteca EA_TLB basta adicioná-la no arquivo fonte. Um exemplo de implementação utilizando as classes da biblioteca pode ser observado no Quadro 4, onde foi adicionado a biblioteca EA_TLB, implementado a criação e abertura de um repositório de dados de um projeto do EA.

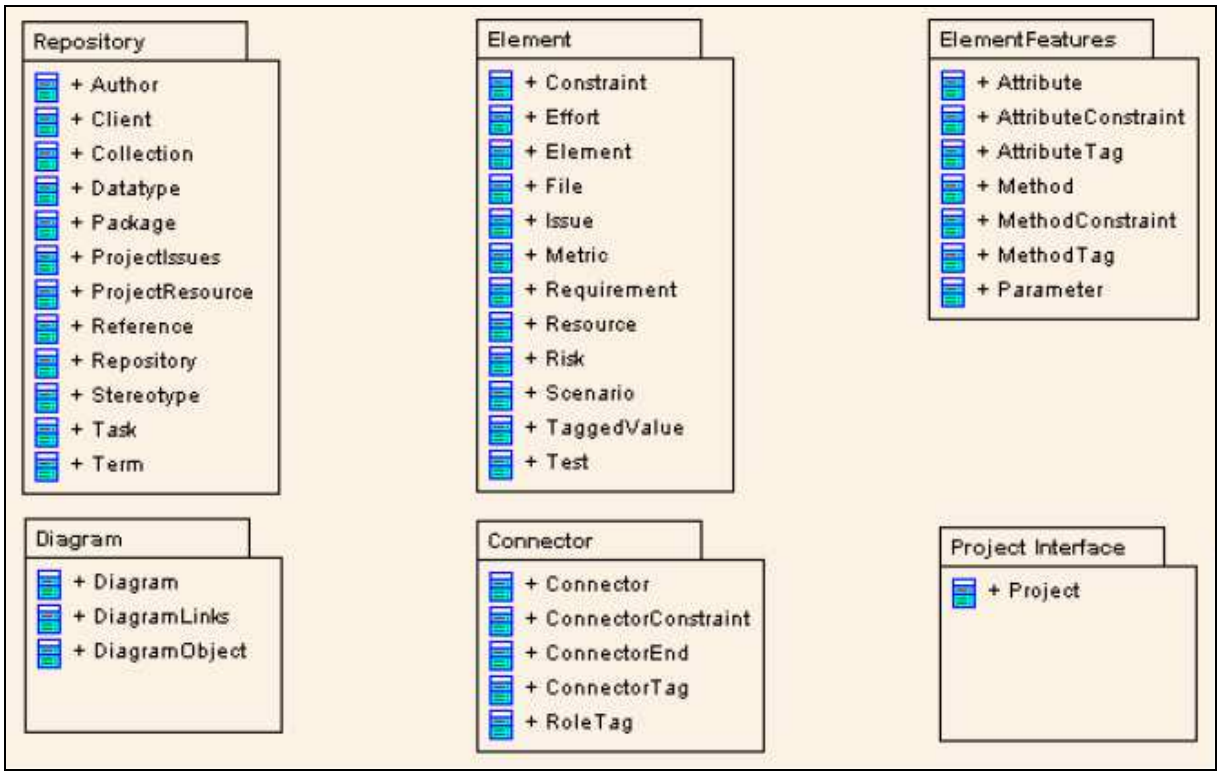
```
implementation

uses EA_TLB;

{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
var
    r : TRepository;
begin
    r := TRepository.Create(nil);

    try
        r.OpenFile('D:\temp\EAtmp.eap');
        r.ShowWindow(SW_SHOW);
    except
        r.CloseFile;
        raise;
    end;
end;
```

Quadro 4 – Acessando o repositório de dados

Segundo Sparx Systems (2005, p. 1147), a integração entre o EA e outras ferramentas é feita sobre um único repositório de dados, permitindo que as informações sejam atualizadas entre as ferramentas em tempo real. O repositório de dados engloba toda a coleção de modelos, pacotes, elementos, etc.

A Figura 7 ilustra um diagrama com os principais elementos da interface e seu conteúdo associado que podem ser criados através da automação de interface (SPARX SYSTEMS, 2005, p. 1169). O diagrama representa um resumo do nível mais elevado da interface de automação para acessar, manipular, modificar e criar elementos no EA. O repositório contém uma variedade de níveis de objetos que permite acesso aos demais elementos (SPARX SYSTEMS, 2005, p. 1170). Estes elementos são:

a) repositório: representa o modelo principal e prove a inclusão de outros elementos;
b) elementos: é a unidade de estrutura básica;
c) características dos elementos: são os atributos e operações sobre um elemento;
d) diagramas: é a representação gráfica do que contém no modelo;
e) conectores: prove o relacionamento entre os elementos;
f) interface do projeto: XML baseado nos elementos do EA para interface através de outras ferramentas baseadas em ActiveX COM *clients*.




Fonte: Sparx Systems (2005, p. 1169).

Figura 7 – Estrutura principal da Interface de Automação do EA

Um modelo é uma propriedade especial do repositório o nível mais alto de um projeto no EA. O modelo é o topo da raiz do projeto, o nó principal que através dele é possível acessar qualquer elemento da sua hierarquia, conforme ilustrado na Figura 8. É possível incluir novos modelos e excluí-los, porém, é obrigatório que haja pelo menos um modelo como raiz (SPARX SYSTEMS, 2005, p. 1176).

O pacote é um atributo do repositório utilizado para manter outros elementos. Um pacote também é um elemento que pode ser associado a outros elementos como, por exemplo, relacionar com um modelo (SPARX SYSTEMS, 2005, p. 1187). A Figura 8 ilustra um pacote com vários elementos conectados.

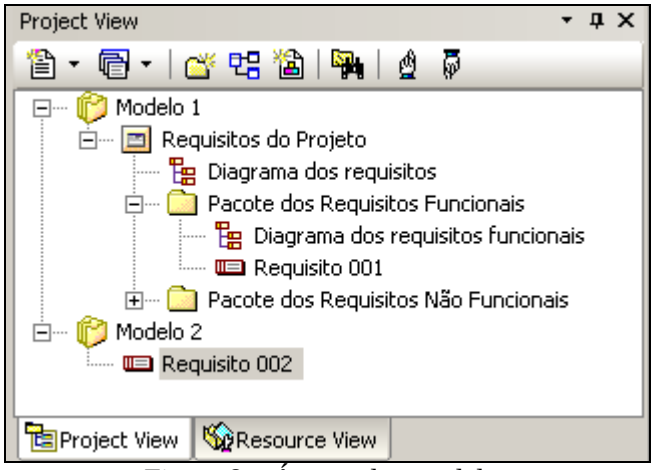



Figura 8 – Árvore dos modelos

No EA um elemento é definido através de uma classe com inúmeros atributos que qualificam o elemento, existem diferentes tipos de elementos, como por exemplo, um elemento pode ser do tipo requisito. Os elementos podem estar contidos em diagramas e relacionados com outros elementos através de conectores. Basicamente, um elemento é qualquer item inserido em um modelo (SPARX SYSTEMS, 2005, p. 1195). Na Figura 8 o elemento denominado Requisito 001 é do tipo requisito.

Um diagrama pode conter vários elementos e vários pacotes, e podem estar associados entre si através de relacionamento. Dentro de um pacote do diagrama, é possível criar outros diagramas (SPARX SYSTEMS, 2005, p. 1225). Na Figura 9 observa-se um diagrama com dois pacotes.

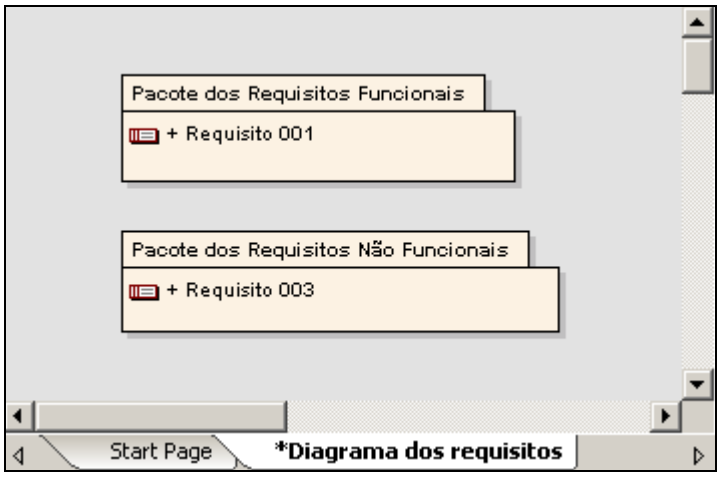



Figura 9 – Diagrama dos requisitos

Conectores também são definidos através de classes. Um conector representa uma classe que relaciona um elemento a outro elemento (SPARX SYSTEMS, 2005, p. 1218). Existem vários tipos de conectores, como por exemplo, um conector de agregação. Um relacionamento de agregação demonstra que um elemento contém ou é composto de outro elemento (SPARX SYSTEMS, 2005, p. 420).

Em suma, o EA disponibiliza na biblioteca *Enterprise Architect Object Model* uma grande quantidade de interfaces para realizar o acesso através de outras ferramentas, permitindo a integração entre ferramentas. No caso do gerenciamento de requisitos, é possível criar uma ferramenta para ERS integrada com o EA que opere sobre o mesmo repositório de dados acrescentando uma interface própria para esta finalidade.

# 3 DESENVOLVIMENTO DO TRABALHO

Neste capítulo são apresentados os tópicos referentes a confecção deste trabalho. A primeira seção descreve os requisitos funcionais e não funcionais. Na segunda seção é apresentada a especificação da ferramenta, para os casos de usos e diagrama de classe optou-se por representá-los através de diagramas com base na UML. Ainda nesta seção, são demonstradas as ferramentas para a definição da estrutura de dados da ferramenta em XML. A terceira seção aborda sobre a implementação, as técnicas e ferramentas utilizadas e a operacionalidade da ferramenta. A quarta seção explana sobre os resultados da ferramenta em relação as ferramentas similares.

## 3.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

Nas subseções seguintes são apresentados os requisitos funcionais e não funcionais utilizados para o desenvolvimento deste trabalho. A identificação destes requisitos teve como base o estudo de referencial teórico (conforme apresentado no capítulo 2) e também a análise de ferramentas correlatas (descritas na seção 2.4). Além destes aspectos, foi considerado fundamental para a definição dos requisitos as características apontadas no documento da IEEE-830-1998.

### 3.1.1 Requisitos funcionais

Os principais requisitos funcionais desta ferramenta incluem:

a) cadastrar os dados das seções do documento de requisitos;
b) permitir abrir o projeto do EA para importar os requisitos;
c) cadastrar os tipos de requisitos desejados para o levantamento de requisitos;
d) cadastrar os requisitos do projeto levantados pelo o usuário;
e) cadastrar os valores do atributo estado do requisito que podem ser, por exemplo, proposto, aprovado e obrigatório;
f) gerenciar os requisitos permitindo vincular requisitos com outros requisitos e casos

de usos, viabilizando a rastreabilidade (funcionalidade primordial para o gerenciamento de requisitos);

g) sinalizar ao usuário, as alterações realizadas no requisito que possui vínculo com casos de uso ou outro requisito;
h) permitir gerar os requisitos no projeto do EA;
i) emitir o relatório do documento de especificação de requisitos.

### 3.1.2 Requisitos não funcionais

Os principais requisitos não funcionais desta ferramenta incluem:

a) disponibilizar o documento de ERS em conformidade com o modelo IEEE-830-1998;
b) utilizar a técnica “Estrutura de Árvore”, que mostrará os requisitos em forma de hierarquia para manipular os requisitos na área de trabalho;
c) permitir portabilidade com Windows 2000/XP;
d) utilizar controles ActiveX para acessar a interface do EA e manipular seus objetos;
e) utilizar ícones representativos para identificar a situação do requisito: criado (indica que ainda não possui vínculo com outros elementos (caso de usos ou requisito)), alterado (indica que sofreu alguma alteração da sua descrição) e vínculo (indica que possui vínculo com outros elementos).

## 3.2 ESPECIFICAÇÃO

Esta seção apresenta os diagramas e ferramentas utilizadas para especificação da ferramenta.

### 3.2.1 Casos de uso

Na Figura 10 são apresentados os casos de uso com a descrição das funcionalidades da ferramenta das quais o usuário está interagindo.

