Documentação de Atividades de Planejamento Ambiental Centrada em Bancos de Dados

Silvania Maria de Resende

Dissertação de Mestrado

Instituto de Computação
Universidade Estadual de Campinas

Documentação de Atividades de Planejamento Ambiental Centrada em Bancos de Dados

Silvania Maria de Resende[1]

Fevereiro de 2003

Banca Examinadora:

- Profa. Dra. Claudia Bauzer Medeiros
  Instituto de Computação, UNICAMP (Orientadora)
- Prof. Dr. Jansle Vieira Rocha
  Faculdade de Engenharia Agrícola - UNICAMP
- Prof. Dr. Célio Cardoso Guimarães
  Instituto de Computação - UNICAMP
- Prof. Dra. Ariadne M. B. Rizzoni Carvalho
  Instituto de Computação - UNICAMP (Suplente)

[1]**Apoio financeiro da FAPESP, processo 01/03893-9**

Documentação de Atividades de Planejamento Ambiental Centrada em Bancos de Dados

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por Silvania Maria de Resende e aprovada pela Banca Examinadora.

Campinas, 26 de fevereiro de 2003.

Profa. Dra. Claudia Bauzer Medeiros
Instituto de Computação, UNICAMP
(Orientadora)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Resende, Silvania Maria de

R311d Documentação de atividades de planejamento ambiental centrada em bancos de dados / Silvania Maria de Resende -- Campinas, [S.P. :s.n.], 2003.

Orientador : Claudia Bauzer Medeiros

Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Documentação. 2. Banco de dados. 3. Fluxo de trabalho 4. Sistemas de hipermídia. 5. Trabalho – Aspectos ambientais. I. Medeiros, Claudia Bauzer. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

TERMO DE APROVAÇÃO

Tese defendida e aprovada em 26 de fevereiro de 2003, pela Banca examinadora composta pelos Professores Doutores:

Prof. Dr. Jansle Vieira Rocha
FEAGRI - UNICAMP

Prof. Dr. Célio Cardoso Guimarães
IC - UNICAMP

Profa. Dra. Claudia Maria Bauzer Medeiros
IC – UNICAMP

Substitua pela folha com a assinatura da banca

À minha família, com carinho.

"Sempre que se produz um novo conhecimento também se inventa um novo peculiar caminho. Quando olhamos para trás é que nos damos conta disso."

Marisa Vorraber Costa

Resumo

O processo de planejamento ambiental é uma tarefa complexa que cobre aspectos variados, envolve uma série de etapas e é alimentado por vários conjuntos de dados. Normalmente exige a cooperação de equipes multidisciplinares que discutem várias alternativas de planejamento, considerando por exemplo questões referentes ao uso ou recuperação de recursos ambientais. Um dos grandes problemas desse processo é a falta de documentação associada. Como em qualquer atividade cooperativa, a documentação é importante para a revisão, manutenção e evolução do plano e para a comunicação entre os projetistas, dentre outros fatores.

O objetivo desta dissertação é resolver parcialmente este problema, através da especificação e implementação parcial de um ambiente para gerenciar, de forma unificada, três tipos de documentos: descrição do produto final do planejamento (documentos O QUE), descrição do processo usado para obter o produto final (documentos COMO) e descrição das razões que estão por trás das decisões para se chegar a este produto (documentos PORQUE). Estes documentos foram especificados visando armazenamento e gerenciamento em um banco de dados. Documentos O QUE são representados através de estruturas de hipermídia, documentos COMO através de *workflows* científicos e o PORQUE é baseado em estruturas de *design rationale*.

As principais contribuições desta pesquisa são: (a) especificação centrada em bancos de dados das estruturas dos documentos O QUE, COMO e PORQUE; (b) especificação do ambiente para gerenciá-los, visando facilitar trabalho cooperativo na área de planejamento ambiental; (c) implementação parcial deste ambiente.

Abstract

The environmental planning process is a complex task that covers various aspects, involving a series of steps and is fed by many data sources. Normally, this process demands the cooperation of multidisciplinary teams that discuss many planning alternatives. These alternatives consider, for instance, multiple issues on preservation or recovery of environmental resources. One of the main problems in this process is the incompleteness of associated documentation. As in any cooperative activity, documentation is important for revision, maintenance and evolution of the plan, and for communication among designers.

The goal of this dissertation is to partially solve this problem, through the specification and partial implementation of an environment to manage, in a unified way, three kinds of documents: description of the final product - the plan (WHAT documents), description of the process used to obtain the final product (HOW documents) and description of the reasons behind the decisions of planning (WHY documents). These documents were specified so as to allow them to be stored and managed in a database. WHAT documents are represented through hypermedia structures, HOW documents using scientific workflows, and WHY documents are based in design rationale structures.

The main contributions of this research are: (a) database-centered specification and design of the WHY, HOW and WHAT documents; (b) specification of an environment to support management of these documents, thus fostering cooperative work in environmental planning; (c) partial implementation of this environment.

Agradecimentos

Gostaria de agradecer a Deus por ter sempre iluminado meus passos possibilitando superar as dificuldades e alcançar mais essa vitória.

À minha família, sobretudo aos meus pais e irmã, que sempre me apoiaram e apesar da distância ofereceram o carinho e atenção necessários para superar a saudade.

À minha orientadora Claudia Bauzer Medeiros pela credibilidade e confiança e por ter compartilhado parte de sua vasta experiência tão importante para o meu crescimento profissional e pessoal.

Ao prof. Jansle Vieira Rocha, da FEAGRI, por ter se disposto prontamente a discutir e fornecer alguns exemplos de planejamento ambiental.

Ao prof. Juliano Lopes de Oliveira, da UFG, por ter ajudado a despertar o meu interesse pela pesquisa.

Aos amigos do grupo de banco de dados do IC pela disponibilidade em ajudar a solucionar várias dúvidas, pelas críticas e sugestões tão importantes para a melhoria deste trabalho. Em especial ao Henrique Rocha que contribuiu mais diretamente para que vários resultados se tornassem concretos e ao Thiago Borin Sicchieri pela ajuda na implementação.

À Thaisa, Henrique e Rodrigo, amigos desde a graduação e companheiros de república, pela convivência amiga.

A todos os amigos do IC pelos vários momentos de alegria e descontração que curtimos juntos. Tenho certeza que nossas festas ainda serão lembradas por muito tempo. Em especial às amigas Juliana e Amanda pela amizade, carinho e companheirismo.

Aos professores e funcionários do IC e todos aqueles que direta ou indiretamente contribuíram para a realização deste trabalho.

Este trabalho foi financiado pela FAPESP (processo 01/03893-9). Além disso, recebeu apoio parcial do CNPQ e do projeto SAI do PRONEX-MCT.

Sumário

Lista de Figuras

Capítulo 1

Introdução e Motivação

A importância da atividade de planejamento ambiental vem aumentando nos últimos anos. Inicialmente restrita a questões ecológicas e relativa a problemas em nível global, cobre hoje aspectos variados e pode ser encontrada em diferentes escalas, desde o nível de região urbana até o nível global.

Este tipo de atividade é caracterizada por vários fatores, tais como: **(1) Multidisciplinaridade**, exigindo esforços conjuntos de vários tipos de especialistas; **(2) Participação de equipes**, muitas vezes distribuídas geograficamente, trabalhando síncrona e assincronamente; **(3) Singularidade dos problemas** a serem resolvidos (cada problema é quase que uma situação inédita).

O processo de planejamento ambiental envolve uma série de etapas dentre as quais se destacam: determinação dos problemas a enfrentar em uma determinada área (o **"diagnóstico"**); definição de estratégias para resolvê-los ou minimizá-los a curto, médio e longo prazo (o **"plano"**); e implementação do plano (o **"acompanhamento"**). Este processo é alimentado por vários conjuntos de dados, tanto relativos ao meio físico quanto sócio-econômico e cultural e normalmente é apoiado por software especial para gerenciar tais dados.

O plano ambiental tem como principal resultado dois tipos de documentos:

- Um conjunto de mapas que detalham características da região estudada. Estes mapas descrevem tanto o estado atual da região quanto o estado desejado (alvo do planejamento);
- Um conjunto de diretivas que especificam como levar a cabo o processo de transformação de uma região do estado presente para o estado desejado (ou, alternativamente, como garantir que o estado atual seja mantido).

Em geral, existem várias alternativas de planejamento a ser discutidas pelas equipes. Questões referentes ao uso de recursos e manutenção ou recuperação de determinados

fatores ambientais devem ser consideradas. A partir das discussões são tomadas decisões e uma alternativa é escolhida para implementação. Na verdade, o planejamento ambiental é um processo contínuo, pois é preciso monitorar constantemente uma região, validando ou aperfeiçoando o plano (e consequentemente, refazendo os mapas e mudando as diretivas).

Do ponto de vista de geoprocessamento, um dos grandes problemas do processo de planejamento é a falta de documentação associada. Raramente existem documentos que descrevam como se chegou ao diagnóstico e à solução. Desta forma, ocorrendo um problema semelhante em outra região, é preciso recomeçar todas as análises do início. Além disso, fica difícil detectar erros de concepção ou de metodologia na elaboração do diagnóstico.

Outro aspecto associado a problemas de documentação é a falta de apoio de ferramentas que facilitem documentar o acompanhamento. De novo, este tipo de ferramenta facilitaria a tomada de decisões e a detecção de erros.

A documentação associada ao planejamento ambiental é importante para comunicação entre os projetistas, manutenção e evolução do plano. Quanto mais complexo for o processo, maior o número de pessoas e tecnologias envolvidas e mais altos serão os custos de manutenção e atualização. Consequentemente, maior a necessidade de documentação. O termo "documentação" deve cobrir um amplo espectro, incluindo dados e documentos gerados durante o processo de diagnóstico e construção do plano ambiental. Em particular, estes documentos devem retratar três pontos principais:

1. Dados que descrevem o produto final - o diagnóstico e mapas associados (por exemplo, manuais, metadados, texto) – aqui chamados de documentos O QUE;

2. Dados que descrevem o processo usado para obter o produto final (por exemplo, modelos matemáticos, procedimentos) – aqui chamados de documentos COMO;

3. Dados que descrevem as razões que estão por trás das decisões de planejamento e de acompanhamento – aqui chamados de documentos PORQUE.

Algumas soluções de geração e gerenciamento de documentos têm sido propostas para esses problemas. Estas soluções são entretanto aplicadas isoladamente, com cada tipo de documento gerenciado por um sistema separado e além disso estudado por um campo diferente da Ciência da Computação. A documentação sobre o O QUE é tratada na área de Banco de Dados ou Engenharia de Software, ao passo que documentos COMO são restritos a sistemas hipermídia e CSCW (*Computer Supported Cooperative Work*), e documentos PORQUE são manipulados no contexto de Inteligência Artificial e ciência cognitiva.

O objetivo desta dissertação é a especificação e desenvolvimento parcial de um ambiente para gerenciar, de forma unificada, os diferentes tipos de documentos gerados

durante atividades de planejamento ambiental. Os três tipos de documentos serão armazenados em um banco de dados resultando em uma documentação completa que possa ser facilmente manipulada. A idéia desta iniciativa é integrar e coordenar o trabalho (cooperativo) dos diversos especialistas envolvidos no planejamento ambiental: sociólogos, ecólogos, engenheiros, biólogos e outros. Isto elimina a quebra de continuidade encontrada nos ambientes normais de projeto ambiental, onde a documentação é praticamente inexistente. Quando existe, cada tipo de documento é manipulado separadamente e usa paradigmas de implementação distintos, complicando a comunicação entre os diversos usuários.

Como parte da integração dos documentos, é preciso definir um conjunto de estruturas para representá-los em um banco de dados. Este trabalho propõe a utilização de estruturas de hipermídia para representar documentos O QUE, *workflows* científicos para representar o COMO e *design rationale* para representar o PORQUE. Estes conceitos serão discutidos no capítulo 2. A opção de implementação adotada foi utilizar como ponto de partida um sistema desenvolvido no IC-UNICAMP, denominado WOODSS (*WOrkflOw-based spatial Decision Support System*) [SMRY99]. Este sistema, baseado na noção de *workflows* científicos, captura interações do usuário com um sistema de informação geográfica (SIG). Isto permite documentar os passos de um processo de modelagem de fenômenos usando o SIG. Para acomodar as novas necessidades de documentação propostas nesta dissertação, parte do WOODSS foi reprojetada e reimplementada. O novo projeto e implementação foram feitos em conjunto com Rocha [Roc03], por constituir também a base de implementação de seu trabalho.

As principais contribuições desta dissertação são:

- Especificação centrada em bancos de dados das estruturas dos documentos O QUE, COMO e PORQUE;
- Especificação do ambiente para gerenciá-los, visando facilitar trabalho cooperativo na área de planejamento ambiental. Esta especificação servirá igualmente de base para outros ambientes que visem documentação de trabalho cooperativo;
- Implementação parcial deste ambiente;
- Reengenharia do sistema WOODSS, tornando-o mais modular de modo a facilitar manutenções e extensões futuras.

O restante da dissertação está organizado da seguinte forma. O capítulo 2 apresenta a revisão bibliográfica necessária ao entendimento do trabalho. O capítulo 3 detalha os três tipos de documentos relativos à documentação de atividades de planejamento ambiental, especificando sua estrutura. O capítulo 4 discute aspectos de implementação do ambiente

de gerenciamento unificado de documentação, o qual é baseado no sistema WOODSS [SMRY99]. O capítulo 5 faz um estudo de caso utilizando o sistema proposto para a documentação de um problema de planejamento ambiental real. Finalmente, o capítulo 6 apresenta as conclusões e propõe algumas extensões para este trabalho.

Capítulo 2

Revisão Bibliográfica

Este capítulo apresenta a revisão bibliográfica feita durante o desenvolvimento desta dissertação. A seção 2.1 apresenta trabalhos correlatos ao gerenciamento de documentação em geral. A seção 2.3 apresenta alguns modelos de dados hipermídia. A seção 2.4 apresenta os principais modelos para representação de *design rationale* descritos na literatura e a seção 2.5 faz uma análise comparativa desses modelos. A seção 2.6 caracteriza a atividade de planejamento ambiental.

2.1 Trabalhos Correlatos

Trabalhos correlatos ao gerenciamento de documentação em geral, principalmente no que se refere a trabalho cooperativo, discutem isoladamente os três tipos de documentos mencionados (O QUE, COMO e PORQUE). A documentação sobre o O QUE é tratada na área de Banco de Dados ou Engenharia de Software, ao passo que documentos COMO são restritos a Sistemas de Hipermídia e CSCW, e documentos PORQUE são manipulados nos contextos de Inteligência Artificial e ciência cognitiva [VMJ00]. Em Engenharia de Software, por exemplo, trabalhos sobre documentação procuram sobretudo registrar o processo de evolução e semântica do software [CNF+00, CWL00], visando facilitar manutenções e evoluções futuras.

