C&L: Uma Ferramenta de Apoio à Engenharia de Requisitos

Lyrene Fernandes da Silva
e-mail: lyrene@inf.puc-rio.br

Julio Cesar Sampaio do Prado Leite
e-mail: julio@inf.puc-rio.br

Karin Koogan Breitman
e-mail: karin@inf.puc-rio.br

PUC-RioInf.MCC25/04 Julho, 2004

Abstract. This paper presents an experimental tool to support requirements engineering activities. This tool, called C&L, is based on two techniques to model software requirements, Scenarios and the Extend Lexicon of the Language. C&L supports a cooperative process to edit Scenarios and the Lexicon. It aims to maintain the traceability between requirements and code. C&L is one of the Requirements Engineering Group's research projects at PUC-Rio

Keywords: Requirements engineering, Scenarios, Lexicon, Traceability.

Resumo. Este trabalho apresenta uma ferramenta experimental para dar apoio à engenharia de requisitos. Essa ferramenta, denominada C&L, está centrada nas técnicas de modelagem de requisitos, Cenários e Léxico Ampliado da Linguagem (LAL). C&L é uma ferramenta para edição cooperativa de cenários e léxico preocupada com a rastreabilidade entre requisitos e código. Esta ferramenta é parte de um projeto de pesquisa do grupo de Engenharia de Requisitos da PUC-Rio. Neste projeto temos estudado algumas técnicas baseadas em cenários e léxicos de maneira a construir uma ferramenta que dê apoio ao desenvolvimento de software guiado por requisitos

Palavras-chave: Engenharia de requisitos, Cenários, Léxico, Rastreabilidade.

1. Introdução

A atividade de engenharia de requisitos é responsável por fazer a interface entre os desejos e necessidades dos clientes e a posterior implementação em forma de software. Em outras palavras, faz o mapeamento entre o informal (requisitos) e o formal (modelos e código). Estes requisitos devem guiar o processo de desenvolvimento de maneira a garantir que o software construído corresponda ao desejado. O vínculo entre requisitos e modelos desenvolvidos no decorrer do processo de desenvolvimento é chamado de rastreabilidade. Manter a rastreabilidade é uma tarefa difícil visto que os diferentes modelos evoluem durante o processo e que os requisitos também mudam.

Pesquisadores da área de Engenharia de Requisitos têm desenvolvido métodos, técnicas e ferramentas que dão apoio às tarefas associadas ao processo de requisitos. Entretanto, muitos destes métodos e técnicas não estão em prática no mercado devido, em parte, à carência de ferramentas. No mercado há algumas ferramentas para gerência de requisitos, tais como RequisitePro [RequisitePro], Doors [Doors], e CaliberRM [CaliberRM]. Todas são proprietárias e não fundamentadas em processos ou técnicas específicas. Esta característica é necessária para atender um maior número de usuários, mas por outro lado, dificulta o trabalho do desenvolvedor, pois este tem que definir seu próprio processo, mesmo que ele não esteja apto a fazê-lo devido à ferramenta só permitir uma modelagem parcial dos métodos desejados ou pela falta de integração dela com outras ferramentas utilizadas pela equipe.

A adoção de software livre por empresas está começando a ser considerada uma alternativa devido ao baixo custo na aquisição e acesso ao código fonte. Este acesso permite desenvolver ou melhorar componentes que não satisfaçam as necessidades desejadas. Neste contexto o grupo de Engenharia de Requisitos da PUC-Rio vem trabalhando ao longo de dois anos em um projeto de pesquisa denominado C&L. C&L é uma ferramenta para modelar requisitos através de cenários e léxico. Cenários e Léxico compõem uma estratégia para modelar requisitos utilizando o vocabulário do Universo de Informação (UdI) e focando em situações conforme percebidas pelos clientes e usuários [Leite97]. Esta estratégia utiliza linguagem natural semi-estruturada de maneira a amenizar os problemas resultantes da ambigüidade inerente à linguagem natural.

A ferramenta C&L está em evolução. Atualmente, C&L permite a edição de cenários, léxico e parte de ontologias. Através da ferramenta, os interessados visualizam a rastreabilidade entre cenários, entre termos do léxico e entre cenários e termos do léxico. Este projeto tem sido a base para diferentes provas de conceito de alunos de pós-graduação. C&L tem sido utilizada para validar estudos sobre desenvolvimento de software livre [Holanda04], geração de ontologias a partir do Léxico Ampliado da Linguagem [Breitman03][Felicissimo03], cenários como padrão para comentar o código [Silva03], verificação de cenários [Sayão03], evolução de software, ensino em cursos de engenharia de software [Silva04].

Uma das metas da ferramenta C&L é manter e disponibilizar a rastreabilidade entre requisitos e código. A idéia é integrar cenários e código de maneira a permitir que os diferentes interessados trabalhem em um conjunto compartilhado de informações sem precisar conhecer detalhes que não são de sua responsabilidade. Por exemplo, permitir que os usuários/clientes tenham acesso aos cenários para validar o sistema, o gerente do projeto possa acompanhar o projeto, os engenheiros de requisitos possam modelar os requisitos e comunicá-los aos desenvolvedores, os desenvolvedores possam entender detalhes da arquitetura do software, os testadores possam criar os testes de aceitação, dentre outros.

Neste artigo apresentamos as principais funcionalidades da ferramenta C&L. Organizamos o artigo da seguinte maneira: na Seção 2 descrevemos o contexto em que este trabalho está inserido, apresentando o embasamento teórico que permitiu a construção da ferramenta; na Seção 3 apresentamos as funcionalidades e arquitetura da ferramenta C&L; na Seção 4 descrevemos trabalhos relacionados a este e as vantagens de nossa abordagem; na Seção 5 resumimos nossas contribuições, conclusões e planos para o futuro.

2. Fundamentação Teórica

Pesquisas na área de engenharia de software têm focado técnicas que minimizem o *gap* existente entre o mundo real (Universo de Informação - UdI) e os modelos computacionais que o implementam. Muitas destas técnicas se baseiam na criação de modelos e de linguagens menos ou mais formais, tentando oferecer ao

desenvolvedor uma maneira gradual de transformar requisitos em software, e assim retratar de maneira fiel uma realidade pretendida.

A engenharia de requisitos é responsável por entender o UdI e identificar quais e como as tarefas da aplicação devem ser automatizadas. Mais que isto, é responsável por comunicar estas informações para os outros integrantes da equipe de desenvolvimento e garantir que o software sendo construído é o software requisitado. As atividades da engenharia de requisitos podem ser resumidas em: Elicitação, Modelagem, Análise e Gerência.

Para entender, registrar e comunicar as informações do UdI e para acompanhar o andamento do projeto faz-se uso de modelos. Há vários modelos utilizados pelos engenheiros de requisitos para dar apoio às suas atividades, tais como: lista de requisitos, cenários, casos de uso, glossários, léxico, diagrama de classes, MER, SADT, DFD, matrizes de rastreabilidade, listas de prioridades e i* entre outros. Assim, o engenheiro de requisitos deve selecionar os modelos mais apropriados para o tipo de projeto, processo, e equipe a desenvolver o software.

Glossários ou léxico e cenários ou casos de uso têm sido bastante utilizados em fases iniciais do desenvolvimento de software porque são modelos razoavelmente simples e de fácil entendimento por parte dos usuários. Eles utilizam linguagem natural para representar suas informações. A linguagem natural desempenha um papel fundamental visto que ajuda a comunicação entre diferentes tipos de interessados, ajuda o desenvolvedor a entender o UdI e ajuda o usuário a validar os requisitos do sistema sendo construído.

Neste artigo relatamos uma ferramenta para modelagem de requisitos baseada no uso de cenários e do léxico. A idéia inicial de construir esta ferramenta era permitir que uma equipe de desenvolvedores trabalhasse colaborativamente (via *Web*) na definição dos requisitos de um projeto. Para chegar aos requisitos, a equipe deve ser capaz de identificar a linguagem utilizada no UdI e os cenários percebidos pelos usuários. Assim, a equipe trabalha em colaboração para definir o léxico e os cenários da aplicação.

A utilização de cenários durante o desenvolvimento de software é um tópico que tem ganhado bastante destaque na comunidade de Engenharia de Software [Weidenhaupt98][Rolland98]. O desenvolvimento de software baseado em cenários se apóia no conceito de que a utilização da linguagem do problema (domínio do usuário) é benéfica à interação entre usuários e desenvolvedores [Leite95]. Entendemos por cenários a descrição de situações comuns ao cotidiano dos interessados [Zorman95]. Eles devem levar em conta aspectos de usabilidade e permitir o aprofundamento do conhecimento do problema, a unificação de critérios, a obtenção do compromisso de clientes e/ou usuários, a organização de detalhes e o treinamento de pessoas [Carroll94].

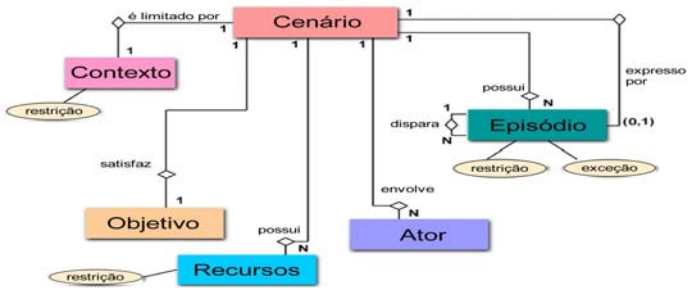

Figura 1. Representação para cenários proposta por Leite [Leite00]

Cada cenário descreve, através de linguagem natural semi-estruturada, uma situação específica da aplicação, priorizando seu comportamento. Cenários podem ser detalhados e utilizados como desenho de maneira a auxiliar a programação. Existem várias propostas para a representação de cenários, desde a mais informal, em texto livre [Carroll94] até representações formais [Hsia94]. Optamos por uma representação intermediária que, ao mesmo tempo em que facilita a compreensão através da utilização de linguagem natural, força a organização da informação através de uma estrutura bem definida. A notação para cenários escolhida é a proposta em [Leite97] e está ilustrada na Figura 1.

Esta estrutura é composta por elementos que expressam o objetivo, o contexto, recursos, atores, atividades bem como restrições e exceções. Descrevemos brevemente esses elementos a seguir:

- Título - é o identificador do cenário.

- Objetivo - é a descrição da finalidade do cenário. Deve ser concreto e preciso.
- Contexto - é a descrição do estado inicial do cenário, através de pré-condições, localização geográfica e/ou localização temporal.
- Recursos – são entidades passivas necessárias à realização do cenário. Devem ser referidos em pelo menos um dos episódios.
- Atores - são entidades (sistema, organização ou pessoa), que se envolvem ativamente no cenário. Devem ser referidos em pelo menos um dos episódios.
- Episódios - seqüência de sentenças simples, condicionais ou opcionais, correspondente a ações e decisões com participação dos atores. Obedecem a uma ordem temporal e utilizam recursos. O conjunto de episódios atende ao objetivo do cenário.
- Restrições – são aspectos não-funcionais que podem estar relacionados ao contexto, recursos ou a episódios específicos.
- Exceções – correspondem a situações que podem impedir que o objetivo do cenário seja atingido e o tratamento correspondente a tal situação.

Os cenários apresentam de maneira objetiva as entidades ativas e passivas necessárias em determinada situação (atores e recursos, respectivamente). Através do contexto é possível identificar em que situação o cenário acontece. Tais informações dão uma visão global do contexto em que o cenário está inserido e sua interação com as entidades do sistema. A descrição do contexto, das restrições, dos episódios e das exceções dá suporte à elaboração de casos de testes. Em particular, as restrições permitem a identificação e a descrição de aspectos não-funcionais. Desta maneira, os cenários podem ser utilizados para validar os requisitos elicitados.

O processo de construção de cenários está relacionado à existência do Léxico Ampliado da Linguagem (LAL). O LAL é um hiper-documento que descreve os símbolos da linguagem do Universo de Informação. É utilizado para facilitar a comunicação e a compreensão de palavras ou frases peculiares ao UdI [Leite93]. Cada termo do léxico possui dois tipos de descrição. O primeiro tipo, noção, é a denotação do termo ou expressão, i.e., seu significado. O segundo tipo, impacto ou resposta comportamental, descreve a conotação do termo ou expressão, i.e., provê informação extra sobre o contexto.

Dicionários e glossários, de modo geral, só representam a denotação dos termos. Como o LAL representa também os relacionamentos entre os termos, através dos impactos, temos uma representação muito mais detalhada da organização do UdI. Os termos do léxico são classificados em quatro categorias: objeto, sujeito, estado e verbo. Há dois princípios fundamentais no LAL. O primeiro é maximizar o uso dos outros termos do léxico quando descrevemos a noção e o impacto de um novo termo, denominado princípio da circularidade. O segundo, é minimizar o uso de termos externos ao UdI, denominado de princípio do vocabulário mínimo.