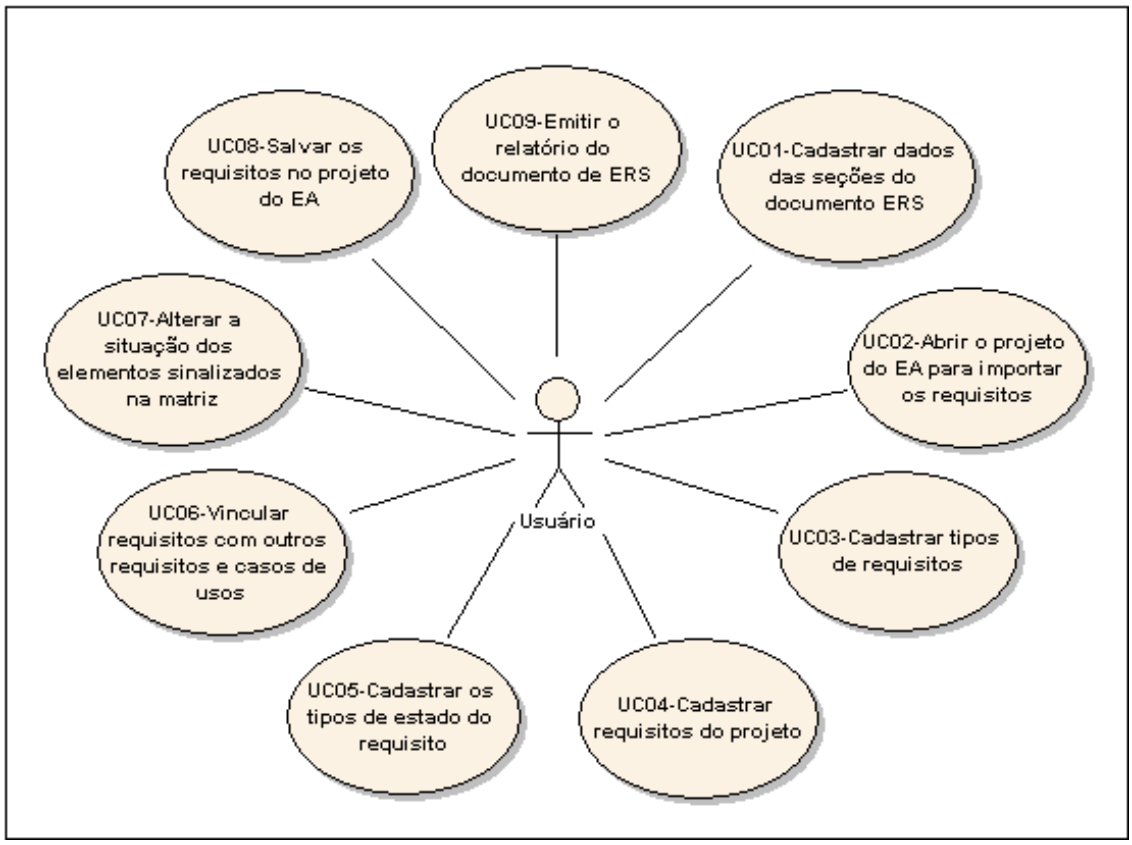



Figura 10 – Diagrama de casos de uso da ferramenta

A seguir, tem-se uma breve descrição dos casos de uso:

a) cadastrar dados das seções do documento ERS: permitir cadastrar as seções do documento. Nesta etapa não é necessário estar conectado ao EA, porém, ao primeiro comando de salvar o documento ou inclusão de um requisito, será solicitado o arquivo do projeto do EA;
b) abrir o projeto do EA para importar os requisitos: cria um vínculo entre o documento e o projeto do EA. Se já existir o vínculo, carrega e atualiza os requisitos no arquivo do projeto do EA. Caso o arquivo do EA já possua o vínculo, verificar a consistência entre os requisitos da ferramenta armazenados em arquivos XML e os requisitos do repositório do EA, se houverem divergências quanto a exclusão de requisitos, alertar o usuário utilizando mensagens de alerta;
c) cadastrar tipos de requisitos: permite cadastrar novos tipos de requisitos, como por exemplo, regras de negócio;
d) cadastrar requisitos do projeto: permite cadastrar os requisitos vinculando ao tipo de requisito definido. Nesta etapa o arquivo do projeto do EA deve estar carregado;
e) cadastrar os tipos de estado do requisito: permite cadastrar novos tipos de estado do requisitos, por exemplo, proposto, aprovado e obrigatório. Nesta etapa o

arquivo do projeto do EA deve estar carregado;

f) vincular requisitos com outros requisitos e casos de uso: permitir relacionar os requisitos com outros tipos de requisitos e casos de uso, estabelecendo a rastreabilidade entre os elementos do projeto. Nesta etapa o arquivo do projeto do EA deve estar carregado;
g) alterar a situação dos elementos sinalizados na matriz: permitir alterar a situação do relacionamento dos elementos na matriz que estejam sinalizados por terem sofrido alteração. Na matriz de rastreabilidade, os elementos relacionados, permanecem marcados até que o usuário confirme a alteração do requisito. Nesta etapa o arquivo do projeto do EA deve estar carregado;
h) salvar os requisitos no projeto do EA: permitir salvar os requisitos verificando a consistência entre os requisitos da ferramenta armazenados em arquivos XML e os requisitos do repositório do EA, se houverem divergências quanto a exclusão de requisitos, alertar o usuário utilizando mensagens de alerta;
i) emitir o relatório do documento de ERS: permitir o usuário emitir o documento completo de ERS, contendo todas as seções cadastradas mais os requisitos específicos separados pelo tipo de requisito.

### 3.2.2 Diagrama de classes

Nesta seção é apresentado o diagrama de classes de projeto para o desenvolvimento da ferramenta, conforme apresentado na Figura 11. Os detalhes das classes estão no Apêndice A.

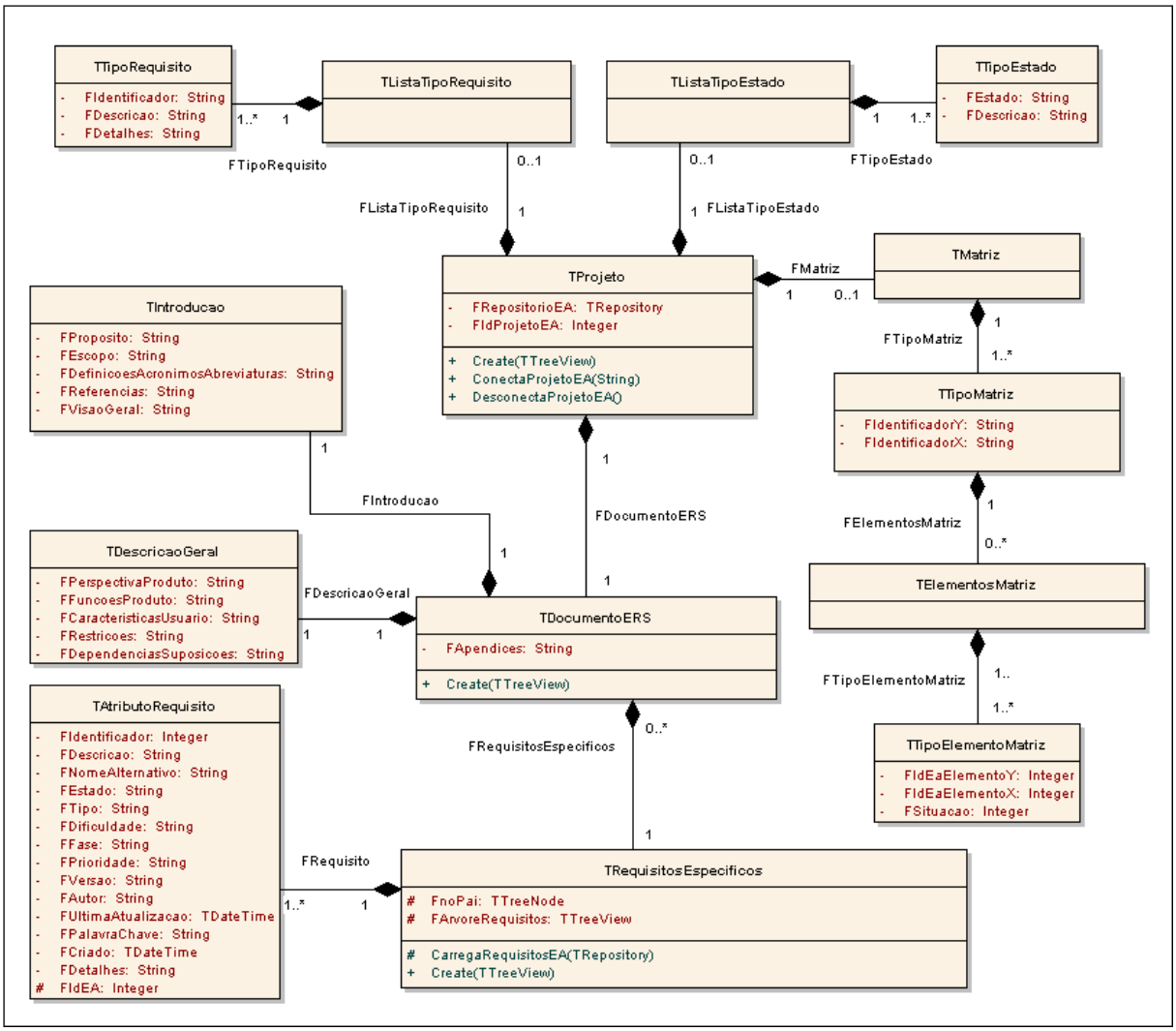



Figura 11 – Diagrama de Classes

Em seguida, uma breve descrição das classes do diagrama:

a) `TProjeto`: classe principal responsável pelo acesso ao repositório do EA e manipular os dados dos atributos do projeto, documento de ERS e da matriz de rastreabilidade;

b) `TMatriz`: classe que mantém uma lista de tipos de matrizes de rastreabilidade, como por exemplo, matriz do tipo RFxRN e matriz do tipo RNFxRF;

c) `TTipoMatrix`: classe que mantém o tipo da matriz de rastreabilidade, como por exemplo, uma matriz do tipo RNFxRN, que define uma matriz de rastreabilidade entre requisitos não funcionais e regras de negócio;

d) `TElementosMatrix`: classe que mantém uma lista de elementos relacionados na matriz de rastreabilidade, como por exemplo, um elemento do tipo requisitos não funcional e outro do tipo regra de negócio;

e) `TTipoElementoMatrix`: classe que mantém os elementos relacionados e a situação

entre estes elementos, como por exemplo, o relacionamento de um requisito funcional com uma regra de negócio;

f) `TDocumentoERS`: classe que mantém as seções do documento de ERS, como por exemplo a seção Descrição Geral;
g) `TDescricaoGeral`: classe que mantém as subseções da seção descrição geral do documento de ERS, como por exemplo, a subseção Restrições;
h) `TIntroducao`: classe que mantém as subseções da seção introdução do documento de ERS, como por exemplo, a subseção Propósito;
i) `TRequisitosEspecificos`: classe que mantém os requisitos da seção requisitos específicos do documento de ERS que provem do repositório do EA, como por exemplo, requisitos funcionais e requisitos não funcionais;
j) `TAtributoRequisito`: mantém os atributos dos requisitos que provem do repositório do EA, como por exemplo, o atributo de prioridade e estado do requisito;
k) `TListaTipoEstado`: classe que mantém uma lista de tipos de estado do requisito, como por exemplo, proposto, aprovado e obrigatório;
l) `TTipoEstado`: classe que mantém os tipos de estado do requisito que provem do repositório do EA;
m) `TListaTipoRequisito`: classe que mantém uma lista dos tipos de requisito, como por exemplo, requisito funcional e não funcional;
n) `TTipoRequisito`: classe que mantém os tipos de requisitos que provem do repositório do EA, com por exemplo, regra de negócio.

### 3.2.3 XML Schema

Segundo World Wide Web Consortium (2007), *Extensible Markup Language* (XML) é uma linguagem de marcação utilizada na internet para prover o compartilhando de informações. A linguagem *XML Schema Definition* (XSD) descreve a estrutura de um documento XML. A XSD é uma alternativa a linguagem *Document Type Definition* (DTD), já que a DTD não é baseada no formato XML (WORLD WIDE WEB CONSORTIUM, 2007). De modo geral, um XSD define:

a) tipos de dados dos atributos do documento XML;

b) estrutura do documento XML;

c) conformidade entre o esquema e o documento XML.

No desenvolvimento deste trabalho foi gerado a partir do software XMLSpy o arquivo XSD para manter as seções do documento de ERS, bem como os atributos necessários para relacionar o documento de ERS com o projeto do EA, tendo como base o diagrama de classes. As figuras a seguir, ilustram os diagramas modelados no XMLSpy para geração do arquivo XSD. A Figura 12 apresenta o diagrama do projeto e sua estrutura.