Um dos poucos trabalhos que trata do gerenciamento de documentação de forma unificada é o de Voisard *et. al.* [VMJ00], relativo ao ambiente WHOW. O WHOW é centrado em um banco de dados orientado a objetos e propõe o gerenciamento unificado de documentos dos tipos O QUE, COMO e PORQUE, relativos a artefatos de engenharia. Nessa proposta, documentos do tipo O QUE são representados por textos estáticos associados a cada objeto que constitui um artefato, documentos do tipo COMO são formados por *workflows*, e o *design rationale* de um artefato (PORQUE) é documentado através de grafos denominados WHY-GRAPH. Esse trabalho constituiu uma importante motivação

para a proposta desta dissertação. No entanto, trata-se de um trabalho ainda preliminar, que levanta vários problemas em aberto, e que teve origem em necessidades constatadas com usuários que trabalham de forma cooperativa em aplicações de engenharia.

Outro trabalho que trata do gerenciamento unificado de documentação é o de Peerbocus *et. al.* [PMJV01]. Nesse trabalho documentos O QUE, COMO e PORQUE são associados, de maneira seletiva, às mudanças ocorridas em um banco de dados espaço-temporal com o objetivo de fornecer um explicação mais precisa sobre a evolução de fenômenos geográficos. O artigo concentra-se no gerenciamento de documentação de mudanças no contexto de desenvolvimento de aplicações urbanas. São consideradas mudanças relativas a três níveis de abstração: eventos que ocorrem no mundo real; evolução cartográfica descrevendo as modificações feitas em mapas de diferentes versões; e evolução do banco de dados devido a atualização dos objetos. Cada tipo de documento é especificado através de uma classe. Os objetos de documentação são associados aos objetos espaciais correspondentes e armazenados no mesmo banco de dados, permitindo documentar as razões, os procedimentos e as origens das mudanças. No entanto, as razões, por exemplo, são armazenadas em um objeto PORQUE como um único atributo textual. O mesmo ocorre para a descrição do processo em um objeto COMO. Os propósitos de documentação apresentados nesta dissertação exigem uma especificação mais detalhada destes aspectos.

Quanto ao restante da bibliografia correlata, do ponto de vista de representação e gerenciamento de documentos, existem propostas relativas a: metadados; dados textuais, organizados em grafos de hipertexto/hipermídia; *workflows* e *design rationale*. Estes mecanismos são adequados à especificação da documentação de atividades de planejamento ambiental e serão descritos a seguir.

2.1.1 Metadados

Metadados podem ser definidos simplesmente como dados sobre dados. Uma definição mais completa é fornecida em [DH98]: metadado é um dado associado a objetos, facilitando a seus usuários ter mais conhecimento de sua existência ou características. Um usuário pode ser um programa ou uma pessoa.

Exemplos de metadados são: dados auto-descritivos tais como título, autor e resumo de um documento; cabeçalho de um e-mail; dados do catálogo de uma biblioteca.

Nesta dissertação o principal interesse é em metadados geográficos. Além dos aspectos descritivos ou convencionais que são comuns em diversas categorias de metadados, os metadados geográficos preocupam-se também com informações sobre localização geográfica, tempo de validade e qualidade dos dados.

A utilização de metadados geralmente implica na utilização de um padrão de metadados existente ou na criação de um novo padrão. Um padrão define um conjunto de

metadados a ser adotado na descrição de características dos objetos de um domínio. O uso de padrões de metadados facilita a obtenção, compartilhamento, integração e transferência de dados.

Um dos padrões de metadados mais utilizados em sistemas de informação geográfica é o *Content Standard for Digital Geospatial Metadata* (CSDGM) desenvolvido pelo *Federal Geographic Data Committee* (FGDC) [Fedb]. O CSDGM visa fornecer um conjunto comum de terminologias e definições para a documentação de dados geoespaciais [Feda]. O padrão estabelece os nomes e componentes dos elementos a serem utilizados, as definições destes elementos, e informações sobre os valores que serão fornecidos para os elementos.

A figura 2.1 ilustra os grupos de metadados do padrão CSDGM, versão 2 – 1998. Cada grupo é composto por elementos de dados. Por exemplo, o grupo "Informação de Identificação" contém os campos "Descrição"(descrição textual do conjunto de dados) e "Palavras-chave"(palavras ou frases que representam o conjunto de dados), dentre outros.

Uma discussão detalhada sobre outros padrões de metadados, incluindo a definição completa e exemplo de utilização de um padrão de metadados geográficos encontra-se em [Fag99].

2.1.2 Estruturas de Hipermídia

Hipermídia representa uma abordagem para gerenciamento de informações na qual os dados são armazenados em uma rede de nós conectados por ligações. Um nó geralmente representa um conceito ou uma idéia, podendo conter qualquer tipo de dado multimídia, como textos, gráficos, áudio, vídeo, imagem. As ligações representam relacionamentos entre nós e podem ser unidirecionais ou bidirecionais. O nó no qual uma ligação se origina é chamado origem e o nó no qual uma ligação finaliza é chamado destino. O conteúdo de um nó é apresentado pela ativação das ligações. Tanto nós como ligações podem possuir atributos descritivos [Gon97].

Estruturas de hipermídia são normalmente gerenciadas por sistemas hipermídia em que os principais mecanismos oferecidos são:

- Uma Interface gráfica de usuário, que ajuda os usuários a navegar através de uma grande quantidade de informações;
- Mecanismos de recuperação de informação tais como busca por palavras-chave e busca por autores;
- Uma máquina hipermídia que gerencia a informação sobre nós e ligações;
- Um sistema de armazenamento, como um sistema de arquivos ou um Sistema Gerenciador de Banco de Dados (SGBD).

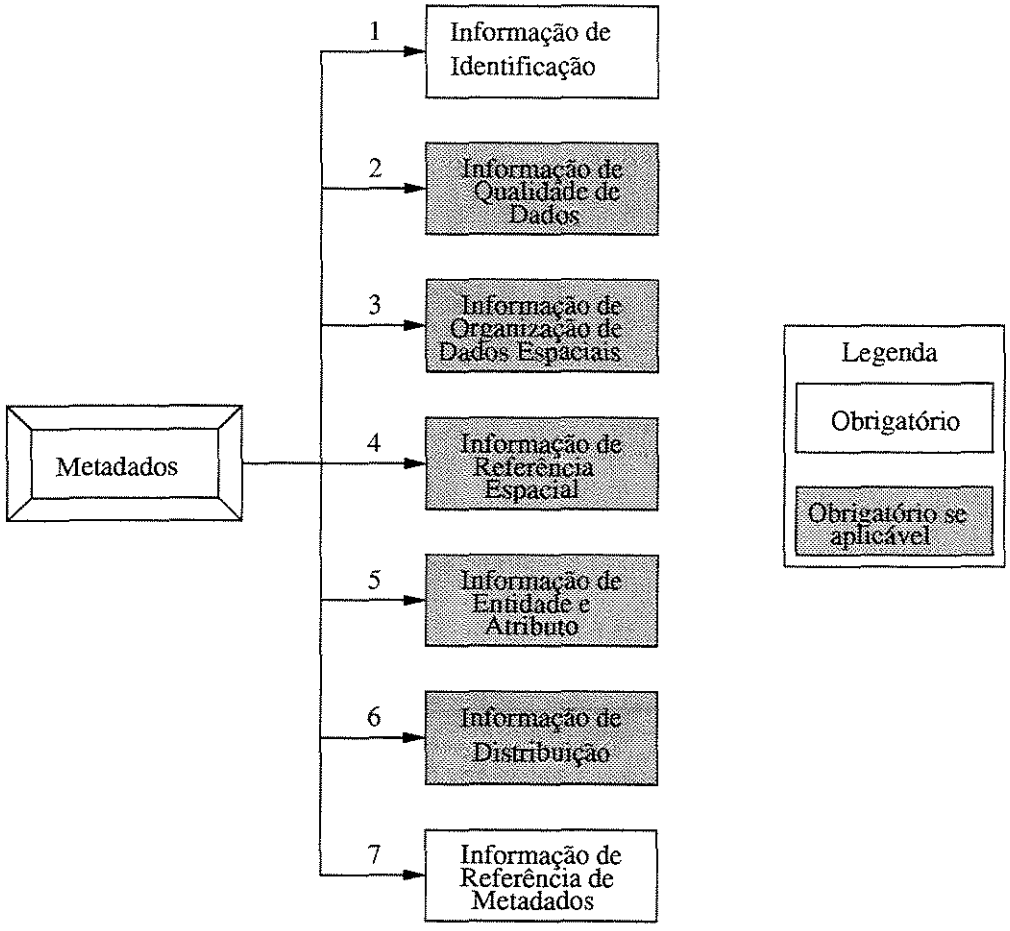


Figura 2.1: Grupos de metadados do padrão CSDGM, versão 2–1998

Hipermídia é uma forma de ligar dados, e portanto pode ser usada para representar documentos. Documentos em hipermídia são chamados hiperdocumentos. Ao contrário da maioria dos documentos tradicionais, que são essencialmente textuais e apresentam uma organização sequencial, hiperdocumentos são não lineares e podem ser acessados de diversas formas por meio de apresentações em janelas e cliques de mouse em *links* [Lau01]. Hiperdocumentos que suportam apenas conteúdo textual em seus nós são chamados de hipertexto.

A tecnologia hipermídia é geralmente utilizada em aplicações que gerenciam documentos dinâmicos tais como bibliotecas digitais [Gon97] ou na Web. Também pode ser utilizada em outros contextos, embora mais raramente, por exemplo, integração de ambientes heterogêneos de desenvolvimento de software [ATW00, FHS+92].

2.1.3 *Workflows*

Um *workflow* representa um tipo de documento dinâmico utilizado para registrar o processo de construção de um objeto. *Workflows* geralmente são empregados em atividades que envolvem a execução coordenada de múltiplas tarefas por diferentes entidades de processamento [RS95]. O processo é representado por conjuntos de passos interligados, onde cada passo constitui uma tarefa ou atividade e representa um trabalho a ser feito. Uma entidade de processamento ou agente executa tarefas, podendo ser uma pessoa ou um software. Através de um papel é possível descrever um determinado (tipo de) agente de acordo com um conjunto pré-estabelecido de habilidades ou conhecimento de contexto necessários à execução de uma tarefa [Bar96]. A especificação de um *workflow* envolve determinar as atividades que o constitui juntamente com suas interdependências, entradas e saídas; os agentes e seus papéis no *workflow* em questão; e os requisitos/restrições de execução de cada atividade.

Workflows geralmente são manipulados por SGWFs (Sistemas de Gerenciamento de *Workflow*). Um SGWF fornece facilidades para definir, criar e gerenciar a execução de *workflows* [Coa96]. Durante a fase de definição, um processo do mundo real é traduzido em uma especificação formal que pode ser processada por um computador. Em tempo de execução essa especificação é interpretada por um software responsável pela criação de instâncias operacionais do processo, escalonamento das atividades que compõem o processo e invocação dos recursos necessários para a execução de cada atividade [Hol95].

Tradicionalmente *workflows* têm sido usados para automação total ou parcial de processos de negócio, como meio de documentação e de ajuda à coordenação de grupos. Estes *workflows* são denominados *workflows* de negócios. Uma extensão dos *workflows* de negócios são os chamados *workflows* científicos [WWVM96]. Estes são especialmente definidos para documentar procedimentos e experimentos científicos. A documentação de trabalhos científicos requer um tratamento especial pois este tipo de trabalho é caracterizado por um grande grau de flexibilidade e incerteza e pela ocorrência de um grande número de exceções. Enquanto processos de negócios envolvem situações rotineiras e bem conhecidas, experimentos científicos frequentemente lidam com situações indeterminadas em que a sequência de tarefas que os especificam nem sempre é totalmente conhecida a priori. *Workflows* científicos estendem os *workflows* de negócios nos seguintes aspectos [WVM96, Kas01]:

- **Incompletude**: *workflows* científicos podem ser executados mesmo quando incompletos, sendo construídos progressivamente durante sua execução. *Workflows* de negócios, ao contrário, precisam ser totalmente especificados antes de serem executados;

- **Reutilização parcial**: *workflows* científicos são considerados blocos em construção

para especificação de experimentos. Assim, *workflows* parciais podem ser utilizados para estruturar novos *workflows*;

- **Modificação dinâmica**: é possível alterar dinamicamente a especificação inicial de um *workflow* científico com base em resultados obtidos durante sua execução, restabelecendo seu contexto e realizando uma reexecução, possivelmente tomando um novo curso de ação. *Workflows* de negócios, por outro lado, são estáticos e executados de maneira rotineira;

- **Retrocesso**: *workflows* científicos permitem não apenas reexecutar uma atividade mas também retroceder para uma atividade anterior e seguir um caminho de execução alternativo;

- **Execução de processos inválidos**: no domínio científico, os processos decisórios são baseados no mecanismo de tentativa e erro. Assim, ao contrário de *workflows* de negócios, *workflows* científicos servem como meio de documentação de processos bem sucedidos e mal sucedidos. A documentação de processos mal sucedidos é importante para evitar a reincidência dos erros;

- **Especificação a partir da instância**: *workflows* de negócios são especificados para serem executados frequentemente, onde cada execução é uma instância. *Workflows* científicos, por outro lado, podem ser executados uma única vez, como em tentativas sem sucesso. Uma vez que um *workflow* científico pode ser definido dinamicamente, sua especificação é realizada a partir da instância, ao invés da especificação definir a instância, como em *workflows* de negócios.

A principal motivação para o uso de *workflows* em aplicações de negócios é executar trabalhos repetitivos de forma mais eficiente. Em aplicações científicas, a motivação para o uso de *workflows* é controlar experimentos e documentar como eles foram realizados. Estes documentos podem constituir uma importante referência para a realização de novos experimentos similares.

Como atividades de planejamento ambiental constituem um trabalho científico, esta dissertação utiliza o paradigma de *workflows* científicos para documentar COMO estas atividades são realizadas. Um exemplo de iniciativa neste sentido é o sistema WOODSS [SMRY99], desenvolvido no IC-UNICAMP. O WOODSS utiliza *workflows* científicos para monitorar e documentar o processo de geração de mapas em um SIG. O WOODSS constituiu o ponto de partida para a definição e implementação dos documentos do tipo COMO. No entanto, como a representação de *workflow* utilizada pelo WOODSS era bastante simples, ela foi complementada. A nova representação será descrita no capítulo 3.

2.2 *Design Rationale*

Ao longo de um projeto, geralmente existem várias alternativas que podem ser adotadas. Cada alternativa apresenta vantagens e desvantagens. Os projetistas precisam analisar cada opção e escolher a mais adequada, tendo em vista os objetivos a serem atingidos.

Do ponto de vista tradicional, a documentação de um projeto consiste em uma descrição do projeto final [BB00], ou seja, o processo de raciocínio utilizado ao longo do projeto é perdido. O *design rationale* (DR) complementa esta abordagem permitindo que as razões por trás das decisões tomadas em um projeto sejam representadas de maneira estruturada. Ele registra as questões, alternativas e justificativas que foram relevantes para os elementos do projeto [PT98].

