SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito: 15.01.2004

Assinatura:

DocRationale: uma ferramenta para suporte a design rationale de artefatos de software

Simone Domingues Francisco

Orientadora: *Profa. Dra. Renata Pontin de Mattos Fortes*

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências de Computação e Matemática Computacional.

USP – São Carlos
Janeiro/2004

A Comissão Julgadora:

Profa. Dra. Renata Pontin de Mattos Fortes

Profa. Dra. Maria da Graça Campos Pimentel

Profa. Dra. Eliane Martins

Eliane Martins

*“Não se mede o valor de um homem
pelas suas roupas ou pelos bens que
possui. O verdadeiro valor de um
homem é o seu caráter, suas idéias e a
nobreza dos seus ideais.”
(Charles Chaplin)*

*Aos meus pais,
Nélson e Ignez,
e aos meus irmãos,
Newton, Almir e Regiane.*

Agradecimentos

Agradeço à Deus pela vida e à Nossa Senhora Aparecida pela fé que me deu coragem durante os momentos difíceis!

Aos meus pais, Néelson e Ignecz, pela formação que me deram, pela confiança e amor depositados em mim, pela força e pela compreensão de minha ausência em virtude do trabalho!

Aos meus irmãos, Newton, Almir e Regiane, pelo amor e carinho, pelas conversas, pelo incentivo!

Às minhas sobrinhas, Giovana e Graziela, pela alegria, amor e carinho!

À minha orientadora, Renata, pelo apoio e confiança em mim depositada no desenvolvimento deste trabalho!

À Sarita, minha companheira de república, minha amiga, minha “mãezinha”, sempre do meu lado em todos os momentos, principalmente nos de dificuldade no trabalho, me dando conselhos, me orientando, e nos de dificuldade na saúde, cuidando de mim! Valeu mesmo, Sa!!! :)

À Naninha, minha amigona de todas as horas... seu ouvido deve estar com calo de tanto me ouvir :W Eu precisaria mais que uma página para te agradecer tudo... as conversas, o apoio, os conselhos, a alegria, a diversão... e, principalmente, a sua amizade!!!! Nanis, você é demais!!!!!!! :D

À Thaise, minha “vizinha” de computador, companheira de conversas e academia, minha amiga! Valeu por tudo!!! :)

Ao pessoal da Academia Vibração, Thaise, Glaucia, Terence, Marília, Alex (Tio), Viviane, Vanessa, Adriana, Márcia, Osnete, Bira, Fernando, Alê, André Rocha, Michele, Tânia, Helena e Erika pelos momentos de descontração e alegria! Vocês são *show*! E viva os *Bodys*!!!! :D

Ao Ades, André (Primo), KLB (Sandro), Auri, André Freire, Débora e Cláudia pelo auxílio no meu trabalho.

Ao pessoal do Labes, Ades, Thaise, André (Primo), Mateus, Gelza, Tati, Ellen (Boo), Andrea (Dinda), Erika, Auri, KLB (Sandro), Alê, Maris, Elisa, Débora, André Rocha, Bira, Valter (Vagner), Lu, Osnete, Tânia (Toinha), Fabiano, Otávio e Marco pelo companheirismo, alegrias e apoio. Valeu pelo ambiente animado!!! :D

Aos amigos dos outros laboratórios, Fernando, Hermes, Douglas, Kali e Ju, sempre presentes.

Aos meus avós (*in memoriam*), cunhadas, tios e primos próximos que participaram da minha formação e, mesmo com a distância que agora nos separa, sempre encontraram um jeitinho de reunir todos! :)

Aos amigos, Maria Julia, Agnes, Dagmar e Erik, que mesmo de longe sempre me deram uma força!

Ao CNPq e aos meus pais pelo apoio financeiro.

E a todas as pessoas que contribuíram direta ou indiretamente para a realização deste trabalho.

Sumário

Resumo	xiii
Abstract	xv
1 Introdução	1
1.1 Objetivo	3
1.2 Organização do Trabalho	3
2 Fundamentos sobre <i>Design Rationale</i>	5
2.1 Considerações Iniciais	5
2.2 Definição	6
2.3 Benefícios e Dificuldades de DR	6
2.4 Sistemas de DR	8
2.5 Abordagens para construção de Sistemas de DR	11
2.5.1 Abordagem orientada por processo	11
2.5.2 Abordagem orientada por característica	12
2.5.3 Abordagem híbrida	13
2.6 Esquemas de Representação	13
2.6.1 IBIS	17
2.6.2 PHI	20
2.6.3 QOC	21
2.6.4 DRL	23
2.6.5 Comparação dos esquemas de representação da perspectiva de argumentação	25
2.7 Captura de DR	26
2.8 Recuperação de DR	29
2.9 Considerações Finais	31
3 Suporte a DR	33
3.1 Considerações Iniciais	33
3.2 Suporte direto ou indireto a DR	34
3.2.1 Suporte direto a DR	34

3.2.2	Suporte indireto a DR	40
3.3	CoTeia	41
3.4	GroupNote	45
3.5	Considerações Finais	48
4	DocRationale	49
4.1	Considerações Iniciais	49
4.2	Requisitos	50
4.3	O Suporte a DR na DocRationale	53
4.4	Modelagem	53
4.5	Arquitetura	56
4.6	Implementação	59
4.7	O Produto Desenvolvido	61
4.8	Considerações Finais	71
5	Conclusões	73
5.1	Contribuições	74
5.2	Trabalhos Futuros	74
5.3	Considerações Finais	75
	Referências Bibliográficas	77
A	Modelagem da ferramenta DocRationale	83
A.1	Diagrama de casos de uso	83
A.2	Esquemas, ADVs e ADVcharts	84
A.2.1	Esquema Conceitual	84
A.2.2	Esquemas Navegacional e Contexto Navegacional	86
A.2.3	ADV's	87
A.2.4	ADVcharts	95

Lista de Figuras

2.1	Arquitetura geral de um sistema de DR	10
2.2	Modelo IBIS (Conklin & Begeman, 1988)	19
2.3	Modelo PHI (Wiegeraad, 1999)	21
2.4	Exemplo PHI (Hu et al., 2000)	22
2.5	Modelo QOC (Wiegeraad, 1999)	23
2.6	Modelo DRL (Wiegeraad, 1999)	25
3.1	Interface do gIBIS (Conklin & Begeman, 1988)	35
3.2	Matriz de decisão (Wiegeraad, 1999)	36
3.3	Reunião de projeto (Hammond et al., 2002)	39
3.4	MER da CoTeia (Arruda Jr. & Pimentel, 2001)	42
3.5	Página Inicial da CoTeia – lista de todas as <i>swikis</i> criadas e disponibilizadas para leitura e acesso	42
3.6	Exemplo de Página da CoTeia	43
3.7	Edição de Página na CoTeia	44
3.8	Criação/Edição de páginas (Arruda Jr. et al., 2002)	44
3.9	Modelagem Conceitual do GroupNote (Pimentel et al., 2001)	45
3.10	Tela da ferramenta WebNote: apresentação das anotações e operações sobre as mesmas	46
3.11	Inserção de uma anotação na ferramenta WebNote	46
3.12	Tela inicial da anotação a partir da CoTeia	47
3.13	Tela de inserção de uma anotação a partir da CoTeia	47
3.14	Tela com informações sobre uma determinada anotação a partir da CoTeia	48
4.1	Informações a serem manipuladas na ferramenta DocRationale	52
4.2	Diagrama de casos de uso da ferramenta DocRationale	54
4.3	Esquema Conceitual da ferramenta DocRationale	55
4.4	Visão Geral da integração da DocRationale com a CoTeia e GroupNote	57
4.5	Arquitetura da ferramenta DocRationale	58
4.6	Ligações entre DocRationale , CoTeia e GroupNote	59
4.7	Tela com identificação de partes da ferramenta DocRationale	62
4.8	“Porta de entrada” da ferramenta DocRationale	62

4.9	Tela de <i>login</i>	63
4.10	Tela de projetos	64
4.11	Tela de artefatos	65
4.12	Tela de projeto	66
4.13	Tela de equipe	66
4.14	Tela de questões	67
4.15	Tela de artefato	68
4.16	Tela de projeto na CoTeia	68
4.17	Tela de artefato na CoTeia	69
4.18	Tela de questão no GroupNote	70
4.19	Tela de <i>logout</i>	70
A.1	Diagrama de casos de uso da ferramenta DocRationale	84
A.2	Esquema Conceitual da ferramenta DocRationale	85
A.3	Esquema Navegacional da ferramenta DocRationale – visão do administrador de projetos, do membro e do visitante	86
A.4	Esquema Navegacional da ferramenta DocRationale – visão do administrador de usuários	86
A.5	Esquema de Contextos de Navegação da ferramenta DocRationale	88
A.6	ADV da ferramenta DocRationale	89
A.7	ADVs encontrados no ADV da ferramenta DocRationale	90
A.8	ADVs de mais baixo nível da ferramenta DocRationale – Listar I	92
A.9	ADVs de mais baixo nível da ferramenta DocRationale – Listar II	93
A.10	ADVs Inserir/Editar/Localizar e Excluir	94
A.11	ADVs Autorizar/Esqueceu a senha e Registro	94
A.12	ADVs Projeto e Artefato	94
A.13	ADV Finalizar	95
A.14	ADVs Páginas CoTeia e GroupNote	95
A.15	ADVchart Login	96
A.16	ADVchart Logout	96
A.17	ADVchart Listar Projetos	97
A.18	ADVchart Listar Artefatos	98
A.19	ADVchart Listar Questões	99
A.20	ADVchart Listar Equipe	100
A.21	ADVchart Listar Fases/Atividades	101
A.22	ADVchart Listar Usuários	102
A.23	ADVchart Listar Candidatos	103
A.24	ADVchart Inserir/Editar/Localizar	103
A.25	ADVchart Excluir	104
A.26	ADVchart Finalizar	104
A.27	ADVchart Projeto	105
A.28	ADVchart Artefato	105

Resumo

Uma grande quantidade e variedade de artefatos é gerada durante o processo de desenvolvimento de um *software*. A documentação por meio dos diversos artefatos de tal processo é importante para proporcionar um bom conhecimento do *software*, além de tornar menos complicada a sua manutenção e reuso. Tais documentos, em geral, não registram informações adicionais relativas às alternativas, escolhas e decisões – *Design Rationale* (DR) – feitas durante a elaboração de cada artefato. O armazenamento e recuperação de DR dos artefatos de *software* tornaria mais simples a sua manutenção e facilitaria seu reuso, já que possibilita que o *software* torne-se mais completo de informações. O presente trabalho apresenta a ferramenta **DocRationale** que captura, armazena e recupera DR de artefatos de *software*. A **DocRationale** enfatiza a colaboração entre os desenvolvedores e a anotação estruturada por meio de uma simplificação de um modelo de representação de DR.

Abstract

During the software development process a large number of different artefacts is generated. Documentation is an essential activity, since it promotes a better software understanding and eases its maintenance and reuse. In general, the documents do not record additional information related to the alternative options, choices and decisions – in other words the Design Rationale (DR) – evaluated during the design of an artefact. The storage and retrieval of DR software artefacts would further improve its maintenance and reuse potential since the software would be easier to understand. This work presents a tool – **DocRationale** – to capture, store and retrieve design rationale of software artefacts. This tool promotes collaboration between software developers and uses structured annotation to simplify the DR conceptual model.

Introdução

As atividades da área de Engenharia de *Software* em geral envolvem uma grande quantidade e variedade de tipos de artefatos, os quais são gerados durante o processo de desenvolvimento de *software*. Um artefato pode ser visto como qualquer informação gerada, alterada ou usada no decorrer do processo de desenvolvimento (Jacobson et al., 1999). Sendo assim, um artefato pode estar na forma de um modelo, um elemento do modelo (uma classe, um subsistema, etc), um documento, um arquivo contendo o código fonte, um arquivo com o código executável, etc. Geralmente, tais documentos gerados são relacionados a outros artefatos, relatando informações sobre eles. Nesses documentos, muitos tópicos são repetidos em diferentes níveis de detalhamento e têm uma relação entre si (Bombani et al., 2000). Com isso, o uso de hiperdocumentos se apresenta como uma boa alternativa para o armazenamento e a recuperação das informações contidas nesses documentos.

O armazenamento e a recuperação da documentação de *software* visam minimizar os altos custos da manutenção de *software* e por isso constituem atividades básicas contidas nos diversos métodos e técnicas da Engenharia de *Software* desde os primeiros tempos. De fato, os documentos gerados durante o processo de desenvolvimento de *software* são deveras importantes por permitir a redução de muitos problemas na manutenção do *software* e facilitar o seu reuso (Pfleeger, 2001).

Uma das linhas de pesquisa que tem proporcionado alternativas de suporte ao controle das informações provenientes da documentação de projetos é denominada “Gerenciamento

de Conhecimento” (Rus & Lindvall, 2002), que possui contribuições da área de Inteligência Artificial. Observa-se que as noções de DR são elementos que compõem as informações de ferramentas de conhecimento. O foco deste trabalho é o de um suporte ao processo de *software*, por meio de colaboração na elaboração dos artefatos de *software*. Assim, foram pesquisadas com mais detalhes, as razões de projeto (*Design Rationale*), referentes à documentação no processo de *software*, mas sem os mecanismos de Inteligência Artificial necessários ao Gerenciamento de Conhecimento.

O acesso aos documentos é necessário para uma boa manipulação e utilização do *software*. Dessa forma, pesquisas constantes têm sido realizadas visando garantir a fidelidade das informações contidas nos documentos em relação ao estado do *software* em utilização.

As informações relativas à decisão, que em geral ocorre durante a elaboração dos documentos, muitas vezes são perdidas devido a falta de seu registro. A ausência desses registros pode levar a uma repetição de erros anteriormente cometidos. O registro das informações de decisão se mostra útil para decisões de outros projetos, pois erros e alternativas já investigadas seriam melhor reaproveitadas (Souza et al., 1998), uma vez que muitas soluções discutidas e adotadas em um projeto, podem ser relevantes a outros. Além disso, o esforço exigido para manutenção poderia ser reduzido. Um outro aspecto é que, em geral, um grande número de desenvolvedores pode estar envolvido em um único processo de produção de *software*. Ou seja, são várias pessoas fazendo consulta aos mesmos documentos, os quais são elaborados por parte dos desenvolvedores. Dessa forma, por mais bem escrito que esteja um documento, quem não o escreveu pode vir a ter dificuldades de entender porque determinada escolha ou decisão foi feita daquela maneira. Por isso, é interessante o armazenamento das razões - *Design Rationale* - que determinaram a elaboração daquele documento e da forma como foi conduzida, facilitando a compreensão desse por terceiros, ou mesmo para aqueles que o escreveu, visando a releitura para uma possível alteração após algum tempo.

Diversas definições são encontradas para *Design Rationale* - (Burge & Brown, 2000; Gruber & Russell, 1991; Hu et al., 2000; Souza et al., 1998), mas todas seguem a mesma idéia. De forma geral, neste trabalho adotamos a concepção de que *Design Rationale* - DR - são as informações obtidas durante a elaboração de cada artefato do projeto, como uma “receita” indicando todos os passos tomados, as discussões e alternativas investigadas, justificando o artefato resultante.

Em vista das vantagens do uso de DR, estudos (Conklin & Begeman, 1988; Hammond et al., 2002; Kato et al., 2002; Lee, 1990; Myers et al., 1999; Robbins & Redmiles, 1998) têm sido realizados para a elaboração de sistemas que ofereçam suporte ao armazenamento de DR, visando sua recuperação para auxiliar o entendimento do projeto e registrar experiências obtidas em suas implementações. Em geral, os sistemas de DR propostos na literatura ou

acarretam muito esforço ao desenvolvedor na captura de DR e facilitam sua recuperação, ou acarretam pouco ou nenhum esforço ao desenvolvedor e dificultam sua recuperação.

1.1 Objetivo

O objetivo deste trabalho é construir uma ferramenta **DocRationale** - que forneça suporte a DR, armazenando os artefatos gerados durante o ciclo de vida do *software* e suas respectivas “discussões” e decisões - DR.

Tendo em vista o estudo sobre DR e sobre os sistemas de apoio a DR realizado durante este trabalho, procurou-se elaborar a **DocRationale** de forma simples, visando sua utilização pelos desenvolvedores. Buscando um equilíbrio entre as opções encontradas durante o estudo, a ferramenta **DocRationale** foi desenvolvida de maneira que acarreta um certo esforço aos desenvolvedores, fazendo uso de um esquema simples de representação de DR e, que por ser estruturado, possa vir a facilitar a recuperação do DR armazenado.

Pode-se dizer que tal ferramenta visa proporcionar:

- Acesso aos artefatos gerados durante o processo do *software*;
- Acesso ao *Design Rationale* (DR);
- Melhor organização e clareza das idéias que levaram às decisões de cada fase de desenvolvimento, auxiliando não somente à equipe de projeto, mas também cada desenvolvedor sozinho;
- Melhor diálogo entre os desenvolvedores, fazendo com que os mesmos entendam melhor o que cada um argumenta a cada fase do desenvolvimento do *software*;
- Menor probabilidade de erros passados serem cometidos novamente;
- Auxílio a novos integrantes da equipe de projeto;
- Redução de problemas durante a manutenção.

1.2 Organização do Trabalho

Esta dissertação apresenta a revisão bibliográfica necessária para este trabalho de mestrado e a ferramenta **DocRationale** desenvolvida.

O Capítulo 2 apresenta a revisão bibliográfica sobre *Design Rationale*. Nesse capítulo encontram-se as definições, os benefícios do uso de DR e os fatores que dificultam sua

adoção. Também são mostrados os sistemas de suporte a DR, as abordagens e os esquemas de representação utilizados pelos sistemas de DR, além das formas de captura e recuperação de DR. No Capítulo 3, são descritas ferramentas/sistemas/propostas de suporte direto ou indireto a *Design Rationale*, especialmente a ferramenta CoTeia e o serviço GroupNote que foram utilizados no desenvolvimento da **DocRationale**. O Capítulo 4 mostra a ferramenta de suporte a *Design Rationale* - **DocRationale** – apresentando os seus requisitos e integração com a CoTeia e GroupNote, sua arquitetura, informações sobre sua modelagem e implementação, além da exibição das telas da ferramenta. O Capítulo 5 mostra as contribuições deste trabalho e os trabalhos futuros. No Apêndice A é apresentada a modelagem da ferramenta **DocRationale**.

Fundamentos sobre *Design Rationale*

2.1 Considerações Iniciais

A importância da documentação no processo de *software* é incontestável, pois visa a proporcionar um bom conhecimento do *software*, além de tornar menos complicada a sua manutenção e facilitar o seu reuso. Os documentos gerados durante o processo de desenvolvimento de *software* registram as informações referentes às decisões finais de determinada fase (Burge & Brown, 2000); em geral, os desenvolvedores não têm costume, cultura ou hábito de registrar cada alternativa investigada.

As informações sobre as opções disponíveis e as decisões que levaram a uma determinada escolha se registradas, como é feito com a decisão final, tornaria o *software* mais fácil de entender. Além disso, facilitaria a sua manipulação, seja para manutenção, reuso ou quando da entrada de novos integrantes na equipe responsável pelo *software*.

Design Rationale (DR) oferece, justamente, o suporte para armazenamento dessas informações adicionais aos artefatos gerados durante o processo de *software*, visando a sua recuperação, para auxiliar o entendimento e tirar proveito da experiência de seu projeto.

As próximas seções apresentam os fundamentos sobre DR encontrados na literatura. A seguir, as definições de DR, os benefícios do seu uso e os fatores que dificultam sua adoção. Do ponto de vista do estado da arte, são apresentadas descrições, abordagens para construção e esquemas de representação de sistemas de DR, formas de captura e de recuperação de DR.

2.2 Definição

Diversas definições do termo *Design Rationale* são encontradas na literatura. Algumas delas são apresentadas a seguir.

Para Gruber & Russell (1991), DR é definido como o conjunto de explicações que responde a questões sobre um projeto. Essas explicações são geradas a partir de conhecimento dos artefatos e das atividades de projeto.

Segundo Souza et al. (1998), DR é a documentação das decisões de projeto com suas respectivas justificativas, as opções consideradas, as avaliações e argumentação que levaram a determinada decisão.

De acordo com Hu et al. (2000), DR é a justificativa da decisão tomada para elaboração de um artefato ou parte dele, incluindo deliberações, razões¹, alterações e todas as decisões intermediárias tomadas no processo do projeto desse artefato.

Já para Burge & Brown (2000), DR consiste das decisões elaboradas durante o processo do projeto e as razões que levaram a essas decisões.

Em vista dessas definições, pode-se observar que existe um consenso de que DR se constitui de informações sobre as decisões tomadas durante o projeto; portanto, considerou-se para construir DR, as informações que se apóiam nas diferentes alternativas consideradas no processo de tomada de decisão para o desenvolvimento de cada artefato de um projeto.

2.3 Benefícios e Dificuldades de DR

Um grande número de benefícios pode ser obtido a partir de DR armazenado. Dentre eles, os mais destacados na literatura são apresentados a seguir.

DR proporciona maior visibilidade das idéias, uma metodologia estruturada para as decisões e a segurança de que as decisões não estão sendo feitas em uma base arbitrária. Quando associado ao projeto, DR pode ajudar a avaliar as conseqüências de uma alteração no sistema e decidir se as partes do sistema devem ser adaptadas ou refeitas (Monk et al., 1995).

A limitação de memória humana é um outro motivo para se fazer o armazenamento de DR. Esse armazenamento permite que, depois de certo tempo, informações sobre as decisões feitas, o porquê delas, e quais alternativas foram consideradas, sejam lembradas, possibilitando que erros já cometidos sejam evitados e a limitação da memória humana

¹No dicionário Michaelis (Michaelis, 2002) o termo razão possui os sinônimos: raciocínio, pensamento, opinião, julgamento, explicação, motivo, argumento. Neste texto, razão e raciocínio muitas vezes são utilizados indistintamente.

não seja sua principal causa. Também é comum que soluções discutidas e adotadas em um projeto possam ser relevantes para adoção em outro. Assim, se as razões de projeto gravadas puderem ser facilmente acessadas, os desenvolvedores de novos projetos poderão ser beneficiados das discussões e das decisões realizadas em projetos anteriores (Souza et al., 1998).