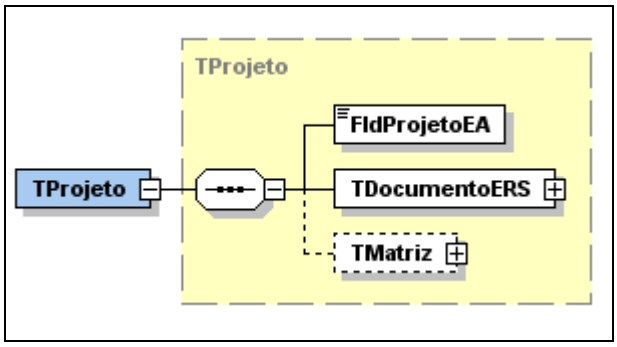



Figura 12 – Diagrama do projeto

A Figura 13 apresenta o diagrama do documento de ERS e sua estrutura.

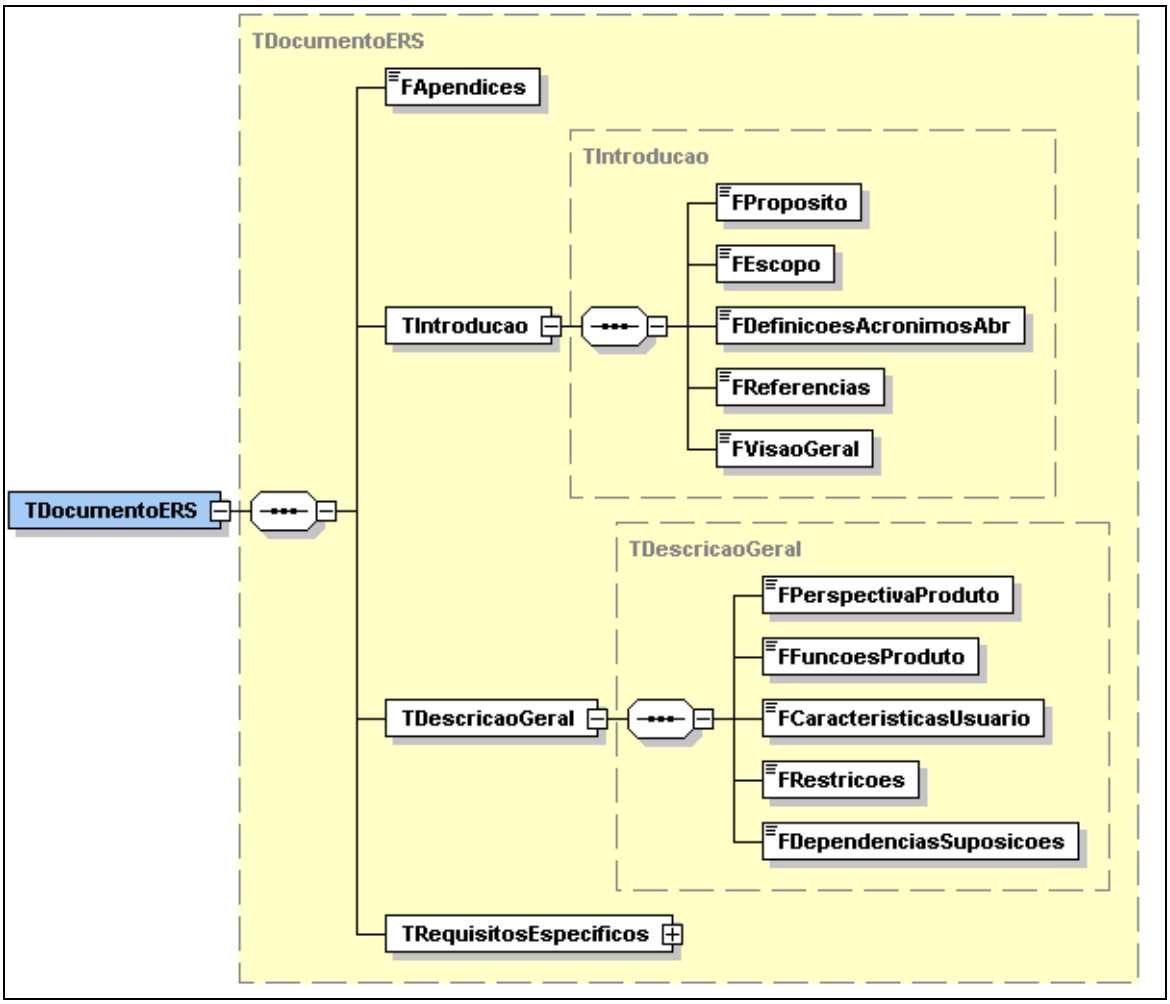



Figura 13 – Diagrama do documento de ERS

A Figura 14 apresenta o diagrama dos requisitos específicos do documento de ERS e sua estrutura.

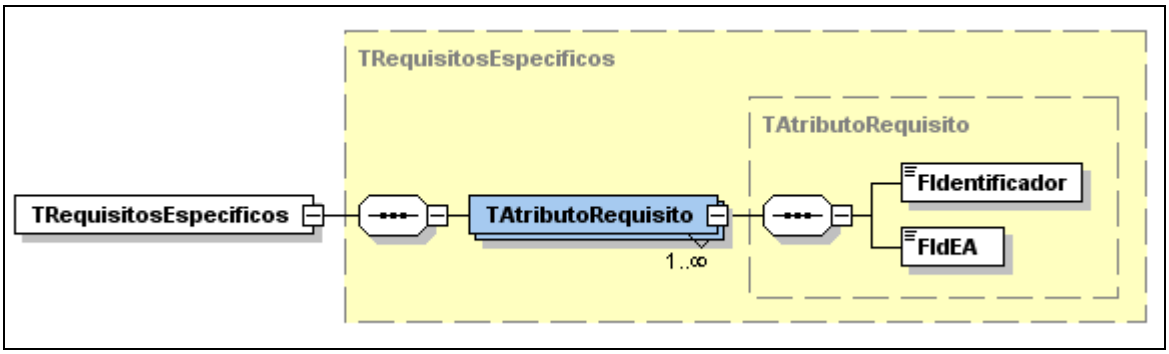



Figura 14 – Diagrama dos requisitos específicos

A Figura 15 apresenta o diagrama da matriz de rastreabilidade do documento de ERS e sua estrutura.

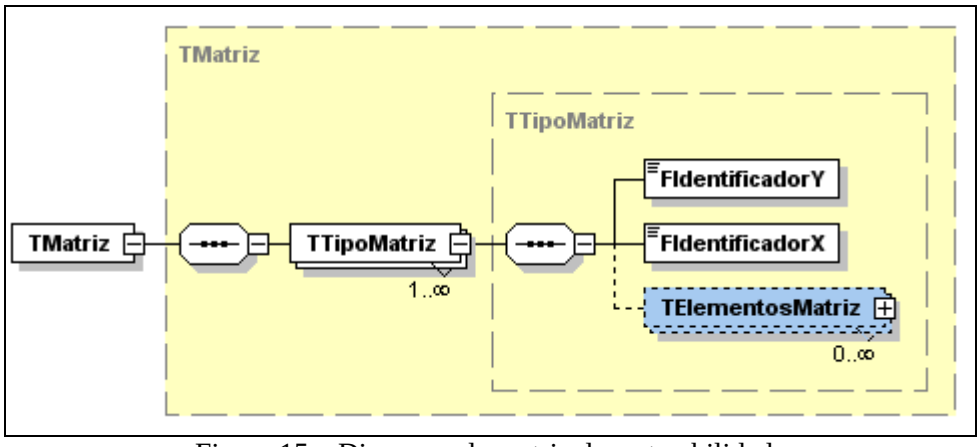



Figura 15 – Diagrama da matriz de rastreabilidade

A Figura 16 apresenta o diagrama dos elementos da matriz de rastreabilidade do documento de ERS e sua estrutura.

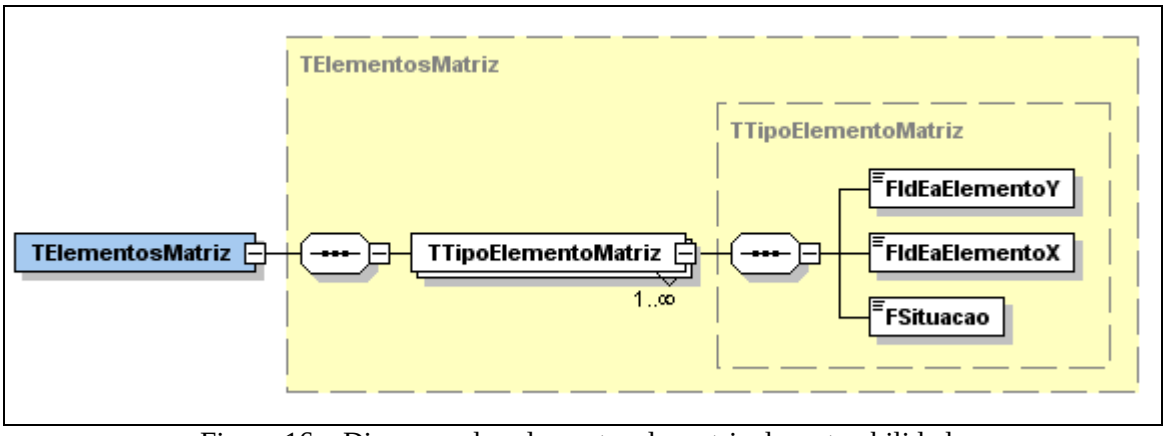



Figura 16 – Diagrama dos elementos da matriz de rastreabilidade

Adicionalmente, a estrutura do arquivo XSD gerada através dos diagramas pode ser consultada no Apêndice B.

### 3.2.4 XML Mapper

Segundo Borland Software Corporation (2001), XML Mapper é utilizado para mapear um XSD e criar um *data packet* para ser utilizado em num c*lient dataset*. Um *data packet* é um arquivo no formato XML que contém a estrutura de um arquivo XSD, porém, esta estrutura é específica para utilizar num *client dataset*. Um *client dataset* é um componente disponível no software Borland Delphi 6.0 que permite manipular dados a partir de um arquivo, como por exemplo, um documento XML que contenha uma estrutura de dados *data packet*. Além de conter a estrutura dos dados, o *data packet* persiste os dados no próprio arquivo (BORLAND SOFTWARE CORPORATION, 2001).

Na Figura 17 é apresentada a ferramenta XML Mapper para geração do *data packet*. A estrutura do *data packet* está no Apêndice C.

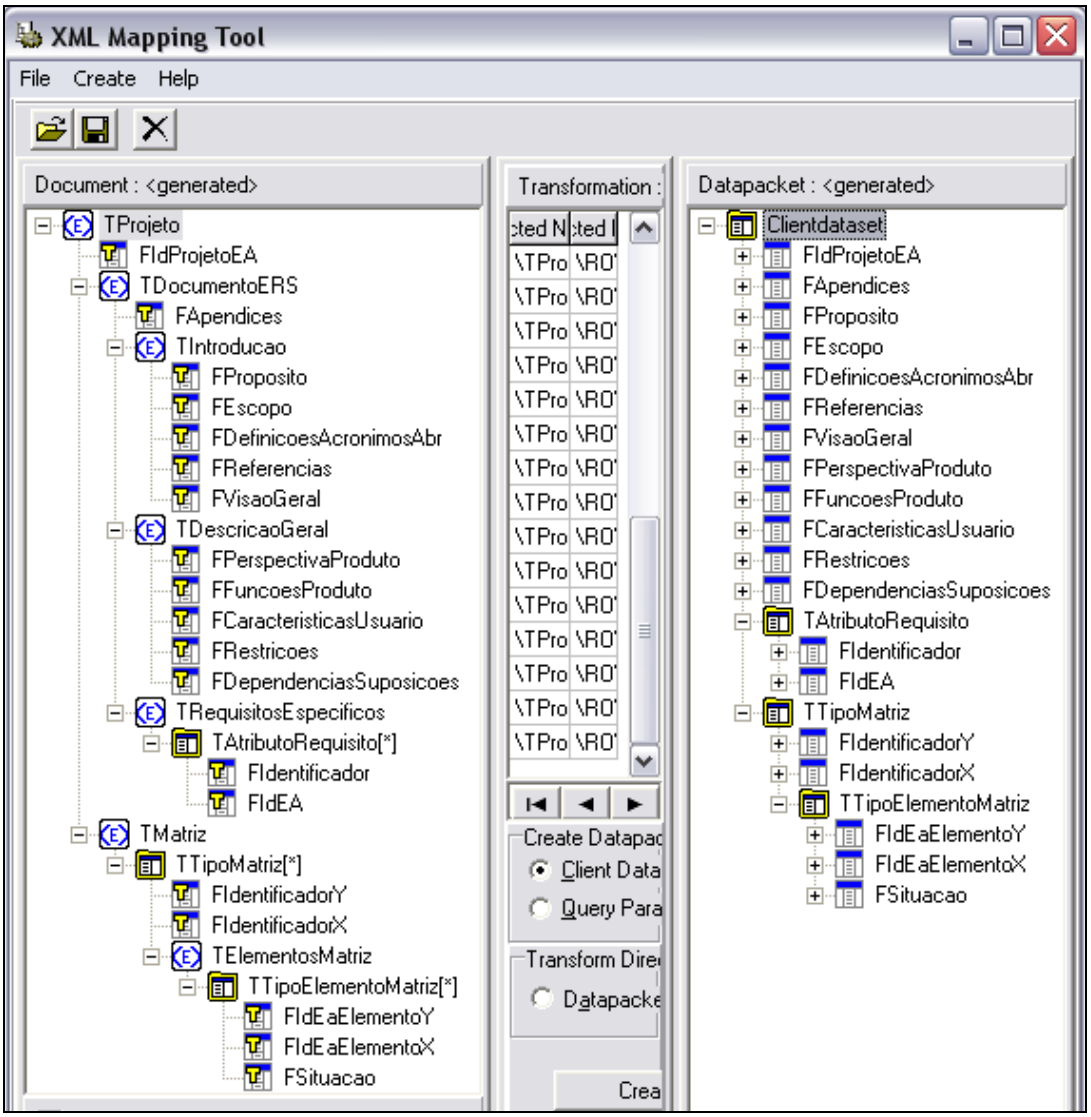



Figura 17 – XML Mapper

## 3.3 IMPLEMENTAÇÃO

Esta seção apresenta as técnicas e ferramentas utilizadas para o desenvolvimento da ferramenta, bem como a descrição da operacionalidade.