O DR representa perspectivas de argumentação, documentação e comunicação [BB00]. A argumentação e documentação são representadas pela decisões de projeto e pelas razões por trás delas. A diferença é que o objetivo da documentação é fornecer uma explicação para pessoas que não participaram do projeto, enquanto que a argumentação tem também o objetivo de estruturar a abordagem do problema utilizada pelo projetista. A representação estruturada das decisões facilita a comunicação entre grupos de projetistas que podem estar distribuídos geograficamente.

Como apontado em [MYBM96], é importante documentar o *rationale* porque um objeto precisa ser entendido por uma grande variedade de pessoas que têm que lidar com ele - projetistas, usuários alvo, instrutores, e pessoal de manutenção. Para estas pessoas, não é apenas o objeto que é importante, mas também outras questões - como mudá-lo, como lançá-lo no mercado, e assim por diante, pois há muitas formas de trabalhar com um objeto.

Para esta dissertação são especialmente interessantes dois dentre os vários tipos de DR definidos em [MC96b]: (i) descrição dos relacionamentos entre um objeto sendo projetado, seu propósito, as restrições contextuais em realizar o propósito e a conceitualização do projetista; e (ii) a documentação das razões para o projeto em si, os estágios ou passos do processo de projeto, e a história do projeto e seu contexto.

O DR vem sendo pesquisado e utilizado principalmente nas áreas de Inteligência Artificial [PT98, Pol98, BB00], Engenharia de Software [MSPD95, PV96, HE00] e Interfaces Homem-Computador [AYM89].

Um exemplo de uso do DR em Inteligência Artificial é descrito em [BB00] que apresenta o InfoRat, um sistema que faz inferências sobre o DR a fim de detectar inconsistências e completude, e ainda verificar o impacto de mudanças em um projeto. O InfoRat faz dois tipos de inferência: sintática e semântica. A inferência sintática procura por inconsistências tais como requisitos cujas metas não foram representadas e metas ou sub-metas sem alternativas selecionadas. A inferência semântica observa os argumentos contra e a

favor das alternativas. As discrepâncias procuradas são: alternativas selecionadas cujos argumentos contra têm peso maior que os argumentos a favor; alternativas selecionadas mas que não representam a melhor escolha; e alternativas selecionadas que possuem argumentos contraditórios. O domínio alvo do InfoRat é o projeto de software. Ele pode ser integrado, por exemplo, com ferramentas CASE que reportam problemas, mostrando as razões porque o projeto exigiu modificações, bem como quais mudanças foram feitas.

Em Engenharia de Software, vem sendo investigado o emprego do DR para a confecção de projetos bem elaborados. A captura do *rationale* contribui principalmente para a manutenção e reuso do software. Em [PV96], por exemplo, é apresentado um *framework* que usa DR e padrões de projeto para desenvolver sistemas de software reutilizáveis. O projeto de reuso de software envolve a aplicação de vários tipos de conhecimento sobre um sistema existente em um novo sistema, a fim de reduzir o tempo e o custo de desenvolvimento e de manutenção do novo sistema. O conhecimento reutilizado envolve conceitos que são registrados pelo DR tais como: conhecimento do domínio/contexto, decisões de projeto e história do projeto.

Os benefícios de se empregar DR em projeto de interfaces são semelhantes àqueles obtidos em Engenharia de Software. O *rationale* explica porque uma inteface foi construída de uma determinada forma [AYM89]. O DR é utilizado nesta dissertação sob esta mesma perspectiva. A seção 2.4 apresenta as principais abordagens de representação de DR descritas na literatura.

2.3 Modelos de Dados Hipermídia

Atualmente existe um grande número de sistemas hipermídia voltados para diferentes domínios. Osterbye & Will [OW96] classificam os sistemas hipermídia em cinco categorias:

- **Sistemas Monolíticos**: têm um único módulo que lida com todos os aspectos do sistema (armazenamento, gerenciamento e apresentação dos dados). KMS [AMY88] e StorySpace 1 [Ber02] pertencem a esta categoria;

- **Hiperbase**: sistemas caracterizados por um módulo de armazenamento que manipula conteúdo e estrutura e por um gerenciador de sessão que dá suporte aos visualizadores na manutenção do conteúdo e estrutura dos dados. Um exemplo é o Sepia [SHH+92];

- **Sistemas de ligações embutidas**: representam um caso especial de hiperbase com apenas dois módulos, armazenamento e execução. Esta abordagem não separa conteúdo e estrutura. Um exemplo é a WWW (*World Wide Web*) [BLCL+94], onde o armazenamento é feito em servidores WWW e a execução em navegadores

(por exemplo, Netscape e Internet Explorer). As ligações são embutidas dentro dos documentos HTML através de *tags* especiais contendo a URL do documento referenciado [Hel01];

- **Servidores de Ligações**: mantêm apenas a estrutura da rede hipermídia em uma base de ligações. O armazenamento e apresentação do conteúdo são feitos por ferramentas externas, caracterizando esses sistemas como abertos. Multicard [RS92] e Chimera [ATW00] pertencem a esta categoria;

- **Hiperbase aberta**: combina as abordagens de hiperbase e servidores de ligações. O módulo de armazenamento é responsável por armazenar a estrutura e pode também armazenar o conteúdo ou deixar o armazenamento do conteúdo como responsabilidade de ferramentas externas. DHM [GT94], HyperDisco [WL96] e Hyperform [WL97] pertencem a esta categoria.

Diferentes sistemas hipermídia utilizam diferentes modelos de dados. Os modelos diferem quanto ao conjunto de abstrações utilizadas para representar a rede hipermídia, quanto à organização dessas abstrações e consequentemente quanto às funcionalidades fornecidas para o usuário. Há ainda diferenças quanto à nomenclatura utilizada para denominar abstrações equivalentes. Por exemplo, a abstração comumente conhecida como nó é chamada de *frame* em KMS, *object* em Chimera, *component* em DHM e *node* em Sepia, Multicard e Hyperdisco. Para uma representação formal e comparação de diferentes modelos de dados hipermídia ver [Whi02].

Apesar das diferenças, a maioria dos modelos de dados hipermídia são variações do paradigma básico nó-e-ligação no qual dados multimídia (nós) são interrelacionados por ligações constituindo um grafo direcionado. Esses modelos utilizam três conceitos principais [Gon97]:

- **Abstrações de baixo nível**: entidades estruturais básicas que constituem o modelo. Correspondem aos nós e ligações, que podem ser refinados através da especificação de atributos. Definições de ligações podem suportar diversas características diferenciadoras como tipo, direção, âncoras e relacionamentos com o nó. Âncoras são, em geral, definidas como extremos de ligações e referenciam alguma parte do conteúdo de um nó;

- **Abstrações de alto nível**: permitem agregação dos componentes básicos e hierarquias de generalização/especialização. Abstrações adicionais podem incluir versões e contextos;

- **Operações**: Incluem, geralmente, operações básicas de visualização, edição, inserção e eliminação de dados. Vários modelos definem outras operações particulares.

Exemplos são operações para manter restrições de integridade, tais como: assegurar a unicidade do objeto; desabilitar a definição recursiva de objetos complexos; e impedir a presença das chamadas ligações *dangling*, isto é, ligações que não apontam para lugar nenhum.

A constatação da existência de aspectos em comum entre os sistemas hipermídia levou à criação de modelos de referência. O mais utilizado desses modelos é o Dexter [HS90, HS94]. Grønbæk & Trigg [GT94] propuseram importantes extensões para o modelo Dexter resultando no modelo DHM (*DeVise Hypermedia Model*). Esses modelos serão descritos a seguir.

2.3.1 Modelo de Referência Dexter

O modelo de referência Dexter [HS90, HS94] é uma tentativa de capturar, de uma maneira formal, as principais abstrações encontradas em sistemas hipermídia. O objetivo do modelo é fornecer uma base para comparar diferentes sistemas e uma infra-estrutura para o desenvolvimento de padrões de interoperabilidade e intercâmbio entre esses sistemas. O modelo é resultado de dois *workshops* em hipertexto, que reuniram discussões sobre os sistemas hipermídia mais representativos da época. O primeiro ocorreu em 1988 em Dexter Inn em New Hampshire, dando origem ao seu nome.

O modelo divide um sistema hipermídia em três camadas: **camada interior aos componentes**, **camada de armazenamento** e **camada de tempo de execução**. O principal foco do modelo é a camada de armazenamento, que modela a rede básica de nós e ligações. A figura 2.2 mostra essas camadas e as interfaces de comunicação entre elas.

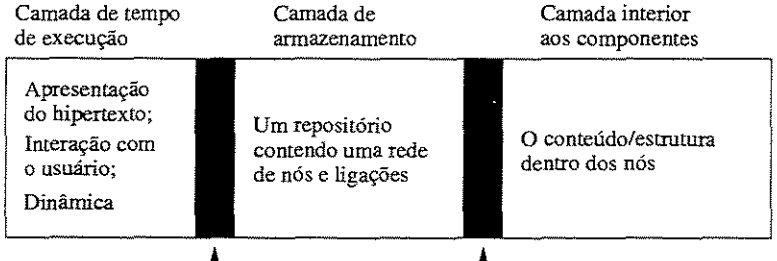


Figura 2.2: As camadas do modelo Dexter e suas interfaces

Camada Interior aos Componentes

Esta camada preocupa-se com o conteúdo e estrutura interna dos componentes da rede hipermídia. Ela é propositadamente não elaborada dentro do modelo, por causa da grande variedade de tipos de dados que podem ser incluídos nos componentes, por exemplo, textos, gráficos, animações e imagens. O modelo Dexter assume que padrões específicos para descrever a estrutura interna de tipos de dados particulares (por exemplo, SGML) serão usados em conjunto com ele.

Camada de Armazenamento

Esta camada descreve a organização de uma rede hipermídia como um "banco de dados" composto por um conjunto de componentes interconectados por ligações relacionais. A camada possui ainda duas funções: uma função de mapeamento (*resolver*) e uma função de acesso aos dados (*accessor*). Essas duas funções em conjunto são responsáveis pela recuperação dos componentes.

A entidade básica desta camada é chamada **componente**. Um componente pode ser um **átomo**, uma **ligação** ou uma **entidade composta** por outros componentes. Os componentes atômicos correspondem ao que é tipicamente conhecido como nó em sistemas hipermídia, atuando como repositórios genéricos de dados. Ligações são entidades que representam relacionamentos entre componentes. Elas são formadas por duas ou mais especificações de extremos, cada uma das quais se referindo a um componente completo ou parte dele. Um componente composto pode conter átomos, ligações ou outras composições. A única restrição é que deve ser um grafo acíclico, ou seja, não pode fazer referência a si próprio direta ou indiretamente.

Todo componente possui um identificador global único (UID). A função de mapeamento é responsável por mapear uma especificação de componente em seu UID correspondente, o qual é passado para a função de acesso que então recupera o componente.

Um componente é, na verdade, uma entidade complexa formada por um componente base (átomo, ligação ou composição) e um conjunto de informações que descrevem as propriedades estruturais do componente. As informações do componente contêm uma sequência de âncoras, uma especificação de apresentação que indica como o componente deve ser apresentado, e um conjunto de pares arbitrários atributo/valor. Quando o componente é uma ligação, seu conteúdo é um conjunto de especificadores. Esses conceitos serão definidos a seguir. A figura 2.3 ilustra a estrutura de um componente.

Âncoras são entidades de endereçamento indireto utilizadas para endereçar partes de um componente individual. Elas constituem a interface ente a camada de armazenamento e a camada interior aos componentes, mantendo a separação lógica entre essas camadas. Uma âncora tem duas partes: um identificador e um valor. O valor especifica alguma localização, região, item ou subestrutura dentro do componente. O identificador da âncora

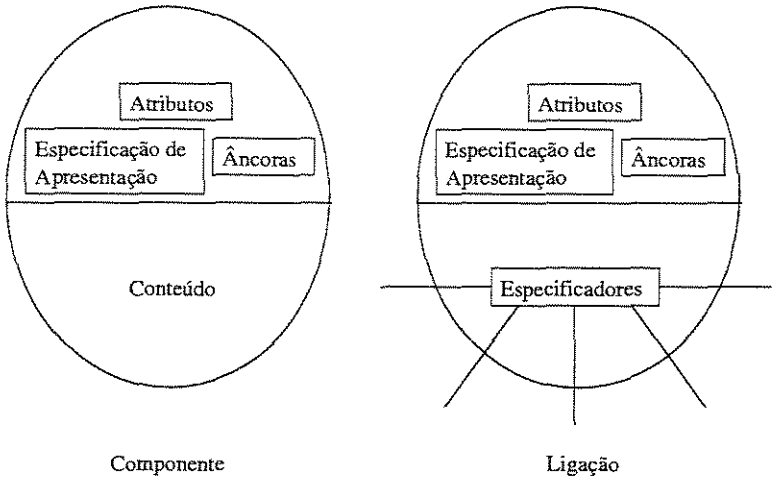


Figura 2.3: Estrutura dos componentes na camada de armazenamento

a identifica unicamente dentro do escopo de seu componente.

O mecanismo de identificação da âncora pode ser combinado com o mecanismo de especificação de componentes de forma a especificar os extremos de uma ligação. Isso possibilita que ligações sejam estabelecidas entre componentes completos e também entre localizações específicas dentro de componentes. No modelo Dexter, esta combinação é capturada por uma entidade chamada **especificador**. Um especificador especifica um componente e uma âncora dentro do componente que serve como extremo de uma ligação. Além disso, ele possui dois campos adicionais: uma **direção** e uma **especificação de apresentação**. A direção indica se o extremo especificado é origem ou destino de uma ligação, ambos ou nenhum deles (codificados respectivamente por FROM, TO, BIDIRECT e NONE). A especificação de apresentação constitui a interface entre a camada de armazenamento e a camada de execução. Ela permite especificar, dentro da camada de armazenamento, como um componente deve ser apresentado para o usuário.

As definições anteriores permitem descrever a estrutura de uma ligação de uma forma mais precisa. Uma ligação é formada por um sequência de dois ou mais especificadores. Isso possibilita que ligações tenham uma aridade arbritária, embora ligações binárias sejam as mais comuns em sistemas hipermídia. Ligações direcionais são tratadas através do campo direção do especificador.

O campo **atributo** pode ser usado para anexar propriedades arbitrárias a um componente. Por exemplo, palavras-chave podem ser anexadas a um componente usando múltiplos atributos "palavra-chave". Da mesma forma, tipos de componentes podem ser implementados no modelo pela adição de um atributo "tipo" a cada componente cujo valor deve ser uma especificação de tipo apropriada.

Em adição ao modelo de dados, a camada de armazenamento define um conjunto de operações para acessar e/ou modificar os dados e as estruturas hipermídia. Essas operações incluem adição e remoção de um componente, modificação do conteúdo e da informação auxiliar de um componente (tais como suas âncoras e atributos), recuperação de um componente dado seu UID e operações para auxiliar a determinação de interconectividade da rede hipermídia, dentre outras.