Grande parte do esforço da fase de manutenção se concentra no entendimento do sistema para que as alterações sejam feitas corretamente. DR pode auxiliar na obtenção do conhecimento de como e porquê o sistema foi projetado de tal maneira. Além disso, DR pode ser utilizado para verificação e geração de documentação, sendo que a revisão de DR, após sua elaboração, pode ser usada para se encontrar erros no desenvolvimento (Souza et al., 1998).

Através de DR, é possível que haja melhor comunicação entre os membros da equipe de projeto, entre os projetistas e os usuários e, entre a atual equipe de projeto e as futuras que queiram construir ou reusar partes de projetos já desenvolvidos. DR é também uma ferramenta importante para um novo membro da equipe que queira ou precise se inteirar do projeto (Souza et al., 1998).

Dessa forma, entre os benefícios estudados pode-se resumir:

- Redução da arbitrariedade das decisões e alternativas de projeto;
- Melhor avaliação das consequências de alterações no sistema;
- Recuperação das decisões feitas;
- Utilização das razões arquivadas para beneficiar outros projetos;
- Auxílio na manutenção do sistema;
- Possibilidade de se encontrar erros no desenvolvimento do sistema;
- Melhor comunicação entre as pessoas envolvidas direta ou indiretamente no projeto.

Porém, mesmo com toda essa lista de vantagens, os sistemas de DR têm sido muito pouco adotados na prática. Entre os itens que se apresentam como obstáculos, podem-se citar (Hu et al., 2000):

- Dificuldade encontrada pelos desenvolvedores na busca de DR: quando o sistema adota a abordagem orientada por processo (a ser visto na seção 2.5.1), uma possível recuperação do DR armazenado pode ser feita por meio de navegação. Essa navegação pode se tornar cansativa ou mesmo não ser bem sucedida, se a quantidade de informação armazenada for muito grande, uma vez que a navegação disponibiliza toda a informação sem qualquer tipo de discriminação;

- Diferença do formato encontrado entre a informação fornecida pelos desenvolvedores e a representação do DR: quando a captura de DR não é automática, os desenvolvedores podem ficar em dúvida sobre que tipo de nó alocar para determinada informação e como classificá-la de acordo com os tipos de nós do sistema. Se a captura é automática, o computador é responsável por fazer essa escolha e, para isso, é necessário o uso de sistemas auxiliares com “inteligência” para tal tarefa;
- Os sistemas não suprem as necessidades específicas de cada empresa: por mais que o sistema envolva as mais diversas necessidades, sempre existem algumas necessidades particulares a cada empresa que não são entendidas;
- Tempo gasto para o armazenamento do DR: seja durante a “discussão” ou depois da mesma, os sistemas exigem um certo tempo das pessoas para a captura e armazenamento do DR, tempo este que poderia estar sendo utilizado em outras atividades do desenvolvimento do projeto;
- Interferência na progressão natural das atividades de projeto: é um grande desafio conseguir fazer uma captura de informações sem que a mesma venha a interferir nas atividades habituais do desenvolvimento do processo.

Pode-se observar que na maioria das desvantagens que se apresentam não há uma concentração de problemas que estejam relacionados às atividades de captura, representação ou recuperação do DR. Mas, neste trabalho, as atividades de captura e armazenamento foram mais investigadas de forma a subsidiar a identificação das funcionalidades principais da ferramenta a ser desenvolvida.

2.4 Sistemas de DR

Quando DR se refere à descrição de como e porquê as decisões de projeto são realizadas durante o processo de desenvolvimento, os sistemas de DR visam a fornecer suporte por meio de algum auxílio ao processo, capturando o conhecimento, as suposições e as razões das decisões de projeto, de modo que essas informações possam ser úteis aos desenvolvedores no futuro.

Segundo Hu et al. (2000), um sistema de DR tem por objetivo fazer os desenvolvedores pensarem sobre o projeto e discutirem dentro de uma determinada estrutura da representação de conhecimento.

O sistema de DR não é um sistema que isolado proporcione as funcionalidades esperadas; ele pode ser classificado como uma tecnologia que visa a enriquecer o projeto auxiliado por

computador (*Computer Aided Design* - CAD) e outras atividades da engenharia. Para melhor aproveitamento do sistema, é preciso integrá-lo a outros sistemas de suporte ao projeto, como por exemplo combiná-lo com os sistemas *Knowledge-Based Decision Support* (KBDS), *Computer-Aided Design and Manufacturing* (CAD/CAM) ou *Computer-Aided Software Engineering* (CASE) (Hu et al., 2000).

De modo geral, as funcionalidades esperadas de um sistema de DR são a captura, a representação e a recuperação de DR capturado. A captura refere-se à forma de como obter e converter em um formato lógico, os tipos de informação sobre as decisões de projeto. A representação faz uso de um esquema para representar o DR a ser armazenado. E a recuperação é responsável pelo fornecimento do DR adequado ao contexto do projeto (Hu et al., 2000).

A Figura 2.1 mostra, esquematicamente, a arquitetura geral de um sistema de DR. Como pode ser observado, a partir de uma “reunião” dos desenvolvedores é feita a captura do DR de forma automática ou interativa. Essa “reunião” pode ocorrer com a participação síncrona ou assíncrona dos desenvolvedores. O sistema, por meio de um esquema de representação, armazena esse DR capturado num repositório. A partir do DR armazenado, pode-se recuperá-lo por meio de intervenções do tipo entrada e saída. A entrada de uma intervenção pode ser feita por meio de questões, pelas quais os desenvolvedores podem vir a recuperar diversas informações sobre o DR armazenado. A saída obtida das intervenções é o conjunto de decisões sobre o DR requerido, tais como: respostas a consultas, aspectos lógicos de um problema relevante, dados sobre a monitoria do progresso do projeto ou ainda os documentos sobre o artefato projetado. O DR pode ser tanto sobre o projeto em desenvolvimento, quanto sobre projetos anteriores. A recuperação pode ser feita por meio de pesquisa, de navegação, ou de forma automática, ou mesmo de uma combinação dessas três formas.

A Figura 2.1 ainda mostra a integração do sistema de DR com outros sistemas que lhe fornecem suporte (KBDS, CAD/CAM, CASE). Tais sistemas de suporte auxiliam na recuperação de DR, pois envolvem uma base de conhecimento e processamento de raciocínio de projeto. Além disso, contribuem com a representação do conhecimento e com ferramentas para capturar raciocínio de projeto, e para a comunicação durante o processo de projeto (Hu et al., 2000).

As decisões e as razões que foram consideradas durante o processo de tomada de decisão devem ser armazenadas pelos próprios desenvolvedores ou capturadas por um módulo de monitoração do sistema. A captura de DR pode ser feita durante a comunicação dos desenvolvedores, por meio do uso de ferramentas de trabalho cooperativo apoiado por computador (*Computer-Supported Cooperative Work* – CSCW), tais como o correio eletrônico,

Figura 2.1: Arquitetura geral de um sistema de DR, simplificada de Hu et al. (2000)

as conversas por telefone, e o arquivamento das reuniões de projeto e do conteúdo dos computadores portáteis dos desenvolvedores (Hu et al., 2000).

O DR capturado é organizado, geralmente por meio de regras, as quais são determinadas pelo esquema de representação. Em seguida é armazenado no repositório do sistema, com a finalidade de que ele seja utilizado posteriormente. Sendo assim, se um projeto for similar a um projeto existente, o DR do projeto anterior pode ser recuperado do repositório, fornecendo sugestões e idéias relevantes para tarefas do projeto atual. Além disso, consultas

sobre o projeto atual podem ser feitas com a finalidade de esclarecer ou simplesmente observar as decisões anteriormente tomadas (Hu et al., 2000).

As seções, a seguir, descrevem com mais detalhes as abordagens utilizadas na construção dos sistemas de DR, apresentando os esquemas de representação de DR, as formas de captura e as formas de recuperação de DR.

2.5 Abordagens para construção de Sistemas de DR

Segundo Hu et al. (2000), duas principais abordagens predominam para a construção de sistemas de DR: a orientada por processo e a orientada por característica.

A abordagem orientada por processo é usada para dar uma representação histórica dos artefatos, quando o processo possui domínio de informações dinâmico. Assim, é geralmente adotada quando nada está muito bem definido e as alterações acontecem o tempo todo: os problemas não são bem compreendidos e conseqüentemente, os requisitos não estão totalmente definidos e não se sabe qual solução escolher. Dessa forma, essa abordagem não pressupõe um alto grau de padronização dos artefatos.

A abordagem orientada por característica é usada para fornecer uma representação lógica dos artefatos, seguindo regras lógicas e rigorosas de acompanhamento do processo do projeto, quando se tem um alto grau de padronização.

Ainda segundo Hu et al. (2000), utilizando as melhores características de ambas abordagens (abordagem híbrida), o sistema fornece uma estrutura lógica de DR e armazena o histórico do processo de projeto.

As subseções seguintes apresentam as abordagens orientadas por processo, orientada por característica e híbrida, de acordo com Hu et al. (2000).

2.5.1 Abordagem orientada por processo

Na abordagem orientada por processo, DR é tratado como um histórico do processo de projeto, ou seja, o DR é descritivo. Frequentemente, essa abordagem é utilizada em domínios dinâmicos, nos quais os problemas não são bem compreendidos, não se sabe precisamente qual tecnologia utilizar e há pouca ou nenhuma padronização dos artefatos projetados.

Esses aspectos são geralmente encontrados nos estágios iniciais de um projeto, pois há somente uma especificação funcional informal sem nenhuma restrição para o prosseguimento da evolução do projeto. Dessa forma, o DR vai sendo capturado e representado a cada passo da evolução do projeto.

Para a representação das informações contidas no sistema de DR, a abordagem orientada por processo usa um esquema com base em grafos, com nós e *links*. Os nós representam questões, opções e argumentos, e os *links* representam relações entre os nós.

Esse esquema de representação é conveniente para a captura das informações devido a sua estrutura flexível. Entretanto, para que essas informações se tornem acessíveis, um obstáculo é encontrado na conversão da informação capturada em uma estrutura de DR, ou seja, o desafio está na criação das ligações dos nós.

Depois que as informações ficam acessíveis, é possível recuperar o DR por meio de navegação ou pesquisa para responder diversas questões do tipo “quem?”, “por quê?”, “o que?” e “onde?”, auxiliando os desenvolvedores a esclarecerem suas dúvidas.

2.5.2 Abordagem orientada por característica

Na abordagem orientada por característica, as regras e o conhecimento sobre o domínio de informação específico são considerados em cada tomada de decisão de projeto. Uma decisão do projeto, não suportada pelas regras do conhecimento, necessita ser confirmada, acarretando em uma revisão das regras e do conhecimento.

Um sistema orientado por característica é desenvolvido geralmente em um contexto de tarefa particular, usando um estudo empírico. Em geral, estes tipos de sistema contêm bases de conhecimento do domínio que podem ser usadas para suportar o raciocínio automatizado.

Assim, a abordagem orientada por característica exige mais formalidade do que a abordagem orientada por processo na representação das regras e do conhecimento. Tal formalidade possibilita que a recuperação do DR, tanto para consulta quanto para reuso de DR, seja feita mais facilmente.

Além disso, quando os sistemas de DR com essa abordagem são incluídos nos sistemas de projeto, o armazenamento de DR é feito com uma representação formal, que pode ser processada automaticamente. Isso proporciona um suporte efetivo para manipulação de projeto ou re-projeto, tais como avaliação das decisões de projeto e resolução de conflitos.

A abordagem orientada por característica possui algumas restrições quanto ao acesso do DR. Por ser uma abordagem bastante automatizada, o DR disponível satisfaz os requisitos estipulados pelo sistema em determinado momento. Sendo assim, o desenvolvedor fica limitado ao DR apresentado pelo sistema, perdendo a oportunidade de fazer uma busca mais livre por meio de navegação ou de pesquisa por perguntas como “quem?”, “quando?”, “por quê?”, etc.

2.5.3 Abordagem híbrida

Como pode ser observado nas duas últimas seções, tanto a abordagem orientada por processo quanto a abordagem orientada por característica possuem qualidades desejáveis para a manipulação de DR. Entretanto, ambas as abordagens sofrem algum tipo de restrição.

A abordagem orientada por característica requer captura mais formal de DR, o que facilita o seu arquivamento na base de dados, além de proporcionar recuperação mais direta ao DR, adequado à situação de um projeto em um determinado instante de seu desenvolvimento. Mas, essa abordagem possui limitações quanto às partes que podem vir a ser manipuladas, dificultando a busca segundo a preferência do desenvolvedor.

A abordagem orientada por processo é mais informal, o que possibilita maior liberdade na forma de captura e recuperação do DR. Porém, essa abordagem enfrenta dificuldades no armazenamento das informações capturadas de uma forma que o computador “entenda” as mesmas e possa auxiliar em sua recuperação posteriormente.

Assim, como as restrições e qualidades da abordagem orientada por processo e da abordagem orientada por característica são diferentes, propõe-se a combinação das mesmas. Dessa forma, a finalidade da criação de sistemas com abordagem híbrida é superar as limitações individuais e aproveitar as qualidades de cada abordagem. A partir da união das duas abordagens, são gerados sistemas que fornecem uma estrutura lógica para o DR, além de gravarem o histórico do processo de projeto.

2.6 Esquemas de Representação

Os sistemas de DR armazenam informações úteis para a decisão de projeto (suposições, opções, alternativas, etc), assim como a própria decisão de projeto. Tais informações e as decisões armazenadas pelo sistema podem ser consultadas futuramente pelos desenvolvedores, auxiliando-os no progresso do projeto em andamento, ou mesmo no progresso de outros projetos.

A organização dessa grande quantidade e variedade de informação em uma estrutura acessível é uma questão crítica. Hu et al. (2000) afirmam que um bom esquema de representação é essencial para uma recuperação efetiva.

A escolha de qual esquema de representação utilizar pode ser feita observando-se sua perspectiva de DR. Segundo Shipman & McCall (1997), três são as diferentes perspectivas encontradas: argumentação, comunicação e documentação. Mesmo sendo diferentes, essas três perspectivas não são contraditórias, nem totalmente independentes.

Segundo a perspectiva de **argumentação**, DR é tido como a argumentação (os diversos pontos discutidos, os argumentos apresentados e como foram selecionados) que os projetistas usam na formulação e solução de problemas. Essa perspectiva tem por finalidade expressar o raciocínio dos desenvolvedores, particularmente para melhorar os argumentos que eles usam.

De acordo com a perspectiva de **comunicação**, DR é a captura e recuperação de toda e qualquer comunicação que ocorre entre os desenvolvedores, por meio das mais variadas mídias – audio, vídeo, *email*, anotações, desenhos, etc. Essa perspectiva tem por objetivo armazenar toda comunicação possível entre os desenvolvedores, capturando as “discussões” e a evolução do projeto conforme ocorrem.

Sob a perspectiva de **documentação**, DR é o registro da informação sobre decisões de projeto – que decisões são feitas, quando elas foram feitas, quem as fez e porque. A intenção dessa perspectiva é armazenar claramente as decisões que foram tomadas durante a “reunião”, para que a documentação gerada seja usada tanto para compreensão do projeto por pessoas externas à equipe de desenvolvimento quanto para obtenção e defesa de patentes.

Tanto sob a perspectiva de argumentação quanto sob a de documentação, a captura das informações é realizada de forma ordenada, estruturada, o que facilita o seu armazenamento e, conseqüentemente, sua recuperação. Já sob a perspectiva de comunicação, existem dificuldades na recuperação das informações, porque essas não são capturadas ordenadamente, e o armazenamento não é feito de acordo com as necessidades esperadas para uma recuperação adequada.

Das três perspectivas, a que armazena menos informações é a perspectiva de documentação e a que armazena mais é a perspectiva de comunicação. Devido a sua intenção, a documentação faz um resumo claro e direto da “reunião” dos desenvolvedores, registrando apenas as questões que foram discutidas, suas soluções, quando foram feitas, quem as fizeram, sem todos os detalhes. A argumentação, além das informações contidas na documentação, possui informações sobre os argumentos de cada questão levantada, pontos a favor e contra as argumentações. Essas informações armazenadas auxiliam na formação de melhores argumentos, além de possibilitar a detecção e correção de deficiências do projeto. A perspectiva de comunicação armazena todas essas informações e outras que são consideradas desnecessárias pelas perspectivas anteriores. Porém, as informações não são capturadas de forma organizada, pois a comunicação entre os membros da equipe de projeto é bastante livre, desordenada, repleta de retrocessos e toda informação é capturada em sua forma original.

As perspectivas de argumentação e comunicação capturam o DR no momento em que o mesmo está sendo discutido entre os desenvolvedores, garantindo a fidelidade do conteúdo armazenado ao DR real. A perspectiva de documentação faz a captura do DR após a reunião dos desenvolvedores, sendo feita de forma resumida. Isso pode proporcionar alguma

informação armazenada incorretamente, que não esteja de acordo com a realidade, podendo tal informação causar enganos quando essa for recuperada para utilização.

As três perspectivas utilizam diferentes formas de representação de DR. A perspectiva de comunicação pode utilizar todos os tipos de mídias disponíveis para representar o DR e a perspectiva de documentação usa um esquema linear de representação, os documentos. A representação da perspectiva de argumentação é feita por meio de nós (elementos da “discussão”) e *links* (relacionamento entre os elementos) (Bacelo & Becker, 1996), que possibilitam uma maior flexibilidade. Porém, essa última forma de representação acaba acarretando um certo *overhead* devido à necessidade de classificação dos nós e *links* em abstrações e relacionamentos, respectivamente, pré-estabelecidos do modelo de esquema de representação. Para Hu et al. (2000), *Issue-Based Information System* (IBIS), *Questions, Options and Criteria* (QOC), *Decision Rationale Language* (DRL) e *Procedural Hierarchy of Issues* (PHI) são os modelos de esquemas de representação mais conhecidos da perspectiva de argumentação. Esses modelos são muito parecidos, diferenciando-se nos tipos de nós e *links*.

Quanto mais estruturado estiver o DR armazenado no repositório de dados, melhor será sua recuperação. Portanto, sob a perspectiva de argumentação e sob a de documentação, tem-se uma recuperação facilitada. Já a recuperação de DR sob a perspectiva de comunicação é mais problemática devido à falta de estrutura do DR que foi armazenado de forma livre.

O Quadro 2.1 mostra resumidamente as diferenças já comentadas entre as perspectivas de argumentação, comunicação e documentação.

Quadro 2.1: Três Perspectivas de DR

Características das informações	Perspectivas		
	Argumentação	Comunicação	Documentação
Conteúdo	Semi-estruturado	Não estruturado	Estruturado
Nível de detalhamento da informação armazenada em relação à produzida durante as discussões	Médio - analítica e sintetizada	Alto - Analítica	Baixo - sintetizada
Momento de captura	Durante a “discussão”	Durante a “discussão”	Após a “discussão”
Esforço/atividade requerido(a) no momento da discussão	Classificar os nós e <i>links</i>	Não possui	Não possui
Esquema de representação	Nós e <i>links</i>	Diversas mídias	Linear (documentos)

De acordo com Shipman & McCall (1997), os sistemas mais utilizados são aqueles que adotam a perspectiva de documentação.

Os desenvolvedores são resistentes aos sistemas que utilizam a argumentação porque não se sentem motivados a utilizar os esquemas de representação para estruturar seus pensamentos enquanto estão executando suas tarefas habituais de projeto. Tal atividade extra causa sobrecarga considerável sobre os desenvolvedores. Segundo Shipman & McCall (1997), os sistemas de captura de argumentação só obtêm êxito quando existe uma pessoa responsável pelo armazenamento dos argumentos elaborados pelos outros membros da equipe de projeto. Embora sejam raros, existem sistemas sob essa perspectiva que não alocam uma pessoa responsável pelo armazenamento e ainda assim obtêm êxito.

O ponto forte da perspectiva de argumentação é a recuperação. Isso é devido ao seu armazenamento feito com a estruturação necessária para uma boa recuperação. O sucesso da recuperação também está ligado à relação entre nós e *links* que oferece um poderoso recurso para indexação associativa, possibilitando o uso de hipertexto para recuperação do DR relevante em determinado momento.

Enquanto a argumentação exhibe dificuldades para captura, a comunicação tem seu ponto forte justamente nessa atividade. A captura de DR comunicado exige poucas restrições para sua representação (praticamente sem nenhuma estrutura) que se encontra em diferentes formatos: *emails*, áudio, vídeo, anotações, desenhos, etc; além disso, não oferece incômodo aos desenvolvedores como acontece na argumentação. Para que o DR seja representado no repositório basta que esteja em um formato eletrônico. Por exemplo, o *email* já está no formato eletrônico, o áudio e o vídeo podem ser facilmente convertidos em formatos “legíveis” pelo computador porque são próprios para isso e textos escritos à mão ou desenhos podem ser convertidos para formato eletrônico por uso de *scanners*. Entretanto, somente ter o DR em formato eletrônico não é suficiente para que se tenha sucesso na recuperação. É necessário que o DR capturado seja armazenado de uma forma bem estruturada, para assim se reduzir falhas na recuperação do DR obtido sob a perspectiva de comunicação.

As perspectivas de documentação e argumentação são mais difundidas pois a captura é feita após a “reunião” dos desenvolvedores, evitando uma tarefa a mais no momento da “discussão”. Porém, essa captura tardia não preserva a linguagem ou a forma com que as decisões foram feitas, o que pode levar a dúvidas sobre a exatidão do DR armazenado.

Visto que nenhuma das três perspectivas é suficientemente isenta de problemas, Shipman & McCall (1997) propõem o uso da combinação das três perspectivas para que se obtenha melhores resultados. Entretanto, tal proposta não é tão simples de viabilizar, pois cada perspectiva tem uma estrutura própria para representar o DR.

É certo que quanto maior a exigência na estrutura da representação do DR, melhor é a sua recuperação, mas menor é a efetividade que se tem na sua captura e vice-versa. Dessa forma, para que se tenha um sistema bem aceito são requeridas uma captura e

uma recuperação efetivas. Dado que a perspectiva de documentação proporciona uma boa captura e recuperação, mas a quantidade de informação e/ou a precisão dessa informação não é totalmente garantida, as pesquisas têm sido focadas nas outras duas perspectivas, com o objetivo de ter um maior número de informações armazenadas, com mais garantias da precisão das mesmas e para que os desenvolvedores sejam o mínimo possível incomodados durante as suas atividades habituais de projeto.