### 3.3.1 Técnicas e ferramentas utilizadas

Para implementação da ferramenta foi utilizado o software Borland Delphi 6. O *data packet* contendo a estrutura do XSD foi criado a partir da ferramenta XML Mapper. Foi utilizado o componente *client dataset* disponível do Borland Delphi 6 para acessar os dados do *data packet*.

O acesso ao repositório do EA foi realizado através da biblioteca de interface de automação criada a partir do Borland Delphi 6. Em seguida são apresentadas as etapas de criação, acesso e leitura dos dados do repositório do EA através da interface a de automação.

No Quadro 5 é apresentado o código que instancia a variável do tipo `TRepository`, que contém todos os elementos do repositório do EA.

```
constructor TProjeto.Create(AOwner : TTreeView);
begin
    try
        DMbase := TDMbase.Create(Application.Owner);
        FRepositorioEA         := TRepository.Create(nil); // repositorio
        FDocumentoERS          := TDocumentoERS.Create(AOwner);
        FMatriz                := TMatriz.Create;
        FListaTipoRequisito    := TListaTipoRequisito.Create;
        FListaTipoEstado       := TListaTipoEstado.Create;
        FIdProjetoEA           := GetIdProjetoEA;
        FAlertas               := TListaAlerta.Create;
        FDocumentoERS.FProjeto := Self;
        FMatriz.FProjeto       := Self;
    except
        raise;
    end;
end;
```

Quadro 5 – Instanciando repositório do EA

A abertura do arquivo do projeto do EA e a chamada do procedimento que carrega os requisitos na seção requisitos específicos do documento de ERS são apresentados no Quadro 6.

```
procedure TProjeto.ConectaProjetoEA(const p_nomarq: String);
begin
    try
        FRepositorioEA.OpenFile(p_nomarq);
        FConectadoEA := True;
        DMbase.ConnectionEA.Open;
    except
        erro('O arquivo especificado não existe. Conexão com EA falhou.');
    end;

    ValidaVinculoEA;
    CarregaTipoRequisito;
    CarregaTipoEstado;
    FDocumentoERS.FRequisitosEspecificos.CarregaRequisitosEA(FRepositorioEA);
    CarregaXML;
end;
```

Quadro 6 – Conectando ao repositório do EA

O código do Quadro 7 implementa a varredura nos elementos do EA, bem como a chamada para geração da árvore de requisitos. Destaca-se no código a seguir, a utilização do comando da interface de automação `QueryInterface`. Este comando retorna para a variável `l_p` um ponteiro da instância do objeto da interface especificada por `IID_IDualPackage`, que representa um elemento do tipo pasta no EA.

```
procedure TRequisitosEspecificos.CarregaRequisitosEA(const p_repositorioEA : TRepository);
var i : Integer;
    l_u : IUnknown;
    l_p : IDualPackage;
begin
    FArvoreRequisitos.Items.Item[c_noPai].DeleteChildren;
    try
        for i:=0 to Pred(p_repositorioEA.Models.Count) do begin
            l_u := p_repositorioEA.Models.GetAt(i);
            l_u.QueryInterface(IID_IDualPackage, l_p);

            GeraArvore(FnoPai, l_p);
        end;
    finally
        FArvoreRequisitos.FullCollapse;
        FArvoreRequisitos.Refresh;
    end;
end;
```

Quadro 7 – Varrendo a árvore de elementos do EA

Os elementos identificados como requisitos são inseridos na árvore de requisitos, conforme apresentado no Quadro 8. Destaca-se no código a seguir, a leitura dos elementos contidos no parâmetro `p_elementos` do tipo `IDualCollection`, que representa todos os elementos da interface de automação no EA.

```
procedure TRequisitosEspecificos.GeraElementosArvore(const p_noPai : TTreeNode
                                                     ; const p_elementos : IDualCollection);
var i : Integer;
    l_u : IUnknown;
    l_e : IDualElement;
begin
    for i:=0 to Pred(p_elementos.Count) do begin
        l_u := p_elementos.GetAt(i);
        if  l_u.QueryInterface(IID_IDualElement, l_e) = 0 then
            if  l_e.Type_ = c_tipo_requisito then begin
                FRequisito := TAtributoRequisito.Create;
                with FRequisito do begin
                    FIdEA              := l_e.ElementID;
                    FDescricao         := l_e.Name;
                    ...
                    ...
                    ...
                end;
                InsereElementoArvore(p_noPai
                                   , l_e.Name
                                   , c_imagem_requisito
                                   , FRequisito);
            end;
    end;
end;
```

Quadro 8 – Criando a árvore de requisitos através da interface de automação

### 3.3.2 Operacionalidade da implementação

Nesta seção é apresentado um estudo de caso para demonstrar as funcionalidades da ferramenta, para um melhor entendimento, cada etapa é mostrada com uma ilustração correspondente a tela da ferramenta. Para este estudo de caso, foi utilizada a própria ferramenta como exemplo.

Ao abrir a ferramenta um novo documento de ERS é iniciado. Neste momento, a ferramenta não está conectada a nenhum projeto do EA. O usuário poderá iniciar a digitação das seções do documento de ERS, exceto a seção de Requisitos Específicos, que obriga que o usuário esteja conectado a um projeto do EA. Caso o usuário optar em abrir um projeto do EA existente, a ferramenta automaticamente carregará o documento de ERS. Ao abrir o projeto do EA, a ferramenta realiza uma verificação de integridade das informações entre os dados armazenados pela ferramenta e os dados do banco de dados do EA, se houver divergências, mensagens de alerta são listadas na aba Alertas.  A Figura 18 ilustra as mensagens de alerta.


Figura 18 – Verificar alertas

A Figura 19 apresenta as seções do documento a serem cadastradas. A navegabilidade entre as seções pode ser feita de duas maneiras, através da árvore hierárquica do documento ou através das abas e botões contidas na aba Documento ERS. Para ambas das opções, basta clicar sobre a aba para posicionar na seção desejada. A formatação do texto pode ser feita através da barra de ferramenta de formatação, localizada dentro da aba Documento ERS.

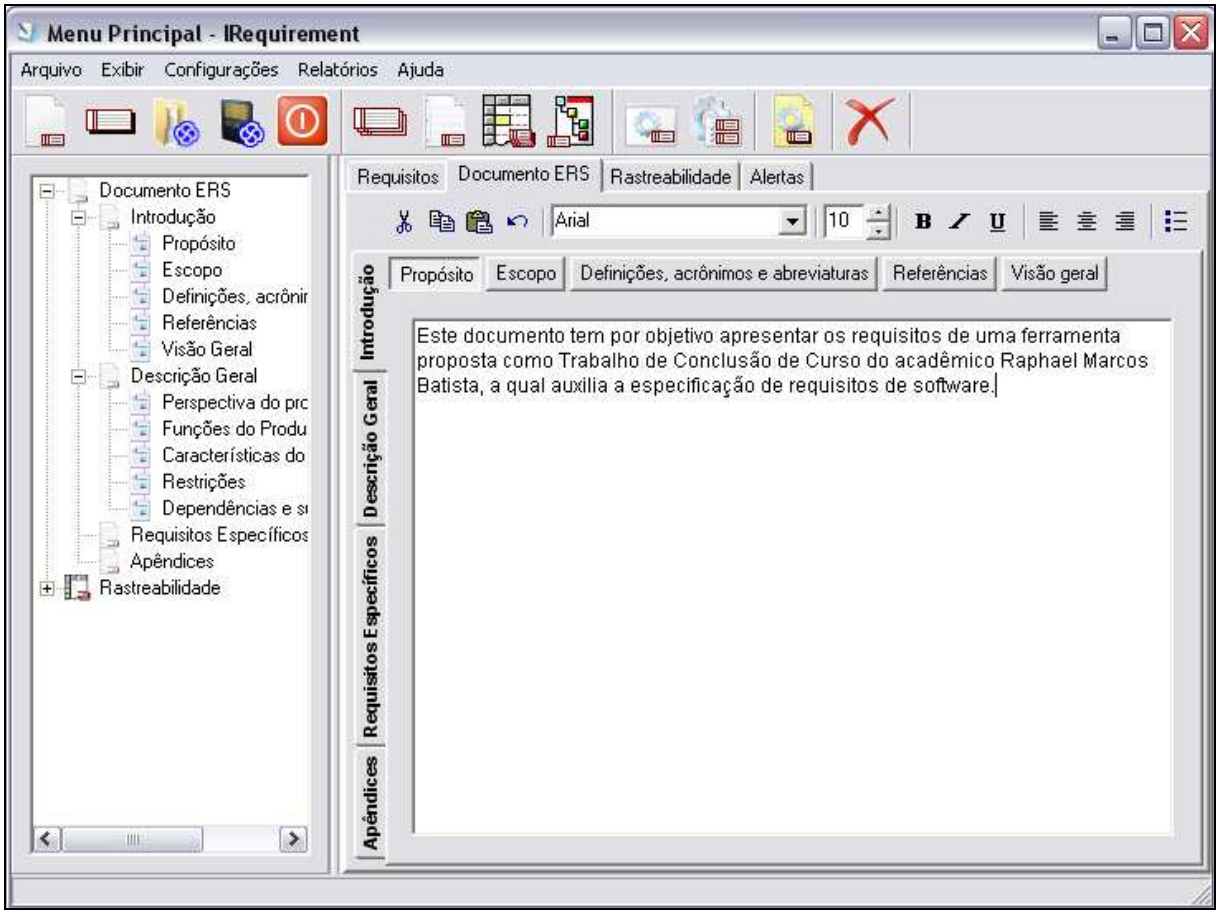



Figura 19 – Cadastrar as seções do documento de ERS

Novos tipos de requisito podem ser cadastrados na ferramenta para auxiliar na identificação dos requisitos. Os tipos de requisito cadastrados estarão concomitantemente disponíveis na ferramenta EA. Para acessar este cadastro, deve ser selecionado no menu configurações a opção Tipos de Requisito. A Figura 20 ilustra os tipos de requisito cadastrados.

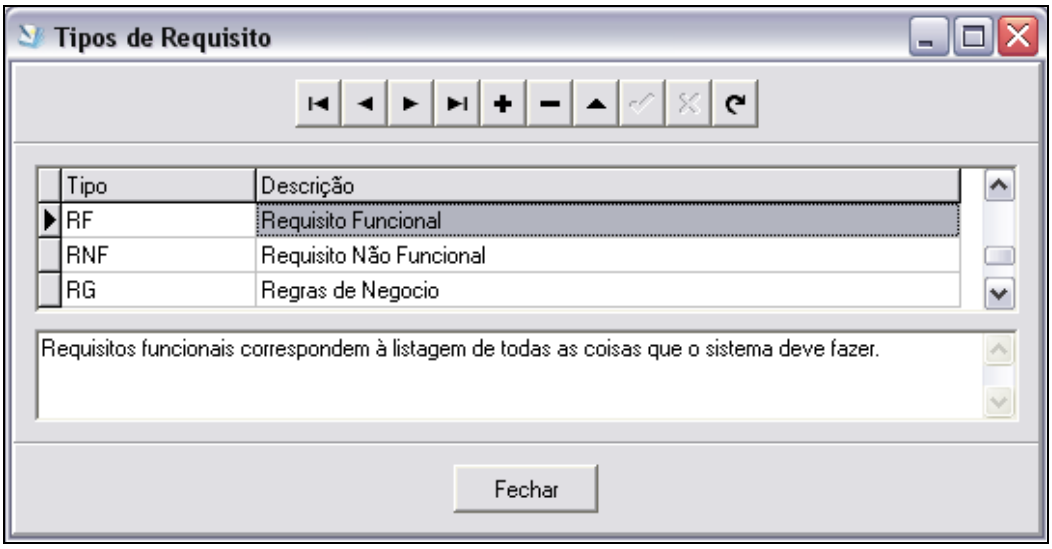



Figura 20 – Cadastrar os tipos de requisitos

No cadastro de tipos de estado é permitido ao usuário definir valores para auxiliar no gerenciamento do projeto, conforme apresentado na Figura 21. Os tipos de estado cadastrados estarão concomitantemente disponíveis na ferramenta EA.