Camada de Tempo de Execução

Esta camada é responsável por instanciar os componentes da rede hipermídia e apresentá-los ao usuário. No modelo Dexter uma **instanciação** corresponde a uma cópia de um componente. A cópia é visualizada pelo usuário que pode editá-la e salvá-la. Neste caso as alterações são gravadas na camada de armazenamento. Pode haver várias instanciações simultâneas de um componente. Cada instanciação recebe um identificador de instância (IID) único.

A instanciação de um componente implica na instanciação de suas âncoras. Uma âncora instanciada é denominada **marcador de ligação**. Este marcador é uma manifestação visível de uma âncora em um documento apresentado. Para acomodar este conceito no modelo, uma instanciação corresponde a uma entidade complexa contendo uma instanciação base juntamente com uma sequência de marcadores de ligação e uma função que faz o mapeamento entre os marcadores de ligação e as âncoras que eles instanciam.

Esta camada inclui também uma entidade chamada **sessão** que permite o usuário manipular várias instanciações ao mesmo tempo. Quando um usuário pretende acessar estruturas hipermídia ele deve explicitamente abrir uma sessão. Instanciações são criadas na sessão através da operação **"apresentar componente"**.

O coração da camada de execução é a chamada **função instanciadora** da sessão. Ela recebe como entrada o UID e a especificação de apresentação de um componente e retorna uma instanciação desse componente como parte da sessão.

A função instanciadora é também o núcleo da operação **"apresentar componente"** que por sua vez é o núcleo da operação **"seguir ligação"**. O resultado de **"seguir ligação"** é a apresentação dos componentes que são o destino da ligação em questão.

2.3.2 *DeVise Hypermedia Model (DHM)*

O DHM [GT94] é um sistema hipermídia aberto baseado no modelo Dexter. Ele possui um projeto de componentes orientado a objetos utilizando um Sistema Gerenciador de Banco de Dados Orientado a Objetos (SGBDOO) para armazenar os objetos persistentes. O modelo propõe extensões para os elementos ligações, âncoras e componentes/composições

de Dexter. As principais extensões serão apresentadas a seguir.

Ligações

O DHM suporta a noção explícita de ligações chamadas *dangling*. Estas ligações têm zero ou um extremo, relaxando a semântica do modelo Dexter, que é restrita a ligações com pelo menos dois extremos. No DHM as ligações *dangling* podem aparecer como resultado de algum processo sobre os objetos hipermídia como, por exemplo, exclusão de extremos de ligações (componentes ou âncoras) ou indisponibilidade de objetos relevantes referenciados pelo conteúdo do componente. Ligações *dangling* podem também ser criadas intencionalmente como *placeholders* quando seus extremos ainda não existem.

A noção de direcionalidade de uma ligação também é estendida. O DHM considera três tipos de direção: direção semântica, que representa um relacionamento semântico entre os componentes; direção de criação, que corresponde à ordem em que os extremos foram criados; e direção de percurso, que especifica como a ligação deve ser percorrida – se da origem para o destino, o inverso ou ambos.

Âncoras

O DHM estende o modelo de âncoras de Dexter de várias formas. Primeiro, ele usa referências dinâmicas (apontadores) ao invés de IDs de âncoras. Isto significa que os especificadores de ligação apontam diretamente para as âncoras do componente, evitando a necessidade de utilizar a função de acesso aos dados.

O DHM distingue três tipos de âncoras, independentes do tipo de componente que as inclui: âncoras para componente completo, âncoras marcadas e âncoras não marcadas. As âncoras para **componente completo** suportam o caso de extremos de ligações não ancorados dentro de um componente. Todas as ligações com extremos componente-completo compartilham uma única âncora que aponta para o componente por inteiro.

Âncoras **marcadas** são associadas com objetos particulares inseridos no conteúdo do componente. Estes objetos são chamados marcadores de ligação no Dexter, podendo ser visíveis ou não. O DHM apresenta marcadores de ligação para o usuário, em componentes textuais, como regiões contornadas. Um *click* na região marcada invoca a ação de seguir as ligações da âncora correspondente.

Âncoras **não marcadas** não possuem marcadores de ligação. Sua localização dentro de um componente deve ser computada. Um exemplo de âncora não marcada em DHM é a chamada âncora de palavra-chave, suportada por componentes textuais. Sua criação requer salvar uma cópia da *string* selecionada como valor da âncora. Ao invocar a operação seguir ligação a partir de uma seleção que não corresponde a um marcador de ligação, é verificado se a seleção é igual ao valor de alguma âncora de palavra-chave. Se sim, a ligação correspondente à âncora é seguida.

Componentes e Composições

O DHM oferece um conjunto de estruturas para componentes e composições mais rico que o Dexter, suportando construções complexas para aplicações particulares. Dentre as extensões propostas destaca-se a distinção feita entre componentes cujo conteúdo é gerenciado (armazenado) pelo banco de dados e aqueles cujo conteúdo é gerenciado por aplicações externas. No DHM um objeto de dados pode ser ancorado em um componente de duas formas: (1) o componente mantém os dados propriamente ditos como parte de seu conteúdo e (2) o componente (denominado *wrapper*) mantém apenas uma referência para os dados, por exemplo, o nome de um arquivo. No segundo caso, o DHM suporta ligações para documentos criados com aplicações como Microsoft Word ou Excel. A operação **"seguir ligação"** executa automaticamente a aplicação apropriada sobre o arquivo referenciado.

2.4 Modelos de Representação de *Design Rationale*

Esforços de pesquisa na área de representação de DR visam alcançar os seguintes objetivos [HPP+00]: ajudar a organizar a argumentação; rastrear decisões; manter a consistência na tomada de decisão; comunicar o raciocínio ao longo do projeto; e ajudar a responder indagações.

As principais abordagens de representação de DR descritas na literatura usam notações gráficas semi-formais baseadas em grafos. Nestes grafos, nós representam conceitos que organizam o processo de raciocínio e arestas mostram relacionamentos entre os nós. As representações são consideradas semi-formais porque os conceitos e relacionamentos entre eles são formalmente estruturados mas o conteúdo de cada nó são sentenças informais. Ferramentas de hipertexto são frequentemente usadas em conjunto com estas representações, que tornam-se difíceis de gerenciar à medida que o número de nós e arestas cresce.

Os modelos de representação de DR mais utilizados atualmente são *Issue-Based Information System (IBIS)*, *Procedural Hierarchy of Issues (PHI)* e *Design Space Analysis (DSA)*. Outro modelo relevante para este trabalho é o *Proteus*, desenvolvido para auxiliar o projeto de software. Estes modelos serão descritos a seguir.

2.4.1 *Issue-Based Information System (IBIS)*

O IBIS foi a primeira proposta específica para representar DR. O modelo IBIS procura capturar as questões que surgem ao longo das discussões de projeto, juntamente com as várias posições (ou alternativas) que surgem em resposta a essas questões, e os argumentos a favor ou contra as posições. Sua representação original [KR70] compreende

um grafo direcionado com três tipos de nós: Questões (*Issues*), Posições (*Positions*) e Argumentos (*Arguments*). As ligações entre os nós podem ser de oito tipos. Por exemplo, uma Posição <Responde-a> uma Questão. Argumentos relacionam-se com Posições através de ligações <Sustentam> ou <Opõem-se>. Questões podem <Generalizar> ou <Especializar> outras Questões e podem também <Questionar> ou <Ser Sugeridas> por outras Questões, Opções ou Argumentos. A figura 2.4 mostra o vocabulário de IBIS especificando os relacionamentos legais do modelo.

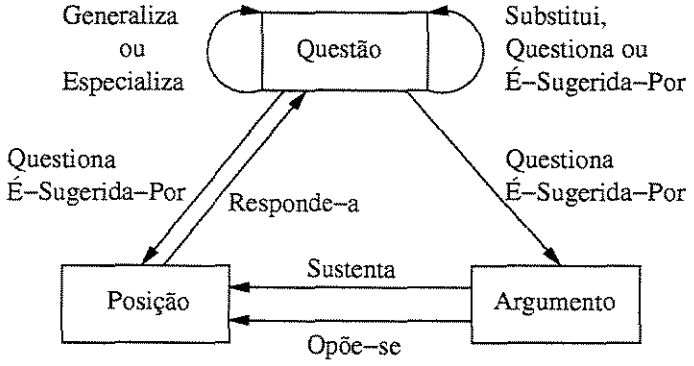


Figura 2.4: Vocabulário e transições do IBIS. Adaptada de [CB88].

O IBIS representa a argumentação ao longo de um projeto como uma árvore, cuja raiz é a questão principal que dá origem à discussão. Os filhos diretos dessa Questão são as Posições que tentam solucioná-la. Argumentos são filhos das Posições que sustentam ou opõem-se. A árvore cresce à medida que Questões secundárias (e Posições e Argumentos relativos a elas) surgem para questionar/expandir a discussão sobre a Questão da raiz.

Uma implementação gráfica do modelo IBIS foi feita no gIBIS (*graphical IBIS*) [CB88], uma ferramenta de hipertexto designada para suportar a construção colaborativa de esquemas IBIS por equipes conectadas em uma rede local. Para permitir maior flexibilidade, a implementação de gIBIS estendeu o modelo IBIS acrescentando (1) um tipo de nó e ligação Outro (*Other*) como um mecanismo de escape para usuários que poderiam não conseguir expressar alguma idéia dentro do *framework* IBIS; (2) um nó do tipo Externo (*External*) para gerenciar itens externos ao IBIS tais como especificação de requisitos ou código fonte; e (3) a possibilidade de que as ligações de generalização e especialização ocorram também entre Posições e entre Argumentos.

2.4.2 *Procedural Hierarchy of Issues (PHI)*

O PHI [McC91] é uma extensão do IBIS que busca representar mais precisamente a estrutura do processo de projeto. As principais mudanças são: alteração da estrutura que

relaciona Questões, Posições e Argumentos; e forma de resolver as Questões.

PHI substitui o conjunto de relacionamentos entre Questões do IBIS por um único relacionamento denominado Serve *(Serves)*. Uma Questão A <Serve> uma Questão B se a resolução de A ajuda a resolver B. O relacionamento <Serve> é implementado indiretamente por dois relacionamentos: <Subquestão> *(SubQuestion)* e <Antecedente> *(Antecedent)*. A é uma SubQuestão de B se B foi levantada primeiro. A é um Antecedente de B se A foi levantada primeiro. Na prática, referências explícitas a relacionamentos do tipo Antecedente são usualmente omitidas, pois estas ficam implícitas no texto das Questões que as contêm.

Em PHI, as Posições do IBIS são denominadas Respostas *(Answers)*. Respostas podem ser especializadas através de relacionamentos <SubRespostas> *(SubAnswers)* constituindo uma hierarquia de SubRespostas. Os Argumentos que sustentam ou opõem-se às Respostas também podem ser especializados em SubArgumentos. Ao contrário de IBIS, PHI não classifica os argumentos em a favor ou contra determinada Resposta.

Questões podem ser resolvidas de duas formas em PHI: deliberação ou decomposição. A deliberação, como no IBIS, corresponde à discussão das Respostas alternativas para uma Questão com base em Argumentos e a escolha de uma alternativa como solução. A decomposição em SubQuestões é utilizada para Questões que não podem ser resolvidas diretamente. Neste caso, a resolução da Questão é formada pelas resoluções de suas SubQuestões.

A estrutura de PHI é quase hierárquica (uma questão pode ter mais que um antecedente), sendo geralmente representada como uma descrição aninhada. A figura 2.5 ilustra esta notação.

2.4.3 *Design Space Analysis (DSA)*

A DSA, proposta em [MYBM96], coloca o alvo do projeto em um espaço de possibilidades e procura explicar porque um artefato particular foi escolhido dentre estas possibilidades. Ela usa uma notação chamada QOC (*Questions, Options,* and *Criteria*) para representar o espaço de projeto ao redor de um artefato. Os principais componentes (nós) dos grafos QOC são as Questões, Opções e Critérios. As Questões são perguntas que identificam problemas chave que devem ser endereçados pelo projeto. As Opções fornecem possíveis respostas para as Questões, ou seja, representam soluções alternativas. Os Critérios são bases de avaliação e escolha entre as Opções. Os Critérios representam propriedades desejáveis do artefato e elucidam os objetivos do projeto. Os relacionamentos entre Opções e Critérios são denominados Avaliações (*Assesments*) e podem ser de dois tipos: avaliação positiva (a Opção satisfaz o Critério) ou avaliação negativa (a Opção não satisfaz o Critério). Outro componente de QOC são os Argumentos. Eles podem se relacionar

QUESTÃO:
Questão?
SUBQUESTÕES:
SubQuestão1?
RESPOSTAS:
1 Resposta
SUBRESPOSTAS:
1: SubResposta1
ARGUMENTO:
Argumento
2: SubResposta2
2 Resposta2
ARGUMENTOS:
1: Argumento1
2: Argumento2
SubQuestão2?
RESPOSTA:
Resposta
ARGUMENTO:
Argumento
SUBARGUMENTOS:
1: SubArgumento1
2: SubArgumento2

Figura 2.5: Exemplo de argumentação em PHI

com Questões, Opções, Critérios e Avaliações sustentando ou opondo-se à sua presença ou caracterização. Argumentos podem também sustentar ou desafiar outros Argumentos.

A notação QOC está ilustrada na figura 2.6. Cada Questão é ligada às Opções que a responde. Opções são ligadas aos critérios que as avaliam e podem ser ligadas a questões consequentes que as seguem. A Opção escolhida como solução de cada Questão é destacada por um retângulo. Os Argumentos são ligados aos elementos que justificam.

O *design space* é o conjunto de relacionamentos conceituais ou dimensões, usado para comparar projetos e formas alternativas de instanciação de conceitos [Shu91]. A DSA permite descobrir dimensões importantes em um espaço (Quais são as Questões?), explorar o espaço de alternativas (Opções) que definem os espaços locais ao redor das dimensões, e justificar porque um ponto em um espaço local é melhor que outro no contexto explorado (através de Critérios e Argumentos).