Observa-se um grande interesse pela perspectiva que possua captura e recuperação efetivas, sem o envolvimento direto dos desenvolvedores na captura do DR, mas que este seja estruturado depois e não durante sua captura. Para isso, Shipman & McCall (1997) identificam algumas características que essa nova perspectiva deveria ter: usar a perspectiva de comunicação para capturar o DR; usar a perspectiva de argumentação para recuperar o DR; e, usar uma “ponte”, como uma formalização incremental, para conectar essas duas perspectivas, a fim de estruturar o DR depois que este for capturado pelo uso de computadores, com o mínimo de interferência possível dos desenvolvedores.

Com as características almeçadas por Shipman & McCall (1997), tal perspectiva poderia ser a base de um sistema que na captura integrasse a maior variedade de formas de mídias, com um repositório para toda e qualquer comunicação da equipe de projeto, além de usar várias formas de indexação com base nas tarefas e conteúdos, visando a uma recuperação mais adequada à necessidade do momento. Então, as funções principais do sistema seriam a captura por meio da perspectiva de comunicação, a estruturação das informações capturadas e a recuperação por meio da perspectiva de argumentação.

As informações nessa nova perspectiva combinada seriam as mesmas encontradas na perspectiva de documentação, além de muitas outras, com vantagens: o esforço do projetista pode ser substancialmente reduzido, seu tempo pode ser economizado e a precisão das informações pode ser maior.

Na literatura, foram encontrados diversos trabalhos sobre captura de DR que abordam a perspectiva de argumentação e as diferentes formas para armazenamento das informações. A seguir são detalhados os quatro esquemas² de representação mais difundidos, seguindo a perspectiva de argumentação: IBIS, PHI, QOC e DRL.

2.6.1 IBIS

Issue-Based Information System (IBIS) foi proposto como um modelo simples de argumentação, semi-estruturado, desenvolvido por Horst Rittel no início da década de 70 (Conklin & Begeman, 1988).

²Neste trabalho os termos esquema de representação de DR e modelo de representação de DR são utilizados indistintamente.

Esse modelo tem o princípio de que durante o processo do projeto de soluções para problemas complexos é fundamental a comunicação entre os desenvolvedores, pois estes trazem para a “discussão” suas experiências e pontos de vista que podem vir a solucionar as questões do projeto (Conklin & Begeman, 1988). Sendo assim, o modelo IBIS tem a finalidade de fornecer suporte à estruturação da “discussão”, de forma que as informações capturadas e estruturadas possam auxiliar os desenvolvedores a resolverem suas questões.

Neste modelo, com base em questões, a representação da argumentação é feita por meio de três abstrações: questões (problemas), posições (respostas) e argumentos (prós e contras) (Burge & Brown, 2000). Essas abstrações são relacionadas entre si para representar a “discussão” dos desenvolvedores, por meio de relacionamentos pré-determinados pelo modelo (Bacelo & Becker, 1996). Os relacionamentos (*links*) entre essas abstrações são mostrados na Figura 2.2.

O objetivo é capturar as questões que são levantadas durante todo o desenvolvimento do projeto, juntamente com as várias posições que são apresentadas em resposta às questões e os argumentos prós e contra as posições. A principal preocupação do modelo recai sobre as questões-chave do problema do projeto, as quais podem ter muitas posições. Cada uma das posições, por sua vez, pode ter um ou mais argumentos que a suportam (concordam com ela) ou que fazem objeção à mesma (são contra ela) (Hu et al., 2000).

Como pode ser observado na Figura 2.2, existem sete tipos de *links* no modelo IBIS:

- Uma POSIÇÃO “responde a” uma QUESTÃO;
- Um ARGUMENTO “suporta” uma POSIÇÃO;
- Um ARGUMENTO faz “objeção a” uma POSIÇÃO;
- Uma QUESTÃO “generaliza” ou “especializa” outra QUESTÃO;
- Uma QUESTÃO “substitui”, “questiona” ou “é sugerida por” outra QUESTÃO;
- Uma QUESTÃO “questiona” ou “é sugerida por” uma POSIÇÃO;
- Uma QUESTÃO “questiona” ou “é sugerida por” um ARGUMENTO.

Uma “discussão” é requisitada sempre que um problema, interesse ou pergunta (questão) trazer opiniões diversas entre os desenvolvedores, sendo necessária uma decisão para que o desenvolvimento do projeto possa prosseguir (Conklin & Begeman, 1988).

Geralmente, a “discussão” inicia quando um dos desenvolvedores coloca em um nó QUESTÃO o enunciado de um problema. A partir desse nó QUESTÃO a equipe de desenvolvimento

Figura 2.2: Modelo IBIS (Conklin & Begeman, 1988)

vai criando nós POSIÇÃO com propostas de solução e nós ARGUMENTO que apóiem esses nós POSIÇÃO. Esses nós POSIÇÃO podem concordar ou discordar entre si. Para cada nó POSIÇÃO, os desenvolvedores devem inserir também um nó ARGUMENTO que suporte seu nó POSIÇÃO. Além disso, os desenvolvedores podem colocar outros nós ARGUMENTO que dêem suporte, ou sejam contrários, a qualquer nó POSIÇÃO já inserido. Também, novas questões levantadas durante a “discussão” podem ser inseridas (novo nó QUESTÃO) e devem ser ligadas àqueles nós que mais diretamente as sugeriram (Wiegendaad, 1999).

O IBIS é considerado um modelo de “discussão” muito poderoso pelos usuários de ferramentas que utilizam esse modelo em sua implementação. Tais usuários trabalham tanto sozinhos como em equipe. Os usuários que trabalham sozinhos afirmam que a organização das idéias, em termos de QUESTÃO-POSIÇÃO-ARGUMENTO, os ajudam a prestar mais atenção na partes mais difíceis e críticas do problema, e também os ajudam a perceber a falta de consistência dos pensamentos mais rapidamente. Os desenvolvedores que trabalham em equipe dizem que a estrutura imposta na “discussão” é muito útil e serve para mostrar opiniões pessoais, sinalizações de concordância e retórica inteligente, além de tornar as suposições e definições mais claras (Conklin & Begeman, 1988).

No entanto, algumas restrições são também apontadas no modelo IBIS. Por ser um modelo simples, com apenas três tipos de nós (QUESTÃO, POSIÇÃO e ARGUMENTAÇÃO), diversas informações acabam sendo armazenadas num único nó (classificadas em um tipo) (Monk et al., 1995). Por exemplo, o nó ARGUMENTO, além dos argumentos em pró ou contra certo nó POSIÇÃO, armazenados explicitamente, acaba armazenando de forma implícita os requisitos, as restrições e os objetivos. O modelo IBIS não tem um nó específico para requisitos e restrições (critérios) que são impostos para a escolha da solução, nem um nó específico para expressar o objetivo que se deseja alcançar (Wiegendaad, 1999).

Além dessa restrição, o modelo IBIS, por não ser hierárquico, gera uma rede complexa quando armazena muitos nós. Muitas vezes não se sabe qual questão foi armazenada primeiro, a qual questão determinado argumento pertencia inicialmente, etc (Wiegeraad, 1999). Não há um suporte para identificação das questões que afetaram uma particular decisão (Shum, 1991). Para este problema, uma solução tem sido proposta: dividir a rede grande e complexa, em redes menores e mais simples (Wiegeraad, 1999).

Também não se sabe identificar questões triviais que poderiam ser evitadas (Shum, 1991), diminuindo assim o número de nós da rede.

Visando solucionar ou minimizar tais problemas, outros modelos foram definidos.

2.6.2 PHI

Procedural Hierarchy of Issues (PHI) é um modelo de argumentação desenvolvido por McCall (McCall (1986)³ apud Shum (1991)) que estende o modelo IBIS, ampliando o escopo do conceito “questão” e alterando a estrutura que relaciona os seus tipos de nós (Lu et al., 1999).

O modelo PHI tem uma estrutura quase hierárquica (Shum, 1991) e representa as informações capturadas utilizando uma notação um pouco diferente da notação do modelo IBIS: QUESTÃO, RESPOSTA e ARGUMENTOS, além de SUB-QUESTÃO (construída a partir de uma questão), SUB-RESPOSTA (construída a partir de uma resposta) e SUB-ARGUMENTO (construída a partir de um argumento) (Wiegeraad, 1999). Os tipos de nó SUB-QUESTÃO, SUB-RESPOSTA e SUB-ARGUMENTO indicam um nível a mais na especificação com relação aos tipos de nó QUESTÃO, RESPOSTA e ARGUMENTO, respectivamente. Sendo assim, existem diferentes níveis de especificação e detalhamento da informação.

O PHI simplifica a relação entre seus nós por meio do uso de somente uma relação chamada “serve” (Hu et al., 2000), como pode ser observado na Figura 2.3. A relação “serve” é definida da seguinte maneira (Shum, 1991): a QUESTÃO A “serve” a QUESTÃO B, se e somente se a resolução da QUESTÃO A influenciar a resolução da QUESTÃO B. A relação “serve” pode ser expressa de outra maneira, por exemplo, A “é uma sub questão de” B, ou como B “é uma antecedente de” A, se B apareceu primeiro que A. Da mesma forma, RESPOSTA e ARGUMENTO podem ser definidos.

Além da possibilidade de se saber a ordem em que as idéias surgem, PHI usa o método de decomposição para, por meio dos sub-nós, restringir mais o conteúdo de cada

³McCALL, R. Issue-serve systems: A descriptive theory of design. *Design Methods and Theories*, v. 20, n. 8, p. 443-458, 1986

nó, proporcionando um armazenamento menos implícito das informações. Também evita o levantamento de questões triviais e irrelevantes que atrapalham a compreensão da rede de nós montada (Hu et al., 2000). Isso porque PHI considera que se um problema não pode ser mostrado de acordo com a hierarquia, então este problema é considerado irrelevante (Shum, 1991).

Outra vantagem do modelo PHI é que as questões, sub-questões, respostas, sub-respostas, argumentos e sub-argumentos podem ser apresentados em formato linear (Hu et al., 2000), como pode ser observado na Figura 2.4. Obviamente, no caso em que um argumento, por exemplo, servir para mais de uma resposta no formato linear ele deverá ser repetido, assim como toda sub-rede que lhe estiver associado.

Figura 2.3: Modelo PHI (Wiegeraad, 1999)

O modelo também fornece relações de dependência entre a resolução das questões e considera os prós e contras das respostas. Além disso, modela a estrutura das tarefas do processo do projeto e as informações úteis para as tarefas (Hu et al., 2000).

Embora a estrutura quase hierárquica aumente a expressividade de PHI, algumas relações importantes ainda não podem ser facilmente expressadas (Lu et al., 1999), como os critérios, objetivos, etc.

2.6.3 QOC

A análise de espaço de projeto, *Design Space Analysis* - DSA (MacLean & McKerlie, 1995), usa o modelo semi-formal *Questions, Options and Criteria* (QOC) para representar o DR capturado (MacLean et al. (1991)⁴ apud Burge & Brown (2000)).

Enquanto os sistemas/protótipos que usam os modelos IBIS e PHI têm o propósito de capturar o histórico das deliberações, os sistemas que usam o modelo QOC enfatizam o

⁴MACLEAN, A.; YOUNG, R.; BELLOTI, V.; MORAN, T. Question, options, and criteria: Elements of design space analysis. *Human-Computer Interaction*, v. 6, n. 3-4, p. 210-250, 1991

Figura 2.4: Exemplo PHI (Hu et al., 2000)

desenvolvimento sistemático de um espaço de opções de projeto estruturado por questões (Hu et al., 2000).

Outra diferença do QOC em relação aos modelos IBIS e PHI é o armazenamento do espaço de critérios (Lu et al., 1999). Tal espaço de critérios possibilita a captura explícita das restrições do projeto, facilitando a resolução do problema em questão.

O QOC representa o espaço do projeto por meio de três tipos de nós (Hu et al., 2000), conforme mostra a Figura 2.5: QUESTÃO (identifica as questões chave para estruturação do espaço de opções), OPÇÃO (possíveis soluções) e CRITÉRIO (base da avaliação e escolha entre as opções). Os possíveis *links* para estes tipos de nós são:

- Uma OPÇÃO “responde a” uma QUESTÃO;
- Uma QUESTÃO “é consequência de” uma OPÇÃO;
- Um CRITÉRIO faz uma “avaliação positiva” da OPÇÃO;
- Um CRITÉRIO faz uma “avaliação negativa” da OPÇÃO.

Assim que um nó QUESTÃO é inserido, vários nós OPÇÃO são colocados procurando solucionar a questão. Para avaliar se as opções colocadas são válidas, são usados os critérios.

Figura 2.5: Modelo QOC (Wiegaraad, 1999)

Os critérios avaliam as opções para verificar qual opção é adequada à questão. Sendo assim, tal avaliação pode ser positiva ou negativa para a opção. Uma mesma opção pode ter um determinado critério a seu favor e outro contra. Outras questões podem ser colocadas se uma opção trazer alguma dúvida. Dessa forma, o nó QUESTÃO é ligado ao nó OPÇÃO que gerou a dúvida.

O QOC possui vantagens com relação aos aspectos de inovação e reuso. É relativamente fácil para o desenvolvedor criar um QOC para a “engenharia reversa” de parte do sistema e preservá-la para uso futuro (Hu et al., 2000).

2.6.4 DRL

Decision Rationale Language (DRL) é uma extensão do modelo IBIS, que envolve um número maior de abstrações e relacionamentos (Bacelo & Becker, 1996). Pode ser visto como um “avanco” a mais para a extensão que PHI propõe sobre o modelo IBIS pela generalização da estrutura hierárquica em relações mais complexas e mostra alguns outros elementos, especialmente aqueles do espaço dos critérios. O DRL é mais expressivo para a representação da decisão do projeto, pois seus objetos capturam vários relacionamentos entre os elementos de decisão (Lu et al., 1999).

Como observado na Figura 2.6, o modelo DRL é composto pelos seguintes nós (Lee, 1990): PROBLEMA DE DECISÃO (representa o problema da escolha da alternativa que melhor satisfaça o objetivo), OBJETIVO (propriedades que uma opção ideal deve ter – critérios), ALTERNATIVA (possíveis soluções), ALEGAÇÃO (argumentos a favor ou contra um objetivo ou uma outra alegação), QUESTÃO (dúvidas/perguntas), PROCEDIMENTO e GRUPO.

Comparando-se os tipos de nós de DRL aos tipos adotados nos modelos IBIS, PHI e QOC, pode-se observar que no modelo DRL o espaço da questão, o espaço de opção, o espaço

do argumento e o espaço do critério são, respectivamente, representados pelo problema de decisão, alternativa, alegação e objetivo.

As possíveis ligações entre os nós do modelo DRL são (Lee, 1990):

- Um PROBLEMA DE DECISÃO “é uma sub-decisão de” um outro PROBLEMA DE DECISÃO;
- Um OBJETIVO “é um sub-objetivo de” um PROBLEMA DE DECISÃO ou um outro OBJETIVO;
- Uma ALTERNATIVA “atinge” um OBJETIVO;
- Uma ALTERNATIVA “é uma alternativa para” um PROBLEMA DE DECISÃO;
- Uma ALEGAÇÃO “suporta” uma ALTERNATIVA ou uma outra ALEGAÇÃO;
- Uma ALEGAÇÃO “é contra” uma ALTERNATIVA ou uma outra ALEGAÇÃO;
- Uma ALEGAÇÃO “pressupõe” outra ALEGAÇÃO;
- Uma ALEGAÇÃO “responde” uma QUESTÃO;
- Uma ALEGAÇÃO “é um resultado de” um PROCEDIMENTO;
- Uma ALEGAÇÃO “é um membro de” um GRUPO;
- Uma QUESTÃO “questiona” ou “influencia” uma ALEGAÇÃO;
- Um PROCEDIMENTO “é uma resposta para” uma QUESTÃO;
- Um PROCEDIMENTO “é um subprocedimento de” um PROCEDIMENTO;
- Um GRUPO “é um conjunto de possíveis respostas para” uma QUESTÃO.

A argumentação do DRL inicia com o problema de decisão. A partir deste problema, um número de sub-objetivos podem ser definidos para resolvê-lo e alternativas para tais objetivos são determinadas (Monk et al., 1995). Para essas alternativas surgem alegações a favor ou contra e para as alegações, surgem as questões ou outras alegações. Um grupo de alegações pode vir a responder às questões, assim como alegações também o podem.

Como o DRL possui maior número de nós e de *links* entre esses nós, em modelo possibilita uma “discussão” de fácil compreensão entre desenvolvedores separados geograficamente ou no tempo (*collaborative computer network*) porque a informação é capturada explicitamente em cada nó (Wiegendaad, 1999).

Figura 2.6: Modelo DRL (Wiegaraad, 1999)

Outro benefício obtido do maior número de tipos de nós e *links* está relacionado com a captura explícita dos critérios e objetivos (Wiegaraad, 1999), que facilita muito as decisões, pois os desenvolvedores conseguem ver de forma mais clara aonde devem chegar para solucionar determinado problema. Entretanto, esse grande número de tipos de nós e *links* exige, durante o armazenamento, cuidados para a correta delimitação da informação correspondente a cada tipo de nó disponível.

2.6.5 Comparação dos esquemas de representação da perspectiva de argumentação

O IBIS é o modelo mais antigo para representação de DR, além de ser o mais simples e servir de base para os demais. Esse modelo possui três tipos de nós que são nomeados por aquilo que representam: QUESTÃO, POSIÇÃO e ARGUMENTO.

O modelo PHI representa a informação por meio de QUESTÃO, RESPOSTA e ARGUMENTO, além de SUB-QUESTÃO, SUB-RESPOSTA e SUB-ARGUMENTO. Como consegue separar melhor a informação capturada, o PHI é proposto com vantagens sobre IBIS.

Os tipos de nós do QOC são: QUESTÃO, OPÇÃO e CRITÉRIO, sendo que o nó QUESTÃO é equivalente ao nó QUESTÃO dos modelos IBIS e PHI e o nó OPÇÃO é equivalente ao nó POSIÇÃO e RESPOSTA dos modelos IBIS e PHI, respectivamente. O nó CRITÉRIO é a vantagem do modelo QOC em relação aos modelos IBIS e PHI, porque possibilita a captura explícita dos critérios, os quais são importantes para uma melhor e mais precisa tomada de decisão. Porém, a informação correspondente ao nó ARGUMENTO dos modelos IBIS e PHI acaba sendo armazenada no nó OPÇÃO, no qual os argumentos ficam implícitos.

O modelo DRL possui um maior número de tipos de nós em relação aos modelos IBIS, PHI e QOC, o que possibilita uma separação mais adequada da informação capturada, ou seja, uma representação mais explícita da informação. Assim como o QOC, o DRL também armazena explicitamente os critérios no nó OBJETIVO. Uma desvantagem do modelo DRL é a exigência de uma delimitação da informação de acordo com os tipos de nó.

Entre os quatro modelos, PHI é o que possibilita um único tipo de *link* “serve” e DRL é o que tem o maior número de tipos de *links*, proporcionando mais relacionamentos entre os diversos tipos de nós, fornecendo uma maior flexibilidade.

Os relacionamentos entre os quatro modelos citados estão resumidos no Quadro 2.2, que descreve suas vantagens e suas limitações.

2.7 Captura de DR

A captura do DR deve ser feita durante todo o processo do projeto, por meio da obtenção de argumentos, opções, decisões, etc. e a posterior construção de uma estrutura também deve ser feita para que o DR possa ser armazenado no repositório, a fim de que seja possível a sua recuperação (Hu et al., 2000; Lee, 1997).

Dessa forma, de um modo geral, a captura de DR consiste em dois passos: o registro do conhecimento e a construção de razões de projeto (Lee, 1997). O primeiro passo está relacionado com a captura de todo e qualquer tipo de informação que for possível durante o processo do projeto. O segundo passo é a extração, a organização e o armazenamento do conhecimento de razões, com base em um esquema de representação de DR e nas limitações do domínio do projeto (Hu et al., 2000).

Quadro 2.2: Modelos de DR (Lu et al., 1999)

	IBIS	PHI	QOC	DRL
Conceitos correspondentes	Questão	Questão	Questão	Problemas de Decisão
	Posição	Resposta	Opção	Alternativa
	Argumento	Argumento	-	Alegação
	-	-	Critério	Objetivo
Vantagens	Modelo simples	Definição ampla das questões, respostas e argumentos; captura das relações de dependência entre as questões, respostas e argumentos	Expressa explicitamente os relacionamentos do critério	Representação expressiva; captura de relações complexas
Limitações	Muita informação capturada implicitamente	Um único tipo de relação: "serve"	Argumentos capturados implicitamente	Necessidade de uma delimitação da informação de acordo com os tipos de nós

A determinação de quais informações capturar durante o projeto e de como capturá-las depende de quais informações deseja-se obter na recuperação. Por exemplo, algumas capturas são feitas com a finalidade de responder às futuras questões dos desenvolvedores (Hu et al., 2000). Os problemas mais identificados durante a captura são (Hu et al., 2000): (i) a pouca informação ou informação incorreta impossibilita a criação de uma representação de DR; (ii) se as atividades dos desenvolvedores forem desviadas por causa da captura, eles certamente vão desistir de fornecer informação.

Sob a perspectiva dos usuários, de acordo com Hu et al. (2000), existem duas possíveis formas de captura de DR: automática e com base na intervenção do usuário.

Na captura automática, geralmente é utilizada a perspectiva de comunicação para a obtenção de DR entre os membros da equipe de desenvolvimento. Normalmente essa captura tem o auxílio de ferramentas CSCW, tais como gravadores de fitas cassetes, telefone, filmadoras, correio eletrônico, entre outras opções, para a obtenção de informação verbal. A comunicação escrita, tais como anotações, rascunhos, e esquemas desenhados, também pode ser capturada. A vantagem dessa forma de captura é o não envolvimento direto dos desenvolvedores na tarefa de captura da informação. A desvantagem refere-se ao fato da informação armazenada, por meio de uma representação informal, não ser estruturada,

o que acaba gerando muitas vezes insucesso na recuperação, quando da consulta dessas informações.

Na captura baseada na intervenção do usuário, a perspectiva de documentação é a mais utilizada (Hu et al., 2000). Essa captura registra o histórico das atividades de projeto tais como as decisões feitas pelos desenvolvedores, quando as fizeram, quem as fez e porquê. A vantagem dessa forma de captura é que a informação armazenada, semi-formal ou formalmente, possui uma estrutura, possibilitando uma melhor recuperação. A desvantagem refere-se ao fato de que os usuários são desviados de suas atividades usuais, causando um atraso nessas atividades, além de exigir um esforço maior.