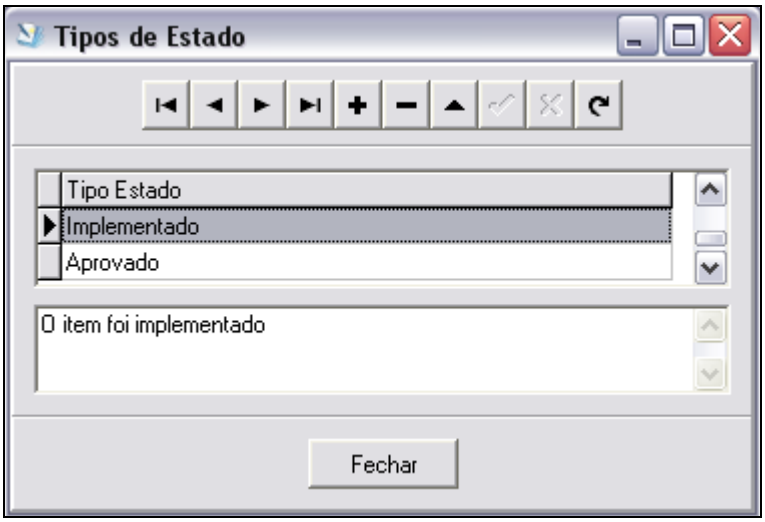



Figura 21 – Cadastrar os tipos de estado

Os requisitos cadastrados recebem um número seqüencial e único por tipo de requisito. O tipo do requisito concatenado com o identificador representará o requisito na matriz de rastreabilidade e no documento de ERS. Os demais atributos devem ser preenchidos para completar as informações do requisito, conforme demonstrado na Figura 22.

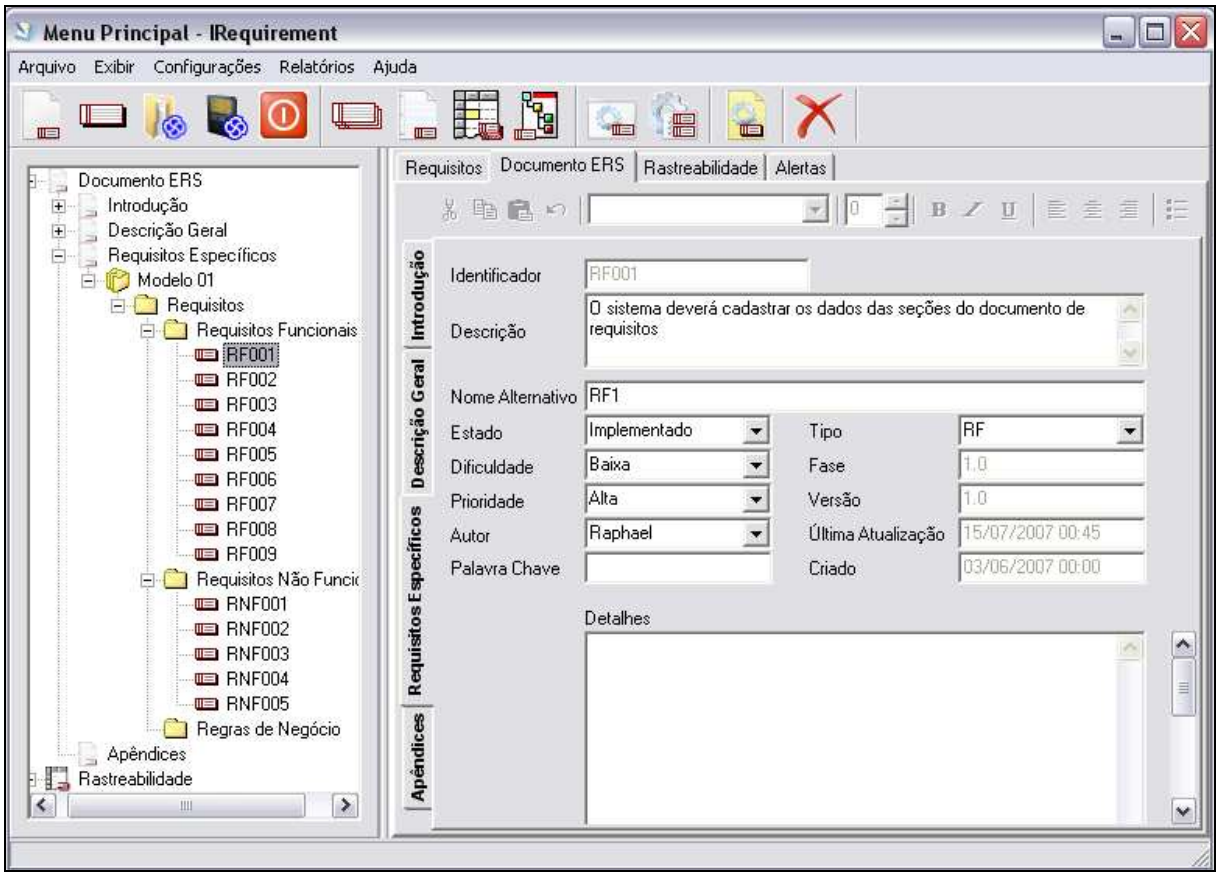



Figura 22 – Cadastrar os requisitos

A ferramenta dispõe de uma funcionalidade que possibilita a configuração de diversos tipos de matriz de rastreabilidade (Figura 23). É permitido configurar a referência cruzada entre os tipos de requisito e caso de uso. Tanto na configuração da linha quanto da coluna, a lista de tipos de requisito e a opção de seleção do caso de uso são mutuamente exclusivas. A opção de seleção por caso de uso importará todos os casos de uso definidos no repositório do EA.

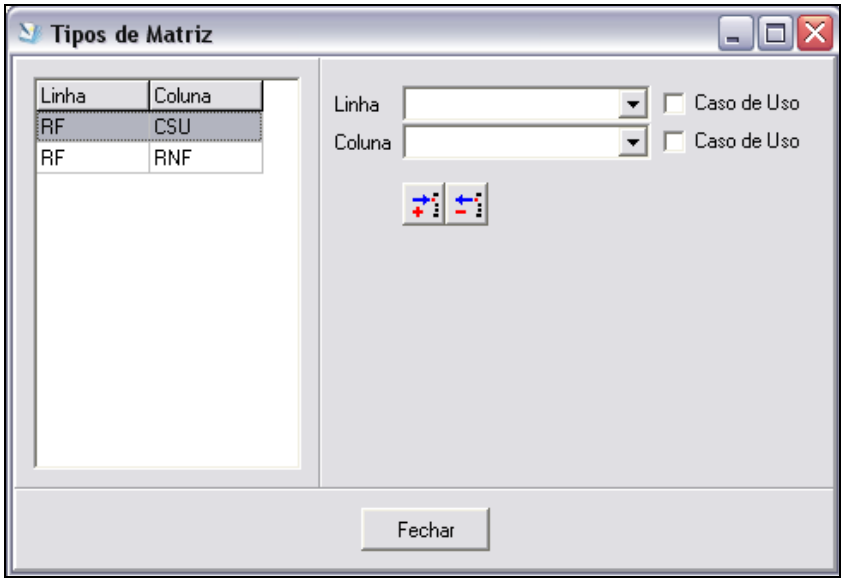



Figura 23 – Tipos de Matriz

Na matriz de rastreabilidade os requisitos e casos de uso são visualizados nas linhas e colunas de acordo com o tipo de matriz configurada pelo usuário. A descrição do requisito ou caso de uso selecionado é apresentada logo abaixo da matriz (Figura 24) visando facilitar a utilização da ferramenta.

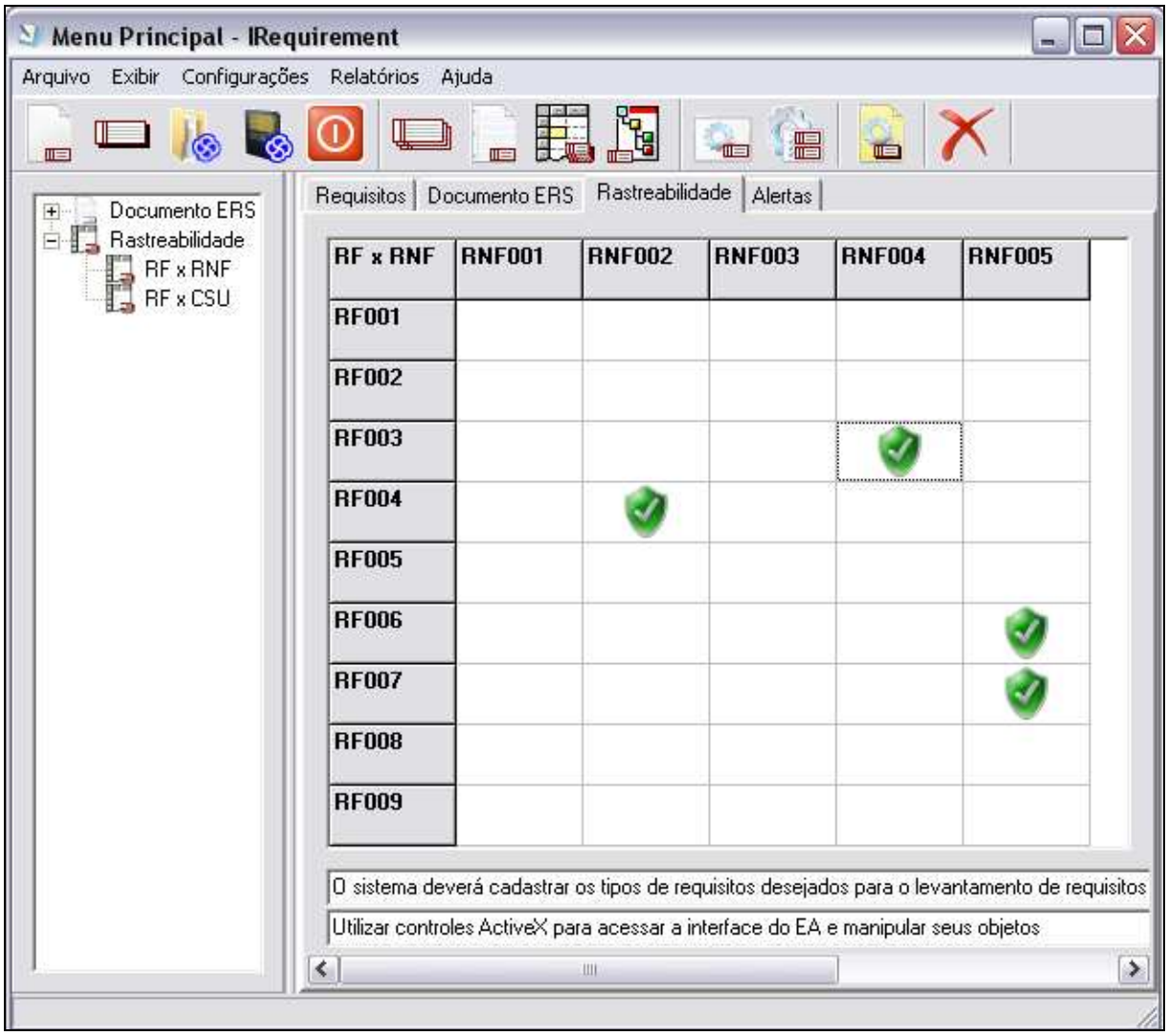



Figura 24 – Relacionar os elementos na matriz de rastreabilidade

Para criar um relacionamento entre os elementos da matriz de rastreabilidade, basta clicar com o botão direito do mouse e escolher a opção vínculo. Uma imagem representativa indicará o vínculo do relacionamento. Para marcar o vínculo como alterado, basta selecionar a opção alterado. Esta opção é automaticamente atribuída ao vínculo quando algum dos elementos vinculados for alterado. Esta funcionalidade não é executada quando o elemento é alterado na ferramenta EA ou se o elemento for um caso de uso. Para desvincular um relacionamento deve-se clicar na opção criado. A Figura 25 ilustra os elementos relacionados na matriz.