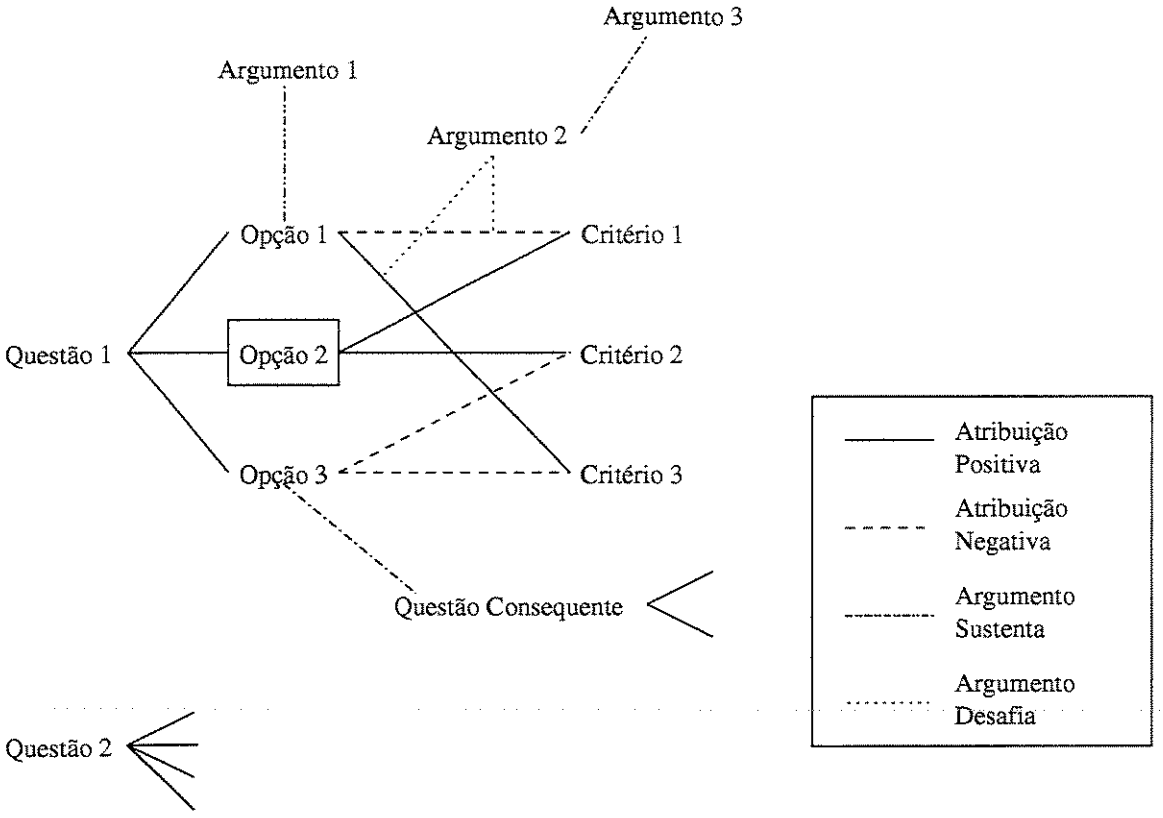


Figura 2.6: Notação QOC, usada para representar a *Design Space Analysis*. Adaptada de [MYBM96].

2.4.4 *Proteus*

O *Proteus* [MSPD95] é um modelo voltado para documentar e gerenciar o *rationale* de um projeto de software. Ele adapta conceitos existentes nos modelos descritos anteriormente, buscando aproximá-los do vocabulário já conhecido pelos engenheiros de software. Além disso, ele incorpora novos conceitos que integram análise de riscos ao DR. O modelo *Proteus* define treze tipos de entidades:

1. Requisito (*Requirement*): um requisito que deve ser satisfeito pelo projeto;
2. Problema (*Problem*): pergunta que especifica o problema correspondente ao requisito;
3. Soluções candidatas (*Candidate solutions*): possíveis soluções para o problema;
4. Solução provável (*Probable solution*): solução candidata avaliada como sendo a melhor solução;

5. Solução validada (*Validated solution*): solução provável depois da validação pela análise de riscos;

6. Elemento de projeto (*Design Element*): elemento de projeto que implementa um requisito;

7. Metas (*Goals*): aspectos do problema que as soluções candidatas devem endereçar;

8. Risco (*Risk*): risco que pode tornar uma solução provável inaceitável;

9. Ação (*Action*): ação que pode ser executada para reduzir o risco;

10. Ação pendente (*Remaining action*): ação que não pode ser completada durante o projeto;

11. Matriz de decisão (*Decision matrix*): tabela usada para avaliar os méritos de cada solução candidata. A matriz de decisão pode ser calculada para cada Problema a partir de pesos atribuídos às Metas do Problema (colunas da tabela) e às Soluções Candidatas (linhas da tabela);

12. Argumentos contrários (*Object arguments*): argumentos contra uma solução candidata;

13. Argumentos favoráveis (*Support arguments*): argumentos a favor de uma solução candidata.

Este amplo conjunto de tipos de entidades foi criado para refletir diretamente o processo da engenharia de software [MSPD95]. O modelo deve ser instanciado para cada decisão de projeto.

A figura 2.7 ilustra o modelo *Proteus* mostrando a representação gráfica projetada para as entidades e os relacionamentos existentes entre elas.

2.5 Comparação entre os Modelos de Representação de *Design Rationale*

O aspecto central da representação do *rationale* de um projeto gira em torno das questões que formalizam as discussões surgidas ao longo do projeto, das diferentes alternativas que respondem a estas questões e dos argumentos a favor ou contra às alternativas. Esta idéia é representada no IBIS de forma clara e simples através das entidades Questões, Posições e Argumentos e dos relacionamentos <Responde-a>, <Suporta> e <Opõe-se>. Estes elementos básicos de IBIS possuem correspondentes diretos em todos os modelos

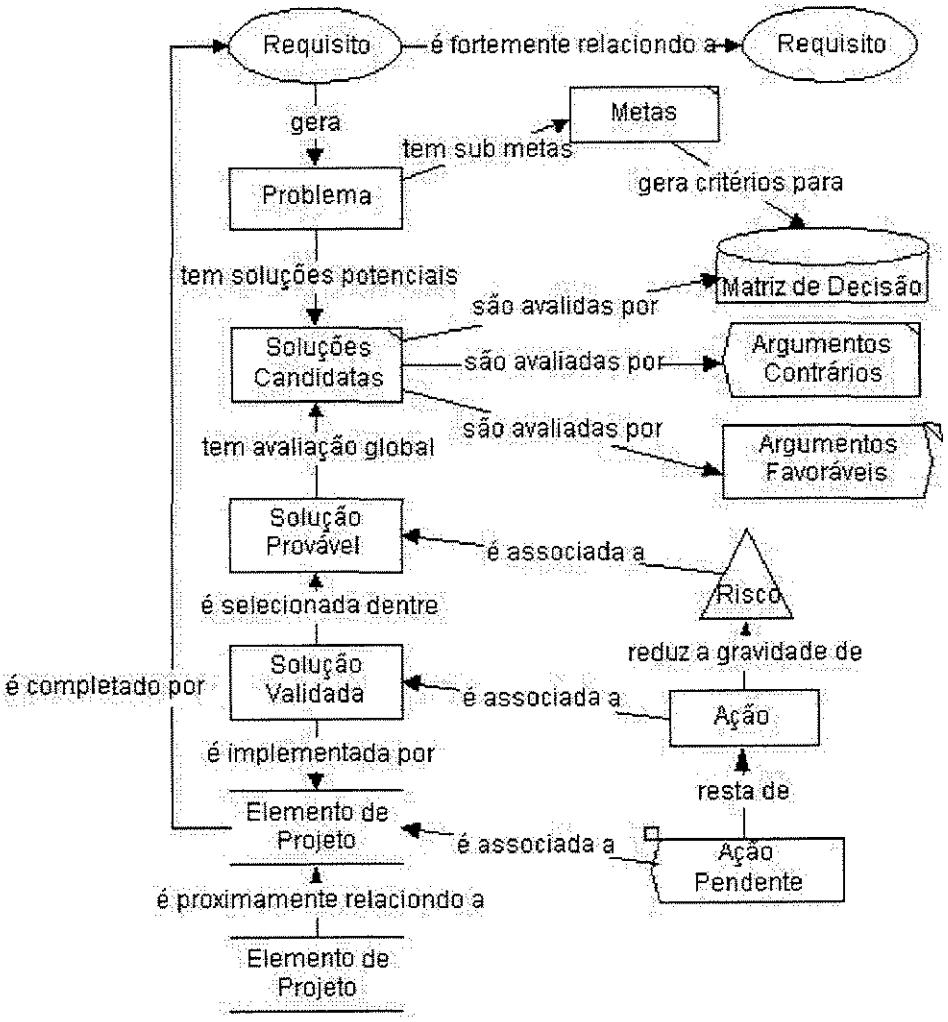


Figura 2.7: Modelo *Proteus*. Adaptada de [MSPD95].

descritos na seção 2.4, com exceção dos relacionamentos <Suporta> e <Opõe-se> que não são explicitamente diferenciados em PHI.

Além dos elementos básicos, os modelos DSA e *Proteus* representam também conceitos referentes a propriedades desejáveis do artefato que devem ser endereçadas pela solução do problema. Em DSA estas propriedades são representadas pelos Critérios e no *Proteus* pelas Metas. Apesar de representarem a mesma idéia, estes elementos são disponibilizados de forma diferente nos dois modelos. Em DSA, os Critérios são relacionados com as Opções que respondem a uma Questão. No *Proteus*, as Metas são diretamente relacionadas com o Problema (questão). A tabela 2.1 mostra a correspondência entre as entidades dos quatro modelos apresentados.

IBIS e PHI	**DSA**	**Proteus**
Questão (*Issue*)	Questão (*Question*)	Problema
Posição	Opção	Solução Candidata
Argumento	Argumento	Argumento Contrário e Argumento Favorável
	Critério	Meta

Tabela 2.1: Correspondência entre entidades de diferentes modelos de representação de *design rationale*

O *Proteus* possui entidades que não têm correspondentes nos outros modelos. Enquanto IBIS, PHI e DSA começam a formalizar um discussão através de questões, o *Proteus* representa um nível de abstração anterior à elaboração das questões, através dos Requisitos. O conceito de requisito é familiar em Engenharia de Software e a sua representação facilita a expressão do DR pelos engenheiros de software. O *Proteus* representa também o Elemento de Projeto que implementa um Requisito. Assim, o modelo estabelece uma ponte entre um requisito levantado durante a análise de requisitos, as discussões surgidas em busca de uma solução para o problema e o elemento do projeto de software que atende cada requisito. Outro aspecto representado pelo *Proteus* é a análise de riscos. A consideração dos riscos é importante não só em Engenharia de Software mas em projetos de engenharia em geral. O *Proteus* permite ainda calcular uma matriz de decisão para cada Problema.

A forma como as entidades são relacionadas também varia de um modelo para outro. QOC e *Proteus* apresentam um conjunto de relacionamentos simples com semântica intuitiva. Os relacionamentos entre Questões do IBIS são menos intuitivos e mais difíceis de ser modelados principalmente para iniciantes. O PHI minimiza este problema relacionando Questões através de um único relacionamento (<Serve>). Questões relacionadas por <Serve> formam uma hierarquia que possibilita a resolução de uma Questão indire-

tamente por decomposição. Outro diferencial de PHI em relação aos outros modelos é o suporte a hierarquias de Respostas e de Argumentos.

2.6 Planejamento Ambiental

O planejamento ambiental é um campo multidisciplinar que recebe contribuições de várias áreas tais como biologia, engenharia, geografia, geologia, hidrologia, arquitetura, dentre outras. Cada área trata problemas de planejamento ambiental sob uma perspectiva diferente, por isso não há uma definição única do que é planejamento ambiental [Ort84].

"Planejamento é o ato ou processo de elaborar ou executar planos; especificamente: o estabelecimento de metas, políticas, e procedimentos para uma unidade econômica ou social em planejamento de cidades ou de negócios. Um plano é um método projetado para fazer alguma coisa ou atingir um objetivo. Ele sempre implica em formulação mental e as vezes em formulação gráfica. Planejar significa ter em mente e projetar a realização de um plano ou programa" [Dicionário Webster].

Segundo Franco [Fra00], planejamento ambiental é todo planejamento que parte do princípio da valoração e conservação das bases naturais de um dado território como base de auto-sustentação da vida e das interações que a mantém, ou seja, das relações ecossistêmicas. Para isso, a ação humana sobre os ecossistemas deve seguir três princípios: preservação, recuperação e conservação do meio ambiente.

De acordo com [Rod94] o principal objetivo do planejamento ambiental é garantir as condições ecológicas para o desenvolvimento efetivo da produção social, e todas as atividades da população, através do uso racional e da proteção dos recursos do meio ambiente.

De um modo geral, o planejamento ambiental pode ser visto como um processo que visa explorar os recursos naturais eficientemente e ao mesmo tempo minimizar os efeitos da ação humana na degradação do meio ambiente. Este processo é complexo e envolve os meios físico, socio-econômico e cultural.

A atividade de planejamento ambiental normalmente é feita por equipes compostas por diferentes especialistas, que trabalham conjuntamente, buscando integrar as diferentes visões de cada área a um problema particular e resolvê-lo da melhor forma possível.

Os problemas de planejamento podem ocorrer em nível local, regional e global [SCP97]. Problemas locais, como por exemplo em nível municipal, referem-se geralmente a questões pontuais, que abrangem pequenas áreas e ecossistemas. Neste caso, o planejamento deve gerar soluções também pontuais e sempre que possível internas ao território estudado. Problemas regionais, como por exemplo uso adequado dos recursos hídricos de uma bacia hidrográfica que engloba vários municípios, centralizam seus estudos no entendimento da dinâmica regional bem como dos problemas ambientais que envolvem os municípios.

Quando se planeja em nível global, esforços costumam ser centralizados na discussão de um problema central: a qualidade ambiental como um complexo integrado. Neste caso, o planejamento é voltado à obtenção de políticas ambientais amplas, que interferem na organização política, econômica e administrativa de conjuntos de países.

O trabalho de planejamento é fortemente baseado na utilização de dados geográficos e geralmente é apoiado por softwares especiais para gerenciar tais dados, por exemplo, Sistemas de Informação Geográfica (SIGs). Esse trabalho pode seguir alguma metodologia com etapas bem definidas ou ser um processo iterativo de construção de cenários e avaliação de resultados, sem seguir uma metodologia clara [OPM97].

Em [OPM97] é apresentada uma metodologia de projeto de aplicações ambientais que divide o processo de planejamento ambiental em cinco etapas básicas: definição de objetivos; modelagem; integração de informação, prognóstico e identificação de alternativas; tomada de decisão; e definição de diretrizes e políticas. Cada etapa compreende várias tarefas que podem ser realizadas por diferentes meios (automático ou manual). Essa é uma metodologia global. Na prática, as etapas e o conjunto de tarefas adotados podem variar de acordo com os objetivos do planejamento.

A definição dos objetivos compreende as seguintes tarefas: descrição do problema; delimitação da área geográfica de interesse; definição de categorias, fatores e parâmetros pertinentes; e seleção de escalas de trabalho. A definição de fatores e parâmetros corresponde à determinação de temas geográficos (por exemplo, relevo e vegetação) que serão considerados e dos atributos relevantes (parâmetros) para cada tema.

A modelagem corresponde à modelagem do processo. Em um primeiro estágio, o usuário identifica e analisa as funções a serem aplicadas, e em um segundo estágio define como executá-las de acordo com o conjunto de algoritmos e modelos adotados. O resultado desta etapa é a especificação de um conjunto de cenários alternativos.

A integração de informação, prognóstico e identificação de alternativas envolve a execução do modelo especificado na etapa de modelagem e a integração dos diferentes cenários. Neste ponto, os usuários podem identificar diferentes estratégias de planejamento que levam ao objetivo desejado.