Lee (1997) classifica as possíveis formas de captura de acordo com a participação dos usuários e do sistema. Em ordem crescente de participação do sistema na captura, tem-se: reconstrução, armazenamento e repetição, metodologia de produto, aprendizagem e geração automática.

Na captura por reconstrução, os desenvolvedores produzem o DR sem o uso do sistema, utilizando somente seus conhecimentos (Lee, 1997). A captura da informação pode ser feita por meio de entrevistas com os envolvidos no projeto ou por meio de filmadoras, gravadores de fitas cassetes, etc, além da possibilidade da engenharia reversa. A estruturação da informação acaba sendo feita depois da “reunião” o que traz a vantagem de uma reflexão mais cuidadosa na representação e disposição do DR.

Na captura por armazenamento e repetição, a informação é capturada como ela “surge” durante a “reunião” (Lee, 1997). A informação pode ser capturada sincronamente, por exemplo por vídeo conferência, ou assincronamente, por exemplo por *email*. Os sistemas com representação informal ou semi-formal geralmente usam esse tipo de captura evitando uma excessiva sobrecarga e interrupção das atividades habituais dos desenvolvedores.

Na captura por metodologia de produto, a informação surge naturalmente do processo do projeto de acordo com algum método (Lee, 1997). Os passos do método são essencialmente tipos diferentes de refinamentos e reformulação que o sistema captura e usa para geração das explicações. O custo não é uma desvantagem, pois o mesmo é pequeno. O desafio dessa forma de captura é fornecer um método que realmente ajude o usuário sem impor uma excessiva sobrecarga ou restrições.

Na captura por aprendizagem, o sistema gera as razões essencialmente pela “observação” da “reunião” e pelas questões feitas aos desenvolvedores se o sistema não “entender” ou “discordar” das ações dos desenvolvedores (Lee, 1997). Tanto o sistema quanto o usuário são beneficiados com essa interação: o usuário do sistema é beneficiado se o sistema estiver certo; o sistema “aprende” algo se ele estiver errado. Além disso, as razões são capturadas no contexto de uso, quando o conhecimento relevante está “fresco” na cabeça dos desenvolvedores

e novos conhecimentos podem ser relacionados com os antigos. Embora essa forma de captura seja viável, deve ser criada uma base de conhecimento inicial rica o suficiente para entender muitas das ações dos usuários e fazer questões adequadas quando algo ocasionar dúvidas.

Na captura por geração automática, o sistema gera o DR automaticamente (Lee, 1997). O custo não é excessivamente alto, além de manter a consistência e a atualização da razão. Porém, há um alto custo inicial devido à compilação do conhecimento necessário para a construção da razão.

Independente da forma de captura de DR, em geral, tem-se por objetivo provocar o mínimo de interferência possível na progressão normal das atividades do projeto (Hu et al., 2000). Dessa forma, os principais problemas investigados são: como capturar o DR sem tirar os desenvolvedores de suas atividades comuns para mais uma tarefa de documentação; como resolver os conflitos que possam vir a aparecer devido a um novo conhecimento capturado que viole um outro conhecimento previamente capturado.

2.8 Recuperação de DR

A recuperação de DR é de incontestável utilidade, pois permite que, por meio de experiências passadas armazenadas, novos problemas venham a ser solucionados e até mesmo evitados. Porém, como dito anteriormente, a consulta ou o reuso de DR são efetivos se o DR for armazenado de uma forma que possibilite a sua correta recuperação (Hu et al., 2000).

A recuperação é determinada pelo esquema de representação e pelos requisitos do domínio da engenharia a que pertence o conhecimento utilizado (Hu et al., 2000). O interesse na recuperação de DR varia de acordo com a intenção de uso da mesma. Por exemplo: na fase inicial de um projeto, pode-se desejar uma consulta a casos de projetos similares; ou durante o processo de projeto, pode-se desejar a recuperação de critérios, regras e opções feitas em projetos anteriores; ou após o processo de projeto, pode-se querer uma consulta para auxílio na produção dos documentos.

Com relação à recuperação ser iniciada pelo sistema ou pelo usuário, de acordo com Lee (1997), os sistemas de DR podem ser classificados como: sistemas com iniciativa do usuário ou sistemas com iniciativa própria.

No sistema com iniciativa do usuário, o usuário decide quais partes do DR examinar ou quando ou como as visualizar (Lee, 1997). Esses sistemas devem auxiliar os usuários a tomarem conhecimento da informação que existe e tornar fácil o acesso às partes desejadas.

No sistema com iniciativa própria, é o sistema quem decide quando e como apresentar que partes do DR (Lee, 1997). Tais sistemas devem possuir conhecimento suficiente para fazer decisões inteligentes.

Segundo Hu et al. (2000), diferentes estratégias podem ser utilizadas na recuperação de DR, dependendo do interesse que se tem em sua recuperação. Entre as estratégias de recuperação têm-se: navegação de DR arquivado, recuperação baseada em pesquisa, acionamento automático e estratégias de recuperação híbridas.

Por meio da navegação de DR, os projetistas podem verificar o DR, passando de um nó a outro pelos *links*. Para a abordagem orientada por processo, a navegação proporciona o histórico do projeto. Já na abordagem orientada por característica, a navegação é feita considerando-se a estrutura da característica. Para um artefato complexo, uma grande quantidade de informação é armazenada durante o processo do seu projeto, o que dificulta a busca por respostas específicas por meio da recuperação pela estratégia de navegação.

A recuperação com base em pesquisa se apresenta mais eficiente que a recuperação com base em navegação, pois proporciona uma recuperação de acordo com o interesse dos projetistas. As pesquisas devem ser perguntas “e - se?”, que podem ser respondidas explorando-se diferentes opções, ou perguntas “por quê?”, que são respondidas por meio do retorno pelos *links* com a finalidade de se encontrar a argumentação ou razão por trás da decisão.

No acionamento automático, é feita uma monitoração do desenvolvimento do projeto, de acordo com as correspondentes regras e critérios do projeto. Um monitor é utilizado para observar e verificar o processo de projeto, e comparar as decisões feitas com as regras e critérios da biblioteca de DR ou da base de conhecimento. Se diferenças forem encontradas, o DR será mostrado automaticamente.

As estratégias de recuperação híbridas fazem uso de uma combinação das estratégias de navegação, recuperação e acionamento automático, proporcionando uma recuperação conveniente e eficiente.

Para essas estratégias, alguns problemas são encontrados. Quando o projeto é orientado por processo, organizado cronologicamente, é preciso um gerenciamento para a grande quantidade de DR e um aumento da usabilidade humana e da flexibilidade computacional. Facilidades efetivas são necessárias para fornecer aos usuários a informação requerida sem a necessidade de navegação por todo o espaço das razões, tais como a ocultação seletiva dos detalhes desnecessários e a reorganização da informação. Para recuperar as deliberações e razões no processo do projeto, às vezes, é desejável fornecer um conjunto de questões que se relacionam às várias necessidades já conhecidas do usuário. Essas perguntas podem ser configuradas usando as razões de projeto arquivadas por projetos anteriores ou pela extração de analogias de situações prévias de projeto por meio do raciocínio com base em casos.

2.9 Considerações Finais

A documentação de *software* bem elaborada proporciona um bom conhecimento do mesmo. Isso traz diversas vantagens sendo as principais: melhor utilização do *software*, manutenção menos complicada, melhor possibilidade de reuso de partes do *software* e uso de documentos de fases anteriores para um melhor desenvolvimento do *software* nas fases seguintes.

Geralmente, os documentos gerados durante o processo de *software* só possuem informações referentes às decisões finais de cada fase, ignorando as alternativas, argumentos, critérios entre outras informações que levaram a tais decisões, ou seja, não se registra o desenvolvimento do raciocínio que determinou tais decisões, o qual denomina-se *Design Rationale* (DR).

Se o DR fosse armazenado, o *software* seria mais fácil de entender-se, o que favoreceria ainda mais a sua manipulação (manutenção, reuso, entendimento do *software*, etc).

Neste capítulo foram apresentados os fundamentos sobre o tema DR: as definições, os benefícios, as dificuldades, os sistemas, as abordagens para construção de sistemas, os esquemas de representação, a captura e a recuperação de DR.

No próximo capítulo são apresentados alguns sistemas de suporte direto ou indireto a DR, estudados para observação dos aspectos relevantes para orientação do desenvolvimento da ferramenta **DocRationale**.

Suporte a DR

3.1 Considerações Iniciais

Uma grande variedade de sistemas e projetos de suporte a DR tem sido desenvolvida para atender a diversas áreas de domínio de conhecimento, orientadas ou não a um domínio específico (Hu et al., 2000), devido ao interesse na obtenção e uso de DR, em busca de auxílio no desenvolvimento ou reuso de projetos.

Os sistemas de DR oferecem suporte para a captura, armazenamento e recuperação de DR, visando a auxiliar o entendimento dos projetos e registrar as experiências obtidas em toda sua elaboração, servindo para futuros projetos.

Foram estudados alguns sistemas encontrados na literatura para que fossem conhecidas suas estratégias de funcionamento, em busca da construção da ferramenta apresentada nesta dissertação, **DocRationale**.

Os sistemas, projetos e serviços de suporte a documentação de *software* e trabalhos colaborativos foram estudados devido ao domínio da ferramenta **DocRationale**. Tais sistemas, projetos e serviços também fornecem suporte a DR, embora de forma indireta, pois as informações que armazenam em suas bases também são DR e se estruturadas de alguma forma podem ser tão úteis quanto as informações de DR capturadas diretamente.

Uma tecnologia bastante utilizada nos sistemas, projetos e serviços estudados é a hipermídia devido às características de leitura não-linear, diferentes formas de navegação, exploração da informação entre outras (Cybulski & Reed, 1992).

Nas seções a seguir, são descritos em linhas gerais, os sistemas mencionados. Foi dado mais destaque à ferramenta CoTeia e ao serviço GroupNote, pois os mesmos foram utilizados para o desenvolvimento deste trabalho.

3.2 Suporte direto ou indireto a DR

Diversos sistemas têm sido desenvolvidos buscando a captura e recuperação de DR, sob os mais variados domínios de informação (Hu et al., 2000). Existem, ainda, sistemas que fornecem suporte à documentação de *software* ou aos trabalhos colaborativos que, de certa forma, fornecem suporte a DR indiretamente. Dessa forma, alguns sistemas encontrados na literatura, tanto de suporte direto como de suporte indireto a DR, foram estudados e são descritos a seguir. Os sistemas de suporte direto a DR descritos a seguir são o gIBIS, SIBYL, JANUS, HOS, PHIDIAS, RCF, IDIMS e Metaglué. O domínio dos sistemas é diversificado possibilitando uma visão geral do funcionamento de sistemas que visam à captura de DR. Os sistemas e/ou serviços de suporte indireto a DR estudados são HyperCase, HyperDev, SASHDoc, CoTeia e GroupNote. Esses sistemas e/ou serviços auxiliam na documentação de *software* ou na colaboração entre usuários.

3.2.1 Suporte direto a DR

O sistema gIBIS (*graphical Issue-Based Information System*) (Conklin & Begeman, 1988) não foi originalmente projetado para um domínio específico. Fornece apoio à decisão em grupo por meio do uso da tecnologia de hipertexto e do modelo IBIS para representar a atividade de argumentação dos desenvolvedores. O gIBIS almeja facilitar, capturar, estruturar e gravar a ‘discussão’ entre esses desenvolvedores, construindo uma rede de forma a representar tal ‘discussão’ em três tipos básicos de nós: **questão**, **posição** e **argumento**. O nó **questão** deve conter a questão que é um problema identificado para o projeto. O nó **posição** é um nó filho de um nó *questão* e deve conter uma posição que propõe uma solução particular à questão. O nó **argumento** é um nó filho de um nó *posição* que deve conter um argumento que dá suporte ou faz objeção à posição.

A interface básica do gIBIS está dividida em quatro janelas (Figura 3.1): *navegador gráfico*, *janela de índice de nós*, *painel de controle* e *janela de fiscalização*.

Figura 3.1: Interface do gIBIS (Conklin & Begeman, 1988)

O *navegador gráfico* apresenta a estrutura gráfica do modelo IBIS, em duas visões: visão global - toda a estrutura construída; e visão local - parte dela com mais detalhes de nós e links. A *janela de índice de nós* fornece uma visão ordenada hierarquicamente dos nós da atual rede IBIS. O *painel de controle* mostra os botões que estão disponíveis em cada momento, pois é sensível ao contexto. Já a *janela de fiscalização* mostra o conteúdo de um nó quando selecionado no *navegador gráfico*.

SIBYL (Lee, 1990) é um sistema de apoio à decisão em grupo, também não dedicado a um domínio específico, que conta com representação e gerenciamento de aspectos qualitativos do processo de decisão, tais como os objetivos, as alternativas e a avaliação dessas alternativas de acordo com os objetivos. O SIBYL é uma extensão do sistema de hipertexto simples gIBIS, somente com tipos sintáticos (como nó de texto e nó de imagem), com uma base de conhecimento desenvolvida, onde todo o conhecimento é representado formalmente para que o sistema faça o processamento automático (Lee, 1990). Para a criação e gerência da rede de argumentação, o sistema SIBYL adota o modelo DRI.

A interface com o usuário fornece uma barra, na parte superior da janela, com os tipos de nós disponíveis a cada momento. Quando selecionado um dos tipos de nós, é disponibilizado, na parte restante da janela, um quadro com campos a serem preenchidos referentes ao tipo de nó escolhido. Além disso, os tipos de nós disponíveis na barra são alterados de acordo

com a nova situação. Todos os nós criados vão sendo associados direta ou indiretamente ao primeiro nó criado (**problema de decisão**), formando a rede de argumentação. Para a visualização da rede de argumentação, tem-se a opção do navegador que mostra graficamente os nós que fazem parte da mesma.

Os nós **alternativa** e **objetivo** também podem ser visualizados por meio da matriz de decisão (Figura 3.2) que indica se uma solução foi conseguida ou não. Essa matriz de decisão é composta das células da primeira linha, correspondente aos objetivos, das células da coluna mais a esquerda, correspondente às alternativas, e das células restantes (células de avaliação), que correspondem a atual avaliação da alternativa (mesma linha) de acordo com o objetivo (mesma coluna). A matriz de decisão vai sendo atualizada à medida que os desenvolvedores vão discutindo as questões, gerando alegações a favor ou contra as alternativas ou outras alegações, questões que podem vir a influenciar ou questionar as alegações, colocando objetivos adicionais ou retirando os desnecessários e assim por diante. Os possíveis valores que as células de avaliação assumem variam durante toda a ‘discussão’ desde ‘não resolvido’, passando por ‘alto’, ‘médio’ e ‘baixo’, sendo que esses três últimos representam o grau de concordância da alternativa com o objetivo associado à célula de avaliação.

Objetivos: Alternativas:	Objetivo 1	Objetivo 2	Objetivo 3
Alternativa 1	Alegação		Alegação
Alternativa 2			Alegação
Alternativa 3			

Figura 3.2: Matriz de decisão (Wiegeraad, 1999)

JANUS é um sistema para o ambiente de projeto de cozinhas planejadas (Robbins & Redmiles, 1998), elaborado a partir da integração dos sistemas CRACK e VIEWPOINTS (Fischer et al., 1989). Tais sistemas incluem uma ferramenta crítica que possui uma crítica de projeto. Essa crítica de projeto é um mecanismo de interface inteligente para o usuário que analisa um projeto no contexto da decisão em questão e fornece um *feedback* para ajudar o desenvolvedor a melhorar o projeto. O *feedback* da crítica deve informar os erros do projeto, apontar o que não estiver completo, sugerir alternativas ou oferecer recursos heurísticos (Robbins, 1998). JANUS inclui, além da ferramenta de crítica, um ambiente hipertexto de documentação estruturado por argumentação para facilitar a captura do conhecimento do projeto (Hu et al., 2000).

HOS (*Hyper-Object Substrate*) é um sistema de hipermídia com características comuns de sistemas com base em conhecimento, não orientada a um domínio específico. Fornece um ambiente de suporte a projetos cujos desenvolvedores não se encontram no mesmo espaço físico, fazendo uso das redes de computadores (Hu et al., 2000). Utiliza agentes para suportar a comunicação entre os desenvolvedores e para relembrar os desenvolvedores do progresso das tarefas de projeto (Shipman & McCall, 1997).

A captura do DR combina a comunicação natural e uma estrutura de argumentação da seguinte forma: toda a comunicação é capturada, mas somente parte dela é selecionada e convertida para uma representação formal com a finalidade de ser utilizada (Burge, 1998). Para conseguir manter a captura da comunicação natural juntamente com uma estrutura que possa ser bem utilizada na recuperação da informação armazenada, HOS faz uso de um processo denominado Formalização Incremental (Shipman & McCall, 1997). Este processo permite que a formalização seja guiada pela demanda, além de ser ativamente suportado pelo sistema, não necessitando de esforços dos desenvolvedores para colocar a informação na representação mais estruturada.

PHIDIAS é um sistema de hipermídia usado para projetos gráficos 2D e 3D que adota o modelo PHI. Sua estrutura de nós-*links* com base em grafos facilita as relações entre o conhecimento semi-formal contido no DR com o conhecimento mais formal do sistema (Hu et al., 2000). Combina várias mídias de comunicação com as informações de argumentação e estruturas CAD. A estrutura de argumentação pode incluir informações de áudio, vídeo, gráficos, texto, as quais devem ser conectadas diretamente ao artefato de projeto e a seus componentes. PHIDIAS usa uma linguagem de pesquisa (PHIQL) e críticas de projeto para suportar a recuperação e uso das informações armazenadas (Shipman & McCall, 1997).

RCF (*Rationale Construction Framework*) (Myers et al., 1999) é um sistema experimental para obtenção de DR das operações feitas com a ferramenta CAD comercial MicroStation95, sem a interferência nas atividades habituais dos desenvolvedores do projeto. O foco do RCF está na fase de projeto detalhado, já que ferramentas CAD são utilizadas para a geração de diagramas esquemáticos de artefatos. Sendo assim, o DR capturado pelo RCF é justamente aquele encontrado durante a execução das atividades da fase de projeto detalhado. Tal DR capturado é o histórico dessas atividades. Para conseguir esse histórico, o sistema faz o monitoramento das interações dos desenvolvedores com a ferramenta CAD.

O sistema RCF organiza o DR capturado por meio: de informações que relatam o quê o desenvolvedor fez e quando fez; e do raciocínio do desenvolvedor (razões relacionadas às suas intenções) ao realizar determinada ação. Para a captura desse raciocínio é feito uso de um conjunto de metáforas de projeto, o qual descreve operações do desenvolvedor constituindo episódios significativos das ações que o mesmo realizou. A metáfora de projetos serve de base

para se deduzir as intenções do desenvolvedor, fazendo a ligação entre as operações realizadas e suas explicações. Segundo Myers et al. (1999), não é conveniente a geração automática de explicações para todos os aspectos de um projeto, já que os desenvolvedores fazem muitas decisões críticas que não são explícitas nos projetos e em seus processos. Myers et al. (1999) acreditam que os métodos do RCF deveriam ser complementados por métodos interativos da aquisição de DR para que os desenvolvedores pudessem estender e corrigir as informações capturadas automaticamente.

IDIMS (*Integrated Design Information Management System*) (Kato et al., 2002) é um protótipo de sistema para Web, testado no domínio de sistemas de satélites, tendo por finalidade a captura de DR da comunicação da equipe de desenvolvido ocorrida por *email*.

Os documentos gerados durante o desenvolvimento de um sistema são adicionados pelos usuários no repositório de documentos do IDIMS, em formato PDF ou XML. Tem-se interesse no formato XML, pois nesse formato é acrescentado DR em suas *tags*. Dessa forma, se o usuário não fornecer o documento em formato XML, o PDF é transformado em XML e então os dois formatos são armazenados no repositório. Sendo assim, quando é solicitada a visualização de um documento armazenado em PDF e XML, o usuário pode escolher em qual formato deseja observar o documento. Caso deseje ver o documento em formato XML ou só se tenha o documento em XML, é gerado uma página HTML correspondente.

Para representar o DR relacionado aos documentos, IDIMS adota uma versão simplificada com base em modelos como IBIS e QOC. Assumiu-se sua representação por meio de duas construções básicas: questões e decisões. As questões são os problemas encontrados durante o desenvolvimento de sistemas e as decisões são a resolução de tais questões. Tanto as questões quanto as decisões são armazenadas em formato XML no repositório de questões/decisões. As ligações possíveis entre as construções questão e decisão são: decisão ligada a questão; questão ligada a uma decisão; questão ligada a questão; e, decisão ligada a decisão. O IDIMS possui um visualizador de questões e decisões por meio do qual os usuários podem registrar, pesquisar e ver as questões e decisões do respectivo repositório.

IDIMS ainda possui um filtro de *emails* que é responsável pelo processamento das mensagens trocadas pelos membros da equipe. O filtro de *emails* é configurado para receber mensagens enviadas para uma determinada lista. Quando um *email* é enviado para a lista, o filtro transforma a mensagem em um documento XML e armazena-o no repositório de documentos. Em seguida, anotações (questões/decisões) são extraídas do documento XML e armazenadas no repositório de questões/decisões. Então é enviado uma mensagem aos membros da lista com um modelo de resposta que contém uma citação a mensagem original e uma citação de suas anotações.

Metaglue (Hammond et al., 2002) é um sistema com base em agentes para captura e indexação de DR de reuniões de projeto de agentes. O sistema registra, nomeia e rotula cronologicamente todos os eventos do processo do projeto (eventos significantes e eventos menos significantes). Um evento significativo é definido como adição ou exclusão de classes, agentes, gramáticas ou relacionamentos. Eventos menos significativos incluem o reposicionamento de objetos na tela ou a adição, exclusão ou edição de texto. Com o auxílio de uma lousa eletrônica (Figura 3.3) onde os desenvolvedores fazem diagramas UML de próprio punho, o sistema captura, armazena e interpreta tais diagramas em tempo real, o que auxilia a recuperação posterior do histórico do projeto. Além disso, o sistema também visa à captura de áudio e vídeo.