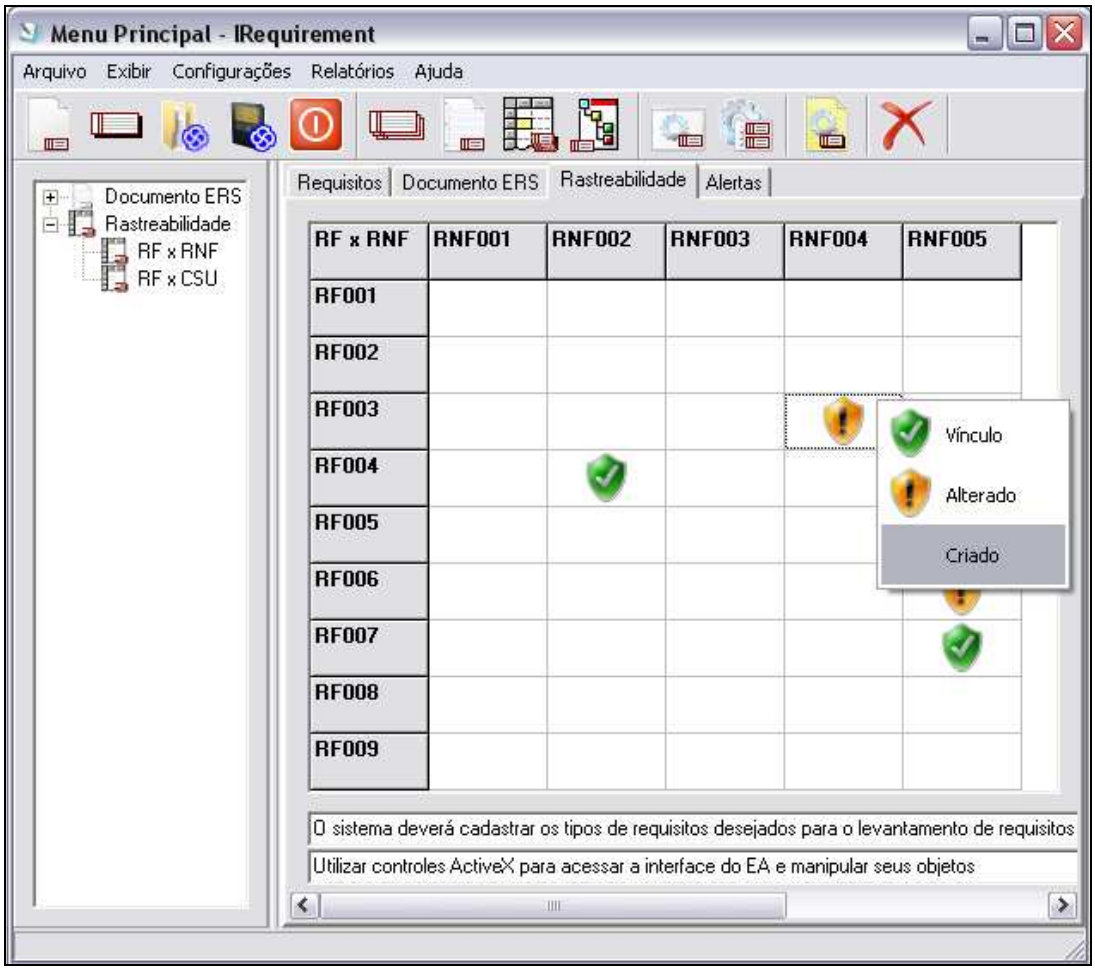



Figura 25 – Elementos relacionados na matriz

Uma listagem completa dos requisitos cadastros pode ser visualizada na aba Requisitos, conforme apresentado na Figura 26.

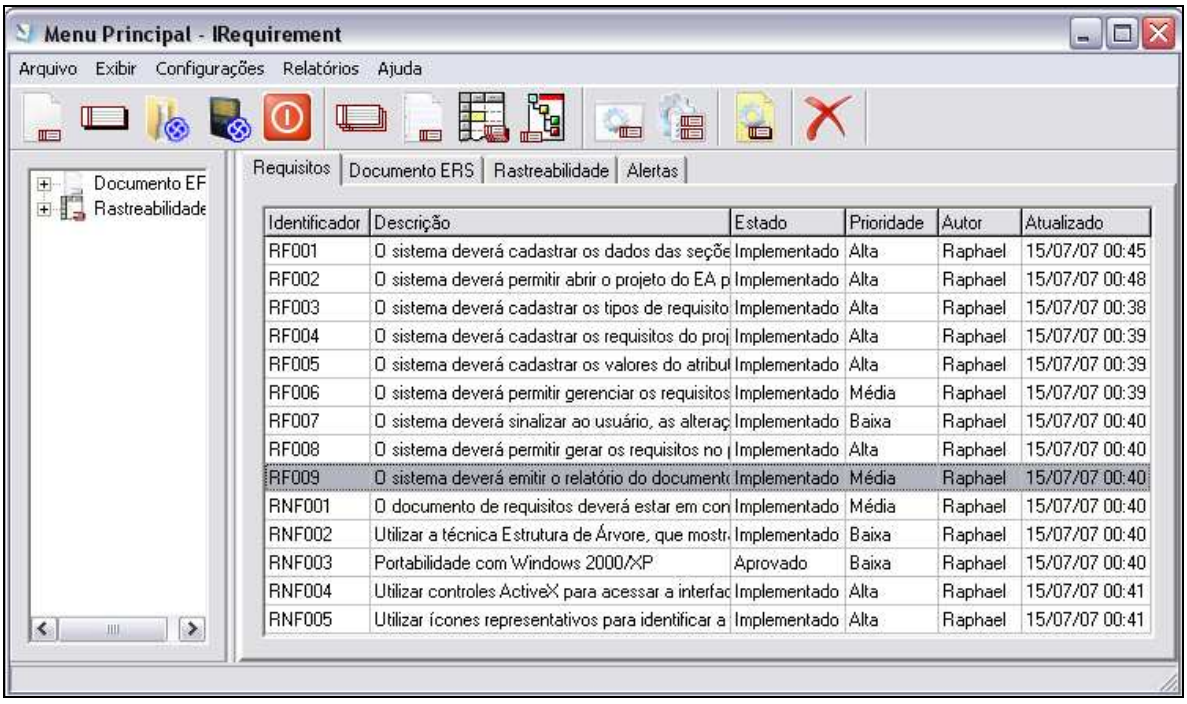



Figura 26 – Visualizar os requisitos cadastrados

O documento de ERS pode ser gerado em diversos formatos (ilustrado na Figura 27). Para isso basta selecionar o tipo do formato na caixa Imprimir para Arquivo, informado também o caminho e o nome do arquivo onde será gerado o documento.

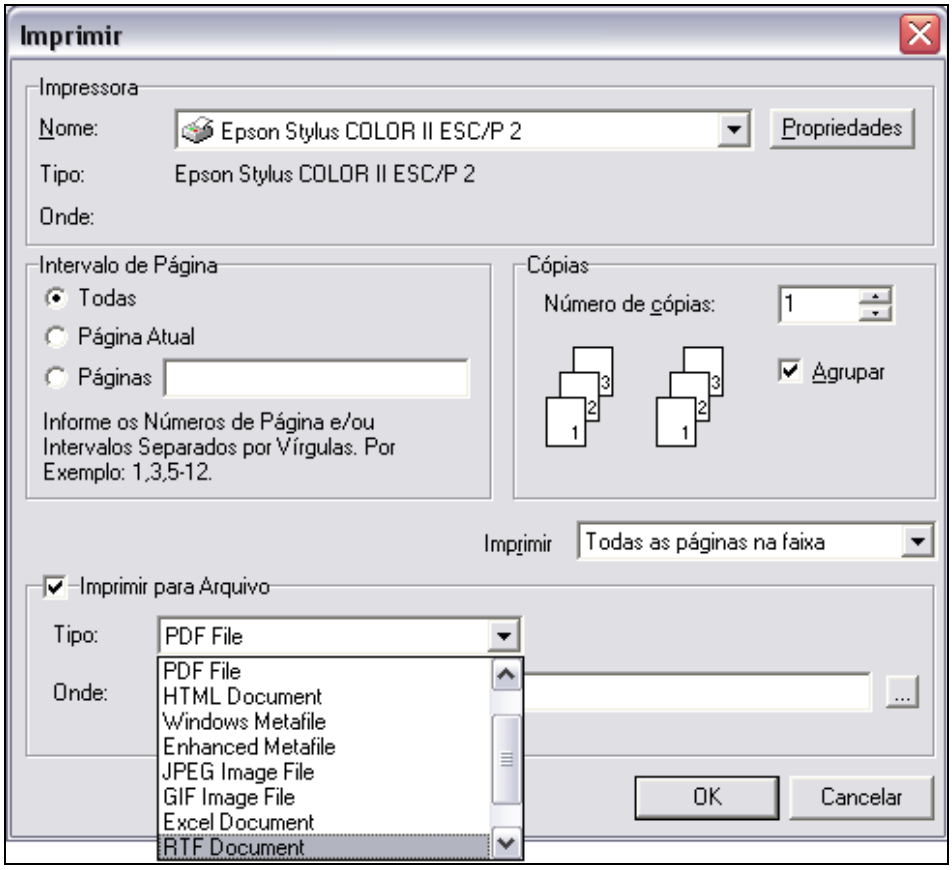



Figura 27 – Gerar o documento de ERS

No Apêndice D encontra-se um exemplo do documento de ERS gerado pela ferramenta, no formato JPEG.

## 3.4 RESULTADOS E DISCUSSÃO

Em relação aos trabalhos correlatos apresentados na fundamentação teórica, o Quadro 9 apresenta o comparativo entre a ferramenta desenvolvida e os trabalhos correlatos.

| Ferramenta / Característica | Documentação de Requisitos | Requisitemanager | SPRES | RaQuest | IRequirement |
|---|---|---|---|---|---|
| Rastreabilidade dos Requisitos | Não | Sim | Sim | Sim | Sim |
| Finalidade de Uso | Acadêmica | Acadêmica | Acadêmica | Comercial | Acadêmica |
| Padrão do documento de ERS | Próprio | IEEE-830-1993 | IEEE-830-1998 e RUP | Próprio | IEEE-830-1998 |
| Plataforma | Windows | Web | Windows | Windows | Windows |
| Integração com Ferramentas CASE | Não | Não | Não | Sim | Sim |
| Histórico de Alterações | Não | Sim | Sim | Sim | Não |

Quadro 9 – Comparativo entre as ferramentas

Esta ferramenta alcançou seus objetivos e agregou, em relação as ferramentas correlatas, as principais características e funcionalidades que uma ferramenta de gerenciamento de requisitos deve possuir, tais como:

a) rastreabilidade dos requisitos: através da matriz de rastreabilidade;
b) documento de ERS: em conformidade com o modelo IEEE-830-1988;
c) integração com ferramentas CASE: integrada com a ferramenta Enterprise Architect;
d) histórico de alterações: não possui histórico de alterações dos requisitos.

A ferramenta permite que o usuário configure de maneira flexível os vários tipos de matriz de rastreabilidade. Isso é alcançado através da alternância dos tipos de requisito e casos de uso na linha e coluna da matriz. Além desta facilidade de configuração, também é possível identificar os elementos relacionados na matriz que possuem vínculo e os que sofreram algum tipo de alteração em seus atributos, auxiliando a análise de impacto e o gerenciamento de mudanças no projeto de software.

O modelo IEEE-830-1998 apresenta um conjunto de práticas para a confecção de um documento de ERS a fim de realizar uma especificação de requisitos de software de boa qualidade. Este modelo foi adotado por esta ferramenta. O modelo de documento de ERS desta ferramenta é organizado em seções e subseções, em cada umas delas o usuário descreve os itens identificados da etapa de especificação em questão. Por exemplo, na subseção propósito é delimitado o propósito do documento de ERS. O modelo de documento adotado pela ferramenta fica em conformidade com o definido na cláusula 5 do modelo IEEE-830-1998 que descreve as partes do documento de ERS que foram apresentas na subseção 2.2 deste trabalho. A seção índice não foi implementada por estar definida no modelo como uma seção opcional no documento de ERS.

Além de diretrizes quanto ao formato de um documento de ERS, o modelo IEEE-830-

1998 também discorre sobre características desejáveis em um requisito, como por exemplo, correto, não ambíguo, completo, consistente, dentre outros. Em relação a estes aspectos citados pela IEEE, as subseções apresentadas na cláusula 5 que definem a forma de especificação das subseções do documento de ERS, referente as seções de introdução, descrição geral e requisitos específicos, não foram incorporadas nesta ferramenta, visto que são informações de livre contexto, em que somente o analista de sistemas com base em seus conhecimentos técnicos será capaz de formular e descrever tais especificações.

Ainda no IEEE-830-1998 é apresentada uma série de modelos para organizar a seção de requisitos específicos em um documento de ERS. O modelo adotado por esta ferramenta é semelhante ao modelo apresentado na seção A.1 do documento. De modo geral, este modelo é organizado por tipo de requisito, que pode ser entre outros, requisito funcional, não funcional ou regra de negócio. Este modelo foi organizado de forma que para cada requisito especificado no documento de ERS, é listada a descrição, o atributo de estado, o atributo de prioridade e a rastreabilidade do requisito. Na rastreabilidade são listados os requisitos vinculados ao requisito.