A tomada de decisão ocorre quando os especialistas escolhem uma estratégia de planejamento, selecionando um cenário do conjunto de alternativas produzido no passo anterior.

Finalmente, a definição de diretrizes e políticas corresponde à especificação de normas e procedimentos a serem tomados a fim de implementar a solução escolhida na etapa anterior. Estes procedimentos podem ser de natureza técnica, jurídica ou administrativa.

Essa metodologia não dá suporte à etapa de acompanhamento. O acompanhamento envolve a avaliação periódica dos resultados podendo levar à revisão e aperfeiçoamento do plano. Desta forma, o planejamento ambiental é um processo contínuo que pode não terminar nunca.

2.7 Resumo

Este capítulo apresentou os principais conceitos necessários ao entendimento do texto. O capítulo detalhou a base teórica necessária para a especificação dos documentos O QUE (estruturas de hipermídia), COMO (*workflows*) e PORQUE (*design rationale*). Além disso, apresentou uma caracterização da atividade de planejamento ambiental destacando uma metodologia genérica para auxiliar este tipo de atividade.

O próximo capítulo especifica a estrutura para representar cada tipo de documento. Os documentos serão gerenciados, de forma unificada, através de um banco de dados.

Capítulo 3

Especificação de Estruturas para Representação dos Documentos

Conforme mencionado na introdução, a dissertação considera três tipos de documentos relativos à documentação de atividades de planejamento ambiental: O QUE, COMO e PORQUE. Documentos O QUE são representados por estruturas de hipermídia, documentos COMO por *workflows* científicos e documentos PORQUE por *design rationale*. Este capítulo detalha a especificação desses três tipos de documentos constituindo um dos principais resultados da dissertação. A notação utilizada é o diagrama entidade relacionamento pois os documentos são gerenciados por um banco de dados relacional. A seção 3.1 especifica os documentos O QUE, a seção 3.2 especifica o COMO e a seção 3.3 especifica o PORQUE. Finalmente a seção 3.4 integra os tipos de documentos.

3.1 Modelo de Dados Hipermídia para Documentação do O QUE

Os documentos O QUE representam uma descrição do plano ambiental. A escolha de hipermídia para documentar o O QUE deveu-se a dois fatores principais:

- Possibilidade de organizar os documentos de uma forma hierárquica e não linear facilitando a interação do usuário com os mesmos;
- Possibilidade de incorporar diversos tipos de dados multimídia tais como gráficos e imagens (por exemplo, mapas e fotos de satélite) aos documentos.

A documentação O QUE relativa a planos ambientais é relativamente simples. O volume de dados é bem definido e o seu gerenciamento geralmente pode ser centrado em um único banco de dados. Questões relativas ao gerenciamento de grandes redes

hipermídia tais como suporte a ligações dinâmicas encontram-se fora do escopo deste trabalho. Para uma discussão sobre essas questões ver [Gon97].

O modelo de dados hipermídia projetado para documentar o O QUE é baseado no modelo de Dexter e em algumas extensões propostas pelo DHM. Esses modelos são descritos nas seções 2.3.1 e 2.3.2, respectivamente. O modelo de Dexter foi escolhido por ser um padrão de referência utilizado em vários sistemas hipermídia e por utilizar um conjunto de elementos estruturais bem definidos. A figura 3.1 ilustra o diagrama entidade-relacionamento do modelo proposto. As entidades definidas pelo modelo são:

- **Hiperdocumento**: agregação de um conjunto de nós interrelacionados constituindo um documento hipermídia. Todo hiperdocumento possui um nó principal ou raiz que é o primeiro nó apresentado para o usuário quando o um hiperdocumento é acessado. A partir do nó raiz o usuário poderá navegar pela rede hipermídia ativando as ligações e acessando os demais nós;

- **Nó**: representa um conjunto de dados multimídia. Todo nó possui um nome que o identifica unicamente. Geralmente possui também uma lista de âncoras. O modelo diferencia dois tipos de nós: textual e não textual. Essa classificação foi inspirada na definição de componentes *wrapper* feita em DHM;

- **Nó textual**: nó cujo conteúdo é um conjunto de caracteres armazenados no banco de dados;

- **Nó não textual**: nó cujo conteúdo são dados complexos (não textuais) que não podem ser facilmente armazenados no banco de dados. Neste caso, o atributo conteúdo do nó contém apenas um nome de arquivo, constituindo uma referência para o conjunto de dados. Um nó não textual pode ser referenciado por um nó textual. Isso ocorre quando o usuário deseja inserir, por exemplo, imagens ou gráficos no texto;

- **Âncora**: identifica uma localização dentro de um nó. Toda âncora possui um nome que a identifica unicamente dentro do nó. As âncoras aqui definidas correspondem às âncoras *marcadas* de DHM. Uma âncora pode ser visível ou não visível;

- **Âncora visível**: âncora que representa uma ligação e cujo valor aparece destacado no nó que a contém;

- **Âncora não visível**: âncora que indica uma região específica dentro de um nó, mas que não aparece destacada para o usuário. A figura 3.2 ilustra os dois tipos de âncora. Os termos sublinhados (introdução, 3.1, 3.2 e 3.3) são âncoras visíveis que representam ligações. Os títulos de cada seção representam âncoras não visíveis. Ao clicar nas ligações 3.1, 3.2 ou 3.3 haverá um deslocamento, dentro do próprio documento, para as respectivas seções;

- **Ligação**: unidade semântica para a navegação. Este modelo define apenas ligações binárias, logo toda ligação tem dois extremos – origem e destino;

- **Extremo**: entidade que modela referências (origem e destino) para ligações. Um extremo refere-se a um nó completo ou a uma âncora não visível dentro do nó. A utilização de âncoras não visíveis evita que haja ligações entre ligações. A opção de referenciar nós completos evita a necessidade de se definir uma âncora *default* para todo nó.

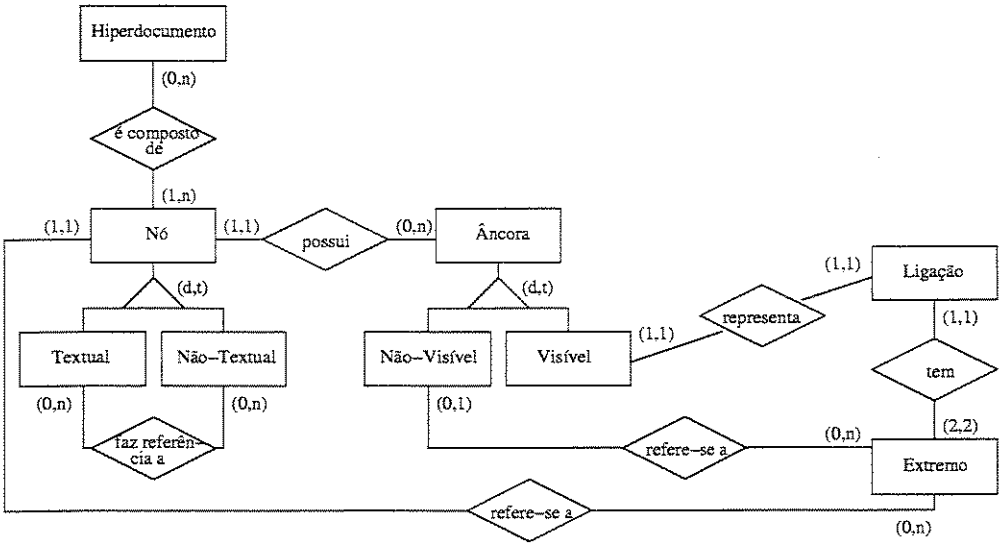
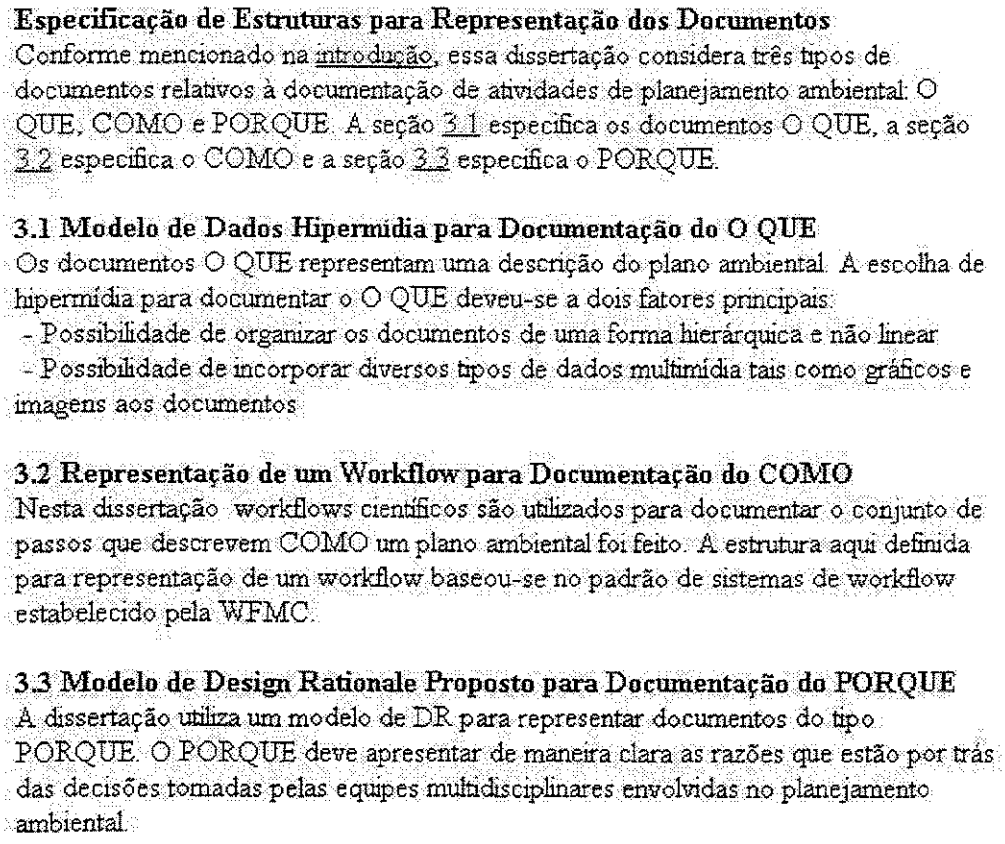


Figura 3.1: Modelo de dados hipermídia para a documentação do O QUE

Além das especializações inspiradas em extensões popostas pelo modelo DHM – especialização de nó em textual e não textual e especialização de âncora em visível e não visível, vale destacar as seguintes diferenças do modelo de dados hipermídia aqui proposto com relação ao Dexter: (1) adaptação de nomenclatura e (2) ligação direta de extremo para nó. No Dexter todo extremo refere-se a uma âncora dentro de um nó. Sendo assim, para se referir a um nó como um todo é preciso definir uma âncora para o começo do nó. Como pode ser observado na seção 2.3.1, o Dexter é um modelo bem mais complexo e define outros conceitos que não são empregados nesta dissertação.

O conjunto de operações definidas para esse modelo inclui apenas as operações básicas de acesso, atualização, adição e remoção de nós (conjuntos de dados) e de estruturas de navegação (âncoras, ligações e extremos). Todas as ligações são bidirecionais, ou seja, é possível navegar da origem para o destino (avançar) e do destino para a origem



Especificação de Estruturas para Representação dos Documentos
Conforme mencionado na introdução, essa dissertação considera três tipos de documentos relativos à documentação de atividades de planejamento ambiental: O QUE, COMO e PORQUE. A seção 3.1 especifica os documentos O QUE, a seção 3.2 especifica o COMO e a seção 3.3 especifica o PORQUE.

3.1 Modelo de Dados Hipermídia para Documentação do O QUE
Os documentos O QUE representam uma descrição do plano ambiental. A escolha de hipermídia para documentar o O QUE deveu-se a dois fatores principais:
- Possibilidade de organizar os documentos de uma forma hierárquica e não linear
- Possibilidade de incorporar diversos tipos de dados multimídia tais como gráficos e imagens aos documentos

3.2 Representação de um Workflow para Documentação do COMO
Nesta dissertação workflows científicos são utilizados para documentar o conjunto de passos que descrevem COMO um plano ambiental foi feito. A estrutura aqui definida para representação de um workflow baseou-se no padrão de sistemas de workflow estabelecido pela WFMC.

3.3 Modelo de Design Rationale Proposto para Documentação do PORQUE
A dissertação utiliza um modelo de DR para representar documentos do tipo PORQUE. O PORQUE deve apresentar de maneira clara as razões que estão por trás das decisões tomadas pelas equipes multidisciplinares envolvidas no planejamento ambiental.

Figura 3.2: Âncoras visíveis e não visíveis

(voltar). Não são definidas restrições de integridade adicionais além daquelas garantidas pelo SGBD.

3.2 Representação de um *Workflow* para Documentação do COMO

Nesta dissertação *workflows* científicos são utilizados para documentar o conjunto de passos que descrevem COMO um plano ambiental foi feito. O paradigma de *workflows* científicos foi escolhido para documentar o COMO porque ele vem sendo utilizado para documentar como procedimentos científicos foram realizados e a atividade de planejamento ambiental pode ser considerada um procedimento científico. A estrutura aqui definida para representação de um *workflow* baseou-se no padrão de sistemas de *workflow* estabelecido pela WFMC (*Workflow Management Coalition*) [Wor].

A WFMC é formada por um grupo de companhias que uniu-se para endereçar o problema da inexistência de padrões que permitissem que diferentes SGWFs pudessem trabalhar juntos. Eles verificaram que diferentes SGWFs disponíveis no mercado focavam em funcionalidades particulares mas possuiam um conjunto de características comuns. A identificação dessas características comuns viabilizou a construção de um modelo de implementação geral para um sistema de *workflow*. Este modelo padrão, chamado Modelo de Referência de Workflow (*Workflow Reference Model*), fornece uma base comum para o desenvolvimento de cenários de interoperabilidade entre diferentes sistemas de *workflow* [Hol95]. O modelo considera um SGWF como um conjunto de componentes genéricos e estabelece um conjunto de interfaces e formatos de troca de dados entre esses componentes. Como este projeto utiliza *workflows* para documentação, a preocupação principal é com o conteúdo do *workflow*. Questões relativas à execução de um *workflow* estão fora do escopo deste trabalho. O Modelo de Referência de Workflow define uma especificação padrão para o conteúdo de um *workflow* através da interface de definição de processo [Coa99]. Esta interface estabelece um meta-modelo que descreve entidades comumente usadas em uma definição de processo, seus relacionamentos e atributos. O meta-modelo está ilustrado na figura 3.3.