O sistema Metaglue faz uso de vários agentes, sendo dois agentes principais: *Design Meeting Agent* e *Tahuti*. O agente *Design Meeting Agent* é responsável pela aquisição do histórico do projeto, fazendo uso de áudio, vídeo e serviços disponíveis para captura dos desenhos/caracteres da tela. Além disso, esse agente também inicializa o agente *Tahuti*. O agente *Tahuti*, um agente de reconhecimento de diagramas, reconhece os tipos de diagramas UML elaborados durante a reunião de projeto pelos desenvolvedores e produz os eventos significantes com base nos diagramas. Esses eventos são usados para posicionar os vídeos e áudios, visando a uma recuperação rápida da informação específica.

Figura 3.3: Reunião de projeto (Hammond et al., 2002)

Do estudo realizado sobre esses sistemas que oferecem suporte direto a DR, observa-se uma preocupação relativa às formas de possibilitar meios naturais, a cada novo sistema, para captura de informações de DR, propiciando facilidades aos desenvolvedores. Na próxima

seção, são descritos outros sistemas, de suporte indireto a DR e que também favorecem mecanismos facilitadores de captura e estruturação de DR.

3.2.2 Suporte indireto a DR

HyperCase (Cybulski & Reed, 1992) é um sistema que combina interface com base em hipertexto com um repositório de documentos com base em conhecimento e tem por finalidade dar apoio aos desenvolvedores de *software* no gerenciamento de projetos, análise de sistemas e codificação. É formado por três sub-sistemas: HyperEdit (interface gráfica) ferramenta que permite a criação, modificação e navegação de documentos; HyperBase (base de conhecimento) repositório de hipertexto com base em conhecimento para documentos de *software*; e, HyperDict (dicionário de dados) dicionário de dados com objetivo de controlar a integridade e a consistência do banco de dados no qual todos os documentos são armazenados.

Uma limitação do sistema HyperCase é a centralização do controle de documentação nos gerentes e nos engenheiros de *software* diretamente envolvidos no projeto, enquanto que outros tipos de usuários ficam com pouco suporte para manipulação dos documentos.

HyperDev (Maidantchik & Xexéo, 1996) é um projeto de sistema hipertexto orientado a objetos cujo domínio é de Engenharia de *Software*. Seus principais objetivos são: promover melhor compreensão dos requisitos; evitar o adiamento de decisões; permitir a identificação de características essenciais do *software*; e ser usado como base para manutenção e extensões futuras.

O HyperDev usa *guidelines* para auxiliar os desenvolvedores na construção e manutenção de *software*, definindo o processo com base no padrão de engenharia de *software* ESA PSS-05 e no método OMT (*Object Modeling Technique*) (Maidantchik & Xexéo, 1996).

SASHDoc (Sistema de Autoria e Suporte Hipermídia para Documentação de *Software*) (Silva, 2000) é um protótipo de sistema hipermídia para a produção de documentos de *software*. É composto por dois módulos: um de autoria e outro de navegação. O módulo de autoria é responsável pela elaboração dos documentos de *software* e o módulo de navegação é responsável pela exploração das informações que compõem esses documentos.

Os documentos de *software* do SASHDoc possuem um conjunto de propriedades, cuja finalidade é classificar tais documentos de modo a auxiliar seu entendimento, controlar suas versões e auxiliar na recuperação e navegação dos documentos. Para a exploração de informações dos hiperdocumentos, as propriedades principais são (Silva, 2000): (i) “abstração” para documentos de *software* no SASHDoc, baseia-se no nível de detalhamento do conteúdo do documento acerca do *software*; (ii) “visibilidade” que visa a apresentar as características do *software* sob diferentes pontos de vista; “formalidade” que refere-se à classificação do

documento de acordo com a quantidade de notações matemáticas e lógica formal presentes no documento; (iii) e “funcionalidade” que possibilita a criação de outros documentos além daqueles que descrevem formalmente o *software*.

A seguir, são descritos com mais detalhes a ferramenta CoTeia e o serviço GroupNote, pois os mesmos foram utilizados no desenvolvimento deste trabalho.

3.3 CoTeia

A CoTeia (Arruda Jr. et al., 2002) é uma ferramenta hipermídia colaborativa assíncrona de criação e edição de páginas Web. É uma nova versão da CoWeb (Guzdial (1999)¹ *apud* Arruda Jr. et al. (2002)) implementada sobre plataforma Linux, utilizando servidor Apache, linguagem PHP, especificações XML para estruturação e apresentação dos documentos e banco de dados MySQL, entre outras tecnologias. As características da CoWeb de simplicidade e de autoria colaborativa de documentos Web foram mantidas na CoTeia.

As principais características da CoTeia estão relacionadas a: simplicidade de uso, já que os usuários não têm a necessidade de conhecer a linguagem de marcação HTML para editar suas páginas; a edição das mesmas é feita no próprio *browser*; ausência de estrutura pré-definida das páginas; e funcionalidades que facilitam a colaboração de informações entre os usuários, como *chat*, anotações e mapa de repositório.

A Figura 3.4 (Arruda Jr. & Pimentel, 2001) mostra o modelo entidade-relacionamento da CoTeia. Como pode ser observado, a CoTeia é composta por um conjunto de *swikis*, sendo cada *swiki* um repositório de páginas. Além disso, pode-se dizer que cada *swiki* pode ter diversos arquivos relacionados a ela e somente a ela. Tais *swikis* só podem ser criadas por um administrador e a partir de então utilizadas pelos usuários comuns.

A CoTeia possui diversas funcionalidades, sendo que a criação e edição de páginas são as funcionalidades base para a ferramenta (Arruda Jr., 2001b). Essas duas funcionalidades permitem aos usuários a criação e edição de suas páginas na própria ferramenta de navegação, dispensando conhecimento prévio de linguagens de marcação. As outras funcionalidades da CoTeia são (Arruda Jr., 2001a; Arruda Jr. et al., 2002): gerenciamento de *swiki* – o administrador cadastra o responsável pela *swiki* e controla o tipo de acesso ao repositório de páginas; últimas modificações – lista a última versão de cada página alterada de uma *swiki*; histórico – acesso ao conteúdo das últimas versões de cada página; pesquisa – pesquisa por título, palavra-chave e/ou conteúdo das páginas de uma *swiki* específica ou de todas as *swikis*; *upload* – submissão de arquivos ao servidor; ajuda – página com informações da CoTeia e sua

¹GUZDIAL, M. Supporting learners as users. *The Journal of Computer Documentation*, v. 23, n. 2, p. 3-13, 1999

Figura 3.4: MER da CoTeia (Arruda Jr. & Pimentel, 2001)

sintaxe; recursos de navegação – mapa da *swiki* pelo qual o usuário pode navegar por todas as páginas; edição concorrente de documentos – controle de concorrência quando um ou mais usuários querem editar a mesma página; serviço de *chat* – os usuários trocam informações em tempo real; e, serviço de anotações – pode ser considerado como uma extensão do espaço colaborativo, visto que os usuários podem trocar e compartilhar idéias.

Sua página inicial, Figura 3.5, mostra uma lista com todas as *swikis* disponíveis. Toda vez que o administrador cria uma nova *swiki*, uma nova entrada é criada para a mesma nessa lista. Então, os usuários podem começar a criar e editar as páginas de sua nova *swiki*, colaborativamente.

Figura 3.5: Página Inicial da CoTeia – lista de todas as *swikis* criadas e disponibilizadas para leitura e acesso

A partir da página da *swiki*, as páginas vem sempre acompanhadas de um menu na parte superior que disponibiliza aos usuários as funcionalidades já mencionadas anteriormente e no restante da página encontra-se seu conteúdo, conforme exibido na Figura 3.6.

Figura 3.6: Exemplo de Página da CoTeia

Para editar a página corrente é só selecionar a opção *edit* do menu. Logo em seguida, é aberta a tela de edição, como exibida na Figura 3.7.

Quando o usuário pressiona o botão *Salva* (Figura 3.7), inicia-se um caminho a ser percorrido pelas informações editadas no formulário até que a mesma chegue ao usuário na forma de página como na Figura 3.6. Essa geração dinâmica de hiperdocumento pode ser observada na Figura 3.8 (Arruda Jr. & Pimentel, 2001; Arruda Jr. et al., 2002). Pela figura, observa-se que as informações recebidas dos formulários preenchidos/alterados pelo usuário são enviadas ao processo *CriaXML* (1). Esse processo une tais informações com os dados dos usuários (como, *login* ou *ip* da máquina) (3) e informações do ambiente (como, data de criação da página ou título da *swiki*) (4). Outra função do processo *CriaXML* é converter as *tags*, inseridas pelo usuário no formulário, em caixa baixa para que as mesmas fiquem de acordo com o DTD, além de converter entidades (como acento e símbolos especiais) em entidades HTML e validá-las pelo DTD. Quando houver *tags* de ligação (*<link>*), uma consulta à base de dados (5) é feita para que tal ligação seja definida. Como resultado dessas operações, são armazenadas na base de dados essas informações e um arquivo XML é gerado (6). Após o término do processo *CriaXML*, o processo *GeraXHTML* é iniciado (7) e tem a responsabilidade de validar o documento XML associando-o a uma gramática DTD (8) e após validado, associando-o a uma folha de estilo XSLT (9) para tratamento da apresentação. Em seguida, é feito o processamento do XML validado e folha de estilo

Figura 3.7: Edição de Página na CoTeia

XSLT por meio do Java XT para produzir o arquivo XHTML (10). Com o arquivo XHTML é finalizado o processo *GeraXHTML* (11) e esse arquivo é enviado ao *browser*, podendo ser visualizado pelo usuário (12).

Figura 3.8: Criação/Edição de páginas (Arruda Jr. et al., 2002)

A CoTeia vem sendo utilizada no ICMC – USP, desde 2001, por diversos cursos de graduação e pós-graduação como fonte de informação de projetos e disciplinas, como meio

de escrita colaborativa de artigos, como agenda de reuniões e como repositório de documentos (Arruda Jr., 2003).

3.4 GroupNote

O GroupNote (Izeki, 2001) é um serviço aberto de anotações colaborativas na Web. Foi implementado, assim como a CoTeia, sobre plataforma Linux, utilizando servidor Apache, linguagem PHP, especificações XML e banco de dados MySQL, entre outras tecnologias. Tal implementação deu-se como uma API (*Application Programming Interface*), possibilitando, dessa forma, reuso e combinação do serviço GroupNote com outras aplicações.

A Figura 3.9 (Pimentel et al., 2001) exhibe o modelo conceitual do serviço GroupNote. De acordo com a Figura, as anotações estão dispostas em pastas (*folders*), sendo que cada pasta, além de ser composta por anotações, pode ser composta também por outras pastas, denominadas subpastas. Tem-se que cada anotação pode gerar outra anotação, como que uma anotação pode ser derivada de outra anotação pré-existente. Também pode ser observado na figura que, cada anotação possui um usuário responsável por seu cadastramento, e tais usuários estão organizados em grupos.

Figura 3.9: Modelagem Conceitual do GroupNote (Pimentel et al., 2001)

Assim, pode-se dizer que o GroupNote é composto por quatro APIs (Pimentel et al., 2001): *API Annotation*, *API Folder*, *API User* e *API Group*. A *API Annotation* é responsável pelo gerenciamento das anotações e a *API Folder* pelo gerenciamento das pastas. Já as APIs *User* e *Group* têm a responsabilidade do gerenciamento dos usuários e seus grupos formados.

Com a finalidade de utilizar e testar os recursos fornecidos pelo GroupNote, a ferramenta WebNote (Izeki, 2001) foi desenvolvida. A ferramenta WebNote consiste de pastas compostas

por anotações que podem ser manipuladas colaborativamente. A Figura 3.10 exibe tela típica da ferramenta WebNote, onde o usuário pode, além de visualizar as anotações, inserir, excluir, mover, copiar anotações.

Figura 3.10: Tela da ferramenta WebNote: apresentação das anotações e operações sobre as mesmas

A Figura 3.11 mostra o formulário de inserção de uma nova anotação da ferramenta Webnote. As informações solicitadas são título, três palavras-chave, título e determinação da permissões para cada tipo de usuário sobre a anotação a ser inserida.

Figura 3.11: Inserção de uma anotação na ferramenta WebNote

O serviço GroupNote também foi integrado à CoTeia (Pimentel et al., 2001) para prover à mesma a possibilidade de fazer anotações sobre cada página aumentando ainda mais o fator de colaboração da CoTeia. A Figura 3.12 exibe a tela da anotação a partir da CoTeia. Como pode ser observado, na tela inicial tem-se a opção de criação de uma nova anotação, além da visualização da árvore das anotações existentes de determinada página.

Figura 3.12: Tela inicial da anotação a partir da CoTeia

A Figura 3.13 exibe o formulário que deve ser preenchido, quando a inserção de uma nova anotação é solicitada. É necessário apenas o preenchimento de um título e conteúdo.

Figura 3.13: Tela de inserção de uma anotação a partir da CoTeia

Para ter acesso ao conteúdo de uma anotação, representada na árvore da Figura 3.12, basta selecioná-la que a tela com seus detalhes é aberta (Figura 3.14). Nessa tela, as informações disponíveis, referentes a anotação, são título, autor, grupo, data de criação e conteúdo. Além disso, ainda fornece as opções de responder a essa anotação ou criar uma nova.

Figura 3.14: Tela com informações sobre uma determinada anotação a partir da CoTeia

3.5 Considerações Finais

Muitos sistemas são propostos na literatura para dar suporte a DR, visando à captura, armazenamento e recuperação de DR. Cada qual faz uso dos mais variados recursos, como bases de conhecimento, agentes, ferramentas de crítica, etc, em busca de bons resultados.

Sistemas de apoio a documentação de *software* e a trabalhos colaborativos propostos na literatura, devido aos recursos que disponibilizam, também podem ser usados para captura de DR desde que sejam ajustados para tanto.

A partir do estudo realizado, foi possível a identificação dos aspectos mais relevantes que orientaram o desenvolvimento da ferramenta proposta neste trabalho. Entre os aspectos identificados, destaca-se a adoção de um sistema de hipermídia que forneça um suporte efetivo às representações de DR gerados durante a elaboração dos artefatos de *software*. As diferentes formas de interação, descritas dos sistemas de suporte direto e indireto a DR, também possibilitaram a perspectiva dos auxílios que devem ser fornecidos aos desenvolvedores considerando-se o problema da sobrecarga de tarefas.

No próximo capítulo é apresentada a ferramenta DocRationale desenvolvida neste trabalho.

DocRationale

4.1 Considerações Iniciais

Pesquisas relacionadas a DR, nos mais variados domínios, preocupam-se com o desenvolvimento de métodos e representações apoiadas por computador para a captura, armazenamento, manutenção, recuperação e reuso das decisões tomadas durante o desenvolvimento de um projeto.

Os sistemas de suporte a DR propostos na literatura que, em geral, acarretam muito esforço ao desenvolvedor na captura de DR, facilitam sua recuperação; os que acarretam pouco ou nenhum esforço ao desenvolvedor, dificultam sua recuperação. Sendo assim, é interessante que um sistema de suporte a DR faça uma combinação das soluções propostas pelos sistemas existentes, de forma que, mesmo acarretando certo esforço aos desenvolvedores, proporcione uma boa recuperação de DR.

A captura e recuperação de DR no processo de desenvolvimento de *software* é útil, visto que a documentação do *software* pode possibilitar um bom conhecimento do mesmo e tendo-se acesso ao DR, o *software* pode ficar ainda mais compreensível.

A ferramenta **DocRationale**, proposta neste trabalho, tem por finalidade permitir a captura, armazenamento (focos principais deste trabalho) e recuperação (apenas navegação simples) de DR relacionados aos artefatos de *software*. A ferramenta tem uma representação simples de DR para facilitar seu uso e não acarretar tanto esforço aos desenvolvedores.

Não faz uso de técnicas de Inteligência Artificial, muito utilizada em outros sistemas, pois o objetivo da **DocRationale** não é fazer inferências sobre uma base de informações de projetos, mas sim documentar informações importantes de projeto tendo em vista o reuso dessas mesmas informações em outros projetos. Para tanto, a ferramenta privilegia bastante o fator de colaboração entre os membros da equipe de desenvolvimento para obtenção de DR. Além disso, também faz uso de anotações, consideradas um bom meio de captura de DR (Souza et al., 1998), estruturadas por uma representação simplificada do modelo PHI.

As próximas seções apresentam os requisitos, o suporte a DR, a modelagem, a arquitetura e a implementação da ferramenta **DocRationale**.

4.2 Requisitos

Os sistemas de DR têm por objetivo obter o conhecimento, as suposições e as razões das decisões de projeto de modo que essas informações possam ser úteis aos desenvolvedores no futuro. Para tanto, segundo os estudos do Capítulo 2, obtidos os requisitos necessários para a construção de um sistema de DR. Um sistema de DR deve fazer:

- a **Captura do DR** – funcionalidade relacionada a forma de captura do DR obtido durante todo o desenvolvimento;
- a **Representação do DR** – funcionalidade relacionada ao armazenamento do DR capturado para posterior uso;
- a **Recuperação do DR** – funcionalidade relacionada a recuperação do DR capturado e armazenado.

Para que um sistema capture, represente e recupere DR, é ainda preciso que sejam realizadas escolhas. Escolha da abordagem (orientado por processo ou por característica) a ser utilizada pelo sistema, qual o esquema de representação usar (IBIS, PHI, QOC, DRL), de acordo com a perspectiva escolhida (argumentação, comunicação e argumentação), além dos mecanismos utilizados durante a captura e recuperação do DR.

Considerando-se essas escolhas, a ferramenta **DocRationale** foi elaborada com a finalidade de permitir captura, estruturação, armazenamento e recuperação de DR relacionados aos artefatos de *software*. A **DocRationale** foi construída sob a abordagem orientada por processo, pois pensou-se em projetos que seriam inseridos na ferramenta sem que se tivesse seu domínio, ou seja, projetos que ainda não se tivesse certeza de quais DRs seriam capturados e recuperados. Assim, abriu-se a possibilidade do usuário inserir/excluir fases,

atividades e artefatos durante todo o desenvolvimento, mesmo podendo fazer uso de um modelo inicial, já que tal modelo também pode sofrer modificações com a inserção/exclusão.

Em busca de uma recuperação satisfatória, para a captura e armazenamento do DR, a **DocRationale** adotou uma combinação das perspectivas de comunicação e de argumentação. Sob a perspectiva de comunicação, a ferramenta possibilita que todas as formas de comunicação digital (arquivos de áudio, vídeo, *email*, etc) sejam anexadas, de forma organizada, complementando a documentação do processo. Para a perspectiva de argumentação foram utilizadas as características do esquema de representação PHI (*Procedural Hierarchy of Issues*) ((McCall, 1986) **apud** (Shum, 1991)). A adoção de PHI foi devido a sua simplicidade na representação de DR e sua estruturação para armazenar DR capturado de forma hierárquica e ordenado cronologicamente, além do fato de ter sido adotada a abordagem por processo. Na **DocRationale**, o esquema foi simplificado para utilizar apenas três tipos de nós: questões, posições (respostas) e argumentos. Dessa forma, tem-se uma captura com intervenção do usuário. Para a recuperação, foi estabelecida a navegação, pois para que houvesse outras opções de recuperação, mais estudos seriam necessários.

Além disso, para que a ferramenta **DocRationale** fornecesse suporte ao DR de artefatos de *software* verificou-se a necessidade de que, para cada projeto iniciado, diversos artefatos deveriam ser elaborados, os quais seriam determinados por uma série de questões, posições e argumentos, além de outros arquivos que representem o DR. Também constatou-se que cada projeto possui uma equipe composta por desenvolvedores e um gerente responsável pelo desenvolvimento. Tais requisitos, relativos à estruturação das informações na **DocRationale**, podem ser melhor visualizados na Figura 4.1.

Outro requisito foi o controle de acesso, pois não seria interessante que qualquer pessoa pudesse ter acesso às informações armazenadas. Para restringir o acesso às informações da **DocRationale**, além de cadastrados, os usuários deveriam ter permissão de acesso de acordo com a sua função. Assim, quatro tipos de usuários foram definidos: (i) **Administrador de usuários** – gerencia os usuários da **DocRationale**; (ii) **Administrador de Projetos** – gerencia projetos, artefatos e questões, e navega sobre cses; (iii) **Membro do Projeto** – insere e atualiza artefatos e insere questões, e navega sobre projetos, artefatos e questões; (iv) **Visitante** – apenas navega sobre projetos, artefatos e questões.

Com relação aos requisitos não-funcionais, decidiu-se pelo desenvolvimento da ferramenta para a Web, para que os usuários pudessem acessá-la remotamente, possibilitando que mesmo que não estejam no mesmo espaço físico ao mesmo tempo, pudessem trabalhar no desenvolvimento de seu sistema. Além disso, determinou-se que a ferramenta fosse implementada com um servidor Web estendido com um interpretador PHP e um servidor de banco de dados relacional, o MySQL.

Figura 4.1: Informações a serem manipuladas na ferramenta **DocRationale**

A linguagem utilizada PHP (*Hypertext Preprocessor*) foi projetada para a Web, possui sintaxe semelhante a das linguagens C, Perl e Java. Com essa linguagem, é possível a criação de aplicações Web dinâmicas, o que disponibiliza a interação com os usuários por meio de formulários, parâmetros de URL e *links*. O código PHP é interpretado no servidor Web, gerando HTML, o qual é enviado para o usuário, ou seja, além de não ser necessário instalar nenhum recurso extra no cliente, evita-se que a interação com banco de dados e aplicações existentes no servidor fiquem expostas para os usuários, possibilitando a garantia de segurança na aplicação. Enquanto nas linguagens C ou Perl, os *scripts* CGI (*Common Gateway Interface*) têm que gerar todo o código HTML ou ler de outro arquivo; na linguagem PHP, o código é embutido no próprio código HTML. Uma característica relevante da linguagem PHP é o fato de dar suporte a diversos banco de dados, além da simplicidade para a geração de consultas. Entre os SGBDs (Sistemas Gerenciadores de Banco de Dados) suportados estão Adabas D, dBase, Empress, Informix, Ingress, InterBase, FrontBase, mSQL, MySQL, Oracle, PostgreSQL e SyBase. PHP também suporta diversos serviços através dos protocolos IMAP, SNMP, NNTP, POP3, HTTP e *sockets* (Simão et al., 2003).