Uma vantagem no momento de especificar os requisitos é a possibilidade de importar e exportar os requisitos da ferramenta EA. Esta vantagem abrange e enaltece o gerenciamento de requisitos de software.

A integração desta ferramenta com o EA vai ao encontro do conceito de I-CASE , que proporciona ao usuário uma interface própria para o gerenciamento de requisitos e, concomitantemente, a utilização de todas as funcionalidades da ferramenta EA baseada na UML para modelagem de um projeto de software. Para garantir a fidelidade das informações, foi criada nesta ferramenta uma área de alertas, que exibe de forma transparente as alterações feitas nos requisitos e casos de uso através da ferramenta EA, desde que estejam vinculados a esta ferramenta.

Para adequar esta ferramenta ao conceito de gerenciamento de mudanças, no que se refere a análise do problema e especificação da mudança, é necessário a implementação de um controle do histórico de alterações dos requisitos, bem como das seções do documento de ERS. Visto que o foco desta ferramenta está no gerenciamento dos requisitos e possui integração com o EA, o gerenciamento de mudanças no que se refere ao custo e implementação da mudança (apresentado na subseção 2.1.3.1) se fará na ferramenta EA, que contempla os recursos necessários para esta etapa, como por exemplo, permitindo integração com ferramenta de controle de versão (a citar, *Concurrent Version System* (CVS)).

# 4 CONCLUSÕES

O estudo realizado da ferramenta EA sobre a interface de automação possibilitou a criação desta ferramenta CASE integrada com EA. A possibilidade de criar e acessar a biblioteca de automação através do Borland Delphi 6 utilizando controles ActiveX foi essencial para desenvolvimento da ferramenta.

O conceito de I-CASE aplicado nesta ferramenta permite que os requisitos criados no EA e mantidos no seu repositório possam ser manipulados através desta ferramenta. Esta ferramenta veio suprir a pouca flexibilidade que o EA possui na etapa de especificação de requisitos de software. O projeto da interface desta ferramenta, que é específico para o gerenciamento de requisitos, permite maior produtividade no gerenciamento de requisitos em relação a ferramenta EA.

As definições da engenharia de requisitos foram incorporadas nesta ferramenta a fim de proporcionar o êxito na especificação de requisitos de software. Um documento de ERS é gerado em formato RTF e pode ser salvo em arquivo neste formato ou em outros formatos, como por exemplo, PDF e HTML.

O documento de ERS está em conformidade com o modelo IEEE-930-1998. A possibilidade de criar novos tipos de estado atende as especificações do modelo do documento de ERS da IEEE-930-1998.

Embora o EA forneça uma biblioteca completa para acesso a seu repositório, algumas informações e recursos adicionados nesta ferramenta não puderam ser incorporados ao repositório do EA. Esta limitação foi contornada utilizando um arquivo em formato XML para manter as informações e atender aos objetivos deste trabalho. As informações das seções e subseções do documento de ERS são salvas neste arquivo XML, bem como as matrizes de rastreabilidade. Nesta etapa foram utilizadas técnicas e ferramentas específicas para modelagem e criação da estrutura de arquivo no formato XML.

A rastreabilidade entre requisitos e casos de uso através da matriz de rastreabilidade permite realizar o gerenciamento de mudanças no projeto de software. Cada mudança é representada no relacionamento entre os elementos da matriz.

Ferramentas similares de gerenciamento de requisitos foram analisadas e seus aspectos avaliados a fim de proporcionar um diferencial competitivo para esta ferramenta.

Os conceitos e técnicas empregados neste trabalho proporcionaram o desenvolvimento desta ferramenta atendendo com sucesso aos objetivos propostos.

## 4.1 EXTENSÕES

Para implementações futuras sugere-se o desenvolvimento focado na segurança dos usuários da ferramenta através de controle de acesso e histórico de alterações. No cadastro de usuário poderia definir os direitos de acesso ao documento de ERS, como por exemplo, direito de incluir, alterar e excluir um requisito de software ou seção do documento, assim como manter um histórico da navegação do usuário pelo documento e as operações que ele realizou.

Outra sugestão é a implementação de uma funcionalidade semelhante a um fórum de discussão, para cada requisito cadastrado, o usuário poderá postar seus questionamentos e sugestões sobre o objetivo e finalidade do qual o requisito se destina, este questionamento também poderá ser respondido por outros usuários.

Também sugere-se o estudo da *Information Technology Infrastructure Library* (ITIL) visando adequar a ferramenta desenvolvida para apoiar ao processo de gerência de mudança proposto pelo ITIL.

# REFERÊNCIAS BIBLIOGRÁFICAS

BORLAND SOFTWARE CORPORATION. **Delphi enterprise**: help. Version 6.0. [S.l.], 2001. Documento eletrônico disponibilizado com o Ambiente Delphi 6.0.

CASTRO, Jaelson Freire Brelaz de; GIMENES, Itana Maria de Souza; MALDONADO, José Carlos. Uma proposta de plano pedagógico para a matéria engenharia de software. In: CURSO DE QUALIDADE DE CURSOS DE GRADUAÇÃO DA ÁREA DE COMPUTAÇÃO E INFORMÁTICA, 2., 2000, Curitiba. **Anais...** Curitiba: Champagnat, 2000. p. 251-270.

DAVIS, Alan M. **Software requirements:** objects, functions, and states. Englewood Cliffs: Prentice Hall, 1993. 521 p.

FIORINI, Soeli T; STAA, Arndt von; BAPTISTA, Renan Martins. **Engenharia de software com CMM.** Rio de Janeiro: Brasport, 1998. 346 p.

INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS. **Recommended practice for software requirements specifications**. Revision 830. New York: IEEE: 1998.

JOSÉ, Odair. **Ferramenta de apoio a documentação de requisitos de software**. 2002. 70 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

KOTONYA, Gerald; SOMMERVILLE, Ian. **Requirements engineering:** processes and techniques. Chichester: John Wiley & Sons, 1998. 282 p.

MARQUARDT, Luciano. **Ferramenta web para gerenciamento de requisitos de software**. 2004. 86 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) – Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

PAULA FILHO, Wilson de Padua. **Engenharia de software:** fundamentos, métodos e padrões. 2. ed. Rio de Janeiro: LTC, 2003. 602 p.

PETERS, James F; PEDRYCZ, Witold. **Engenharia de software:** teoria e prática. Tradução Ana Patrícia Machado de Pinto Garcia. Rio de Janeiro: Campus, 2001. 602 p.

PFLEEGER, Shari Lawrence. **Engenharia de software:** teoria e prática. 2. ed. Tradução Dino Franklin. São Paulo: Prentice Hall, 2004. 537 p.

PRESSMAN, Roger S. **Engenharia de software**. 5. ed. Tradução Mônica Maria G. Travieso. Rio de Janeiro: McGraw-Hill, 2002. 843 p.

QUIRINO, Julio Cesar Chrystopher. **Ferramenta CASE para especificação de requisitos de software observando IEEE-830-1998 e RUP**. 2004. 102 f. Trabalho de Conclusão de Curso (Graduação em Ciência da Computação) – Centro de Ciências Tecnológicas da Terra e do Mar, Universidade do Vale do Itajaí, Itajaí.

RAQUEST. **RaQuest purchase**. [S.l.], 2007. Disponível em: <www.raquest.com/products/raq_purchase.htm>. Acesso em: 18 fev. 2007.

SOMMERVILLE, Ian. **Engenharia de software**. 6. ed. Tradução Maurício de Andrade. São Paulo: Addison Wesley, 2003. 592 p.

SPARX SYSTEMS. **Enterprise Architect**: user guide. Version 6.1. [S.l.], 2005. Documento eletrônico disponibilizado com o Ambiente Enterprise Architect 6.1.

WORLD WIDE WEB CONSORTIUM. **Extensible Markup Language (XML)**. [S.l.], 2007. Disponível em: <www.w3.org/XML/>. Acesso em: 27 maio 2007.

## APÊNDICE A – Descrição dos atributos e principais operações das classes do diagrama de classes

Os atributos e principais operações das classes do diagrama de classes são descritos abaixo:

a) `TProjeto`: os principais atributos e operações desta classe são:
   - `FRepositorioEA`: mantém os elementos do repositório do EA obtidos através da biblioteca de interface de automação, como por exemplo, a classe contendo todos os requisitos definidos no EA,
   - `FIdProjetoEA`: contém o número seqüencial e único que relaciona o projeto do EA com o documento XML do projeto correspondente,
   - `Create()`: propaga o *control* de interface do tipo *`TTreeView`* para as classes filhas,
   - `ConectaProjetoEA`: conecta ao repositório de um projeto do EA,
   - `DesconetaProjetoEA`: salva as alterações e desconecta do repositório do projeto do EA;

b) `TDocumentoERS`: os principais atributos desta classe são:
   - `FApendices`: mantém o texto correspondente ao apêndice do documento de ERS,
   - `Create()`: propaga o *control* de interface do tipo *`TTreeView`* para as classes filhas;

c) `TIntroducao`: os principais atributos desta classe são:
   - `FPropósito`: mantém o texto correspondente ao propósito do documento de ERS,
   - `FEscopo`: mantém o texto correspondente ao escopo do documento de ERS,
   - `FDefinicoesAcronimosAbr`: mantém o texto correspondente as definições, acrônimos e abreviaturas do documento de ERS,
   - `FReferencias`: mantém o texto correspondente as referências do documento de ERS,
   - `FVisaoGeral`: mantém o texto correspondente a visão geral do documento de ERS;

d) `TDescricaoGeral`: os principais atributos desta classe são:
   - `FPerspectivaProduto`: mantém o texto correspondente a perspectiva do

produto do documento de ERS,

- `FFuncoesProduto`: mantém o texto correspondente as funções do produto do documento de ERS,
- `FCaracteristicasUsuario`: mantém o texto correspondente as características do usuário do documento de ERS,
- `FRestricoes`: mantém o texto correspondente as restrições do documento de ERS,
- `FDepenciasSuposicoes`: mantém o texto correspondente as dependências e suposições do documento de ERS;

e) `TRequisitosEspecificos`: os principais atributos desta classe são:

- `FnoPai`: número do nodo pai correspondente a seção requisitos do documento de ERS,
- `FArvoreRequisitos`: contém a estrutura em hierarquia dos requisitos definidos no EA,
- `Create()`: cria uma estrutura de árvore e atribui o nodo raiz,
- `CarregaRequisitosEA()`: carrega na árvore de forma hierárquica todos os requisitos definidos no repositório do EA;

f) `TAtributoRequisito`: os principais atributos desta classe são:

- `FIdentificador`: mantém o número seqüencial e único do requisito,
- `FDescricao`: mantém a descrição do requisito,
- `FNomeAlternativo`: mantém a descrição breve do requisito,
- `FEstado`: mantém o estado do requisito,
- `FTipo`: mantém o tipo do requisito,
- `FDificuldade`: mantém a dificuldade do requisito,
- `FFase`: mantém a fase do requisito,
- `FPrioridade`: mantém a prioridade do requisito,
- `FVersao`: mantém a versão do requisito,
- `FAutor`: mantém o nome do autor do requisito,
- `FUltimaAtualizacao`: mantém a data da última atualização do requisito,
- `FPalavraChave`: mantém a palavra chave do requisito,
- `FCriado`: mantém a data de criação do requisito,
- `FDetalhes`: mantém o detalhamento do requisito,
- `FIdEA`: mantém o ID do requisito definido no repositório do EA;

g) `TTipoMatrix`: os principais atributos desta classe são:
   - `FIdentificadorY`: mantém o identificador do tipo do requisito correspondente no EA,
   - `FIdentificadorX`: mantém o identificador do tipo do requisito correspondente no EA;

h) `TTipoElementoMatrix`: os principais atributos desta classe são:
   - `FIdEaElementoY`: mantém o ID do elemento definido no repositório do EA,
   - `FIdEaElementoX`: mantém o ID do elemento definido no repositório do EA,
   - `FSituacao`: mantém a situação do relacionamento do elemento, quando existir o relacionamento e/ou algum elemento for alterado, é exibido um ícone representativo para indicar a situação que pode ser criado, alterado ou vínculo;

i) `TTipoEstado`: os principais atributos desta classe são:
   - `FEstado`: mantém o estado do requisito,
   - `FDescricao`: mantém a descrição do estado do requisito;

j) `TTipoRequisito`: os principais atributos desta classe são:
   - `FIdentificador`: mantém o identificador do tipo do requisito,
   - `FDescricao`: mantém a descrição do tipo do requisito,
   - `FDetalhes`: mantém a descrição detalhada do tipo do requisito.