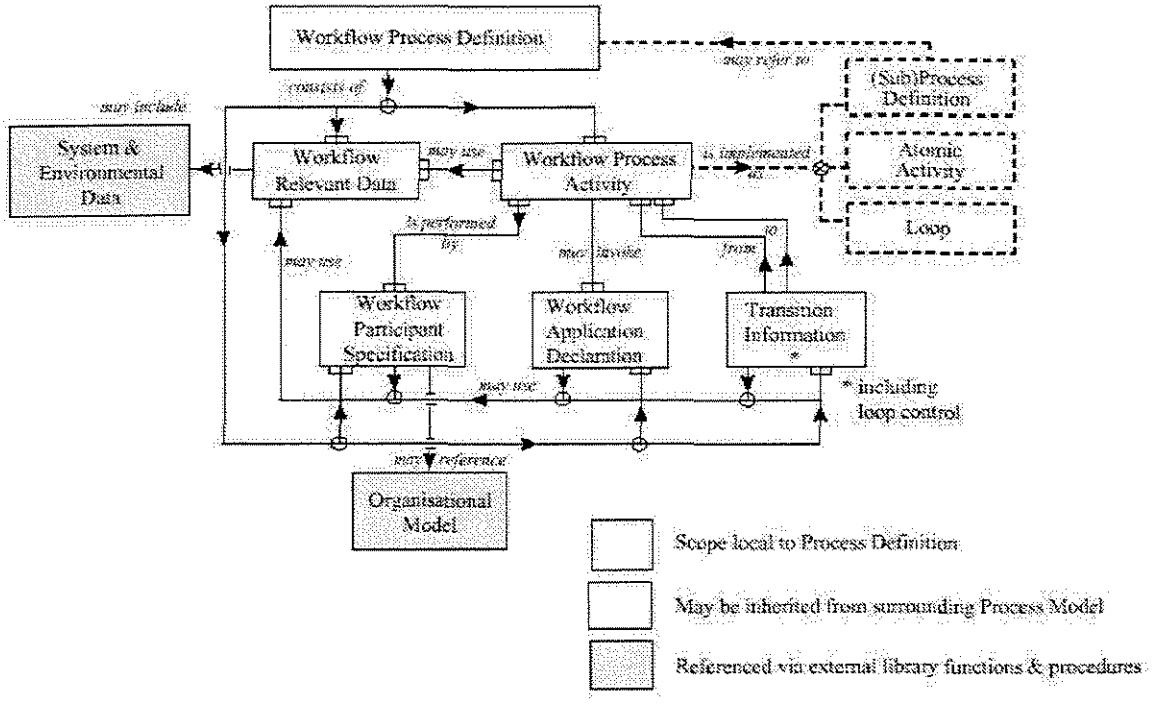


Figura 3.3: Meta-modelo padrão para especificação de um *workflow*. Retirada de [Coa99].

A especificação de um *workflow* para representar documentos do tipo COMO adapta o meta-modelo da figura 3.3 nos seguintes sentidos: remoção de elementos relativos à execução do *workflow*, por exemplo, entidade *Loop*; especificação mais detalhada de certos conceitos através de hierarquias de generalização-especialização, por exemplo, especialização de dependência em dependência de dado e dependência temporal; representação explícita do conceito de agente; adaptação dos relacionamentos; alteração de nomenclatura.

A figura 3.4 ilustra a especificação de um *workflow* proposta para representar um documento do tipo COMO. A especificação está de acordo com a definição de *workflow* fornecida na seção 2.1.3. As entidades definidas pelo esquema são as seguintes:

- **Workflow**: descrição do *workflow* propriamente dito o qual é formado por um conjunto de atividades interligadas;
- **Atividade**: unidade de trabalho a ser realizada. Pode ser uma atividade atômica ou um *sub-workflow*;
- **Atividade atômica**: unidade de trabalho atômica, a ser executada em um intervalo contínuo de tempo;
- **Sub-workflow**: atividade *dummy* que faz referência a outro *workflow* já armazenado no banco de dados. Este conceito facilita o reaproveitamento de especificações em novos projetos;
- **Dependência**: interdependência entre duas atividades. Este trabalho considera dois tipos de dependência – dependência temporal e dependência de dados;
- **Dependência de dado**: dependência entre duas atividades via dado. Uma atividade B depende de uma atividade A via dado se um dado de saída de A constitui um dado de entrada de B;
- **Dependência temporal**: dependência entre duas atividades com relação ao tempo de execução. Se uma atividade B possui uma dependência temporal em relação a uma atividade A, a execução de B só pode ocorrer após o término da execução de A;
- **Dado**: dado utilizado como entrada para o processamento de uma atividade ou resultado da execução de uma atividade;
- **Papel**: Descrição de um conjunto de agentes que podem executar uma atividade, ou seja, funcionalidade capaz de executar uma tarefa;

- **Agente**: entidade de processamento que executa uma atividade. Um agente pode ser um usuário ou um software;
- **Software**: sistema responsável pela execução automatizada de uma atividade;
- **Usuário**: pessoa responsável pela execução de uma atividade.

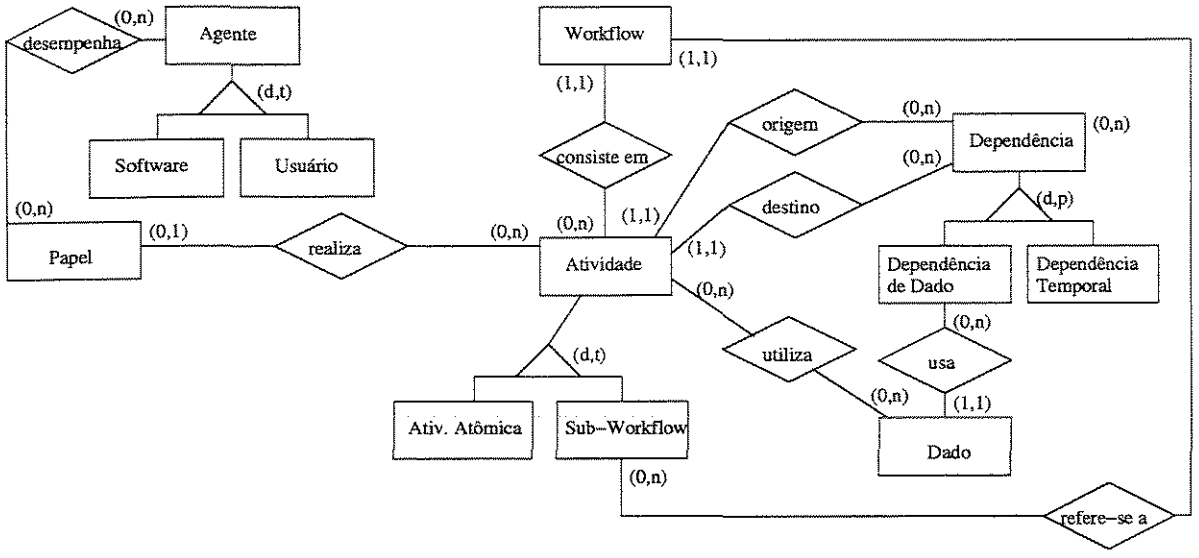


Figura 3.4: Especificação de um documento do tipo COMO

3.3 Modelo de *Design Rationale* Proposto para Documentação do PORQUE

A dissertação utiliza um modelo de DR para documentar PORQUE um plano ambiental foi feito de determinada maneira. A abordagem de DR foi escolhida para representar documentos PORQUE pois esta abordagem permite estruturar discussões, alternativas e justificativas, dentre outros fatores, surgidos ao longo de um projeto.

A análise de modelos de DR da literatura, apresentados na seção 2.4, verificou que todos eles possuem conceitos para representar questões que surgem ao longo do projeto, alternativas que respondem a estas questões e argumentos que sustentam ou opõem-se às alternativas. Destacam-se também conceitos utilizados para representar propriedades desejáveis do artefato, que devem ser endereçadas pela solução. É o caso dos Critérios de QOC e das Metas de *Proteus*.

A forma como os conceitos podem ser relacionados e principalmente o número de relacionamentos varia de um modelo para outro. Alguns relacionamentos tais como <responde>, entre questões e alternativas; e <suporta> ou <opõe-se>, entre alternativas e argumentos são bastante intuitivos. Estes relacionamentos estão presentes em todos os modelos descritos na seção 2.4 de maneira explícita ou implícita. Outros como, por exemplo, <Questiona> e <É-Sugerida-Por> de IBIS (seção 2.4.1) são menos intuitivos, tornando os esquemas gerados mais complexos. Em geral, um maior número de elementos aumenta o poder de expressão de um modelo, mas diminui a sua clareza principalmente para quem não está acostumado com o vocabulário associado.

O modelo de DR aqui proposto para a documentação de atividades de planejamento ambiental utiliza os mesmos conceitos presentes nos modelos descritos na literatura. Houve uma preocupação em organizar estes conceitos de maneira clara e simples. O objetivo é fornecer um arcabouço conceitual que possa ser facilmente utilizado por peritos de diferentes áreas envolvidos no planejamento ambiental.

Um aspecto importante em planejamento ambiental é a consideração dos riscos apresentados por determinadas alternativas de solução. Os riscos podem ser decisivos na escolha da alternativa a ser implementada. Na fase de manutenção ou análise de um plano já existente, a documentação dos riscos pode explicar porque determinada solução não deu certo. Em um documento PORQUE, os riscos poderiam ser incorporados em argumentos contrários às alternativas, mas assim teriam menor destaque e aumentaria a probabilidade deles serem esquecidos pelos projetistas. No modelo proposto, os riscos devem ser documentados para todas as alternativas de solução e não apenas para as soluções prováveis, como no *Proteus*.

A figura 3.5 ilustra o diagrama entidade-relacionamento do modelo proposto. As entidades definidas são:

- **DR**: documento formado pela agregação de questões surgidas durante a discussão de um mesmo problema, mas que não necessariamente estão interligadas;
- **Questão**: pergunta que formaliza uma discussão levantada durante o projeto;
- **Alternativa**: possível solução para determinada questão;
- **Solução**: alternativa escolhida para implementação do projeto;
- **Objetivo**: requisito que deve ser atendido pela solução;
- **Vantagem**: ponto positivo ou argumento a favor de uma alternativa;
- **Desvantagem**: ponto negativo ou argumento contra uma alternativa;
- **Risco**: risco apresentado por determinada alternativa;

- **Ação**: providência que pode ser tomada para minimizar um risco.

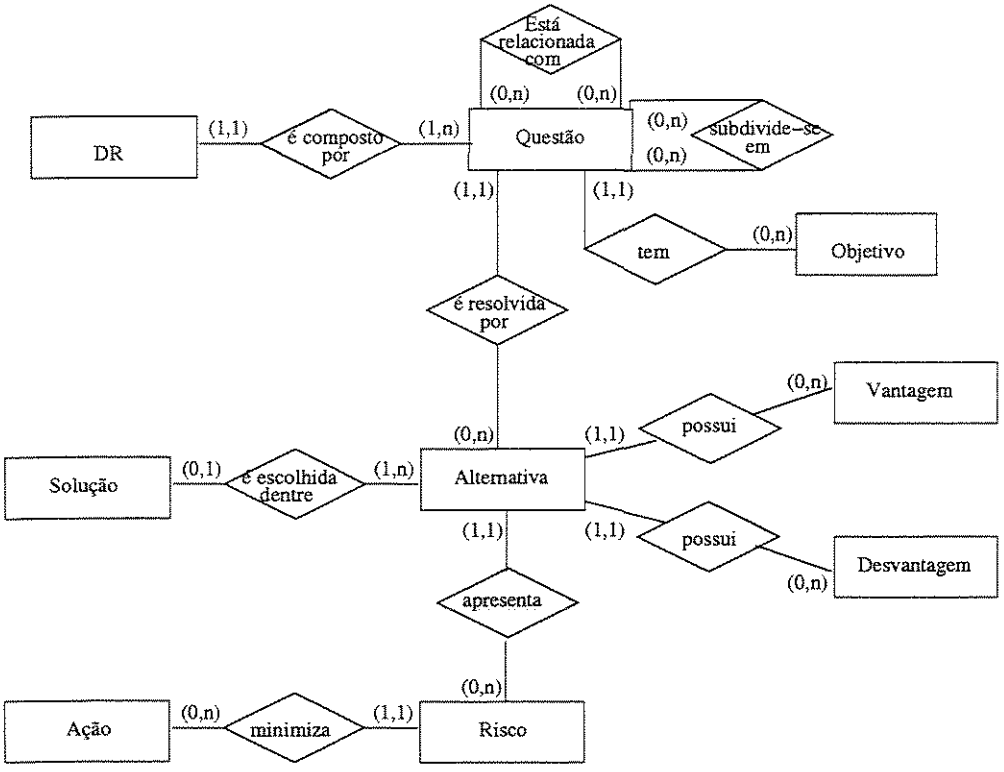


Figura 3.5: Modelo de *design rationale* para a documentação do PORQUE

Esta proposta combina a base do IBIS com parte do modelo proposto pelo *Proteus*. O objetivo é definir um conjunto básico de entidades e relacionamentos que permitam documentar e gerenciar decisões tomadas durante o planejamento ambiental. O significado de cada relacionamento pode ser facilmente compreendido através de seu nome. O relacionamento <Subdivide-se em> permite que Questões complexas possam ser resolvidas indiretamente por decomposição como em PHI (seção 2.4.2).

3.4 Integração dos Três Tipos de Documentos

Conforme descrito no capítulo 2 (seção 2.6) o planejamento ambiental é um processo complexo que pode envolver várias etapas. Cada etapa é caracterizada por um conjunto de procedimentos. Assim, pode ser necessário documentar o O QUE, COMO e PORQUE de cada etapa. A figura 3.6 integra os três tipos de documentos através da noção de documentação por etapas.

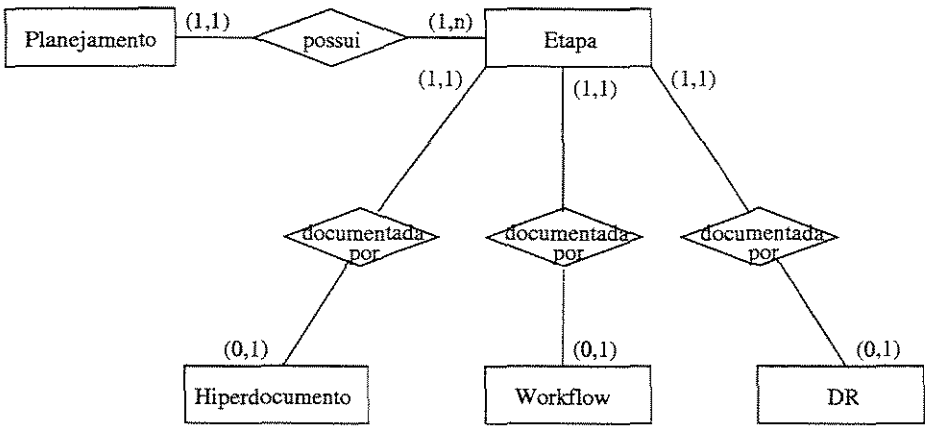


Figura 3.6: Integração dos três tipos de documentos

O diagrama da figura 3.6 apresenta duas novas entidades:

- **Planejamento**: representa uma atividade de planejamento ambiental aplicada a uma determinada região. Esta entidade deve conter um título e uma descrição que descrevem o planejamento de maneira geral;
- **Etapa**: corresponde a uma etapa do processo de planejamento. Esta entidade também deve conter um título e uma descrição que a descreve no contexto do planejamento a que pertence.