MySQL é um gerenciador e servidor de base de dados rápido, multi-tarefa e multi-usuário. Por ser multiplataforma, pode ser executado sobre diversos sistemas operacionais, como Linux, Unix, Windows, FreeBSD, NetBSD e Solaris. Outra vantagem do MySQL é a possibilidade de criação de usuários com diferentes privilégios de acesso aos dados, garantindo que as informações armazenadas sejam acessadas/modificadas apenas pelos usuários autorizados (Seraphim et al., 2003).

Além disso, PHP e MySQL são produtos de código-fonte abertos (Welling & Thomson, 2001), o que possibilita modificações e adaptações de acordo com as necessidades específicas de suas aplicações.

4.3 O Suporte a DR na DocRationale

Para apoiar a principal funcionalidade da **DocRationale**, o suporte a DR, foi estabelecida sua integração com a CoTeia (Arruda Jr. et al., 2002) e GroupNote (Pimentel et al., 2001).

CoTeia, como apresentada no Capítulo 3 é uma ferramenta hipermídia colaborativa assíncrona de criação de páginas *Web*. Possui diversas funcionalidades, sendo as mais interessantes para a **DocRationale**: (i) criação e edição de páginas *Web*, que são feitas no próprio */textitbrowser* via formulário HTML (*HyperText Markup Language*); (ii) histórico, que permite o acesso ao conteúdo das últimas versões de cada hiperdocumento; e (iii) *upload*, que possibilita a submissão de arquivos de qualquer formato ao servidor CoTeia.

GroupNote, também já apresentada no Capítulo 3, é um serviço aberto de anotações colaborativas na *Web* implementado como uma *API* (*Application Programming Interface*). Suas características interessantes para **DocRationale** são: (i) compartilhamento de anotações por grupos de usuários; e (ii) fornecimento da funcionalidade de hierarquia de nós.

Dessa forma, por meio da integração com a CoTeia, é possível a autoria colaborativa e inclusão de anexos (DR). E da integração com o GroupNote, tem-se a qualificação das anotações como questão, posição e argumento contra ou a favor.

4.4 Modelagem

Para a modelagem conceitual dos requisitos da ferramenta **DocRationale**, foi elaborado o diagrama de casos de uso da UML (*Unified Modeling Language*) (Booch et al., 2000), além do esquema conceitual, esquema navegacional, esquema de contexto navegacional, ADVs (*Abstract Data Views*) e ADVcharts da metodologia OOHDM (*Object-Oriented Hypermedia Design Method*) (Rossi, 1996).

O diagrama de casos de uso descreve o modelo de caso de uso em UML, fornecendo uma visão externa da ferramenta e dos objetivos das entidades externas. Dessa forma, o modelo de caso de uso auxiliou a descrição dos requisitos funcionais da **DocRationale**, mostrando as funcionalidades da ferramenta, além das relações da mesma com as entidades externas. A Figura 4.2 apresenta o diagrama de casos de uso da **DocRationale**.

Figura 4.2: Diagrama de casos de uso da ferramenta **DocRationale**

A metodologia OOHDM (Schwabe et al., 1996) tem sido amplamente utilizada para o desenvolvimento de aplicações hipermídia, acompanhando-as desde a análise do domínio até a implementação. Para tanto, está dividida em quatro atividades: modelagem conceitual, modelagem navegacional, projeto da interface abstrata e implementação. Cada uma das atividades gera diversos produtos; os elaborados para a **DocRationale** foram: esquema conceitual (modelagem conceitual); esquema navegacional e esquema de contexto navegacional

(modelagem navegacional); ADVs e ADVcharts (projeto da interface abstrata); e, aplicação em execução (implementação).

O esquema conceitual representa o resultado da análise do domínio da ferramenta. É um diagrama formado por classes representativas da ferramenta, seus relacionamentos (associação, generalização, composição, agregação entre outros) e os subsistemas que as agrupam. O esquema fornece uma visão estática da estrutura de suporte da **DocRationale**, formado por classes e suas associações. A Figura 4.3 apresenta o esquema conceitual da ferramenta **DocRationale**.

Figura 4.3: Esquema Conceitual da ferramenta **DocRationale**

Para a modelagem navegacional, foram elaborados o esquema navegacional e o esquema de contexto navegacional.

O esquema navegacional é um diagrama formado por nós e elos precedentes das classes e relacionamentos definidos no esquema conceitual. Uma aplicação hipermídia pode ter um ou mais esquemas navegacionais, dependendo do número de visões existentes da mesma. No caso da **DocRationale**, com o esquema navegacional foi possível estabelecer o conjunto de nós e elos que fazem parte de duas visões navegacionais distintas.

O esquema de contexto navegacional, estabelecido após o esquema navegacional, foi elaborado para definir em quais contextos haverá navegação entre as informações e que informações serão mostradas. Para a **DocRationale**, foi determinado, por meio do esquema de contexto navegacional, os cios e os nós (objetos navegacionais) que devem estar relacionados de acordo com as funcionalidades levantadas nos casos de uso (Figura 4.2).

Tanto os ADVs quanto os ADVcharts são definidos durante o projeto da interface abstrata. Os ADVs determinam os aspectos estáticos dos objetos de interface, além de mostrar como esses objetos de interface estão relacionados com os objetos navegacionais e como se comportam diante de eventos externos. Os ADVs são utilizados para determinar a aparência dos objetos da ferramenta sem se preocupar com a sua implementação. Os ADVcharts mostram os estados e alterações na interface com o usuário e nos objetos navegacionais que acontecem nos ADVs, ou seja, definem os aspectos dinâmicos da ferramenta. Por necessidade de melhor representar o comportamento das telas da **DocRationale**, foi acrescentada aos ADVcharts a característica de *broadcasting* presente nos StateCharts (Harel et al., 1987). Tal característica adicionada não deve causar problemas, já que os próprios ADVcharts são uma generalização dos StateCharts. Dessa forma, por exemplo, no ADVchart da Figura A.15 foi possível definir-se o encadeamento de eventos/ações que são disparados instantaneamente, como pode ser notado na transição rotulada por “2/3” do estado “Conecta” em “Botão”, que acarreta o disparo da transição rotulada por “3” no subestado paralelo do “Formulário”.

A modelagem da ferramenta **DocRationale**, por ser extensa, é apresentada no Apêndice A.

A implementação da ferramenta **DocRationale**, quarta etapa da metodologia OOHDM, está descrita na Seção 4.6. Para que seja melhor compreendida a implementação da ferramenta, a seção seguinte mostra detalhes de sua arquitetura.

4.5 Arquitetura

Na arquitetura da ferramenta **DocRationale** foram consideradas as informações contidas nas Seções 4.2 (Requisitos), 4.3 (Suporte a DR, por meio da integração com GroupNote e CoTeia) e 4.4 (Modelagem dos objetos analisados).

Uma visão geral da arquitetura da **DocRationale** pode ser vista na Figura 4.4. Nessa figura é mostrada a integração da ferramenta **DocRationale** com a CoTeia e GroupNote por meio dos blocos unidos. Também pode ser observado que tanto a ferramenta **DocRationale** quanto a CoTeia fazem uso das anotações do GroupNote, assim como a **DocRationale** faz uso das páginas e anexos da CoTeia.

Figura 4.4: Visão Geral da integração da **DocRationale** com a **CoTeia** e **GroupNote**

A arquitetura da ferramenta **DocRationale** pode ser visualizada na Figura 4.5. Observa-se que tão logo o usuário solicite acesso à **DocRationale**, o módulo *Controle de Acesso* (CA) é acionado para validação e disponibilidade do usuário. Para tanto, o módulo CA utiliza informações contidas na base de dados *Usuários*. Assim que é validado, o usuário é direcionado para as informações a que tem direito de acesso. Se o usuário for um gerente de usuários, o módulo CA faz uma requisição ao módulo *Gerenciamento de Usuários* (GU) que atende essa requisição com o auxílio da informações contidas na base de dados *Usuários*. Porém, se o usuário for qualquer um dos outros três tipos de usuários (administrador de projetos, membro e visitante) uma requisição é feita ao módulo *Gerenciamento de Projetos* (GP). O módulo GP trabalha com as informações contidas na base de dados *Projetos de Software* e, quando necessário, a base de dados *Usuários* (por meio de outros módulos). Durante o funcionamento da ferramenta, o módulo GP faz requisições e atende a requisições dos módulos *Gerenciamento de Equipe* (GE), *Gerenciamento de Fases* (GF), *Gerenciamento de Atividades* (GAt) e *Gerenciamento de Artefatos* (GAR). O módulo GE também envia requisições ao módulo GU, em busca de informações sobre os usuários. O módulo GAR faz requisições tanto para o módulo GAt quanto para o módulo GF, além de atender e fazer requisições ao módulo *Gerenciamento de Questões* (GQ). Os módulos GP e GAR fazem requisições à **CoTeia** por meio do módulo *Tradutor de Dados para a CoTeia* (TDC). Tal módulo serve como um intermediador de troca de informações entre a **DocRationale** e a **CoTeia**. Da mesma forma, o módulo GQ consegue a troca de informações entre **DocRationale** e **GroupNote** por meio do módulo *Tradutor de Dados para o GroupNote* (TDG). Com o auxílio dos módulos tradutores TDC e TDG, a **DocRationale** tem acesso às bases de dados da **CoTeia** e **GroupNote**, necessário para a ligação entre as informações compartilhadas entre os mesmos.

Figura 4.5: Arquitetura da ferramenta **DocRationale**

Módulos específicos da **DocRationale**

O módulo CA é responsável pelo controle de acesso aos diversos módulos da **DocRationale**, CoTeia e GroupNote. Os usuários pertencentes ao grupo *Administrador de Usuários* só acessam o módulo *Gerenciamento de Usuários*. Por outro lado, os usuários dos grupos *Administrador de Projetos*, *Membro* e *Visitante* podem acessar os demais módulos da **DocRationale**, bem como os módulos da CoTeia e GroupNote. Cabe salientar que esses usuários possuem diferentes níveis de permissão nesses módulos.

O módulo GU, conforme mencionado anteriormente, só pode ser acessado por usuários do grupo *Administrador de Usuários*, os quais são responsáveis por cadastrar usuários, autorizar ou rejeitar uma solicitação de cadastro e, ainda, definir qual o papel a ser desempenhado por esse usuário na **DocRationale**.

Os módulos gerenciadores GP, GAr, GQ, GE, GF e GAt são responsáveis pela manutenção de seus respectivos dados, ou seja, inserir, editar e excluir. Esses módulos ainda permitem a realização de pesquisas nas bases de dados. Cabe ressaltar a interdependência entre esses módulos, como por exemplo o fato de que o GQ sempre estar relacionada ao GAr e, esse sempre estar relacionado ao GP.

Os módulos tradutores TDC e TDG são responsáveis pela comunicação entre a **DocRationale** e a CoTeia e entre a **DocRationale** e o GroupNote, respectivamente. Esses

módulos são compostos por funções que recebem dados provenientes da **DocRationale**, os quais são trabalhados e, posteriormente, repassados às funções da CoTeia ou do GroupNote para atualização de suas respectivas bases de dados.

4.6 Implementação

Antes da implementação da **DocRationale**, foram estabelecidas as ligações entre a mesma e a CoTeia e o GroupNote, devido à necessidade de sua integração para utilização dos recursos tanto da CoTeia quanto do GroupNote.

Figura 4.6: Ligações entre **DocRationale**, **CoTeia** e **GroupNote**

A Figura 4.6 mostra uma visão geral da integração de **DocRationale** com a CoTeia e GroupNote de uma perspectiva do conjunto de páginas Web que são disponibilizadas. Nessa figura, pode-se observar que a CoTeia e **DocRationale** possuem telas correspondentes para projetos, artefatos e questões. Isso possibilita o acesso às informações tanto pela **DocRationale** quanto pela CoTeia.

A criação e atualização das páginas encontrados na CoTeia são de responsabilidade da **DocRationale**. Para tanto, foram definidas funções na **DocRationale** que transmitem seus

dados de forma a serem compreendidos por funções da CoTeia e GroupNote, para posterior processamento. Entre as funções elaboradas estão a de criação de *swikis* e páginas, além das de atualização de páginas. Assim, a cada inserção de um projeto ou artefato na base de dados da **DocRationale**, a CoTeia também é atualizada e em **DocRationale** é colocado um *link* para a página correspondente da CoTeia. De forma semelhante, a cada inserção de uma questão na base de dados da **DocRationale**, uma anotação é criada no GroupNote, além de ser feita a atualização da página CoTeia. As anotações podem ser visualizadas por *links*, tanto na **DocRationale** quanto na CoTeia. Todos os arquivos de DR e artefatos são inseridos diretamente na CoTeia, onde ficam armazenados.

Para que a CoTeia e GroupNote funcionassem de acordo com os requisitos da **DocRationale**, algumas modificações tanto na CoTeia quanto no GroupNote foram necessárias. Na CoTeia foram desabilitadas a opção de anotação do menu para todas as páginas e a opção de *upload* na página relacionada aos projetos. Além disso, foi alterado o código do *upload* para que o mesmo pudesse inserir diretamente o arquivo na página do artefato, já que tal página não pode ser editada pelo usuário. No GroupNote, para prover discussões sobre cada questão levantada pelo usuário desenvolvedor, a base de dados de usuários do GroupNote foi substituída pela base de dados de usuários da **DocRationale**. Outra modificação necessária foi a troca dos tipos de nós, existindo apenas dois níveis de discussão: no primeiro nível, uma posição é colocada em relação à questão levantada; no segundo nível, usuários argumentam contrariamente ou a favor à posição. Finalmente, foi inserido recursos de autenticação, sessão e permissão para a CoTeia e GroupNote, bem como para a **DocRationale**, para que apenas usuários autorizados tivessem acesso às informações armazenadas.

Para a implementação dos recursos de autenticação, sessão e permissão foi utilizada a biblioteca PHPLib (PHPLib, 2003), que é um *framework* para desenvolvimento de aplicações PHP que provê um conjunto de classes que auxiliam na manipulação de tais recursos. Para a utilização desse *framework* é necessário definir algumas constantes, como as relacionadas: (i) ao banco de dados (servidor, nome do banco, etc); (ii) à sessão (tabela, método, tempo de vida, etc); (iii) à permissão (*array* de permissões - baseada em *bits*, etc); (iv) à autenticação (tabela, modo, tempo de espera, etc).

Além dessas constantes, foi necessária a criação de duas tabelas: a primeira para armazenar informações de sessões ativas (tabela **active_sessions**) e; a outra para armazenar informações sobre os usuários e suas respectivas permissões (tabela **auth_user**).

Em complemento a essas tabelas, para definir a base de dados da ferramenta **DocRationale**, seguindo os Esquemas Conceitual e Navegacionais elaborados para a mesma, foi feito o mapeamento dos objetos e seus relacionamentos para o modelo relacional. Cada objeto da **DocRationale** foi mapeado para uma tabela diferente e para cada atributo de um objeto

foi definido um campo na tabela correspondente. Quando o relacionamento entre objetos era de 1:N, o mapeamento foi feito adicionando-se uma chave estrangeira (FK - *Foreign Key*) na tabela representante da classe do lado N e, quando o relacionamento era de N:N, o mapeamento foi feito por meio da criação de uma nova tabela, cujos campos são chaves das tabelas representantes das classes relacionadas.

Dessa forma, foram criadas as tabelas **user** (**auth_user** estendida e **prov_user**), **project**, **artefact**, **issue**, **phase**, **activity**, **team**. Para simplificar as pesquisas feitas pela ferramenta em busca dos usuários e/ou candidatos, a tabela **user** que deveria guardar tanto os usuários quanto os candidatos a usuários foi dividida em duas outras tabelas: **auth_user** - mantém apenas os usuários da ferramenta - e **prov_user** - mantém os candidatos a usuários da ferramenta até que os mesmos sejam aceitos e transferidos para a tabela **auth_user** ou rejeitados e eliminados da tabela (**prov_user**). A tabela **project** contém os dados dos projetos descritos nos esquemas, mais o atributo *id_swiki* (FK da CoTeia) que fornece a identificação da *swiki* correspondente ao projeto. A tabela **team** possui os dados determinados nos esquemas, possibilitando que, por meio dessa, chegue-se ao responsável por determinado projeto. As tabelas **phase** e **activity** possui a chave estrangeira de **project** (*id_proj*). A tabela **artefact**, além dos dados já descritos pelos esquemas, possui os atributos *id_proj* (FK de **project**), *id_page* (FK de **page** - classe da CoTeia que indica a página correspondente) e *id_user* (FK de **auth_user**). A tabela **issue** também possui as chaves estrangeiras de **auth_user** (*id_user*), **artefact** (*id_art*) e **annotation** do GroupNote (*id_note*).

4.7 O Produto Desenvolvido

Nas telas de projetos, artefatos e questões, que são semelhantes, as funcionalidades disponíveis são facilmente visualizadas a cada instante. As telas possuem um cabeçalho com o nome da ferramenta e operação que está sendo fornecida no momento. Logo abaixo do cabeçalho encontra-se um menu com todas as operações possíveis e no rodapé podem ser encontradas informações sobre o usuário e sua permissão no sistema e projeto, nome do projeto e do artefato em que o usuário encontra-se. Na parte central da tela aparecem as informações relevantes à opção feita pelo usuário anteriormente, onde encontram-se ícones que levam à CoTeia ou ao GroupNote. Para validar os dados a serem enviados pelos usuários, ainda no lado do cliente, foi utilizada a linguagem JavaScript, uma vez que essa pode ser inserida dentro do código HTML e é facilmente interpretada pela maioria de *browsers* existentes.

A Figura 4.7 mostra uma tela da **DocRationale** com a identificação na tela, das partes mencionadas. No caso, a tela exibida na Figura 4.7 é de questões de um artefato pertencente a um projeto da **DocRationale**.

Figura 4.7: Tela com identificação de partes da ferramenta **DocRationale**

A seguir, são exibidas telas da **DocRationale**, como se um usuário estivesse acessando-a. Dessa forma, a primeira tela a ser visualizada pode ser observada na Figura 4.8. Essa tela serve como uma tela de “boas vindas”, a “porta de entrada” para a ferramenta **DocRationale**.

Figura 4.8: “Porta de entrada” da ferramenta **DocRationale**

A próxima tela a ser vista pelo usuário, Figura 4.9, solicita seu *username* e senha. Apresenta ainda a opção de solicitação de senha caso o usuário tenha esquecido a mesma e se a pessoa não tiver conta na **DocRationale**, é possível requisitar uma.

Figura 4.9: Tela de *login*

Após ter sido efetuado o *login* com sucesso, o usuário passa a navegar pelas informações contidas na ferramenta **DocRationale**. Inicialmente, o usuário é levado até a página que contém a lista de todos os projetos em que o usuário participa (como gerente, membro ou visitante), Figura 4.10. Também nessa tela tem-se disponível um menu com opções gerais do momento, além de um menu específico de operações sobre os projetos listados que encontra-se juntamente com tais projetos na parte central da tela. As opções desse menu específico depende da permissão que o usuário possui. No caso da Figura 4.10, as opções do menu geral são *Alterar Senha* e *Sair* e, como o usuário logado tem permissão de administrador de projetos (observar essa informação no rodapé da tela), as opções do menu específico são *Inserir*, *Localizar*, *Editar* e *Excluir*. Na tabela que lista os projetos, tem-se um *link* para cada projeto que conduz para página correspondente da CoTeia, um campo para seleção de projeto para edição ou exclusão e, ainda, no próprio nome do projeto existe um *link* que leva o usuário para a lista de artefatos desse projeto. Ainda na parte central da tela, tem-se uma legenda na parte inferior à esquerda, indicando através das cores vermelha, azul e preta se os projetos estão atrasados, em andamento dentro do prazo previsto ou já finalizados, respectivamente.

Escolheu-se o projeto GradHelp para mostrar as demais telas da **DocRationale**. Essas telas fazem parte do trabalho de um grupo de alunos de graduação da disciplina de Hipermídia, ministrada no segundo semestre de 2003 do ICMC-USP. O objetivo do projeto da disciplina, no escopo de desenvolvimento de um sistema de informação na Web, é o de construir um aplicativo com base na metodologia OOADM que dê suporte a informações acadêmicas para estudantes de graduação. Para implementação do sistema de informação, adotou-se as tecnologias W3C, e por isso as páginas e documentos relativos ao sistema, deviam ser especificados

Figura 4.10: Tela de projetos

em XML. Todos os artefatos produzidos, durante as fases de desenvolvimento, bem como as discussões relevantes a respeito de sua elaboração foram registrados na **DocRationale**.

O projeto GradHelp, mostrado a seguir, tem como finalidade: a comunicação de datas de provas e trabalhos; e, o compartilhamento de provas anteriores, lista de exercícios e descrição de trabalhos. A Figura 4.11 exibe a lista de artefatos pertencente ao projeto GradHelp. Essa figura segue o mesmo esquema da tela que exibe a lista de projetos. Como pode ser observado na Figura 4.11, o menu geral tem novas opções: (i) *Projetos* (volta para a tela que lista os projetos na qual o usuário participa), (ii) *Projeto* (vai para a tela que fornece mais informações sobre o projeto em questão), (iii) *Equipe* (vai para a lista dos usuários componentes da equipe de desenvolvimento do projeto), (iv) além das opções sempre constantes nesse menu (*Alterar Senha* e *Sair*). No rodapé estão as informações como o nome do projeto em que o usuário está e qual a sua permissão nesse projeto. Na parte central, juntamente com a lista dos artefatos, encontra-se o menu específico com as mesmas opções do menu específico da lista de projetos, só que tais opções são disponibilizadas para as ações sobre os artefatos. Na tabela tem-se a possibilidade de ir para a página da CoTeia correspondente a cada artefato por meio dos *links* e também tem-se o campo para seleção do artefato que deseja-se editar ou excluir (essas opções são possíveis devido a permissão do usuário no projeto), além do *link* no próprio nome que leva o usuário a página com a lista de

todas as questões discutidas durante a elaboração de determinado artefato. E novamente, tem-se a legenda seguindo o mesmo esquema da tela da lista de projetos.

Figura 4.11: Tela de artefatos

Feita a escolha pela opção *Projeto* do menu geral (Figura 4.11), a tela exibida é a que aparece na Figura 4.12. Nessa tela aparecem as informações sobre o projeto em questão: nome, fases (diversas fases que podem existir no projeto), atividades (diversas atividades que podem existir no projeto), data de início, data prevista do término, data do término, responsável pelo projeto (administrador do projeto), condição do projeto (em adamento dentro do prazo previsto, atrasado ou finalizado) e um *link* para a página correspondente do projeto na CoTeia. Se o usuário tivesse permissão de administrador de projetos nesse projeto, poderia ainda editar/excluir as fases/atividades do projeto.