O LAL é escrito em linguagem natural. Desta forma, o próprio usuário, com a ajuda do engenheiro de requisitos, pode escrevê-lo e validá-lo. Através do LAL o engenheiro de requisitos consegue compartilhar, analisar e reutilizar o conhecimento do UdI. A Figura 2 ilustra a representação para léxico proposta em [Leite00].

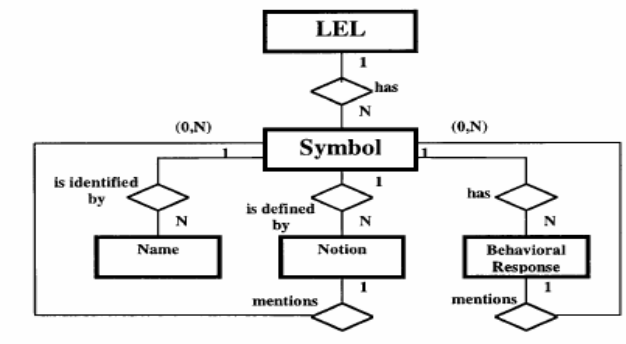

Figura 2. Representação para léxico ampliado da linguagem proposta por Leite [Leite00]

Os termos do LAL são utilizados na descrição dos cenários, retratando uma relação entre elementos do UdI e os requisitos do usuário. A integração entre cenários e LAL permite a rastreabilidade entre termos do LAL, entre cenários, e entre termos do LAL e cenários. Esta rastreabilidade ajuda o engenheiro de requisitos a identificar mais rápido o impacto de mudanças. Mudanças em um termo do LAL (ou seja, algum termo do UdI não estava claro ou foi mal compreendido) podem implicar em mudanças em: outros termos do LAL e nos

cenários que fazem referência àquele. Mudanças em um cenário podem implicar em mudanças: nos cenários que estão direta ou indiretamente relacionados àquele, com maior e com menor freqüência, respectivamente.

Para viabilizar a rastreabilidade entre cenários e código temos trabalhado com a possibilidade de inserir os próprios cenários no código. Desta maneira criamos um documento único com dois tipos de informação: os cenários representando os requisitos, e o código representando a operacionalização destes. Para trabalhar com este documento único é necessário apoio automatizado para tornar possível à extração das diferentes visões deste documento.

Só com o apoio de ferramentas podemos garantir que as mudanças realizadas nos cenários dentro do código serão visíveis para os outros interessados que não estão trabalhando com o código. Também é necessário oferecer um indicativo de mudança para os desenvolvedores. Temos experimentado esta abordagem no contexto de software livre, gerando automaticamente uma documentação a partir do código comentado com cenários. Na Seção 2.1 resumimos esta abordagem.

Um outro tópico de interesse do nosso grupo de pesquisa é a modelagem de ontologias. Em [Breitman03] foi definido um processo de construção de ontologias baseado no LAL. O LAL é utilizado como ponto de partida para a construção da ontologia. Nesta abordagem consideramos a geração de ontologias uma atividade de responsabilidade do engenheiro de requisitos, principalmente quando se trata de ontologias no contexto da *Web* semântica [Hendler01]. Este processo foi implementado no C&L de maneira semi-automática e é resumido na Seção 2.2.

2.1 Enriquecendo o Código com Cenários

Comentar o código com cenários é uma nova abordagem para a construção de código guiada por cenários. Isto fornece a rastreabilidade entre requisitos (registrados através dos cenários) e componentes de código, evidenciando as dependências entre requisitos. Esta simbiose entre cenários e código passa a ser um guia para a construção, manutenção e evolução de programas.

Cenários fornecem uma representação uniforme da informação, promovem o reuso facilitando a identificação de funcionalidades, agilizam a manutenção e a identificação de partes do código que seriam afetadas no caso de mudanças. Além disto, a padronização oferecida pelos cenários permite a automação de parte destas tarefas. As principais vantagens [Silva03] dos cenários quando comparados a comentários não estruturados são:

- cenários são representados através de uma estrutura bem definida, oferecendo organização e padronização na apresentação da informação;
- a estrutura episódica dos cenários facilita a enumeração das funcionalidades encapsuladas em cada porção de código. De forma semelhante, a identificação de episódios similares em diferentes cenários serve de indicativo de redundância de código;
- os cenários são descritos utilizando um subconjunto da linguagem natural, registrado no Léxico Ampliado da Linguagem (LAL). No LAL definimos um número limitado de vocábulos que podem ser utilizados na descrição dos cenários, de forma a garantir a consistência de vocabulário, i.e., utilizar sempre o mesmo termo e minimizar ambigüidades. A utilização de um vocabulário controlado facilita a implementação de mecanismos que permitem a identificação de outros cenários que compartilham os mesmos termos. O casamento destes cenários serve como indicativo de acoplamento entre as respectivas porções de código;
- cenários contém descrições explícitas de pré e pós condições, facilitando a visualização de possíveis encadeamentos entre os cenários e seus respectivos códigos;
- o conjunto de todos os cenários pode servir como documento explicativo do funcionamento do software para o usuário;
- a descrição de cenários em linguagem natural semi-estruturada permite que usuários não especialistas sejam capazes de entendê-los e validá-los.

Nossa proposta é semelhante ao trabalho de Knuth [Knuth84]; a "*web*" referida por ele corresponde às ligações entre cenários e código. Cada módulo do sistema contém código e o cenário correspondente; os cenários de um módulo fazem referências a termos que também são utilizados em outros cenários, que por sua vez estão localizados em outros módulos. Isto possibilita a rastreabilidade entre requisitos (expressos através dos cenários) e código. A ligação entre cenários acontece por referência explícita a sub-cenários ou através de termos constantes no LAL.

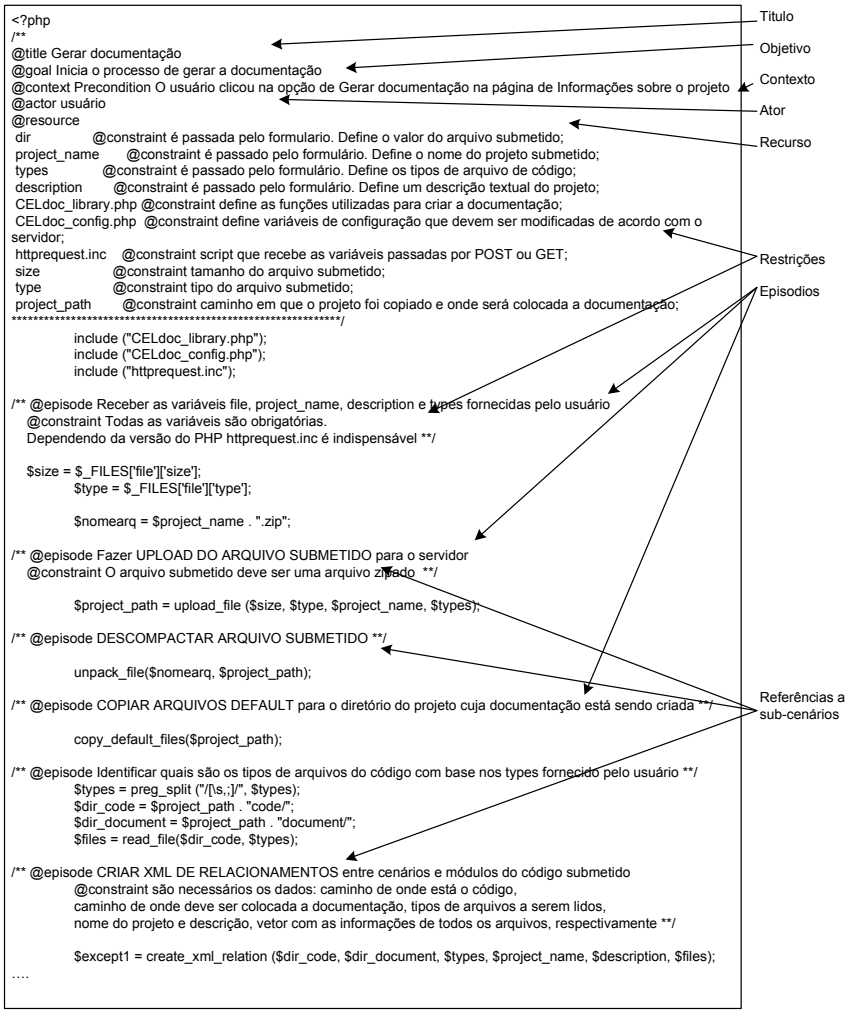

Figura 3. Exemplo de código comentado com cenários

Na Figura 3 ilustramos um exemplo de cenário encapsulado no código. Definimos uma notação para delimitar os elementos do cenário, e.g., *@title, @goal, @context, @constraint, @actor* etc. As referências a outros cenários nós escrevemos em maiúsculo de modo a evidenciar que aquela referência está implementada em um outro componente, mas isto não é levado em consideração para a geração automática dos *links*, ou seja, mesmo que as referências não estejam em maiúsculo os relacionamentos entre cenários e sub-cenários são identificados pelo *parser* descrito na Seção 3.2.

Encapsular as informações oferecidas pelo cenário no código provê uma maneira simples de obter e atualizar estas informações. Desta forma, acreditamos que o programador pode compreender mais rapidamente a funcionalidade de cada módulo e perceber o impacto que mudanças em um módulo provocam no restante do sistema, pois ele tem em mãos informações sobre como este módulo está inserido na arquitetura do sistema.

Esta abordagem propicia maior colaboração durante o processo de desenvolvimento de software livre, visto que neste contexto a qualidade das descrições é crítica, quer sejam descrições de código ou relativas a outros artefatos. Na Seção 3.2 detalhamos como geramos a documentação baseada em cenários automaticamente através de um módulo do C&L.

2.2 Modelar Ontologias a partir do LAL

Nos últimos anos, notamos uma crescente necessidade em categorizar os conceitos do UdI no contexto da *Web*. Esta necessidade é ocasionada, principalmente, devido a grande quantidade de dados disponível na *Web* não dispor de semântica explícita. Isto leva às pessoas a terem um trabalho extra na seleção das informações

relevantes e às máquinas a armazenarem e tratarem os dados redundantes. A categorização taxonômica oferecida pelas ontologias surgiu para mudar este cenário.

Como descrito em [Noy01], as principais razões para se construir ontologias são: compartilhar o entendimento da estrutura da informação entre pessoas e agentes de software; possibilitar o reuso de conhecimento do domínio; tornar as verdades absolutas do domínio explícitas; separar o conhecimento do domínio do conhecimento operacional; e analisar o conhecimento do domínio.

Do ponto de vista conceitual, as ontologias são compatíveis ao LAL porque oferecem uma organização explícita dos conceitos e relacionamentos existentes em um dado Universo de Informação. Ontologias são especificações formais dos termos do UdI e as relações entre eles [Noy01]. Na literatura há várias definições para ontologias. Neste trabalho adotamos a definição proposta em [Maedche02]. Nesta estrutura, uma ontologia é descrita pela tupla:

$O := \{C, R, H^C, rel, A^O\}$:

- C (conceitos) e R (relações) são dois conjuntos disjuntos;
- H^C é uma relação direcionada $H^C \subseteq C \times C$ que é chamada hierarquia de conceitos ou taxonomia. Por exemplo, H^C *(C1, C2)* significa que C1 é um subconjunto de C2;
- *rel* é uma função $rel : R \rightarrow C \times C$ que relaciona os conceitos não taxonomicamente;
- A^O é um conjunto de axiomas expresso em linguagem lógica apropriada.

Em nossa visão, a modelagem de ontologias deve ser realizada durante a fase de requisitos. Para apoiar o desenvolvimento de ontologias, foi proposto em [Breitman03] um processo baseado no LAL. A maior vantagem deste enfoque é poder contar com um método maduro para auxiliar na tarefa de elicitação, modelagem e validação dos conceitos e relacionamentos do Universo de Informação partindo-se da linguagem da aplicação.

O processo de construção do léxico é estruturado e segue princípios sólidos de engenharia de Software e técnicas já estabelecidas para a captura, modelagem e posterior análise da informação modelada [Kaplan00]. O LAL provê a linguagem comum para a comunicação informal entre os interessados no processo de desenvolvimento de software, i.e., clientes, usuários e desenvolvedores, enquanto ontologias fornecem este modelo de modo mais formal, permitindo o compartilhamento de informações entre máquinas e agentes de software.