**APÊNDICE B – XML Schema Definition**

O Quadro 10 apresenta o *XML Schema Defination* gerado através da ferramenta XMLSpy.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- edited with XMLSpy v2007 sp2 (http://www.altova.com) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
	<xs:element name="TProjeto" type="TProjeto"/>
	<xs:complexType name="TProjeto">
		<xs:sequence>
			<xs:element name="FIdProjetoEA" type="xs:integer"/>
			<xs:element name="TDocumentoERS" type="TDocumentoERS"/>
			<xs:element name="TMatriz" type="TMatriz" minOccurs="0"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="TAtributoRequisito" type="TAtributoRequisito"/>
	<xs:complexType name="TAtributoRequisito">
		<xs:sequence>
			<xs:element name="FIdentificador" type="xs:integer"/>
			<xs:element name="FIdEA" type="xs:integer"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="TIntroducao" type="TIntroducao"/>
	<xs:complexType name="TIntroducao">
		<xs:sequence>
			<xs:element name="FProposito" type="xs:string"/>
			<xs:element name="FEscopo" type="xs:string"/>
			<xs:element name="FDefinicoesAcronimosAbr"
type="xs:string"/>
			<xs:element name="FReferencias" type="xs:string"/>
			<xs:element name="FVisaoGeral" type="xs:string"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="TDescricaoGeral" type="TDescricaoGeral"/>
	<xs:complexType name="TDescricaoGeral">
		<xs:sequence>
			<xs:element name="FPerspectivaProduto" type="xs:string"/>
			<xs:element name="FFuncoesProduto" type="xs:string"/>
			<xs:element name="FCaracteristicasUsuario"
type="xs:string"/>
			<xs:element name="FRestricoes" type="xs:string"/>
			<xs:element name="FDependenciasSuposicoes"
type="xs:string"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="TRequisitosEspecificos"
type="TRequisitosEspecificos"/>
	<xs:complexType name="TRequisitosEspecificos">
		<xs:sequence>
			<xs:element name="TAtributoRequisito"
type="TAtributoRequisito" maxOccurs="unbounded"/>
		</xs:sequence>
	</xs:complexType>
	<xs:element name="TDocumentoERS" type="TDocumentoERS"/>
	<xs:complexType name="TDocumentoERS">
		<xs:sequence>
			<xs:element name="FApendices" type="xs:string"/>
```

```
                  <xs:element name="TIntroducao" type="TIntroducao"/>
                  <xs:element name="TDescricaoGeral"
type="TDescricaoGeral"/>
                  <xs:element name="TRequisitosEspecificos"
type="TRequisitosEspecificos"/>
            </xs:sequence>
      </xs:complexType>
      <xs:element name="TTipoElementoMatriz" type="TTipoElementoMatriz"/>
      <xs:complexType name="TTipoElementoMatriz">
            <xs:sequence>
                  <xs:element name="FIdEaElementoY" type="xs:integer"/>
                  <xs:element name="FIdEaElementoX" type="xs:integer"/>
                  <xs:element name="FSituacao" type="xs:integer"/>
            </xs:sequence>
      </xs:complexType>
      <xs:element name="TElementosMatriz" type="TElementosMatriz"/>
      <xs:complexType name="TElementosMatriz">
            <xs:sequence>
                  <xs:element name="TTipoElementoMatriz"
type="TTipoElementoMatriz" maxOccurs="unbounded"/>
            </xs:sequence>
      </xs:complexType>
      <xs:element name="TTipoMatriz" type="TTipoMatriz"/>
      <xs:complexType name="TTipoMatriz">
            <xs:sequence>
                  <xs:element name="FIdentificadorY" type="xs:string"/>
                  <xs:element name="FIdentificadorX" type="xs:string"/>
                  <xs:element name="TElementosMatriz"
type="TElementosMatriz" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
      </xs:complexType>
      <xs:element name="TMatriz" type="TMatriz"/>
      <xs:complexType name="TMatriz">
            <xs:sequence>
                  <xs:element name="TTipoMatriz" type="TTipoMatriz"
maxOccurs="unbounded"/>
            </xs:sequence>
      </xs:complexType>
</xs:schema>
```

Quadro 10 – *XML Schema Defination*

## APÊNDICE C – Estrutura do *data packet*

O Quadro 11 apresenta a estrutura do *data packet* gerado através da ferramenta XML Mapper.

```
- <DATAPACKET Version="2.0">
  - <METADATA>
    - <FIELDS>
        <FIELD attrname="FIdProjetoEA" fieldtype="i4" />
        <FIELD attrname="FApendices" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FProposito" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FEscopo" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FDefinicoesAcronimosAbr" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FReferencias" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FVisaoGeral" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FPerspectivaProduto" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FFuncoesProduto" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FCaracteristicasUsuario" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FRestricoes" fieldtype="bin.hex" SUBTYPE="Text" />
        <FIELD attrname="FDependenciasSuposicoes" fieldtype="bin.hex" SUBTYPE="Text" />
      - <FIELD attrname="TAtributoRequisito" fieldtype="nested">
        - <FIELDS>
            <FIELD attrname="FIdentificador" fieldtype="i4" />
            <FIELD attrname="FIdEA" fieldtype="i4" />
          </FIELDS>
          <PARAMS />
        </FIELD>
      - <FIELD attrname="TTipoMatriz" fieldtype="nested">
        - <FIELDS>
            <FIELD attrname="FIdentificadorY" fieldtype="string" WIDTH="31" />
            <FIELD attrname="FIdentificadorX" fieldtype="string" WIDTH="31" />
          - <FIELD attrname="TTipoElementoMatriz" fieldtype="nested">
            - <FIELDS>
                <FIELD attrname="FIdEaElementoY" fieldtype="i4" />
                <FIELD attrname="FIdEaElementoX" fieldtype="i4" />
                <FIELD attrname="FSituacao" fieldtype="i4" />
              </FIELDS>
              <PARAMS />
            </FIELD>
          </FIELDS>
          <PARAMS />
        </FIELD>
      </FIELDS>
      <PARAMS />
    </METADATA>
    <ROWDATA />
  </DATAPACKET>
```

Quadro 11 – Estrutura do *data packet*

# APÊNDICE D – Documento de especificação de requisitos de software

O Quadro 12 apresenta o documento de ERS gerado através desta ferramenta.

## Documento de Especificação de Requisitos de Software

**1. Introdução**

*1.1 Propósito*

Este documento tem por objetivo apresentar os requisitos de uma ferramenta proposta como Trabalho de Conclusão de Curso do acadêmico Raphael Marcos Batista, a qual auxilia a especificação de requisitos de software.

*1.2 Escopo*

Esta ferramenta permitirá o cadastramento de requisitos funcionais, não funcionais e regras de negócio. Permitirá a integração com a ferramenta EA para importar e exportar os requisitos para esta ferramenta. Será possível a geração de um documento de especificação de requisitos de software em vários formatos, como por exemplo, o formato RTF. Deverá permitir a configuração de vários tipos de matriz

*1.3 Definições, Acrônimos e Abreviaturas*

EA: Enterprise Architect.
ERS: Especificação de Requisitos de Software.
IEEE: Institute of Electrical and Eletronic Engineers.
RF: Requisito Funcional.
RNF: Requisito Não Funcional.
RN: Regras de Negócio.
Requisito Funcional: definem as funcionalidades e serviços que o sistema irá fornecer.
Requisito Não Funcional: definem as restrições sobre as funcionalidades e serviços que o sistema poderá fornecer.

*1.4 Referências*

IEEE-830-1998.
Livros da engenharia de software.
Trabalhos correlatos de ferramentas de gerência de requisitos.

*1.5 Visão Geral*

O documento de ERS é organizado conforme proposto pelo modelo IEEE-830-1998.
Na seção introdução, é apresentado o propósito, escopo, definições, acrônimos, abreviaturas, referências e visão geral.
Na seção descrição geral, são relacionadas a perspectiva e funções do produto, também são descritos as características dos usuários, restrições, dependências e suposições que a ferramenta possuirá. A seção de requisitos específicos contempla os requisitos cadastrados, sendo eles funcionais, não funcionais ou regras de negócio.

**2. Descrição Geral**

*2.1 Perspectiva do Produto*

Proporcionar uma extensão da ferramenta EA para o gerenciamento de requisitos.

*2.2 Funções do Produto*

A ferramenta terá integração com a ferramenta EA. Os requisitos definidos no EA poderão ser importados e manipulados através da ferramenta. A ferramenta permitirá o gerenciamento de mudanças através da matriz de rastreabilidade. Haverá a possibilidade de criar vários tipos de matrizes, para relacionar os tipos de requisitos e casos. A ferramenta permitirá a geração de um documento de ERS de acordo com o modelo IEEE-830-1998. O documento poderá ser salvo em arquivo sob diversos formatos.

*2.3 Características do Usuário*

Para utilizar esta ferramenta os usuários deverão possuir conhecimento técnico em análise de sistemas.

*2.4 Restrições*

Não há histórico de alterações e versões.
Não há controle de acesso dos usuários na ferramenta.

*2.5 Dependências e Suposições*

IEEE lançar uma nova versão do modelo IEEE-830-1998.
A Sparx Systems lançar uma nova versão do EA que não seja compatível com a versão atual de interface de automação.

**3. Requisitos Específicos**

*3.1 Requisito Funcional*

**RF001**

O sistema deverá cadastrar os dados das seções do documento de requisitos
- Estado: Implementado
- Prioridade: Alta

**RF002**

O sistema deverá permitir abrir o projeto do EA para importar os requisitos
- Estado: Implementado
- Prioridade: Alta

**RF003**

O sistema deverá cadastrar os tipos de requisitos desejados para o levantamento de requisitos
- Estado: Implementado
- Prioridade: Alta

**Rastreabilidade:**

**RNF004**

Utilizar controles ActiveX para acessar a interface do EA e manipular seus objetos

**RF004**

O sistema deverá cadastrar os requisitos do projeto levantados pelo o usuário
- Estado: Implementado
- Prioridade: Alta

**Documento de Especificação de Requisitos de Software**

**Rastreabilidade:**
**RNF002**
Utilizar a técnica Estrutura de Árvore, que mostrará os requisitos em forma de hierarquia para manipular os requisitos na área de trabalho

**RF005**
O sistema deverá cadastrar os valores do atributo estado do requisito que podem ser, por exemplo, proposto, aprovado e obrigatório
- Estado: Implementado
- Prioridade: Alta

**RF006**
O sistema deverá permitir gerenciar os requisitos permitindo vincular requisitos com outros requisitos e casos de usos, viabilizando a rastreabilidade (funcionalidade primordial para o gerenciamento de requisitos)
- Estado: Implementado
- Prioridade: Média

**Rastreabilidade:**
**RNF005**
Utilizar ícones representativos para identificar a situação do requisito: criado, alterado e vínculo

**RF007**
O sistema deverá sinalizar ao usuário, as alterações realizadas no requisito que possui vínculo com casos de uso ou outro requisito
- Estado: Implementado
- Prioridade: Baixa

**Rastreabilidade:**
**RNF005**
Utilizar ícones representativos para identificar a situação do requisito: criado, alterado e vínculo

**RF008**
O sistema deverá permitir gerar os requisitos no projeto do EA
- Estado: Implementado
- Prioridade: Alta

**RF009**
O sistema deverá emitir o relatório do documento de especificação de requisitos
- Estado: Implementado
- Prioridade: Média

*3.2 Requisito Não Funcional*

**RNF001**
O documento de requisitos deverá estar em conformidade com o modelo IEEE-830-1998
- Estado: Implementado
- Prioridade: Média

**RNF002**
Utilizar a técnica Estrutura de Árvore, que mostrará os requisitos em forma de hierarquia para manipular os requisitos na área de trabalho
- Estado: Implementado
- Prioridade: Baixa

**Rastreabilidade:**
**RF004**
O sistema deverá cadastrar os requisitos do projeto levantados pelo o usuário

**RNF003**
Portabilidade com Windows 2000/XP
- Estado: Aprovado
- Prioridade: Baixa

**RNF004**
Utilizar controles ActiveX para acessar a interface do EA e manipular seus objetos
- Estado: Implementado
- Prioridade: Alta

**Rastreabilidade:**
**RF003**
O sistema deverá cadastrar os tipos de requisitos desejados para o levantamento de requisitos

**RNF005**
Utilizar ícones representativos para identificar a situação do requisito: criado, alterado e vínculo
- Estado: Implementado
- Prioridade: Alta

**Rastreabilidade:**
**RF006**
O sistema deverá permitir gerenciar os requisitos permitindo vincular requisitos com outros requisitos e casos de usos, viabilizando a rastreabilidade (funcionalidade primordial para o gerenciamento de requisitos)

**RF007**
O sistema deverá sinalizar ao usuário, as alterações realizadas no requisito que possui vínculo com casos de uso ou outro requisito

**Apêndices**



Quadro 12 – Documento de ERS