Caso fosse feita a escolha da opção *Equipe* do menu geral (Figura 4.11 ou Figura 4.12), o usuário teria a sua disposição a tela com a lista dos usuários que formam a equipe de desenvolvimento do projeto, Figura 4.13. Como o usuário em questão tem permissão de membro nesse projeto, ele apenas pode localizar um usuário participante da equipe. Se o usuário possuir o administrador do projeto, pode inserir/edital/excluir usuários da equipe.

Se o usuário escolhesse um dos artefatos da Figura 4.11, a tela a ser exibida seria a da Figura 4.14. Nessa figura encontra-se a lista de questões referentes ao artefato em questão. No menu geral tem-se as opções *Projetos* (lista dos projetos nos quais o usuário participa), *Projeto* (informações sobre o projeto atual), *Artefatos* (lista de artefatos do projeto atual) e *Artefato* (informações sobre o artefato atual), além das opções *Alterar Senha* e *Sair*. Também

Figura 4.12: Tela de projeto

Figura 4.13: Tela de equipe

é possível, por meio do menu específico, inserir, localizar e finalizar questões, desde que o usuário não tenha permissão de visitante que pode apenas localizar questões. No rodapé a informação acrescentada é o nome do artefato. Na parte central da tela, além do menu específico, já mencionado, tem-se a lista das questões com *links* para sua anotação respectiva (GroupNote), onde encontram-se, as posições e argumentos a favor e contra. Existe um campo para seleção da questão que deseja-se finalizar (para finalizar é aberta um janela onde o usuário deve informar a que conclusão a equipe chegou a respeito da questão discutida). Quando finalizada, a questão recebe outro *link* que leva a uma janela com a conclusão a respeito da mesma. Ainda, na parte central existe a legenda com duas possíveis cores para as questões: azul, para questão em discussão, ainda aberta, e preta, para questão finalizada, já concluída.

Figura 4.14: Tela de questões

Feita a escolha pela opção *Artefato* do menu geral (Figura 4.14), a tela exibida é a que aparece na Figura 4.15. Nessa tela aparecem as informações sobre o artefato em questão: nome, fase (a qual pertence o artefato), atividade (a qual pertence o artefato), data de início, data prevista do término, data do término, responsável pelo artefato, condição do artefato (em andamento dentro do prazo previsto, atrasado ou finalizado) e um *link* para a página correspondente do artefato na CoTeia.

A Figura 4.16 mostra a página do projeto GradHelp na CoTeia. Como pode ser observado, na página da CoTeia referente ao projeto encontram-se as informações a respeito do projeto

Figura 4.15: Tela de artefato

e também a lista com todos os seus artefatos. Caso alguma alteração nos dados do projeto seja feita na **DocRationale**, a página da CoTeia também é atualizada, assim como quando há inserção de um novo artefato.

Figura 4.16: Tela de projeto na CoTeia

A Figura 4.17 exibe a página do artefato Esquema Conceitual do projeto GradHelp na CoTeia. Na página podem ser encontradas informações sobre o artefato, questões relacionadas ao mesmo com *links* para o GroupNote onde encontram-se as posições e argumentos a favor e contra, arquivos relacionados ao artefato, além do próprio artefato. Esses arquivos são inseridos na página CoTeia do artefato por meio da opção *uploads* do menu da página. Também é possível ter nessa página uma lista com páginas relacionadas ao artefato, as quais podem ser criadas através do ícone *Nova Página* que encontra-se no corpo da página do artefato. A lista de questões recebe o mesmo *link* que recebe a lista de questões da **DocRationale** quando tal questão é finalizada, disponibilizando a conclusão da questão também na CoTeia.

Figura 4.17: Tela de artefato na CoTeia

A anotação do GroupNote que contém a questão “Que tipo de usuário poderá excluir provas anteriores?”, suas posições e argumentos a favor e contra podem ser vistos na Figura 4.18. Como pode ser observado, as posições ficam num primeiro nível e os argumentos em um segundo nível. No exemplo, essa questão, ela possui duas posições, sendo a primeira com um argumento contra e a segunda com um argumento a favor. Juntamente com a posição ou argumento vem a data, a hora e o nome do usuário que fez sua inserção.

Caso o usuário queira efetuar *logout* é só escolher a opção *Sair* que encontra-se no menu geral da ferramenta **DocRationale**. Assim que é efetuado o *logout*, uma tela aparece com uma

Figura 4.18: Tela de questão no GroupNote

mensagem de agradecimento pelo uso da ferramenta e com um *link* para possível retorno a tela inicial da **DocRationale** caso o usuário queira usar novamente a ferramenta, como pode ser visto na Figura 4.19.

Figura 4.19: Tela de *logout*

As telas para alteração de senha (opção encontrada no menu geral da **DocRationale**), registro (solicitação de cadastro na **DocRationale** que encontra-se na tela de *login*) e solicitação de nova senha por esquecimento da antiga (encontra-se também na tela de *login*) são formulários comuns de preenchimento e sem ênfase no domínio da ferramenta.

Quando o usuário logado tem permissão de administrador de usuário, ele não vê nenhuma dessas telas com conteúdo dos projetos, artefatos ou questões mostradas anteriormente, pois seu acesso é restrito às operações sobre os usuários. O administrador de usuários terá uma tela com a lista de usuários da ferramenta e outra com a lista de candidatos a usuários da ferramenta que seguem o mesmo esquema da tela com a lista de projetos, tornando-se desnecessária a exibição das mesmas.

4.8 Considerações Finais

A documentação de *software* visa a proporcionar um bom conhecimento do *software*, tornar menos complicada a sua manutenção e facilitar o seu reuso (Pfleeger, 2001). Porém, por falta de hábito ou cultura dos desenvolvedores, tais documentos registram somente as decisões finais de determinada fase (Burge & Brown, 2000). Se as alternativas investigadas para cada artefato gerado durante o desenvolvimento do *software* fossem registradas, o *software* tornaria-se mais fácil de manipular, seja para manutenção, reuso ou quando novos integrantes passam a fazer parte da equipe de desenvolvimento do *software*. Em busca de suporte a DR de artefatos de *software*, a ferramenta **DocRationale** foi desenvolvida.

DocRationale é uma ferramenta para a Web que privilegia o fator de *colaboração* entre os membros de uma equipe para a obtenção de DR. Faz uso de armazenamento estruturado de DR que, apesar de causar certo ônus ao usuário (desviando-o de suas atividades habituais), visa a facilitar a sua posterior recuperação, pois sendo armazenado de forma estruturada, a obtenção de informações que realmente sejam úteis aos usuários é facilitada. Dessa forma, torna-se um incentivo, já que o esforço do usuário será recompensado pela utilidade das informações recuperadas. Outro incentivo para o uso da **DocRationale** é a utilização de anotações estruturadas por uma simplificação do modelo PHI que facilita a inserção de DR, em contraste com sistemas que fazem uso de esquemas mais complexos.

Os próximos passos no desenvolvimento da infraestrutura fornecida com a **DocRationale** estão direcionados para um suporte à navegação e facilidade de pesquisas, possivelmente por mecanismos de hipermídia adaptativa, sobre o DR capturado.

Conclusões

A atividade de documentação do processo de *software* tem o objetivo de registrar as informações relevantes do *software* para tornar menos complicada a sua manutenção e facilitar o seu reuso. Diversos artefatos são gerados durante o desenvolvimento do *software* e os documentos, em geral, registram informações referentes às decisões finais de determinada fase, desconsiderando as alternativas investigadas para resolver determinado problema. Entretanto, se as opções, justificativas e decisões que levaram a uma determinada escolha *Design Rationale* (DR) - fossem registradas, o *software* tornaria-se mais compreensível, facilitando a sua manutenção, seu reuso e entendimento por novos integrantes da equipe.

Assim, neste trabalho, foram realizados estudos sobre DR com a finalidade de desenvolver uma ferramenta - **DocRationale** - que desse suporte a DR de artefatos de *software*. A ferramenta **DocRationale** permite a captura, armazenamento e recuperação de DR na Web. Embora não tenham sido utilizadas técnicas de Inteligência Artificial, muito utilizada em outros sistemas de suporte a DR, a **DocRationale** privilegia o fator de colaboração entre os desenvolvedores e faz uso de anotações estruturadas como uma simplificação do modelo PHI para captura e armazenamento de DR, que apesar de causar certo esforço aos desenvolvedores, visa a facilitar a recuperação de informações úteis para os mesmos.

5.1 Contribuições

As contribuições deste trabalho:

- Estudo sobre *Design Rationale* (DR);
- Projeto e implementação da ferramenta **DocRationale**, incluindo alterações e integração com a ferramenta CoTeia e serviço GroupNote;
- Dois artigos publicados: “*Um Sistema de Apoio à Utilização de Design Rationale de Artefatos de Software*” CLEI 2003 (Francisco et al., 2003b) e “**DocRationale: Um Auxílio a Design Rationale na Documentação de Software**” Sessão de Ferramentas do SBES 2003 - (Francisco et al., 2003a);
- Um artigo submetido: “*Supporting Design Rationale for Software Artifacts*” – Ninth IEEE Symposium on Computers and Communications (ISCC 2004);
- Apoio a captura e recuperação de DR por meio da utilização da **DocRationale** por alunos de graduação do curso de Hipermídia (SCE 225) no segundo semestre de 2003. Os vinte e sete alunos formaram oito grupos com uma média de três alunos por grupo para o desenvolvimento de seus projetos. Os oito projetos elaborados pelos alunos geraram quarenta e seis artefatos, sessenta e uma questões e sessenta arquivos. Ou seja, foram construídas cinquenta e quatro páginas na CoTeia e sessenta e uma entradas para anotações no GroupNote utilizados pela **DocRationale**.

5.2 Trabalhos Futuros

Como trabalhos futuros tem-se:

- Investigação de mecanismos para tornar a recuperação de DR mais eficiente pela ferramenta **DocRationale**. Para tanto, a aluna de mestrado, Lisandra Cazassa Fumagalli (Fumagalli, 2003), tem desenvolvido seu trabalho em busca da melhoria da recuperação de DR. Ela tem feito estudos para transformação dos documentos e DRs registrados na **DocRationale** em hiperdocumentos XML. Também irá implementar um *crawler* para percorrer esses hiperdocumentos (textos e *links*) de forma a recuperar a informação que atenda às buscas realizadas pelos usuários;
- Implantação de um controle de versão para a **DocRationale** com o objetivo de proporcionar ao usuário da ferramenta o histórico do registro do DR. Esse trabalho já está

sendo estudado pela aluna de mestrado, Sandra Regina Quadros Moraes da Silva. Ela está fazendo um estudo de uma ferramenta de controle de versões (CVS *Concurrent Versions Systems*) para possivelmente inseri-la na **DocRationale**.

- Investigação da utilização de hiperdocumentos para apoiar os armazenamento e recuperação de artefatos e DR, possibilitando a troca e a busca mais produtiva das informações para as tomadas de decisão necessárias durante o processo de engenharia de requisitos. Para tanto, o aluno de mestrado, Marco Aurélio Graciotto Silva (Silva, 2003), estuda a documentação dos requisitos utilizando a **DocRationale** e a CoTeia com o objetivo de desenvolver uma ferramenta *Wiki* para autoria ágil e colaborativa de documentos de requisitos, aliando mecanismos de gerenciamento de configuração e suporte a DR.
- Investigação da utilidade do uso de DR na captura do conhecimento dos membros das equipes envolvidos na administração e execução das atividades relacionadas a melhoria de processo de *software*. Considerando-se que tal melhoria envolve um aprendizado contínuo de novas metodologias, tecnologias e práticas de gerenciamento e envolve também altos custos para a organização com treinamento de funcionários, consultorias e certificações, torna-se relevante a estruturação do conhecimento adquirido de forma a permitir registrar as experiências daqueles membros que assumem determinados papéis em atividades de melhoria de processo de *software*. A aluna de doutorado, Débora Maria Barroso Paiva, com esse estudo, pretende alcançar o aprimoramento das atividades de reuso e documentação de práticas adotadas no contexto de melhoria de processo de *software*, de ensino e aprendizado de projetos e da comunicação entre os membros de uma equipe de desenvolvimento de *software*.

5.3 Considerações Finais

Neste trabalho foi desenvolvida a ferramenta **DocRationale** que captura, armazena e recupera DR de artefatos de *software* na Web, dando ênfase ao fator de colaboração e as anotações para conseguir seus objetivos.

A ferramenta **DocRationale** ainda apresenta restrições que devem ser tratadas por meio dos trabalhos futuros, melhorando o auxílio aos desenvolvedores de *software* na tarefa de captura e recuperação de DR. Mesmo estando em sua primeira versão, a ferramenta já propiciou um auxílio de interesse aos desenvolvedores, demonstrado pela sua adoção em um curso de graduação, conforme mencionado. Avaliação de usabilidade e testes mais

abrangentes também serão alvo subjacente às pesquisas futuras que continuarão e evoluirão a ferramenta.

Finalmente, o próprio desenvolvimento da ferramenta **DocRationale** foi um exercício prático e de vivência da real necessidade de captura, armazenamento e recuperação do DR dos seus artefatos construídos. Espera-se que, a partir desta dissertação, pontos de discussão e de análise descritos sobre a DocRationale possam incorporar seu DR a ser registrado na própria ferramenta, para sua evolução.

Referências Bibliográficas

ARRUDA JR., C. R. E. Extensão do espaço de colaboração da ferramenta CoTeia. Projeto de Graduação II, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2001a.

ARRUDA JR., C. R. E. Extensão e integração de uma ferramenta colaborativa de edição tipo CoWeb. Projeto de Graduação I, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2001b.

ARRUDA JR., C. R. E. *Context Kernel: um web service para as dimensões de informação de contexto*. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2003.

ARRUDA JR., C. R. E.; PIMENTEL, M. G. C. Projeto e implementação de um sistema colaborativo de edição. *Revista Eletrônica de Iniciação Científica - REIC*, v. 1, n. 2, 2001.

ARRUDA JR., C. R. E.; PIMENTEL, M. G. C.; IZEKI, C. A. CoTeia: Uma ferramenta colaborativa de edição baseada na web. In: *Workshop de Ferramentas e Aplicações do VIII SBMIDIA*, Fortaleza - CE, 2002, p. 371-375.

BACELO, A.; BECKER, K. Uma ferramenta de apoio à discussão e deliberação em grupo. In: *Anais do III Workshop em Sistemas Multimídia e Hiperemídia*, São Carlos - SP, 1996, p. 119-130.

BOMBANI, L.; CIANCARINI, P.; VITALI, F. Sophisticated hypertext functionalities for software engineering. In: *Proceedings of the 22nd International Conference on Software Engineering*, Irlanda, 2000, p. 67-79.

- BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. *UML - guia do usuário*. Editora Campus, 2000.
- BURGE, J. *Design rationale types and tools*. 1998. Disponível em: <<http://www.cs.wpi.edu/Research/aidg/DR-Rpt98.html>>. Acesso em: 15/01/2004.
- BURGE, J.; BROWN, D. C. Reasoning with design rationale. In: *Proceedings of 6th International Conference on Artificial Intelligence in Design*, 2000.
- CONKLIN, J.; BEGEMAN, M. L. gIBIS: A hypertext tool for exploratory policy discussion. *ACM Transactions on Office Information Systems*, v. 6, n. 4, p. 303-331, 1988.
- CYBULSKI, J. L.; REED, K. A hypertext based software engineering environment. *IEEE Software*, v. 9, n. 2, p. 62-68, 1992.
- FISCHER, G.; MCCALL, R.; MORCH, A. I. JANUS: Integrating hypertext with a knowledge-based design environment. In: *Proceedings of Hypertext*, Pittsburgh - PA, 1989, p. 105-117.
- FRANCISCO, S. D.; IZEKI, C. A.; PAIVA, D. M. B.; FORTES, R. P. M. DocRationale: Um auxílio a design rationale na documentação de software. In: *X Sessão de Ferramentas, XVII. Simpósio Brasileiro de Engenharia de Software (SBES2003)*, Manaus, Brasil, 2003a, p. 61-66.
- FRANCISCO, S. D.; IZEKI, C. A.; PAIVA, D. M. B.; FORTES, R. P. M. Um sistema de apoio à utilização de design rationale de artefatos de software. In: *XXIX Conferencia Latinoamericana de Informática (Clei 2003)*, La Paz, Bolívia, em CD-ROM, 2003b, p. 96.
- FUMAGALLI, L. C. Investigação sobre recuperação de design rationale de documentos de software. Qualificação de Mestrado, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2003.
- GRUBER, T. R.; RUSSELL, D. M. *Design knowledge and design rationale: A framework for representation, capture and use*. Relatório Técnico, Knowledge Systems Laboratory, 1991.
- GUZDIAL, M. Supporting learners as users. *The Journal of Computer Documentation*, v. 23, n. 2, p. 3-13, 1999.
- HAMMOND, T.; GAJOS, K.; DAVIS, R.; SHROBE, H. An agent-based system for capturing and indexing software design meetings. In: *Proceedings of 2002 International Workshop on Agents in Design*, Cambridge, EUA, 2002.

- HAREL, D.; PNUELI, A.; SCHMIDT, J.; SHERMAN, R. On the formal semantics of statecharts. In: *Proceedings of 2nd. Symposium on Logic in Computer Science*, Ithaca - N.Y., 1987, p. 54-64.
- HU, X.; PANG, J.; PANG, Y.; ANTWOOD, M.; SUN, W.; C., W. A survey on design rationale: Representation, capture and retrieval. *Engineering with Computers: An Int'l Journal for Simulation-Based Engineering*, v. 16, p. 209-235, 2000.
- IZEKI, C. A. *Anotações colaborativas como hiperdocumentos de primeira classe*. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2001.
- JACOBSON, I.; BOOCH, G.; RUMBAUGH, J. *The unified software development process*. Addison Wesley, 1999.
- KATO, Y.; HORI, K.; TAKETA, K. Capturing design rationale by annotating e-mails. In: *Proceedings of the 6th World Multiconference on Systemics, Cybernetics and Informatics*, Orlando, EUA, 2002.
- LEE, J. SIBYL: A tool for managing design rationale. In: *Proceedings of the Conference on Computer Supported Cooperative Work*, Los Angeles, EUA, 1990, p. 79-92.
- LEE, J. Design rationale systems: Understanding the issues. *IEEE expert/Intelligent Systems and Their Applications*, v. 12, n. 3, p. 78-85, 1997.
- LU, S. C.-Y.; UDWADIA, F.; BURKETT, B.; CAI, J. *Design rationale for collaborative engineering design in the socio-technical framework*. 1999. Disponível em: <<http://impact.usc.edu/cerl/Task2/task2-2-3/DesignRationale.pdf>>. Acesso em: 15/01/2004.
- MACLEAN, A.; MCKERLIE, D. *Design space analysis and use-representations*. Relatório Técnico, Rank Xerox Research Centre, 1995.
- MACLEAN, A.; YOUNG, R.; BELLOTI, V.; MORAN, T. Question, options, and criteria: Elements of design space analysis. *Human-Computer Interaction*, v. 6, n. 3-4, p. 210-250, 1991.
- MAIDANTCHIK, C.; XEXÉO, G. B. *HyperDev: Hypertext tool to support object-oriented software development*. 1996. Disponível em: <http://www.cos.ufrj.br/~lodi/HyperDev/WWW/tableofcontents3_1.html>. Acesso em: 15/01/2004.

- MCCALL, R. Issue-serve systems: A descriptive theory of design. *Design Methods and Theories*, v. 20, n. 8, p. 443–458, 1986.
- MICHAELIS *Michaelis moderno dicionário da língua portuguesa*. 2002. Disponível em: <<http://www.uol.com.br/michaelis/>>. Acesso em: 08/04/2002.
- MONK, S.; SOMMERVILLE, I.; PENDARIES, J. M.; DURIN, B. Supporting design rationale for system evolution. In: *Proceedings of the Fifth European Software Engineering Conference*, Barcelona, Espanha, 1995, p. 307–323.
- MYERS, K. L.; ZUMEL, N. B.; GARCIA, P. Automated capture of rationale for the detailed design process. In: *Proceedings of Eleventh Conference on Innovative Applications of Artificial Intelligence*, Orlando, EUA, 1999, p. 876–883.
- PFLEGER, S. L. *Software engineering: Theory and practice*. 2ed. Prentice Hall, 2001.
- PHPLIB *PHPLib: PHP Base Library*. 2003. Disponível em: <<http://phplib.sourceforge.net/>>. Acesso em: 22/12/2003.
- PIMENTEL, M. G. C.; IZEKI, C. A.; ARRUDA JR, C. R. E. An XML-based Infrastructure Supporting Collaborative Annotations as First-class Hyperdocuments. In: *Proceedings of VII Brazilian Symposium of Multimedia and Hypermedia Systems*, Florianópolis - SC, 2001, p. 173–186.
- ROBBINS, J. E. *Design critiquing systems*. Relatório Técnico, Department of Information and Computer Science, University of California, 1998.
- ROBBINS, J. E.; REDMILES, D. F. Software architecture critics in the argo design environment. *Knowledge-based Systems*, v. 11, n. 1, p. 47–60, 1998.
- ROSSI, G. *Um método orientado a objetos para projeto de aplicações hipermídia*. Tese de Doutorado, Pontifícia Universidade Católica, Rio de Janeiro - RJ, 1996.
- RUS, I.; LINDVALL, M. Knowledge management in software engineering. *IEEE Software*, v. 19, n. 3, p. 26–38, 2002.
- SCHWABE, D.; ROSSI, G.; BARBOSA, S. D. J. Systematic hypermedia application design with oohdm. In: *Proceedings of the ACM Conference on Hypertext*, Washington – USA, 1996, p. 116–128.
- SERAPHIM, E.; DOMINGUES, A. L. S.; SIMÃO, A. S. Tutorial de MySQL. Notas Didáticas do ICMC, n. 60, Série Computação, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2003.