Breitman e Leite em [Breitman03], definem um processo para gerar ontologias a partir do LAL, levando em conta as tarefas de elicitação, modelagem e análise para explicitar e comunicar o conhecimento do UdI. Este processo mapeia os termos do LAL nos elementos da ontologia: os termos do tipo **objeto** e **sujeito** são mapeados em **conceitos**; os termos do tipo **verbo** são mapeados em **propriedades**; os termos do tipo **estado** são mapeados em **conceitos** ou **propriedades**; a **noção** de cada termo é mapeada na **descrição** do respectivo **conceito**; e através da lista de **impactos** de cada termo do léxico mapeia-se o **verbo** em **propriedades** e o **predicado** em **conceitos** ou **restrições dos conceitos**. Na Tabela 1, resumimos o conjunto de mapeamentos entre termos do LAL e elementos da ontologia.

Tabela 1. Mapeamento entre os elementos do LAL e os da ontologia definido por Breitman e Leite em [Breitman03]

***Elementos do* LAL**		***Elementos da Ontologia***
Termo (ou símbolo)		
- Tipo		
- Objeto e Sujeito	→	Conceitos
- Estado	→	Conceito ou axioma
- Verbo	→	Propriedade
- Noção	→	Descrição
- Impacto		
- Verbo	→	Propriedade
- Predicado (termos)	→	Conceitos ou Axiomas

De modo a prover apoio semi-automático ao processo de construção de ontologias a partir do LAL, desenvolvemos um componente para a ferramenta C&L em [Felicíssimo03]. Este componente utiliza como dados de entrada o léxico de um projeto já editado e, gera como saída, uma ontologia em um arquivo do tipo *daml*, padrão [W3Consortium03]. Na Seção 3.1 detalhamos como a geração de ontologias a partir do LAL é apoiada na ferramenta C&L.

3. Descrição da Ferramenta C&L

A primeira versão da ferramenta C&L foi desenvolvida por um grupo de alunos de engenharia de computação da PUC-Rio. Depois desta versão inicial elaboramos um plano de evolução para disponibilizar a ferramenta como software livre. Desde então C&L está disponível no endereço *http://sl.les.inf.puc-rio.br/cel/* para uso e *download*. Na página inicial disponibilizamos a versão atual da ferramenta, o *script* de criação do banco de dados, a lista de erros, artigos relacionados a cenários e léxico, manual do usuário, cenários da ferramenta, arquitetura e casos de teste.

Temos evoluído C&L de maneira a acomodar os resultados de pesquisa do grupo de Engenharia de Requisitos da PUC-Rio. Esta ferramenta é utilizada sistematicamente no ensino de dois cursos, engenharia de software e evolução de software. Em ambos, a ferramenta desempenha papel fundamental por que é empregada no ensino das técnicas de modelagem, cenários e léxico, e é utilizada como estudo de caso no trabalho prático da disciplina.

Como estudo de caso, os alunos utilizam a ferramenta como projeto de software livre para modelar os requisitos e como fonte de componentes reusáveis para novas versões da própria ferramenta. Em [Silva04], descrevemos o método utilizado para ensinar engenharia de software. O método visa despertar o interesse dos alunos em engenharia de software, apresentando os aspectos teóricos e oferecendo uma experiência prática da aplicação destes conceitos. Para a experiência prática adotamos uma filosofia de desenvolvimento cooperativo, calcado no processo *Extreme Programming* (XP).

Com o uso da ferramenta pelos alunos das duas disciplinas realizamos a validação de C&L sobre dois aspectos: sob a visão de usuário e sob a visão de desenvolvedor. Desta forma, temos evoluído C&L de maneira a tentar atender as necessidades levando-se em conta as mudanças solicitadas.

C&L funciona segundo uma arquitetura cliente-servidor em que a parte cliente é apenas a interface em *HTML* visualizada através do *browser*. Todas as solicitações via *browser* são tratadas no servidor *web* onde está instalada a ferramenta. Os dados dos projetos são armazenados em um banco de dados *MySQL*. Na Figura 4 mostramos a arquitetura global de C&L; A interface visualizada no *browser* do cliente foi implementada, principalmente, em *HTML* e as funcionalidades da ferramenta C&L em PHP.

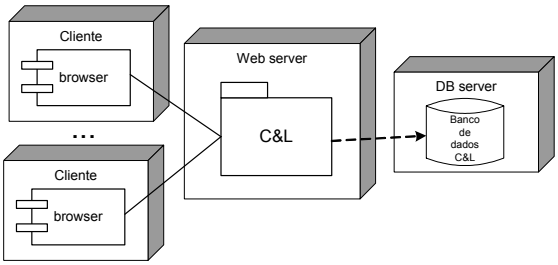

Figura 4. Arquitetura cliente-servidor da ferramenta C&L

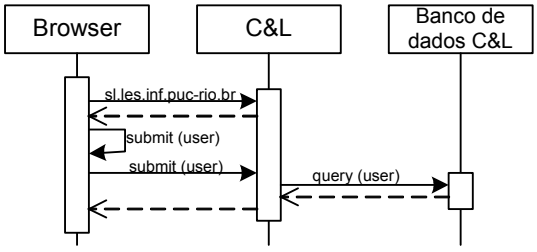

Figura 5. Diagrama de interações entre clientes e servidor

Na Figura 5, ilustramos a seqüência de interações que ocorrem para acessar um serviço oferecido por C&L. Por exemplo, a partir do *browser* o usuário requisita que a página de C&L seja acessada, digitando o endereço *WEB* da ferramenta. O servidor responde mostrando a página de *login* da ferramenta, o usuário submete suas informações, se o formulário não tiver sido preenchido integralmente o usuário recebe uma

mensagem avisando-o, caso o usuário preencha todas as informações a ferramenta busca as informações do usuário no banco e reponde ao usuário com a devida interface.

A interação entre cliente e servidor é realizada através do protocolo *http*. O uso de sessões se faz necessário para atender vários clientes ao mesmo tempo. A gerência destas sessões é realizada pelo APACHE. A ferramenta depende de seu banco de dados, visto que todas as funcionalidades do C&L utilizam informações armazenadas no banco de dados. Os servidores *WEB* e de banco de dados podem está fisicamente em uma mesma máquina ou em máquinas diferentes sendo a conexão realizada através de ODBC.

Na interface com o usuário alguns *scripts* em Java foram implementados apenas para verificar se os formulários foram preenchidos integralmente pelo usuário antes que uma chamada ao servidor seja realizada. Assim, C&L está implementado com as linguagens PHP, HTML, e JavaScript; utiliza a versão livre do banco de dados MySQL e está em um servidor APACHE. Atualmente, a versão disponível funciona inteiramente no *browser* Microsoft Internet Explorer, mas estamos fazendo as devidas alterações para torná-lo disponível em outros *browsers*.

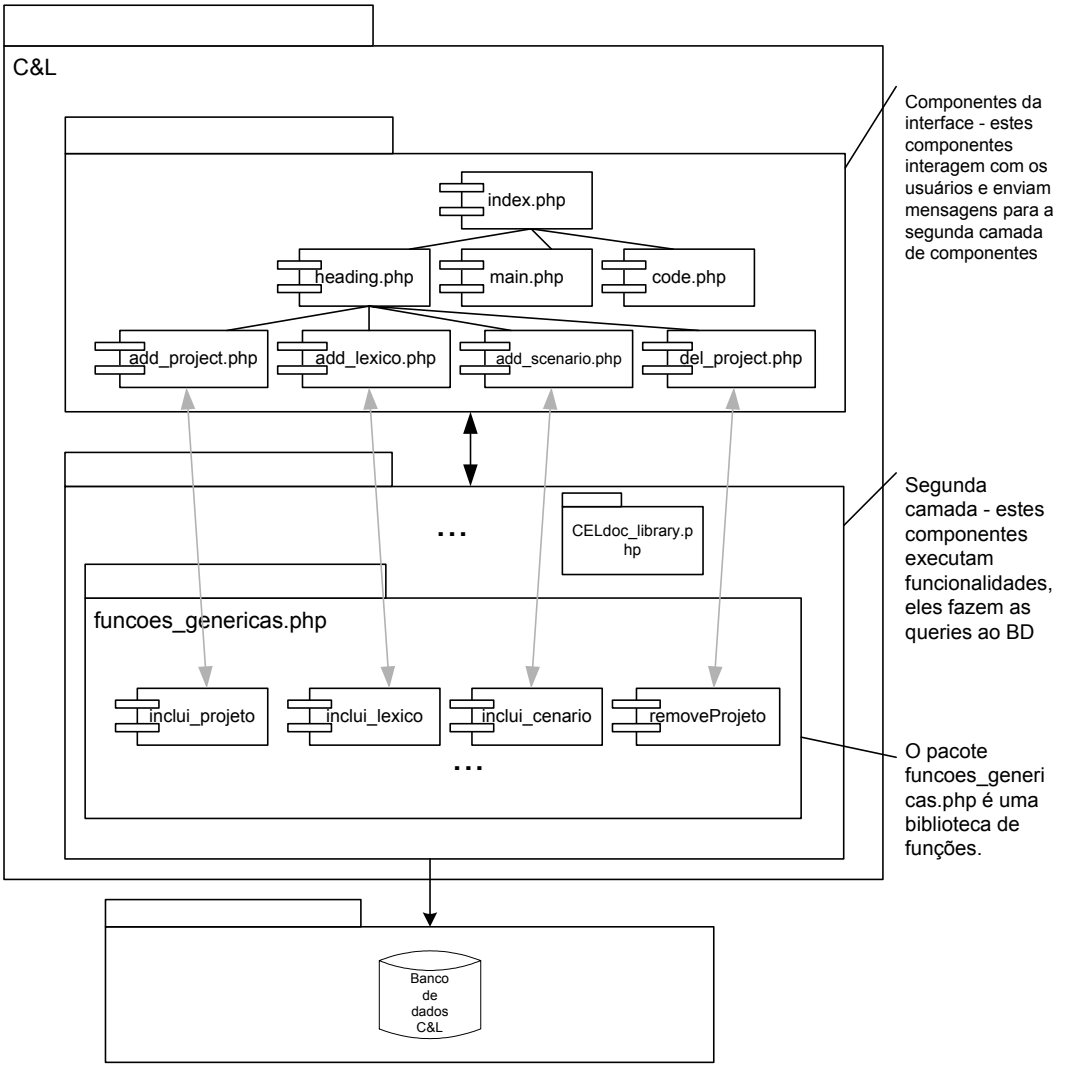

Figura 6. Arquitetura interna do C&L

O pacote C&L, ilustrado nas Figuras 4 e 5, pode ser visualizado conforme uma arquitetura em duas camadas: interface e funcionalidade. A Figura 6 ilustra essa organização para alguns de seus componentes. O processo que temos utilizado no desenvolvimento do C&L e as condições intrínsecas de suas plataformas de software (JavaScript, HTML, PHP) tem dificultado a utilização de uma arquitetura padrão. No entanto, estamos trabalhando para fatorar da parte de funcionalidade os aspectos de banco de dados e também para explicitar os componentes de integração (controle) entre as camadas. O uso dessa estrutura (MVCDB) em código PHP

apresenta alguns desafios, face às restrições de organização da linguagem. É nossa intenção, tratar esse problema nos trabalhos futuros.

C&L está em sua quarta versão, a interface principal da ferramenta é ilustrada na Figura 7. A interface é estruturada em três *frames*: *frame* superior apresenta o menu com o conjunto de funções que podem ser realizadas por usuários comuns e usuários administradores; *frame* do lado esquerdo, apresenta uma estrutura em árvore com os cenários, termos do léxico e conceitos da ontologia criados para um projeto; e *frame* central, onde os cenários e termos do léxico são mostrados para navegação ou alteração.

No *frame* central da Figura 7 ilustramos um cenário. A estrutura do cenário é composta pelos elementos: título, objetivo, contexto, atores, recursos, episódios e exceções. As restrições e os tipos de contextos (pré-condição, localização geográfica e localização temporal) não foram distinguidos dos outros elementos ainda. O cenário corrente é "Organizar reuniões" do projeto Agendador de Reuniões. Os *links* (na figura estão em negrito) são criados ou excluídos automaticamente a medida em que os usuários adicionam ou removem cenários ou termos do léxico, respectivamente.

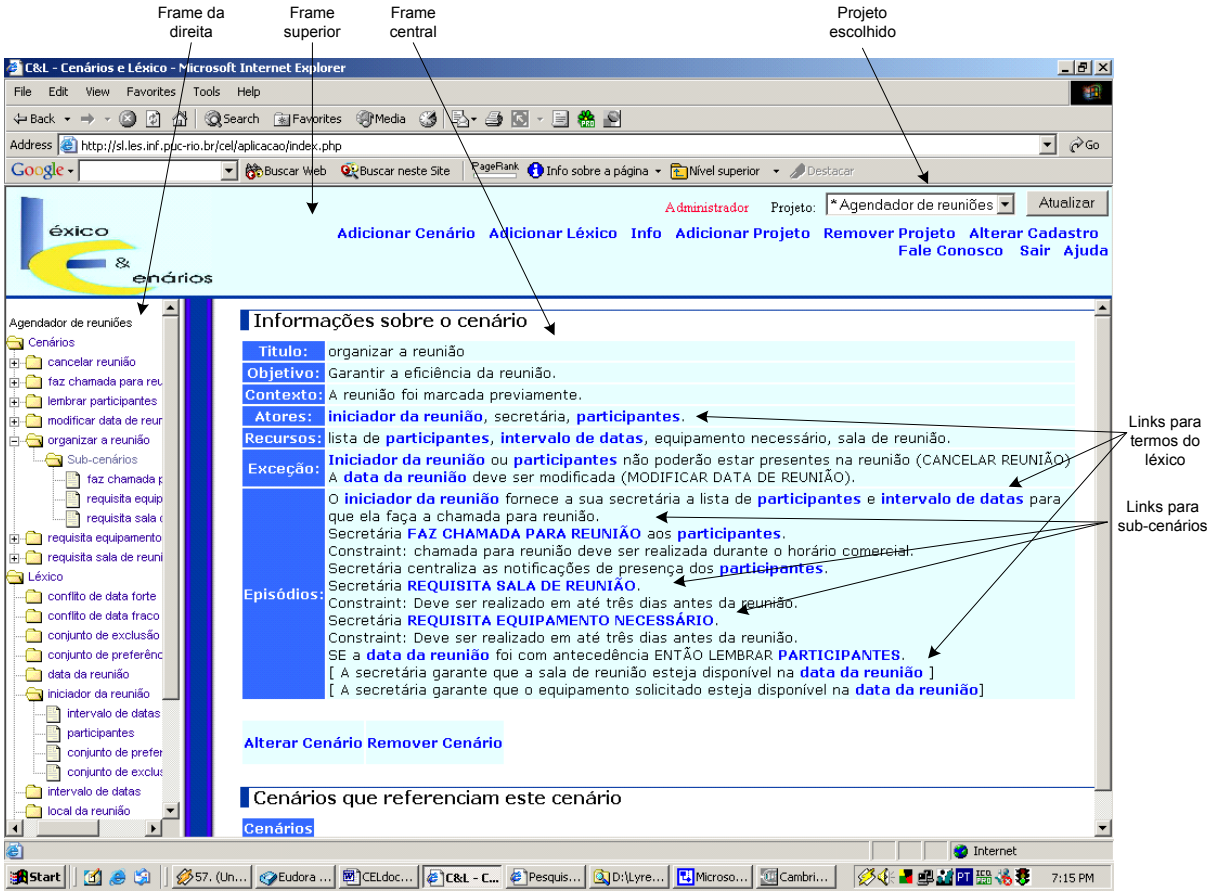

Figura 7. Interface da Ferramenta C&L: navegando pelos cenários

Na Figura 8, ilustramos um termo do léxico deste mesmo projeto, "iniciador da reunião". Os termos do léxico possuem um nome, noção, classificação, impacto e sinônimos. Este termo do léxico é mencionado pelos cenários "Organizar a reunião", "Modificar data de reunião", "Requisitar equipamento necessário" e "Faz chamada para reunião" e pelos termos "Data da reunião", "Intervalo de datas" e "Conjunto de preferência", vistos na parte inferior do *frame* central

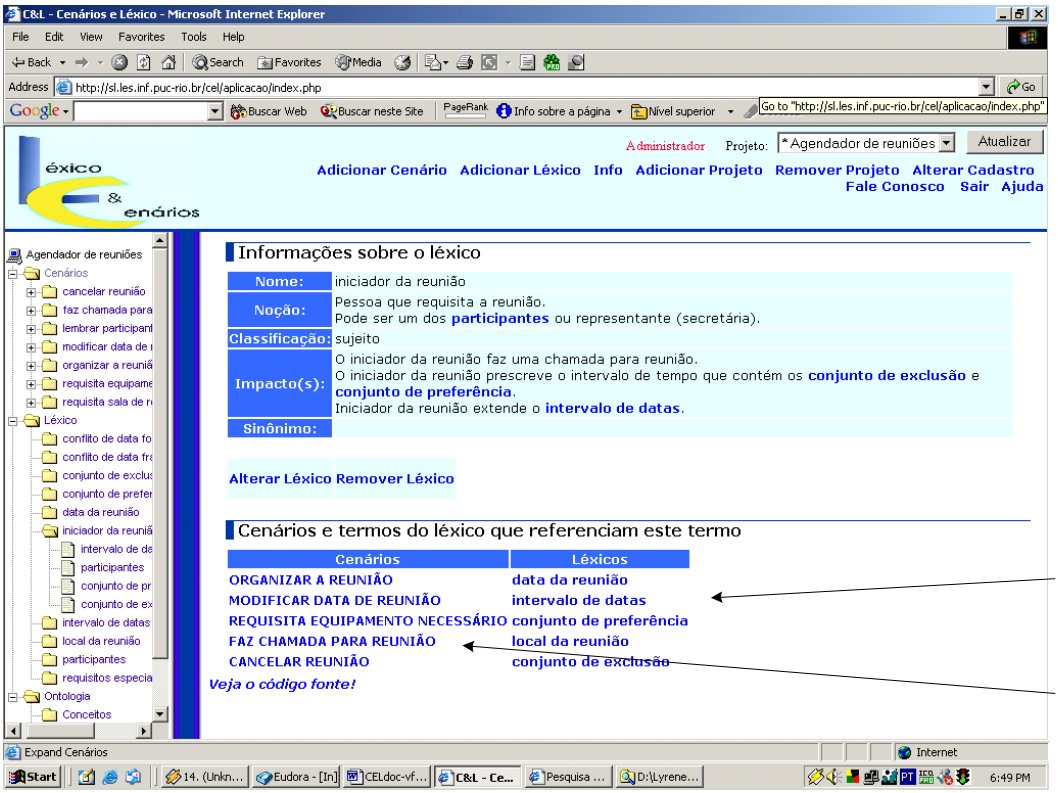

Figura 8. Interface da ferramenta C&L: navegando pelos termos do léxico

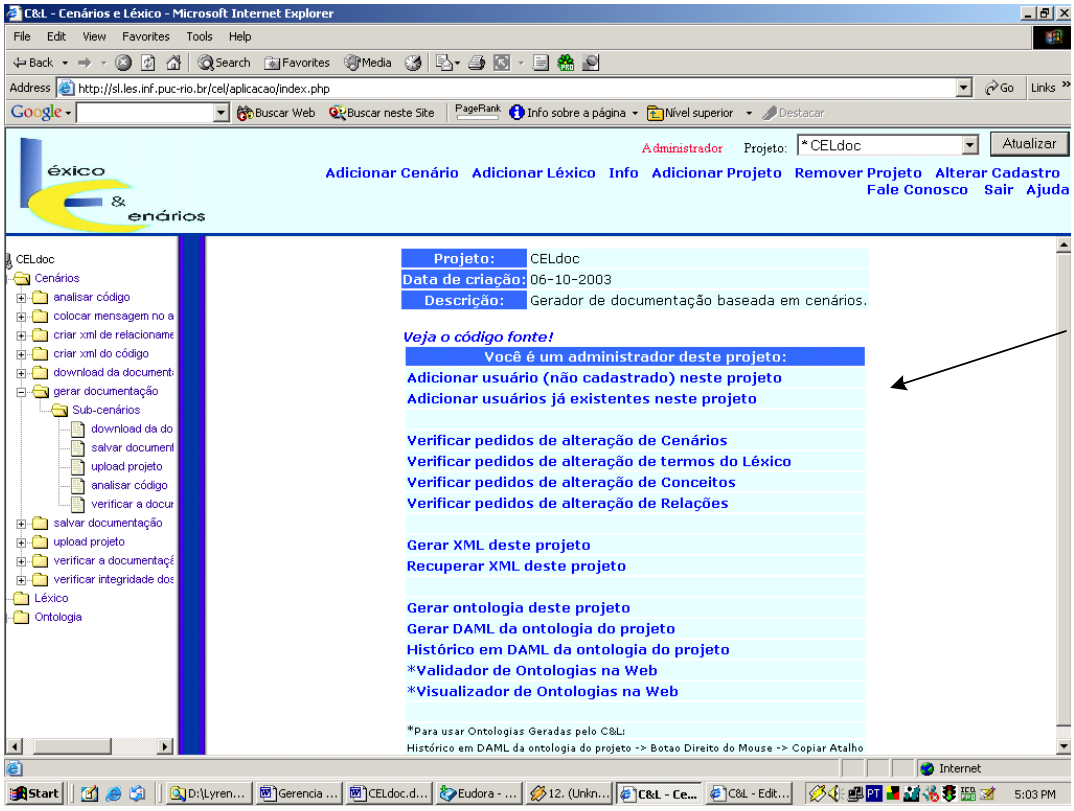

Figura 9. Interface da Ferramenta C&L: menu acessado pelo usuário administrador

Além das funções disponíveis no *frame* superior, usuários administradores podem acessar as funções ilustradas pelo *frame* central da Figura 9. A idéia que rege a modelagem de requisitos no C&L é que uma equipe de desenvolvimento possa trabalhar, em paralelo, descrevendo cenários e termos do léxico no C&L. O usuário

administrador (no C&L é aquele que cria o projeto) é o único que tem o poder de validar os pedidos de alterações. Sendo assim, o administrador do projeto pode associar usuários, estejam eles já cadastrados ou não no C&L, a um projeto.

Cada alteração realizada nos termos do léxico ou nos cenários, antes de ser efetivada, fica em uma lista de pedidos de alteração, aguardando a validação do administrador. Este recebe mensagens eletrônicas avisando quando há pedidos pendentes a serem validados. A geração do XML e da ontologia são duas funcionalidades disponíveis também apenas para o administrador do projeto. A geração do XML disponibiliza um arquivo XML com as informações dos cenários e do léxico de um projeto, na seção 3.4 detalhamos como isto é realizado e o seu propósito. A geração da ontologia foi implementada como um processo semi-automático conforme detalhamos na Seção 3.5.

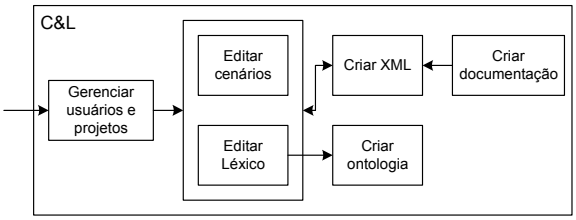

Figura 10. Estrutura funcional da ferramenta C&L

Para explicar quais funcionalidades são oferecidas pela ferramenta C&L, criamos as seguintes categorias: gerenciar usuários e projetos, editar léxico e cenários, gerar ontologia, gerar XML, gerar documentação a partir do código. Na Figura 10 ilustramos as interações existentes entre estas categorias. Nas sub-seções seguintes detalhamos as funcionalidades de cada categoria e como elas estão implementadas.

3.1 Gerência de Usuários e Projetos

Para acessar as funcionalidades da ferramenta C&L é necessário se cadastrar, passar por um processo de autenticação através de *login* e senha. Todo usuário (pessoa cadastrada no C&L) pode criar seus próprios projetos escolhendo a opção Adicionar Projeto do menu no *frame* superior, passando ele a ser o usuário administrador do projeto.

Ao criar um projeto o único usuário relacionado a este é o próprio administrador. Como administrador este usuário pode relacionar outros usuários ou outras pessoas não cadastradas no C&L. No ultimo caso, a pessoa passa a ser um usuário do C&L e passa a ter acesso aos projetos aos quais ele está relacionado. Assim, um usuário pode estar relacionado a vários projetos exercendo em cada um ou o papel de administrador ou de usuário comum.