- SHIPMAN, F.; MCCALL, R. Integrating different perspectives on design rationale: Supporting the emergence of design rationale from design communication. *Artificial Intelligence in Engineering Design, Analysis, and Manufacturing*, v. 11, n. 2, p. 141–154, 1997.
- SIUM, S. *A cognitive analysis of design rationale representation*. Tese de Doutorado, University of York, 1991.
- SILVA, M. A. G. Uma investigação sobre engenharia de requisitos com apoio de hiperdocumentos. Qualificação de Mestrado, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2003.
- SILVA, W. F. *Sistema hipermídia de apoio à documentação de software: Uma investigação sobre a 'mudança de domínio de software'*. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2000.
- SIMÃO, A. S.; SERAPHIM, E.; DOMINGUES, A. L. S. Tutorial de PHP e PHPLib. Notas Didáticas do ICMC, n. 64, Série Computação, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, São Carlos - SP, 2003.
- SOUZA, C. R. B.; WAINER, J.; SANTOS, D. B.; DIAS, K. L. A model and tool for semi-automatic recording of design rationale in software diagrams. In: *Proceedings of the 6th String Processing and Information Retrieval Symposium & 5th International Workshop on Groupware*, Cancun, Mexico, 1998, p. 306–313.
- WELLING, L.; THOMSON, L. *PHP e MySQL: desenvolvimento Web*. Campus, 2001.
- WIEGERAAD, S. *Development of a design history information system*. 1999. Disponível em: <<http://alexandria.tue.nl/extra3/proefschrift/boeken/9902517.pdf>>. Acesso em: 15/01/2004.

Modelagem da ferramenta **DocRationale**

Neste apêndice são apresentados o diagrama de caso de uso da UML e os esquemas conceitual, navegacional e de contexto navegacional, ADVs e ADVcharts da metodologia OOHDm que fazem parte da modelagem da ferramenta **DocRationale**.

A.1 Diagrama de casos de uso

A Figura A.1 mostra o diagrama de casos de uso da **DocRationale**. Nessa figura, observa-se que o Administrador de Usuários é responsável pela inserção, remoção, edição e consulta de usuários da ferramenta. O Visitante pode apenas consultar as informações sobre os projetos, suas equipes, seus artefatos e questões. O Membro da Equipe é capaz de fazer as mesmas operações que o Visitante, além de inserir, remover, editar os artefatos e as questões. O Administrador de Projetos tem as mesmas opções disponíveis para os Membros, além de inserir, remover, editar os projetos e suas equipes.

Figura A.1: Diagrama de casos de uso da ferramenta **DocRationale**

A.2 Esquemas, ADVs e ADVcharts

A seguir, são apresentados os produtos da metodologia OOHDHDM elaborados para a **DocRationale**: esquema conceitual, esquema navegacional, esquema de contexto navegacional, ADVs e ADVcharts.

A.2.1 Esquema Conceitual

A Figura A.2 exibe o esquema conceitual da ferramenta **DocRationale**.

Como a ferramenta tem por objetivo capturar o DR produzido durante o desenvolvimento de um projeto a cada artefato gerado, para cada projeto inserido na **DocRationale**, uma equipe é formada para o desenvolvimento do mesmo, além do estabelecimento das possíveis

Figura A.2: Esquema Conceitual da ferramenta DocRationale

fases e atividades realizadas durante esse desenvolvimento. Informações a respeito do projeto são inseridas em uma *swiki* e constantemente atualizadas. A equipe do projeto é composta por: (i) usuários, já cadastrados na ferramenta e organizados como membros de uma equipe; (ii) um administrador do projeto; (iii) e possíveis visitantes.

Os participantes efetivos do projeto são os dois primeiros, sendo que o terceiro apenas visualiza o desenvolvimento. Durante o desenvolvimento de um projeto, diversos artefatos podem ser gerados, cada qual relacionado a uma fase e uma atividade pré-determinadas do projeto, além de ter a inserção de informações a seu respeito em uma página da CoTeia relacionada. Para um artefato, várias questões podem ser discutidas antes de sua elaboração. O foco principal está justamente nas questões geradas durante a criação de cada artefato. Além das questões, suas possíveis soluções (posições) e argumentos a favor e contra também são importantes e fazem parte do DR a ser capturado, na forma de anotação.

A.2.2 Esquemas Navegacional e Contexto Navegacional

Nas Figuras A.3 e A.4 podem ser observados os esquemas navegacionais da ferramenta **DocRationale**.

Figura A.3: Esquema Navegacional da ferramenta **DocRationale** – visão do administrador de projetos, do membro e do visitante

Figura A.4: Esquema Navegacional da ferramenta **DocRationale** – visão do administrador de usuários

O primeiro esquema navegacional mostra a visão de três tipos de usuários, que embora tenham restrições nas ações, a visibilidade é igual. Ou seja, embora o visitante não possa inserir/alterar/editar artefatos/projetos/questões, ele pode visualizá-los e também fazer busca pelos mesmos. O membro não pode inserir/alterar/editar projetos, mas pode, assim como o visitante, visualizá-los e buscá-los, além de poder inserir/alterar/editar artefatos e questões. O administrador de projeto é o usuário que pode inserir/alterar/editar artefatos/projetos/questões, além de poder navegar sobre eles e fazer busca.

O segundo esquema navegacional exibe a visão do usuário que é administrador de usuários. Esse usuário faz ações apenas sobre a classe *Usuário*, pois é responsável pela inserção/alteração/edição/exclusão de usuários dos tipos citados anteriormente. Também é responsável pela aceitação ou rejeição da solicitação de acesso a **DocRationale** de usuários que não tenham sido cadastrados por ele.

Na Figura A.5 pode ser observado o esquema de contexto navegacional da ferramenta **DocRationale**.

Pelo Esquema de Contextos de Navegação pode-se observar que após a devida validação (verificação se o usuário tem ou não autorização de acesso à ferramenta), ele tem acesso às informações contidas na ferramenta. Se o usuário for um administrador de usuários, seu acesso é direto à funcionalidade que gerencia os usuários que podem acessar e que tipo de acesso têm na ferramenta. Ou seja, o administrador de usuários pode visualizar, inserir, localizar, editar, excluir usuários ou atender à solicitação de acesso a ferramenta.

Se o usuário for um administrador de projetos, membro ou visitante, o acesso é iniciado pela lista de projetos e a partir da escolha de um projeto tem-se acesso à lista de artefatos desse projeto. A partir da escolha de um artefato, tem-se acesso à lista de questões. Como já dito anteriormente, o que difere a visão desses usuários são as ações que os mesmos podem executar. As ações de inserção/edição/exclusão de projetos são possíveis pela equipe de desenvolvimento, as de inserção/exclusão de fases e atividades são exclusivas do administrador de projetos. Já as ações de inserção/edição/exclusão de artefatos e inserção de questões são possíveis tanto para o administrador de projetos quanto para um membro da equipe de projeto. O visitante apenas navega sobre as informações, podendo também fazer pesquisa (busca).

A.2.3 ADVs

As Figuras A.6, A.7, A.8, A.9, A.10, A.11, A.12, A.13 e A.14 apresentam os ADVs que proporcionam uma visão das telas da ferramenta **DocRationale**.

A Figura A.6 mostra o ADV de mais alto nível da ferramenta **DocRationale** representando a estrutura geral dos ADVs. Essa figura apresenta a tela básica da **DocRationale**

Figura A.5: Esquema de Contextos de Navegação da ferramenta DocRationale

composta pelo Menu Principal, Área de Mensagens, Área de Status e por uma área com conteúdo alternado. O Menu Principal possui as opções disponíveis a cada instante da navegação de acordo com o conteúdo alternado. A Área de Mensagens apresenta informações a respeito das operações recém efetuadas pela ferramenta ou erros cometidos pelo usuário, como, “*Questão Incluída!*”, “*Cadastro do Artefato Alterado!*”, “*A data prevista do término não é válida!! Tente novamente!*”. A Área de Status mostra ao usuário o seu *username* e permissão de sistema, além das informações obtidas ao longo da navegação, como o projeto em que está navegando, juntamente com sua permissão de projeto, e até mesmo o artefato explorado no momento. A Área com conteúdo alternado pode exibir informações variadas de acordo com as opções do usuário. Nessa área é possível ver informações sobre Projetos, Artefatos, Questões, Fases, Atividades e Equipe, além de se ter o Login, Logout e Alteração de Senha. Por ser uma área mais complexa, as informações possíveis de tal área são mais detalhadas nos próximos ADVs.

Figura A.6: ADV da ferramenta DocRationale

A Figura A.7 apresenta os ADVs presentes na área com conteúdo alternado da Figura A.6. O ADV Login da Figura A.7(a) é composto por uma área de formulário, onde os usuários devem digitar seu *username* e senha, por um botão para efetuar o *login* - *Fazer Login* - e pelos links *Esqueceu a Senha* e *Registro* onde o usuário poderá solicitar outra senha, caso tenha esquecido a sua, ou fazer seu registro de solicitação de uma conta na **DocRationale**, respectivamente. Os ADVs das Figuras A.7(b), A.7(c) e A.7(d) mostram as possíveis ações que cada um dos objetos tratados pela **DocRationale** têm disponível. O ADV Projetos/Artefatos/Usuários/Equipe, mostrado na Figura A.7(b), tem o maior número

de opções; é possível: Listar, Inserir, Localizar, Editar e Excluir. O ADV Questões é mais restrito quanto ao número de opções; somente é possível Listar, Inserir e Localizar questões. Não seria interessante ter as opções de Editar e Excluir, visto que as questões são DR, e quer-se guardar todo o DR mesmo sendo uma alternativa discutida e descartada posteriormente na solução adotada. O ADV Fases e Atividades (Figura A.7(d)) também possui menos opções: Listar, Inserir e Excluir. Não foi estipulado Editar, pois por tratar-se apenas de um título, tanto para a fase quanto para a atividade, o trabalho praticamente seria o mesmo de excluir e inserir outro novamente. Também não foi disponibilizada a opção Localizar por se tratar apenas de títulos, sem informações adicionais. O ADV Alteração de Senha (Figura A.7(e)) mostra apenas uma opção, a disponibilidade de edição da senha. O ADV Logout, Figura A.7(f), mostra uma área de texto que descreve a finalização do uso da ferramenta, além de um botão (*Login*) fornecendo a possibilidade ao usuário de retornar ao uso da ferramenta.

Figura A.7: ADVs encontrados no ADV da ferramenta **DocRationale**

As Figuras A.8 e A.9 apresentam os ADVs Listar da Figura A.7 compostos por vários tipos de ADVs, como textos vindo de classes e *links*. Como cada Listar tem suas características específicas, os mesmos estão separados. O ADV Listar Projetos, Figura A.8(a),

mostra informações sobre os projetos, além de permitir a edição ou exclusão de determinado projeto, a visualização da lista de artefatos relacionados e da página CoTeia correspondente ao projeto, a possibilidade de inserção de um novo projeto e a busca (localização) de projetos. O ADV Listar Artefatos apresentado na Figura A.8(b) segue o ADV da Figura A.8(a). Mostra a lista de artefatos de um determinado projeto, fornece a possibilidade de edição e exclusão dos mesmos, possibilita a visualização da lista de questões relacionadas e da página da CoTeia correspondente ao artefato e disponibiliza a opção de inserção de um novo artefato e a localização de artefatos. Além dessas opções, é possível visualizar a lista de projetos, a lista de componentes da equipe e informações do projeto proprietário dos artefatos. O ADV Listar Questões da Figura A.8(c) também mostra a lista de questões de um determinado artefato que pertence a um projeto, disponibiliza a visualização da página do GroupNote, a inserção de uma nova questão, a localização de questões e, a possibilidade de finalizar (fechar a discussão de) uma determinada questão. Outras opções desse ADV são a listagem dos projetos, artefatos ou informações sobre o artefato ou o projeto proprietários das questões. O ADV Listar Equipe (Figura A.9(a)) lista os integrantes da equipe de desenvolvimento do projeto, fornecendo a possibilidade de edição, exclusão, localização desses e inserção de um novo integrante. É possível ver a lista de projetos, de artefatos ou informações sobre o projeto proprietário da equipe. Os ADVs de Listar Fase, na Figura A.8(d), e Listar Atividade, na Figura A.8(e), são semelhantes. Ambos possuem a listagem das fases e atividades, respectivamente, além da possibilidade de exclusão de tais entidades e inserção de novos. Tanto o ADV Listar Usuários – Figura A.9(b) quanto o ADV Listar Candidatos – Figura A.9(c) disponibilizam a listagem dos usuários, só que a listagem do primeiro mostra os usuários já ativos e, do segundo mostra os usuários interessados em ter acesso à ferramenta. Ambos têm a opção de exclusão disponível e listagem dos usuários na situação inversa (os candidatos vêm os ativos e vice-versa). A diferença está na possibilidade de editar os usuários no ADV Listar Usnários enquanto que no ADV Listar Candidatos é possível autorizar os usuários candidatos a se tornarem usuários ativos. O ADV Usuários tem duas outras opções que são inexistentes no ADV Candidatos, a de inserção de um novo usuário e a de localização de usuários da ferramenta.

A Figura A.10 apresenta os ADVs de mais baixo nível da ferramenta **DocRationale**. Neles encontram-se as operações de inserção, edição, localização (busca) e exclusão. A Figura A.10(a) mostra o ADV Inserir/Editar/Localizar composto por um formulário, onde o usuário deverá preencher as informações solicitadas conforme a opção, e pelos botões *Confirmar*, para efetuar a operação, e *Cancelar*, para desistir da operação.

Figura A.8: ADVs de mais baixo nível da ferramenta **DocRationale** – Listar I

A Figura A.11 mostra os ADVs Autorizar/Esqueceu a Senha e Registro. Quando for necessário autorizar usuários candidatos a usuários ativos, ou mesmo quando o usuário ativo esquecer a sua senha, é mostrada uma tela como a da Figura A.11(a). Essa Figura possui um ADV *E* listar, pois contém os mesmos componentes que o último. Já a Figura A.11(b) mostra uma tela requisitada quando uma pessoa deseja solicitar seu cadastro na **DocRationale**, preenchendo um registro e tornando-se um usuário candidato, à espera da autorização para

Figura A.9: ADVs de mais baixo nível da ferramenta **DocRationale** – Listar II

ter acesso à ferramenta. Como pode ser observado, o ADV Registro possui o ADV Inserir também por possuir os mesmos componentes.

A Figura A.12 mostra os ADVs referentes a um projeto e a um artefato. No ADV Projeto da Figura A.12(a) tem-se as informações a respeito do projeto, além da possibilidade de visualização de sua página na CoTeia e da listagem dos projetos, artefatos e membros da equipe. Também é possível editar o projeto e finalizá-lo. No ADV Artefato da Figura A.12(b) encontram-se informações sobre o artefato e também a disponibilização da visualização de sua página na CoTeia e de informações sobre o projeto proprietário do artefato, além da listagem dos artefatos e questões e a edição e finalização do artefato.

Figura A.10: ADVs Inserir/Editar/Localizar e Excluir

Figura A.11: ADVs Autorizar/Esqueceu a senha e Registro

Figura A.12: ADVs Projeto e Artefato

A Figura A.13 mostra o ADV Finalizar. Tal ADV mostra uma mensagem a respeito da finalização do projeto ou artefato.

Na Figura A.14 podem ser vistos os ADVs relacionados ao acesso às informações contidas na CoTeia e no GroupNote. A Figura A.14(a) mostra informações contidas na CoTeia

Figura A.13: ADV Finalizar

páginas da CoTeia – apresentadas pelo ADV CoTeia. Tais páginas podem ser tanto de projetos quanto de artefatos. Já a Figura A.14(b) mostra informações contidas no GroupNote – anotações do GroupNote – apresentadas pelo ADV GroupNote. Tais anotações estão diretamente relacionadas com as questões registradas na **DocRationale**.

(a) ADV
Página da
CoTeia

(b) ADV
Página do
GroupNote

Figura A.14: ADVs Páginas CoTeia e GroupNote

A.2.4 ADVcharts

As Figuras A.15, A.16, A.17, A.18, A.19, A.20, A.21, A.22, A.23, A.24, A.25, A.26, A.27 e A.28 exibem os ADVcharts que mostram as interfaces da ferramenta **DocRationale**.

A Figura A.15 mostra o ADVchart Login. Pela Figura pode-se observar que após a transição do estado Exibir Desligado para Exibir Ligado, tem-se a exibição de um formulário que deverá ser preenchido pelo usuário, além do botão Conecta e dos *links* Esqueceu Senha e Registro. Quando é pressionado o botão Conecta, os dados do formulário são autenticados e, caso estejam corretos, a tela volta a ser mostrada no modo Exibir Desligado e a navegação segue pela ferramenta. Porém, se a autenticação não for estabelecida, o formulário volta a ser exibido. Se o usuário esqueceu sua senha, o usuário pode solicitar outra através do *link* Esqueceu Senha ou, se a pessoa ainda não é usuária da ferramenta, pode requisitar cadastro através do *link* Registro. Os dois *links* fazem a tela voltar para o modo Exibir desligado, indo para suas respectivas telas.

Figura A.15: ADVchart Login

Na Figura A.16 pode ser visto o ADVchart Logout. Esse ADV tem um comportamento bem simples, pois quando em modo Exibir Ligado, mostra uma mensagem estática e um *link* que leva o usuário para a tela de login, e coloca o ADVchart em modo Exibir desligado.

Figura A.16: ADVchart Logout

A função listar possui um ADVchart bastante semelhante, mas com algumas particularidades para Projetos, Artefatos, Questões, Fase/Atividade, Usuários e Candidatos. Na verdade, as diferenças estão no direcionamento para outras telas. Dessa forma, será descrito por completo apenas um ADVchart, o ADVchart Projetos. Dos outros ADVcharts Listar são comentados apenas os pontos que os diferem.

O ADVchart Listar Projetos, Figura A.17, mostra que assim que a tela entra no modo Exibir Ligado são mostradas a lista de projetos disponíveis juntamente com um *link* para sua corresp. ndente página na CoTeia, as possibilidades de inserir, editar ou excluir um projeto

e também poder localizar projetos específicos. Além disso, após usar a função localizar, é possível voltar a visualizar todos os projetos através da função listar. Para a exclusão ou edição de um projeto, é necessário que tal projeto seja selecionado através de um botão tipo *radio* e então o botão da respectiva ação seja pressionado.

Figura A.17: ADVchart Listar Projetos

O ADVchart Listar Artefatos, Figura A.18, lista os artefatos de um determinado projeto, previamente escolhido pelo usuário e tem como diferencial, em relação ao ADVchart Listar Projetos, três botões que fornecem as opções de visualização da lista de projetos, de informações do projeto proprietário e da lista dos membros da equipe do projeto.

O ADVchart Listar Questões, Figura A.19, lista as questões de um certo artefato, selecionado pelo usuário. A diferença nesse ADVchart está no fato de não conter as opções de edição e exclusão, e seu *link* ser para uma página do GroupNote e não da CoTeia. Possui a opção de finalização da questão, quando chegou-se a conclusão da mesma, além das possibilidades de ver a lista de projetos, lista de artefatos e também informações do projeto e artefato proprietário das questões.

O ADVchart Listar Equipe, Figura A.20, lista os membros da equipe e possui as mesmas opções do ADVchart Listar Projetos, exceto pela falta do *link* para a página CoTeia e pelo acréscimo de outras três opções, de ver a listagem dos projetos, dos artefatos ou informações do projeto proprietário.

O ADVchart Listar Fases/Atividades, Figura A.21, lista as fases/atividades e, em relação ao ADVchart Projetos, não contém o *link* para a CoTeia nem a opção de localizar e editar.

Figura A.18: ADVchart Listar Artefatos

Entretanto, possui as opções de visualizar a lista de projetos, da equipe, informações do projeto proprietário e a opção de sair da tela, voltando a tela invocadora.

O ADVchart Listar Usuários, Figura A.22, lista os usuários da ferramenta; possui as mesmas opções do ADVchart Projetos, exceto pelo *link* para a CoTeia, além de acrescentar a possibilidade de listagem dos candidatos a usuários da ferramenta.

O ADVchart Listar Candidatos, Figura A.23, lista os candidatos a usuários da ferramenta. Não tem as opções ligação com a CoTeia, localização nem inserção como no ADVchart Listar Projetos, mas possibilita ver a listagem dos usuários.

O ADVchart Inserir/Editar/Localizar, Figura A.24, tem seu comportamento mais simples do que o dos ADVcharts Listar, mas um pouco mais elaborado que os demais ADVcharts. Quando no modo Exibir Ligado, é possui ver na tela um formulário e dois botões, Confirma e Cancela. Se o botão Cancela é acionado, a tela volta para o modo Exibir Desligado. Entretanto, se o botão acionado for o Confirma, é feita uma validação dos dados contidos no formulário, que se não estiverem corretos, volta a mostrar o formulário. Caso o formulário tenha sido preenchido corretamente, é feita a gravação dos dados do formulário na base

Figura A.19: ADVchart Listar Questões

de dados e a tela volta para o modo Exibir Desligado. Contudo, se ocorrer algum erro de gravação, o formulário volta a ser mostrado como no caso de algum erro de preenchimento.

Na Figura A.25 pode ser observado o ADVchart Excluir com os mesmos itens iniciais que o ADVchart Inserir/Editar/Localizar. A diferença entre eles está no comportamento quando o botão Confirma é acionado. É feita a exclusão e a tela volta ao modo Exibir Desligado se tudo estiver correto, ou o formulário volta a ser exibido em caso de ocorrência de erro durante a operação de exclusão.

O ADVchart Finalizar, Figura A.26, é semelhante ao ADVchart Logout A.16, mostrando uma mensagem, quando no modo Exibir Ligado. Porém possui um botão no lugar de um *link*. Esse botão leva a tela ao modo Exibir desligado e volta a tela anterior, a tela que a chamou.

O ADVchart Projeto, Figura A.27, mostra informações sobre o projeto corrente, além de fornecer as opções de editar e finalizar o projeto, listar fases, atividades, projetos, artefatos e equipe. Todas as opções levam a tela para o modo Exibir Desligado, exceto a opção de finalizar o projeto que finaliza o projeto e mostra as informações atualizadas do projeto na tela.

Figura A.21: ADVchart Listar Fases/Atividades

Figura A.22: ADVchart Listar Usuários

Figura A.23: ADVchart Listar Candidatos

Figura A.24: ADVchart Inserir/Editar/Localizar

Figura A.25: ADVchart Excluir

Figura A.26: ADVchart Finalizar

Figura A.27: ADVchart Projeto

Figura A.28: ADVchart Artefato