As funcionalidades agrupadas neste componente são:

- **acessar o sistema** - o usuário acessa o sistema mediante *login* e senha, o sistema gerencia os diferentes tipos de permissões dos usuários, disponibilizando diferentes menus;
- **lembrar senha** - o usuário fornece seu *login* e o sistema envia uma mensagem para seu endereço eletrônico informando qual a senha cadastrada para ele;
- **cadastrar usuário** - o sistema não permite que dois usuários tenham o mesmo *login*;
- **adicionar projeto** - o sistema não permite que o mesmo usuário adicione dois projetos com o mesmo nome;
- **relacionar usuário existente ao projeto** - o sistema disponibiliza uma relação com todos os usuários cadastrados. O usuário seleciona aqueles a serem relacionados ao projeto;
- **relacionar usuário não cadastrado ao projeto** - o sistema disponibiliza interface para que novos usuários sejam adicionados;
- **apagar projeto** - o sistema apaga do banco de dados todas as informações referentes a um projeto;
- **acessar projeto** - no *frame* superior da interface é disponibilizada uma lista com todos os projetos aos quais o usuário corrente está relacionado. Ao selecionar um projeto o sistema disponibiliza a interface ilustrada nas Figuras 4 e 5. A estrutura em árvore disponibilizada no *frame* esquerdo apresenta os cenários, léxico e ontologia do projeto aberto. Na Figura 7 o cenário "Organizar a reunião" foi selecionado, ele é mostrado no *frame* central. Os sub-cenários deste cenário são mostrados em caixa alta no *frame* central e são mostrados como ramos da árvore no *frame* esquerdo. O título dos cenários em caixa alta são *links* que podem ser percorridos para navegação entre cenários. As outras palavras que não aparecem em caixa alta, mas estão grifadas são termos definidos no léxico do mesmo projeto, e também são *links* para navegação. Os cenários

contém *links* para sub-cenários e para termos do léxico. Os termos do léxico contém *links* apenas para os termos do léxico. Isto permite a rastreabilidade entre cenários, entre cenários e termos do léxico e entre termos do léxico. Além desta rastreabilidade, a rastreabilidade avante, própria da definição de Cenários e léxico, a ferramenta C&L disponibiliza a rastreabilidade reversa, ou seja, quais cenários e termos do léxico fazem referência ao cenário corrente ou quais termos do léxico fazem referência ao termo do léxico corrente. Na Figura 8, no *frame* central, ilustramos a rastreabilidade reversa através dos *links* para cenários e termos do léxico que fazem referência ao termo corrente. Todos os *links* são definidos automaticamente pela ferramenta, na Seção 3.2 detalhamos as funcionalidades relacionadas com a geração dos *links*.

3.2 Edição de Léxico e Cenários

Ao escolher um projeto o sistema disponibiliza as funções para editar o léxico e o conjunto de cenários daquele projeto. As informações do léxico são complementares às informações dos cenários, mas não são dependentes um do outro. Um projeto está relacionado a: zero ou um conjunto de cenários e zero ou um léxico. As funcionalidades englobadas neste grupo são:

- **adicionar cenário** - o sistema disponibiliza um formulário para que o usuário adicione as informações do cenário, aquelas definidas na Figura 1. Ao inserir um cenário, a ferramenta procura em todos os cenários do projeto por referências ao novo cenário e cria o link quando encontra. Similarmente, ele procura na descrição do cenário incluído referências aos termos do léxico e aos cenários do projeto e cria os *links* apropriados;
- **alterar cenário** - o sistema disponibiliza o mesmo formulário de adicionar cenário com as informações recuperadas do banco de dados, para que o usuário modifique o desejado. Ao modificar um cenário a ferramenta verifica se todos os *links* antes criados se mantiveram e cria os novos *links* necessários;
- **remover cenário** - o sistema remove todas os *links* para aquele cenário e em seguida remove o próprio cenário do banco de dados;
- **adicionar termo no léxico** - o sistema disponibiliza um formulário para que o usuário escreva as informações do termo, aquelas definidas na Figura 2. Ao inserir um termo, a ferramenta procura em todos os cenários e todos os termos do projeto por referências ao novo termo e cria o link quando encontra. Similarmente ele procura na descrição do termo incluído referências aos termos do léxico e cria os *links* apropriados;
- **alterar termo do léxico** - o sistema disponibiliza o mesmo formulário de adicionar termo no léxico com as informações recuperadas do banco de dados, para que o usuário modifique o desejado. Ao modificar um termo a ferramenta verifica se todos os *links* antes criados se mantiveram e cria os novos *links* necessários;
- **remover termo do léxico** - o sistema remove todas os *links* para aquele termo e em seguida remove o próprio termo do banco de dados;

As funções de adicionar cenário e adicionar termo no léxico são disponibilizadas no *frame* superior da interface, na Figura 9. As funcionalidades de alterar e remover cenários e alterar e remover termos do léxico são disponibilizadas no *frame* central da interface, para o cenário ou termo corrente, respectivamente.

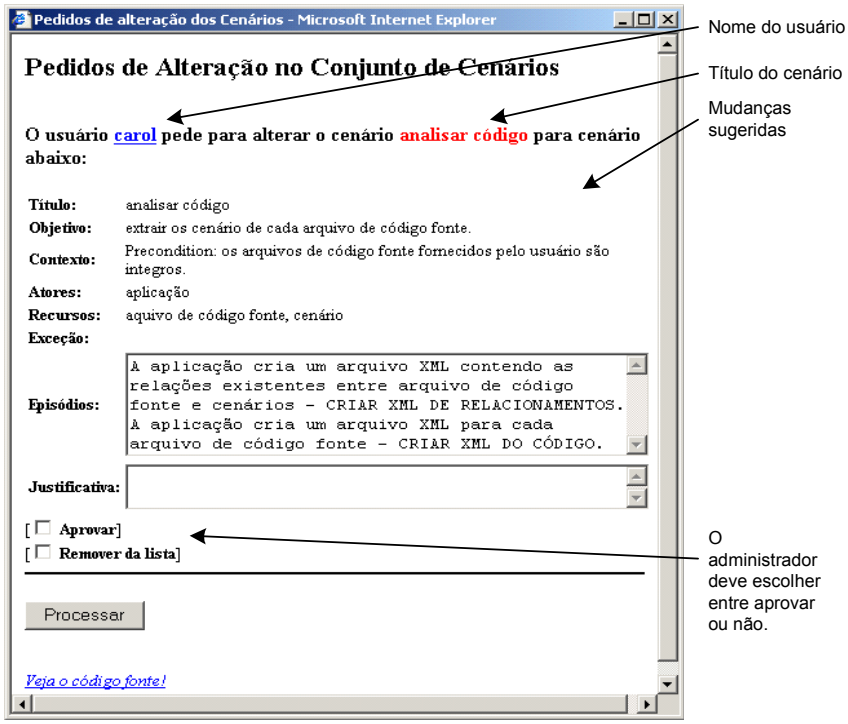

Figura 11. Lista de mudanças acessada pelo usuário administrador que o permite aprovar ou não mudanças requisitadas por outros usuários

Como dissemos anteriormente, as informações editadas só estarão disponíveis para alteração ou visualização quando o usuário administrador aprova as mudanças no projeto, ou seja, elas ficam primeiramente em uma lista de mudanças. Se as mudanças forem feitas pelo próprio usuário administrador então elas são efetivadas imediatamente, à exceção das mudanças do tipo remoção de termo do léxico e remoção de cenário. Na Figura 11, ilustramos a lista de mudanças que o usuário administrador tem acesso para aprovar ou não as mudanças requisitadas.

3.3 Geração de Ontologia

A funcionalidade de "geração de ontologia" é disponibilizada apenas para o usuário administrador no *frame* central acessado através do menu Info do *frame* superior. Para gerar uma ontologia em C&L é necessário que os termos do léxico do projeto tenham sido definidos. O processo para gerar ontologia segue o processo definido em [Breitman03] e explicado em resumo na Seção 2.2.

As funcionalidades deste grupo são:

- **gerar ontologia** - o sistema mapeia os termos do LAL classificados como **objeto** e **sujeito** para **conceitos** da ontologia, a **noção** de cada um é mapeado para a **descrição** do respectivo conceito. Depois disso, o sistema analisa os **impactos** de cada um destes termos; Para isto o sistema identifica o verbo do impacto automaticamente e a partir daí, sugere ao usuário a adição deste **verbo** à lista de **relações** da ontologia, como ilustrado na Figura 12. O usuário deve identificar se na descrição do impacto há **conceitos** ainda não cadastrados na ontologia. Nesta figura ilustramos a interface da ferramenta C&L para o processo de sugestão automática de adição de **propriedade**, descrito anteriormente. Depois de analisar os **impactos** o usuário deve informar se o termo corrente é **disjunto** a algum outro termo, na versão atual de C&L este é o único tipo de axioma gerado. Após a análise dos termos dos tipos **objeto** e **sujeito**, o sistema mapeia os termos do tipo **verbo** para **propriedades** na ontologia gerada; é necessária a intervenção do usuário caso a propriedade ainda não esteja cadastrada. O algoritmo passa então a mapear os termos do LAL do tipo **estado**; cabe ao usuário escolher entre classificar estes termos como **conceito** ou como **propriedade;** Caso a classificação seja **conceito**, então, a ferramenta o conduzirá da mesma forma se o termo fosse originalmente **sujeito** ou **objeto**; caso a classificação seja **propriedade**, a ferramenta o conduzirá como se o termo fosse originalmente verbo. Por fim, o sistema disponibiliza uma interface para que o usuário relacione os conceitos de maneira a estabelecer a hierarquia entre eles.

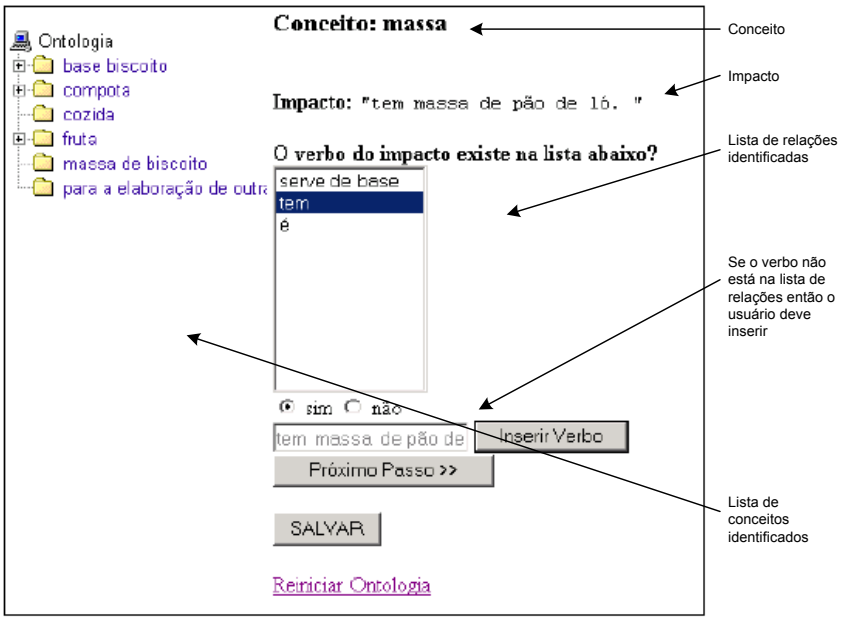

Figura 12. Interface do C&L quanto à extração dos elementos da ontologias do Impacto dos termos do LAL: identificando conceitos e relações

- **rastrear entre conceito da ontologia e termos do léxico** - quando uma ontologia é gerada, seus elementos ficam disponíveis para visualização na estrutura em árvore do *frame* esquerdo, podendo o usuário visualizá-los no *frame* central. Além da descrição dos conceitos da ontologia, no *frame* central é disponibilizada a lista de relações usadas pelo conceito corrente e os conceitos com os quais ele se relaciona, e em sua descrição, se houver referências para os termos do léxico, estas terão *links* para navegação. Na Figura 13, ilustramos a interface do C&L mostrando a rastreabilidade entre conceitos da ontologia e termos do léxico.
- **remover conceito, relação ou axioma da ontologia** - o sistema permite que conceitos, relações e axiomas sejam removidos da ontologia gerada através do *frame* central;
- **gerar daml** - uma vez que a ontologia tenha sido criada, o usuário pode gerar o arquivo daml da ontologia. Este arquivo daml segue o padrão determinado pela 3WC.
- **recuperar ontologia em daml gerada anteriormente** – cada vez que uma arquivo daml é gerado, versionado e armazenado. Nesta funcionalidade o sistema disponibiliza uma lista de arquivos daml gerados anteriormente para que o usuário selecione um para ser exibido.

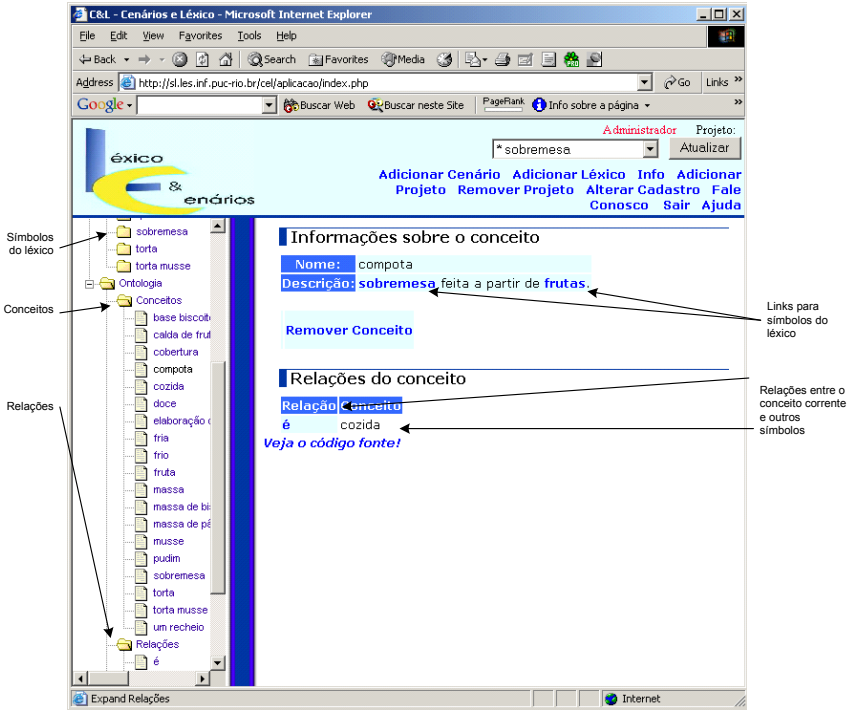

Figura 13. Interface do C&L: rastreabilidade entre ontologia e LAL

3.4 Geração de XML

Esta funcionalidade é disponibilizada pelo menu Info do *frame* superior, no *frame* central e acessada apenas pelo usuário administrador. A geração de XML tem um papel fundamental na ferramenta C&L. Ele serve como protocolo de exportação do C&L para leitura por outras pessoas e de interface com o componente de geração de documentação. As funcionalidades deste grupo são:

- **gerar XML** - o sistema recupera do banco de dados as informações referentes ao léxico e ao conjunto de cenários do projeto e gera um arquivo XML. Este arquivo XML é escrito segundo as regras gramaticais definidas por uma DTD (Data Type Document). Entretanto, como o banco de dados não foi criado já seguindo a estrutura da DTD é necessário identificar e separar seus elementos. Por exemplo, os *links* que representam referências aos termos do léxico e a outros cenários precisam ser identificados e separados de maneira a constituirem as *tags <symbol_name_reference id= "..."> e <scenario_title_reference id="..">* no documento XML, respectivamente. Ao pedir para gerar XML o usuário deve informar qual a versão do XML e se ele deseja o XML formatado ou não. Esta formatação é dada por um arquivo XSL.
- **recuperar XML criado anteriormente** - o sistema disponibiliza uma lista dos arquivos XML gerados do projeto corrente. A partir desta lista o usuário pode visualizar qualquer das versões disponíveis. Esta lista de arquivos XML representa o histórico de mudanças realizadas no projeto;
- **importar XML** - permite que o usuário indique um arquivo XML e extrai dele as informações do léxico e cenários de um projeto incluindo-as no banco de dados. O arquivo XML indicado pelo usuário tem que está de acordo com as regras definidas pela ferramenta. Estas regras são descritas através da DTD, descrita a seguir. Esta funcionalidade está em fase de desenvolvimento;
- **recuperar uma versão do projeto** - as duas funcionalidades anteriores, recuperar XML e importar XML permitem, em conjunto, retornar a uma versão anterior do projeto e reeditá-la;
- **apagar arquivo XML** - ao listar as versões dos arquivos XML geradas para um projeto, o sistema disponibiliza a função de apagar arquivo XML. Esta função remove do banco de dados a versão escolhida do XML.

A DTD criada para léxico e cenários define os tipos de dados permitidos nestes dois artefatos. Nos baseamos em [Leite00] para criar as regras gramaticais. Além disto, acrescentamos às regras as informações referentes à navegação entre cenários e léxico. Nas Figuras 14 e 15 apresentamos as DTDs definidas para o léxico e para o conjunto de cenários, respectivamente.

```
<!ENTITY % sentence "symbol_name_reference | text ">
<!ELEMENT lexicon (symbol)+>
<!ELEMENT symbol ((symbol_name), (synonym)*, (notion)+,
(behavioral_response)+)>
<!ATTLIST symbol
        type (subject | object | verb | state) "object" >
<!ELEMENT symbol_name (#PCDATA)>
<!ATTLIST symbol_name
        id ID #REQUIRED >
<!ELEMENT synonym (#PCDATA)>
<!ATTLIST synonym
        id ID #REQUIRED>
<!ELEMENT text (#PCDATA)>
<!ELEMENT notion (%sentence;)*>
<!ELEMENT behavioral_response (%sentence;)*>
<!ELEMENT symbol_name_reference (#PCDATA)>
<!ATTLIST symbol_name_reference
        id IDREF #REQUIRED>
```

Figura 14. – DTD para o léxico

```
<!ENTITY % sentence "(symbol_name_reference | text)*">
<!ENTITY % episode_sentence "(symbol_name_reference | text |
scenario_title_reference)*">
<!ENTITY % basic_sentence "(simple_sentence | conditional_sentence |
optional_sentence), (exception)*">
<!ELEMENT scenarios (scenario)+>
<!ELEMENT scenario (title, goal, context, (resource)+, (actor)+,
(episode))>
<!ELEMENT title (#PCDATA)>
<!ATTLIST title
        id ID #REQUIRED
>
<!ELEMENT goal %sentence;>
<!ELEMENT context ((geographical_location) | (temporal_location) |
(precondition))+>
<!ELEMENT geographical_location (%sentence;, (constraint)*)>
<!ELEMENT temporal_location (%sentence;, (constraint)*)>
<!ELEMENT precondition (%episode_sentence;, (constraint)*)>
<!ELEMENT text (#PCDATA)>
```

```
<!ELEMENT constraint %sentence;>
<!ELEMENT resource (%sentence;, (constraint)*)>
<!ELEMENT actor %sentence;>
<!ELEMENT episode ((sequential_group) | (non_sequential_group))+>
<!ELEMENT sequential_group ((%basic_sentence;), (%basic_sentence;)+)>
<!ELEMENT non_sequential_group ((pound_symbol), (%basic_sentence;),
(%basic_sentence;)+, (pound_symbol))>
<!ELEMENT simple_sentence (%episode_sentence;, (constraint)*)>
<!ELEMENT conditional_sentence ((if, %episode_sentence;, then,
%episode_sentence;), (constraint)*)>
<!ELEMENT optional_sentence ((bracket_open), %episode_sentence;,
(constraint)*, (bracket_close))>
<!ELEMENT symbol_name_reference (#PCDATA)>
<!ATTLIST symbol_name_reference
        id NMTOKENS #REQUIRED
>
<!ELEMENT scenario_title_reference (#PCDATA)>
<!ATTLIST scenario_title_reference
        id IDREF #REQUIRED
>
<!ELEMENT exception (%episode_sentence;)>
<!ELEMENT pound_symbol EMPTY>
<!ELEMENT if EMPTY>
<!ELEMENT then EMPTY>
<!ELEMENT bracket_open EMPTY>
<!ELEMENT bracket_close EMPTY>
```

Figura 15. DTD para o conjunto de cenários

3.5 Geração de Documentação a partir do Código - C&Ldoc

Este grupo de funcionalidade é um componente externo ao C&L que gera documentação navegável entre cenários e código. Este componente foi desenvolvido segundo as idéias descritas em [Silva03] e apresentadas resumidamente na Seção 2.1. A documentação é criada a partir dos cenários extraídos do código fonte. No código fonte eles são escritos em forma de comentários utilizando alguns delimitadores para identificação de seus elementos. Toda documentação é disponibilizada através de arquivos XML e apresentada de maneira formatada por um XSL. As principais funcionalidades deste componente são:

- **realizar *upload* de um conjunto de arquivos de código** - C&Ldoc disponibiliza um formulário para que o usuário indique em seu computador um arquivo *zipado* contendo todos os arquivos de código fonte do projeto e informe os seguintes dados: nome do projeto, descrição, tipos de arquivos do código fonte. Feito isto, o sistema copia o arquivo *zipado* para o servidor e descompacta o arquivo;
- **analisar os arquivos copiados** - o sistema verifica os arquivos de maneira a encontrar vírus, arquivos corrompidos e identificar os arquivos dos tipos indicados pelo usuário;
- **gerar o rastreamento entre cenários e código fonte** - o sistema identifica o relacionamente entre cenários e código fonte e os registra em um arquivo XML, as infomações identificadas aqui são utilizadas para definir o rastro (navegabilidade) entre cenários e código quando o sistema estiver gerando os outros documentos;
- **gerar documentação** - tendo selecionado os arquivos de código apropriados o sistema inicia o processo de extração de cenários do código. Neste processo, para cada arquivo de código é criado um arquivo XML, a menos que não haja cenários no arquivo de código. Os arquivos XML obedecem a DTD utilizada no C&L para geração de XML. Entretanto, nos arquivos XML aqui gerados tem-se uma informação a mais, o código referente a cada episódio ou exceção do cenário. O código foi inserido no arquivo XML através do tipo CDATA;
- **disponibilizar a documentação para que o usuário faça *download*** - a documentação é zipada e um link para ela é disponibilizado para o usuário.
- **criar um arquivo para registro de erros** - C&Ldoc relata todos os erros ou exceções encontradas no arquivo exceptionsC&Ldoc.txt disponibilizado junto a documentação no final do processo. Este arquivo pode ser utilizado pelo desenvolvedor para detectar que cenários estão incompletos ou incorretos segundo alguns critérios estabelecidos a partir da gramática definida, e que arquivos de código não foram comentados com cenários. Um exemplo deste arquivo de exceções é ilustrado na Figura 16.

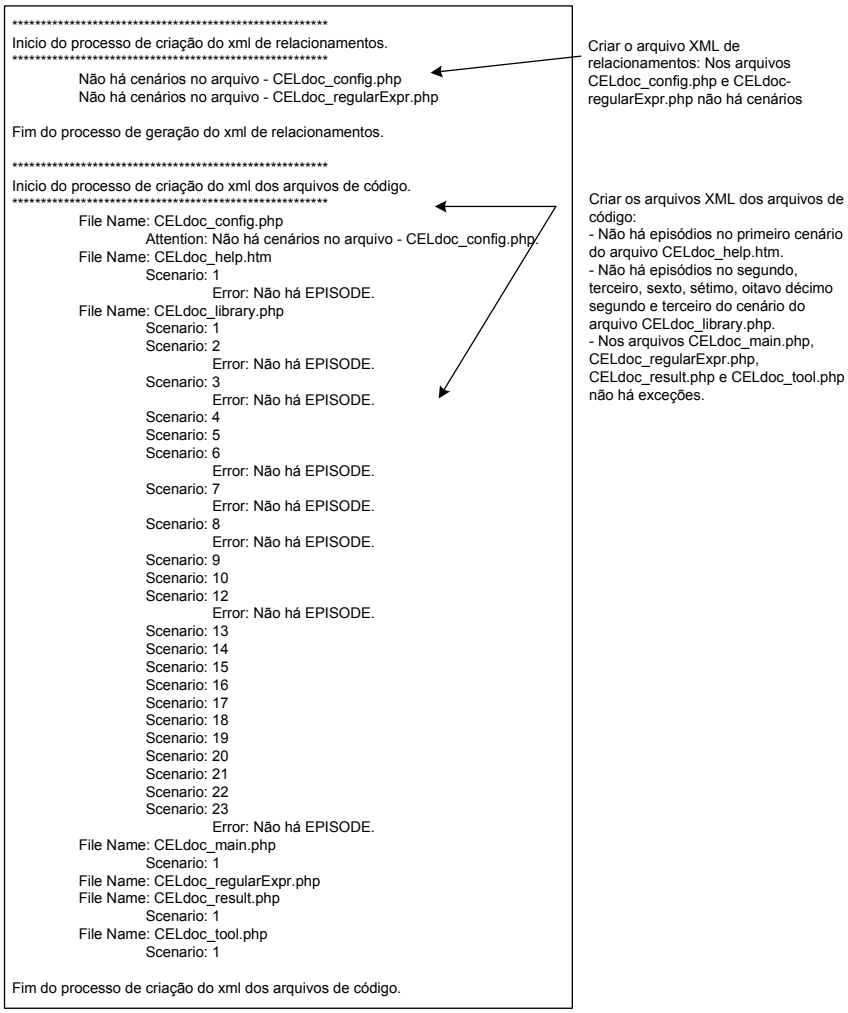

Figura 16. Exemplo do arquivo de exceções gerado pelo C&Ldoc

A documentação gerada é composta dos arquivos seguintes. Os três primeiros itens são arquivos previamente criados, ou seja, para qualquer projeto eles são os mesmos:

1. HTM - O arquivo index.htm integra a documentação. Este é o arquivo por onde o usuário deve começar a navegação pela documentação criada. Ele define os *frame*s da documentação.
2. DTDs - Três DTDs integram a documentação:
   - scenarios.dtd - determina o formato dos arquivos de código comentados com cenários, veja na Figura 15.
   - lexicon.dtd - determina o formato do arquivo com o Léxico Ampliado da Linguagem (LAL) , veja na Figura 14.
   - relationScenarioCode.dtd - determina o formato dos arquivos de relacionamentos entre código e cenários.
3. XSLs - Quatro XSLs integram a documentação:
   - scenarios.xsl - determina o formato que os arquivos de código (nome_do_arquivo_de_codigo.XML) serão apresentados para o usuário (no *browser*).
   - lexicon.xsl - determina o formato que o arquivo do léxico (lexicon.XML) será apresentado para o usuário (no *browser*).
   - home.xsl - determina o formato que o arquivo de abertura da documentação (home.XML) será apresentado para o usuário (no *browser*).
   - relationCodeScenario.xsl - determina o formato que o arquivo de relacionamentos entre código e cenário (relationCodeScenario.XML) será apresentado para o usuário (no *browser*).
   - relationScenarioCode.xsl - determina o formato que o arquivo de relacionamentos entre cenário e código (relationScenarioCode.XML) será apresentado para o usuário (no *browser*).
4. XMLs - No mínimo três XMLs integram a documentação:
   - relationScenarioCode.XML, relationCodeScenario.XML e home.XML. Estes três arquivos estão no formato definido em relationScenarioCode.dtd e possuem basicamente o mesmo conteúdo variam apenas o formato em que são apresentados para o usuário, utilizando relationScenarioCode.xsl, relationCodeScenario.xsl e home.xsl, respectivamente.

- Se no arquivo submetido pelo usuário não contiver arquivos no formato determinado ou não tiver cenários neles então C&Ldoc não gera nenhum outro arquivo XML. Caso contrário, C&Ldoc cria um arquivo XML para cada arquivo de código comentado com cenário. Estes arquivos estão de acordo com a DTD scenario.dtd e são apresentados segundo o arquivo scenarios.xsl.

5. TXT - O arquivo exceptionsC&Ldoc.txt integra a documentação. Ele relata algumas das exceções ocorridas durante o processo de geração da documentação.

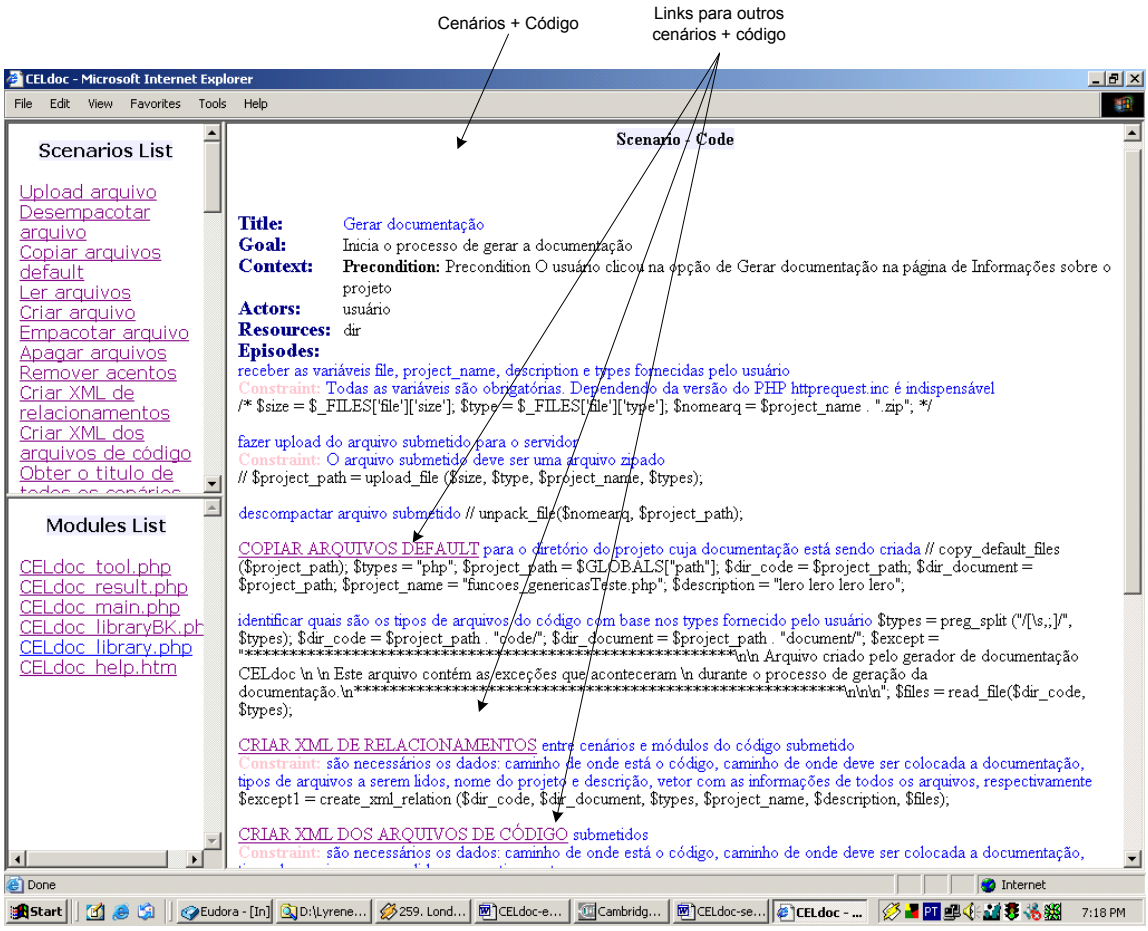

Figura 17. Exemplo de documentação gerada pelo C&Ldoc

Na Figura 17 ilustramos um exemplo de documentação gerada pelo componente C&Ldoc. No lado esquerdo da Figura 17, há a lista de cenários encontrados nos arquivos do projeto e a lista de arquivos de código do projeto. No lado direito está o cenário "Gerar documentação". Este cenário é comentário do módulo CELdoc_tool.php. As informações do cenário são intercaladas com o código. O código mostrado ainda não está no formato apropriado, mas estamos providenciando esta formatação.

4. Trabalhos Relacionados

Apesar de não haver uma ferramenta que dê suporte total à atividade de engenharia de requisitos, há algumas ferramentas que de alguma maneira a apóiam. Algumas ferramentas para requisitos são produtos comerciais, produtos acadêmicos ou experimentais, software livre, outros estão relacionados a uma ou um conjunto de técnicas ou modelos tais como produtos relacionados a ontologias, a casos de uso ou cenários, a glossários e a geradores automáticos de documentação.

Exemplos de ferramentas comerciais específicas para Engenharia de Requisitos são Doors [Doors] e RequisitePro [RequisitePro]. Elas permitem a definição de requisitos e seus atributos como exemplo, tipo do requisito, status, rastreabilidade entre requisitos. Estas ferramentas também possibilitam registrar o *rationale* associado a mudanças em requisitos. Ferramentas como Rational Rose [RationalRose] e Together [Together] que dão apoio ao processo de desenvolvimento, apesar de não focarem na atividade de requisitos dão suporte à modelagem de casos de uso. Entretanto, não oferecem técnicas para melhorar a descrição de casos de uso nem

integração entre estes e o código ou outros diagramas da UML. Ferramentas ainda mais genéricas como editores de texto também são utilizadas nesta atividade. O Microsoft Word, por exemplo, ainda é uma das ferramentas mais utilizadas para apoiar as atividades de Requisitos.

Todas estas ferramentas são produtos com um alto padrão de qualidade com propósitos um pouco diferentes da ferramenta C&L. C&L, além de ser uma ferramenta experimental que atende tanto a modelagem de requisitos quanto a criação do léxico e do rastreamento entre eles, é distribuída como software livre. C&L é fundamentada nas técnicas de cenários e léxico para dar apoio ao desenvolvimento de software.

O crescente interesse pelas técnicas de cenários e léxico tem impulsionado o surgimento de técnicas para o desenvolvimento de software fundamentadas nelas. Exemplos são, validação de requisitos [Sutcliffe02], verificação de requisitos [Duran02][McCoy03][Doorn98][Kaplan00], derivação de objetos [Fresno98], geração de documentação [Silva03], gerência por requisitos [Kroll03][Cockburn00], dentre outras. Essas técnicas estão mais ou menos bem desenvolvidas dependendo do conceito de cenário e léxico considerado. Mesmo para as técnicas bem estabelecidas há poucas ferramentas que dêem apoio e principalmente que permitam a edição de cenários e léxico em conjunto.

REM [REM][Duran02] e RUT [RUT][McCoy03] são ferramentas para gerência de requisitos que permitem a descrição de casos de uso e a verificação dos requisitos, usando *templates* próprios. REM utiliza XML e XSLT como modelos exportados da ferramenta se assemelhando a abordagem utilizada no C&L. RUT está integrada ao Rational Rose e provê algumas métricas para obter informações sobre os relacionamentos existentes entre casos de uso. RUT é uma ferramenta para *Web* desenvolvida usando as tecnologias de software livre PHP e MySQL. Elas estão relacionadas à ferramenta C&L devido ao enfoque em um tipo de cenário e ao uso de algumas tecnologias tais como XML, XSLT, PHP e MySQL. Além disso, ambas tem funcionalidades referentes à verificação de casos de uso, esta é uma funcionalidade que está sendo acrescentada a C&L.

Hear [Petersen00] é uma ferramenta para a descrição de documentação navegável de um baseline que utiliza a definição de cenários e léxico descritas na Seção 2. Ela foi desenvolvida em Java e utiliza o banco de dados MySQL. Esta abordagem foi um ponto de partida para o desenvolvimento da ferramenta C&L. Além das funcionalidades de edição, C&L promove o desenvolvimento colaborativo, automatiza a rastreabilidade entre léxico e cenários, gera ontologias e está integrada com um módulo para gerar documentação automaticamente, como apresentamos na Seção 3.

Quanto à edição de ontologias, existe no mercado uma série de ferramentas. No entanto, não encontramos nenhuma que forneça apoio a um processo completo de forma que pessoas que não são especialistas em ontologias possam desenvolvê-las. A maioria das ferramentas disponível guia o usuário apenas na modelagem do conhecimento do domínio esquecendo que é necessário antes elicitar este conhecimento. Tratam-se, basicamente, de editores de ontologias.

Exemplos são: OilEd [OilEd03] é um simples editor de ontologias. Ele utiliza as linguagens *DAML+OIL*; OntoEdit [OntoEdit03] é um ambiente de engenharia de ontologias que dá suporte a um processo (não automático) para gerar ontologias. No OntoEdit é armazenado o modelo conceitual da ontologia de forma que seja possível fazer a transformação dessa representação conceitual para a maioria das linguagens de representação de ontologias como RDF(s), XML, DAML+OIL ou F-Logic; Protègè-2000 [Protege2000] é um ambiente de plataforma independente e extensível para criação e edição de ontologias e bases de conhecimento; Chimaera [Chimaera03] é um sistema de software que apóia o usuário na criação e manutenção de ontologias distribuídas na *Web*. Suas duas principais funções são: combinação de múltiplas ontologias e diagnóstico de ontologias individuais ou múltiplas.

Estas ferramentas se diferenciam de C&L porque nela temos o LAL como uma base madura para o processo proposto em [Breitman03] de forma a automatizar a geração de ontologias, tendo a intervenção do usuário apenas naquelas em que haja absoluta necessidade. No entanto, ainda falta nesse ambiente uma interface gráfica amigável para a visualização das ontologias geradas e exportadores para as demais linguagens de representação. A idéia é que o C&L seja um *front-end* para outras ferramentas específicas para ontologias.

Quanto à geração automática de documentação, na literatura há algumas propostas para documentar o código fonte. Em [Knuth84], propõe-se a programação literária em que comentários estruturados são inseridos no código fonte e posteriormente tratados por um pré-processador. Estes comentários têm como objetivo facilitar a compreensão de um sistema complexo através da rede de ligações (denominada *WEB*) entre um componente e os outros que estão relacionados a ele diretamente. Staa em [Staa00] define um padrão para comentários e

marcadores tendo em vista a possibilidade de utilizar ou desenvolver ferramentas para geração da documentação técnica a partir dos arquivos de código. Este padrão não só define normas para mostrar a estruturação do programa como também para manter informações históricas de evolução. O uso de arquiteturas para re-construção de documentação foi explorado por Braga [Braga88] na reunião de um navegador de hipertexto com um sistema transformacional; o sistema Documentu possibilita a geração a partir de código legado de uma documentação baseada em hipertexto.

Algumas linguagens de programação possuem aplicativos para apoio à documentação de código fonte. A documentação JavaDoc é gerada pelo uso de um aplicativo da linguagem Java que utiliza comentários e *tags* HTML inseridos no código fonte com esta finalidade. O aplicativo percorre o código fonte identificando seus componentes (classes, interfaces, atributos, métodos), gerando páginas HTML com *hyperlinks* aos objetos, métodos e classes relacionados. A navegação nas janelas de documentação do software é viabilizada através dos *hyperlinks* exibidos nas próprias páginas do JavaDoc. As páginas geradas refletem o trabalho de documentação realizado pela equipe de programação envolvida no projeto. PHPDoc é uma adaptação do JavaDoc para a linguagem PHP. A linguagem PHP possibilita ainda que o código do módulo a ser executado seja exibido ao usuário através de sua função show_source() [Php03].

O gerador de documentação do C&L assemelha-se as propostas de Knuth, Staa, Braga e Javadoc para criar uma documentação baseada em cenários e léxico. Os *hiperlinks* gerados pela ferramenta retratam o rastro existente entre código e cenários. Desta forma, é possível fazer a rastreabilidade entre código e requisitos expressos nos cenários. Ter uma documentação baseada em cenários permite o desenvolvedor entender mais rapidamente a arquitetura do software e o impacto que mudanças em um requisito pode ocasionar em seus módulos. No contexto de software livre, isto pode favorecer a entrada de novos participantes em um projeto.

5. Conclusões e Trabalhos Futuros

Neste artigo apresentamos uma ferramenta experimental que é parte de um projeto de pesquisa na PUC-Rio. A ferramenta C&L dá apoio à atividade de modelar requisitos de software utilizando as técnicas de cenários e léxico. Além disso, C&L está integrada às técnicas de geração de ontologias a partir do LAL, descrita em [Breitman03], e de geração de um documento hiper-textual de rastreabilidade entre código e cenários, descrita em [Silva03].

O uso de C&L viabiliza o trabalho cooperativo entre engenheiros de requisitos, geograficamente separados, sobre um mesmo projeto. A geração automática do rastro entre cenários, entre cenários e léxico, entre termos do léxico, entre léxico e ontologia e entre código e cenários agiliza as tarefas de evolução de software, pois possibilita ao desenvolvedor entender mais rapidamente o UdI, os requisitos do usuário, a estrutura do código e como estes estão relacionados. Com a rastreabilidade o desenvolvedor torna-se mais apto a identificar o impacto que mudanças nos requisitos podem ocasionar no projeto. C&L tem sido utilizada como estudo de caso para vários experimentos do grupo de engenharia de requisitos da PUC-Rio. No momento estamos realizando estudos sobre sua interface, sobre evolução de software, sobre a aplicabilidade do processo *extreme programming* e no ensino da engenharia de software.

Como trabalhos futuros, sugerimos três conjuntos de evoluções: a elaboração de componentes que realizem novas funcionalidades, tais como, geração e visualização de diagramas de cenários, extração da lista de requisitos a partir dos cenários, elaboração de cronogramas, planejamento para evolução do projeto, componentes que dêem suporte ao reuso de projetos ou partes de projetos (cenários e léxico); funcionalidades que apóiem os processos de verificação e validação, tais como, inspeção de cenários e geração de casos de teste; e a melhoria da usabilidade, tais como, exportação de todas as informações do projeto para diferentes formatos, o desenvolvimento de um instalador e configurador de ambiente automático para ferramenta, e algumas melhorias na interface, no tratamento de exceções e na documentação disponível.

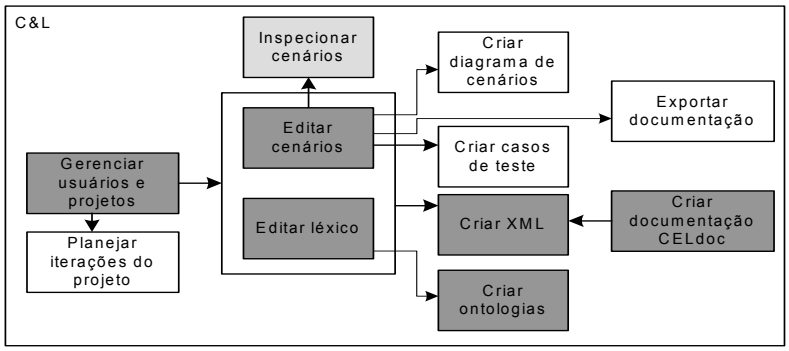

Figura 18. Estrutura funcional de futuras versões do C&L

Ilustramos alguns destes componentes na Figura 18. Os componentes de cor cinza-escuro são os que atualmente estão implementados na versão disponível do C&L, os de cor cinza-claro estão em desenvolvimento, e os de cor branca são os componentes que pretendemos implementar nas próximas versões da ferramenta. Acreditamos que a disponibilização de ferramentas integradas que dêem apoio ao desenvolvimento de software guiado por requisitos é necessária e promove um melhor nível de qualidade aos produtos de software gerados.

Agradecimentos

A ferramenta C&L é resultado do trabalho colaborativo entre os alunos das disciplinas Princípios de Engenharia de Software e Evolução de Software da PUC-Rio (Pontifícia Universidade Católica do Rio de Janeiro), desde o primeiro semestre de 2002

Referências

[Beck99] Beck, K. *Extreme Programming Explained*. 1st edition. Addison-Wesley Publishing Company, 1999.

[Braga88] Braga, C., von Staa, A., Leite, J. Documentu: A Flexible Architecture for Documentation Production Based on a Reverse-engineering Strategy. *Journal of Software Maintenance: Research and Practice*, Vol. 10, Issue 4, p.p. 279 – 303, John Wiley & Sons, Ltd., July 1998.

[Breitman02] Breitman, K., Leite, J. Managing User Stories. International *Workshop on Time Constrained Requirements Engineering*, 2002.

[Breitman03] Breitman, K.K; Leite, J.C.S.P.- Lexicon Based Ontology Construction - 2nd. *International Workshop on Software Engineering for Large Scale Multi Agent Systems* - SELMAS - ACM computer Press, Portland Oregon, 2003.

[CaliberRM] Disponível em http://www.borland.com/caliber/.

[C&L] Disponível em <http://sl.les.inf.puc-rio.br/cel>.

[Chimaera03] Disponível em <http://www.ksl.stanford.edu/software/chimaera>.

[Cockburn00] Cockburn *A. Writing Effective Use Cases*. 1st edition, Addison-Wesley, 2000.

[DBdesigner] Disponível em <http://www.fabforce.net/dbdesigner4/index.php>.

[Doors] <http://www.teLALogic.com/products/doorsers/doors/index.cfm>

[Doorn98] Doorn, J., Kaplan, G., Hadad, G., Leite, J.C.S.P. Inspeccion de Escenarios, *WER 98-Workshop de Engenharia de Requisitos*, Departamento de Informtica PUC-Rio, pp. 58-69, 1998.

[Duran02] Durán A., Ruiz-Cortés A., Corchuelo R. and Toro M. Supporting Requirements Verification Using XSLT. *Proceedings of IEEE Joint International Conference on Requirements Engineering (RE'02)*, pp.165, Essen, Germany, September 09 - 13, 2002.

[Felicissimo03] Felicíssimo C., Silva L., Breitman K., Leite J. Geração de Ontologias Subsidiada pela Engenharia de Requisitos. *Proceedings of WER03 - Workshop em Engenharia de Requisitos*, Piracicaba-SP, Brasil, Novembro 27-28, 2003, pp 255-269.

[Fresno98] Fresno M., Mauco V., Ridao M., Doorn J., Rivero L. Derivación de objetos utilizando LAL y Escenarios en un caso real. *Proceedings of WER98 - Workshop em Engenharia de Requisitos*, Maringá-PR, Brasil, Outubro 12, 1998, pp 89-98.

[Petersen00] Petersen L., Tornabene S., Leonardi M., Doorn J. HeAR, Una herramienta de Adquisición de Requistos. Anais do *WER00 - Workshop em Engenharia de Requisitos*, Rio de Janeiro-RJ, Brasil, Julho 13-14, 2000, pp 38-52.

[Hendler01] Hendler, J. Agents and the Semantic *Web – IEEE Intelligent Systems.* pp.30-37, March/April - 2001.

[Holanda03] Christoph R. Engenharia de software para software livre, *Dissertação de mestrado*. Departamento de Informática da PUC-Rio, 2004.

[IEEE98] IEEE Transactions on Software Engineering, Scenario Management edition, Vol.24, N. 12, Dec 1998.

[Kaplan00] Kaplan, G.; Hadad, G.; Doorn, J.; Leite, J.C.S.P. –Inspección del Lexico Extendido del Lenguaje– *In Proceedings of the Workshop de Engenharia de Requisitos – ER'00* – Rio de Janeiro, Brazil – 2000.

[Kroll03] Kroll P., Kruchten P. *The Rational Unified Process Made Easy: A Practitioner's Guide to Rational Unified Process*, 1st edition, Addison-Wesley, 2003.

[Leite93] Leite, J.C.S.P. and Franco, A.P.M. A Strategy for Conceptual Model Acquisition. *First International Symposium on Requirements Engineering. Proceedings. IEEE Computer Society Press*, 1993. pp. 243-246.

[Leite97] Leite, J.C.S.P, Rossi, G., Balaguer, F., Maiorana, V., Enhancing a requirements baseline with scenarios. In: *Third IEEE International Symposium on Requirements Engineering - RE97*, Proceedings. IEEE Computer Society Press, January, 1997, pp 44-53.

[Leite00] Leite, J.C.S.P., Hadad, G., Doorn, J., Kaplan, G. A Scenario Construction Process. *Requirements Engineering Journal*: Vol. 5, N. 1, Pags. 38 -- 61, (2000), Springer-Verlag London Limited.

[Maedche02] Maedche, A.; Ontology Learning for the Sematic *Web* – Kluwer *Academic Publishers*, 2002.

[McCoy03] McCoy, J. Requirements Use case Tool (RUT). *OOPSLA 2003* conference in Anaheim, CA in October 2003.

[Noy01] Noy, N.; McGuiness, D.; Ontology Development 101 – A guide to creating your first ontology – *KSL Technical Report*, Standford University, 2001.

[OilED03] Disponível em <http://oiled.man.ac.uk/>.

[OntoEdit03] Disponível em <http://www.ontoprise.de/products/ontoedit_en>.

[Protègè2000] Disponível em <http://protege.stanford.edu/>.

[RationalRose] Disponível em <http://www-306.ibm.com/software/awdtools/developer/modeler/ >

[REM] Disponível em <http://rem.lsi.us.es/REM/REM_english_main.html>

[Requisite-Pro] Disponível em < http://www-306.ibm.com/software/awdtools/reqpro/>

[Rolland98] Rolland, C.; Achour, B.; Cauvet, C.; Ralyté, J.; Sutcliffe, A.; Maiden, N.; Jarke, M.; Haumer, P.; Pohl, K.; Dubois, E.; Heymans, P. A proposal for a scenario classification *frame*work. *Journal of Requirements Engineering* vol. (3) Springer Verlag, 1998. pp. 23-47.

[RUT] Disponível em <http://satc.gsfc.nasa.gov/tools/rut/>.

[Sayão03] Sayão, M.; Inspeção de cenários. *Relatório técnico* realizado na disciplina Projeto Final de Programação, Departamento de Informática- PUC-Rio, 2003.

[Silva03] Silva, L., Sayão M., Leite J.C.S.P., Breitman, K. Enriquecendo o Código com Cenários. *Simpósio Brasileiro de Engenharia de Software* (SBES), 2003.

[Silva04] Silva, L., Leite J.C.S.P., Breitman, K. Ensino de Engenharia de Software: Relato de Experiências. *Workshop de Educação em Computação* (XII - WEI), Agosto - 2004.

[Sommerville01] Sommerville, Ian. *Software Engineering*, 6th edition, Addison-Wesley, 2001.

[Staa00] Staa, Arndt von. *Programação Modular: Desenvolvendo programas complexos de forma organizada e segura*. Campus, Rio de Janeiro, 2000.

[Sutcliffe02] Sutcliffe A.G. & Gregoraides A. Validating Functional System Requirements with Scenarios, *Proceedings 11th International Conference on Requirements Engineering*, IEEE Computer Society Press, 181-188, 2002.

[Together] Disponível em <http://www.borland.com/together/ >.

[W3Consortium03] Disponível em <http://www.w3.org>.

[Weidenhaupt98] Weidenhaupt, K.; Pohl, K.; Jarke, M.; Haumer, P. Scenario Usage in system development: current practice IEEE Software Vol. 15 No.2 March, 1998. pp.34-45.