As demais entidades ilustradas no diagrama – Hiperdocumento, *Workflow* e DR – representam, respectivamente, os documentos O QUE, COMO e PORQUE especificados nas seções anteriores.

O sistema de gerenciamento de documentos deverá perguntar ao usuário se ele deseja criar uma documentação geral ou por etapas. No caso de documentação geral, o sistema deverá criar uma etapa geral e armazená-la no banco de dados. Esta etapa poderá posteriormente ser associada a outras etapas, caso o usuário deseje modificar a documentação, registrando os detalhes de etapas distintas. A interface do usuário deve também apresentar os documentos de maneira organizada, permitindo que o usuário navegue entre documentos de diferentes etapas.

3.5 Resumo

Este capítulo especificou uma estrutura para representar cada tipo de documento. Os documentos O QUE foram representados através de hipermídia, os documentos como

através de *workflows* científicos e os documentos PORQUE através de *design rationale*. A especificação foi feita tendo em vista que os documentos serão armazenados em um banco de dados relacional e que o alvo de documentação são atividades de planejamento ambiental. No entanto, essa especificação pode ser aplicada em outros contextos de documentação, principalmente no que tange ao trabalho cooperativo. Os documentos foram integrados de modo a permitir a documentação do planejamento ambiental como um todo e alternativamente suportar a documentação por etapas.

O próximo capítulo apresenta os principais aspectos de implementação desses mecanismos de documentação no sistema WOODSS.

Capítulo 4

Aspectos de Implementação

Este capítulo apresenta os principais aspectos de implementação dos mecanismos de gerenciamento de documentação de atividades de planejamento ambiental especificados no capítulo 3. A implementação foi feita estendendo o sistema WOODSS. A incorporação das novas funcionalidades implicou na reengenharia e reimplementação de parte deste software para adequá-lo às novas necessidades e torná-lo mais modular. Isto permitirá que o trabalho da dissertação possa ser estendido no futuro, aumentando assim sua aplicabilidade. A seção 4.1 apresenta uma visão geral do WOODSS e a seção 4.2 descreve os principais aspectos de reengenharia e reimplementação deste software. O esquema do banco de dados foi colocado em um apêndice, para não sobrecarregar o texto.

4.1 O Sistema WOODSS

O sistema WOODSS (*WOrkflOw-based spatial Decision Support System*) [SMRY99] é um sistema baseado em *workflows* científicos cujo objetivo é apoiar o processo decisório no domínio ambiental. De acordo com a metodologia descrita por Pires [Pir97], o processo decisório ambiental pode ser descrito como uma iteração de quatro passos:

1. Planejamento: envolve a definição dos objetivos da aplicação, da área de estudo e dos modelos a serem utilizados;

2. Inventário: consiste na determinação e coleta dos dados a utilizar;

3. Desenvolvimento: corresponde à implementação, utilizando um Sistema Espacial de Apoio à Decisão [CWP95] ou alguma ferramenta similar, dos modelos selecionados usando os dados levantados na fase de inventário;

4. Avaliação: compreende a interpretação dos resultados e a tomada de decisão.

Os três primeiros passos deste processo correspondem à geração de um conjunto de mapas. Na etapa de avaliação, é feita a análise destes mapas para a tomada de decisão e ajuste dos modelos gerados. Os mapas resultantes da execução de um modelo são usados para realizar diagnósticos ambientais e detalhar as opções a serem utilizadas para solucionar o problema.

Para apoiar a geração de soluções de planejamento ambiental, o WOODSS interage com um SIG monitorando interações dos usuários em atividades de geração de mapas e documentando-as através de *workflows* científicos. Os *workflows* científicos usados no WOODSS têm os seguintes papéis [SMRY99, Kas01]:

- Documentação do processo decisório, com isto auxiliando o planejamento participatório e futuras tomadas de decisão;
- Especificação de alto nível de modelos de simulação de processos ambientais;
- Especificação parametrizada executável de processos de decisão, que pode ser reutilizada e adaptada para novas situações similares.

Um *workflow* criado pelo WOODSS descreve uma sequência de passos utilizada para a geração de um mapa em um SIG e o fluxo de dados entre estes passos, constituindo um documento do tipo COMO. Os *workflows* são armazenados em um banco de dados. Além de criar *workflows* automaticamente a partir da captura de interações do usuário com o SIG, o WOODSS oferece suporte para que o usuário também os crie. O usuário pode criar um *workflow* completo ou recuperar documentos armazenados no banco, modificá-los e gerar novas soluções.

O WOODSS foi desenvolvido no IC-UNICAMP. O seu primeiro protótipo foi implementado como parte da dissertação de mestrado de Seffino [Sef98]. Esse primeiro protótipo era bastante simplificado e apesar de ter sido implementado na linguagem Java^{TM}, o seu projeto não fazia uso apropriado dos conceitos de orientação a objetos, dificultando sua manutenção e extensões futuras. Posteriormente, Kaster [Kas01] propôs um método de recuperação de *workflows* relevantes para a solução de um novo problema, baseado na combinação de técnicas de Inteligência Artificial e Bancos de Dados. Kaster iniciou o processo de reengenharia e reimpletação total do WOODSS, visando principalmente o uso mais apropriado de orientação a objetos. Este trabalho utilizou a linguagem Java^{TM} e o SGBD MySQL, acoplados ao SIG Idrisi [Lab, Eas01]. Ele constitui a base de implementação da proposta desta dissertação.

A arquitetura geral do sistema, proposta por Kaster [Kas01], está ilustrada na figura 4.1. Esta arquitetura divide o WOODSS em cinco módulos: interface do usuário, interface SIG, interface BD, atualizações e consultas.

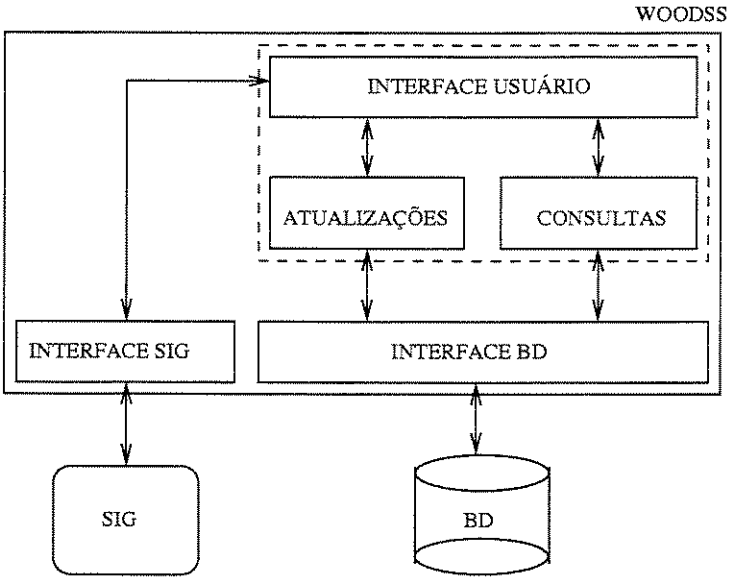


Figura 4.1: Arquitetura do WOODSS. Retirada de [Kas01].

O módulo interface SIG interage com o SIG acoplado ao WOODSS possibilitando a captura e execução de *workflows*. Na captura, as atividades realizadas pelo usuário no SIG são convertidas em *workflows*. No Idrisi as interações do usuário são gravadas em um arquivo de *log* no qual cada linha corresponde a uma função do Idrisi, ou a uma mensagem de erro ou *warning*. No primeiro caso, a linha contém um nome de função e seus parâmetros. A função é convertida para uma atividade do *workflow* e os parâmetros constituem parâmetros da atividade e/ou dados de entrada ou saída da atividade, dependendo de cada função. Na execução é feito o caminho inverso. Um *workflow* do WOODSS é convertido em um arquivo de *macro* que pode ser executado no Idrisi. Estas operações estão ilustradas nas figuras 4.2 e 4.3, respectivamente.

A interação do WOODSS com o Idrisi é feita somente através dos arquivos de *log* (captura) e de *macro* (execução). Essa interação não é completamente automática. Para a captura o usuário deve fornecer o arquivo de *log* a ser processado pelo WOODSS. O *log* contém todas as informações básicas necessárias para a geração do *workflow* (documento COMO) correspondente. Para a execução o usuário deve comandar a criação do arquivo de *macro* pelo WOODSS e em seguida executar esse arquivo no Idrisi. Mais detalhes sobre o processo de interação do WOODSS com o Idrisi podem ser encontrados em [Sef98].

O módulo interface BD recebe pedidos de consulta/atualização, os executa e devolve a resposta para a interface do usuário. O módulo de consultas é responsável pela navegação no banco de dados. O módulo de atualizações é responsável pelas atualizações do banco (inserção, modificação ou exclusão). Por último, a interface do usuário realiza

```
DISPLAY image file forestbool with symbol file QUAL256.
OVERLAY  3*v:\tmp\slopebool.rst*v:\tmp\bufferbool.rst*v:\tmp\combinedtmp.rst
DISPLAY image file combinedtmp with symbol file IDRIS256.
open layer properties of combinedtmp
OVERLAY  3*combinedtmp.rst*forestbool.rst*combined.rst
```

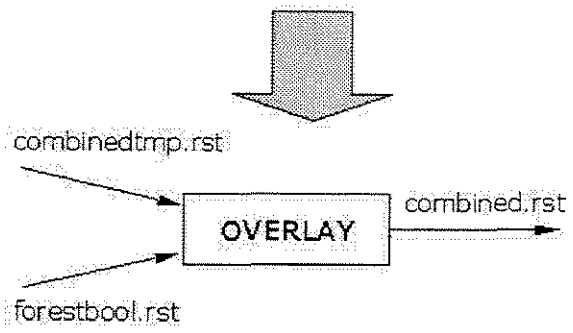


Figura 4.2: Captura de *workflow* a partir de um *log*

a comunicação dos demais módulos com os usuários do sistema.

Para esta dissertação, verificou-se que os módulos já existentes referentes à manipulação de *workflows* poderiam ser reaproveitados para a implementação do gerenciamento dos documentos COMO. Por outro lado, a associação de documentos O QUE aos *workflows* do WOODSS ajudaria na recuperação de processos relevantes para novos planejamentos e a associação do PORQUE auxiliaria na tomada de decisão. Assim, foi decidido que o sistema de gerenciamento de documentação deveria ser baseado em uma extensão do WOODSS, aumentando sua aplicabilidade e abrindo novas possibilidades de extensões futuras. Para acomodar estas novas necessidades, parte do WOODSS foi reprojetada e reimplementada. Este projeto e implementação foram feitos em conjunto com Rocha [Roc03], por constituir também a base de implementação de seu trabalho. As principais mudanças feitas são descritas na próxima seção.

4.2 Aspectos de Reengenharia e Implementação

4.2.1 Descrição Geral das Modificações

A versão do WOODSS [Kas01], usada como base para implementação da proposta desta dissertação, apresentava alguns problemas do ponto de vista de modularidade. Por exemplo, o sistema não separava a manipulação dos *workflows* em memória de sua manipulação no banco de dados. Todas as operações de atualização feitas pelo usuário eram automa-

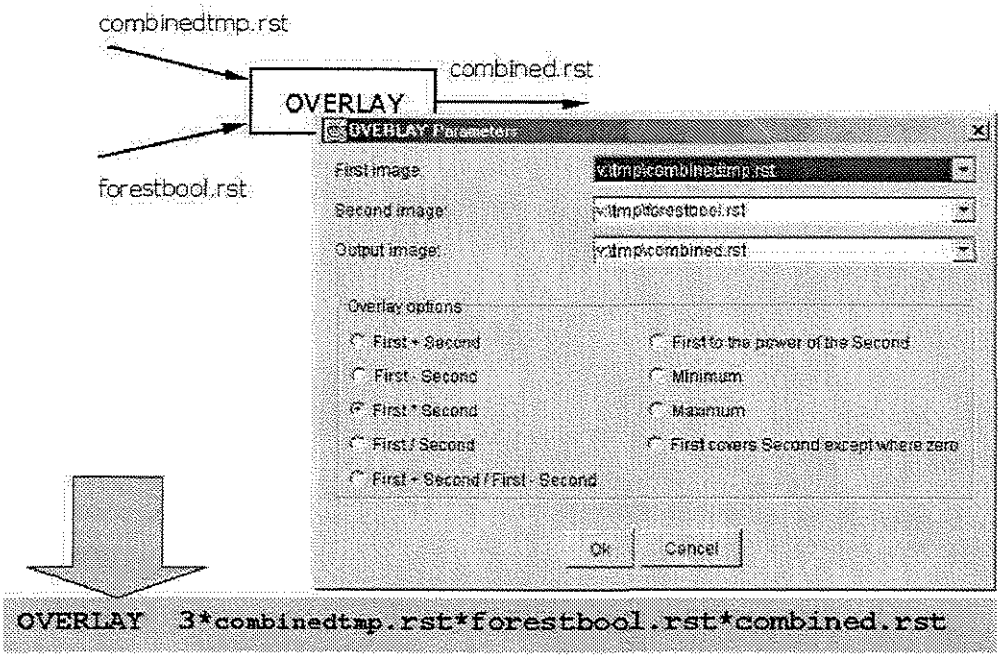


Figura 4.3: Geração de *macro* a partir de um *workflow*

ticamente salvas no banco.

A arquitetura do WOODSS foi ampliada para torná-lo mais modular e adaptá-lo à incorporação dos novos tipos de documentos. A figura 4.4 mostra a nova arquitetura proposta. Esta arquitetura foi projetada de modo a facilitar manutenções e expansões futuras. Os módulos gerenciadores são responsáveis por manter a separação lógica entre os diferentes tipos de documentos e as diferentes interfaces do sistema. O Gerenciador SIG responde pelos procedimentos envolvendo o SIG (captura e execução de *workflows*) com o qual se comunica através da Interface SIG. O Gerenciador de Documentos é responsável por manipular os diferentes tipos de documentos em memória, recebendo requisições de atualização e consulta. Ele é formado por três gerenciadores: Gerenciador de Workflows, responsável pela manipulação dos *workflows* (COMO); Gerenciador de Hipermídia, responsável pela manipulação de estruturas de hipermídia (O QUE) e Gerenciador de DR, responsável pela manipulação do *design rationale* (PORQUE). Cada gerenciador de documentos se comunica com o gerenciador BD correspondente. Os Gerenciadores BD são responsáveis pela manipulação dos documentos no nível do banco de dados. Eles recebem as requisições de serviços que envolvem interação com o banco, comunicando-se com este último através da Interface BD. Por último, a Interface do Usuário traduz os comandos do usuário em requisições de atualização e consulta, que são repassadas para os módulos de processamento dessas requisições. Desta arquitetura, apenas as partes corresponden-

tes ao gerenciamento do O QUE (hipermídia) e PORQUE (*design rationale*) não foram implementadas.

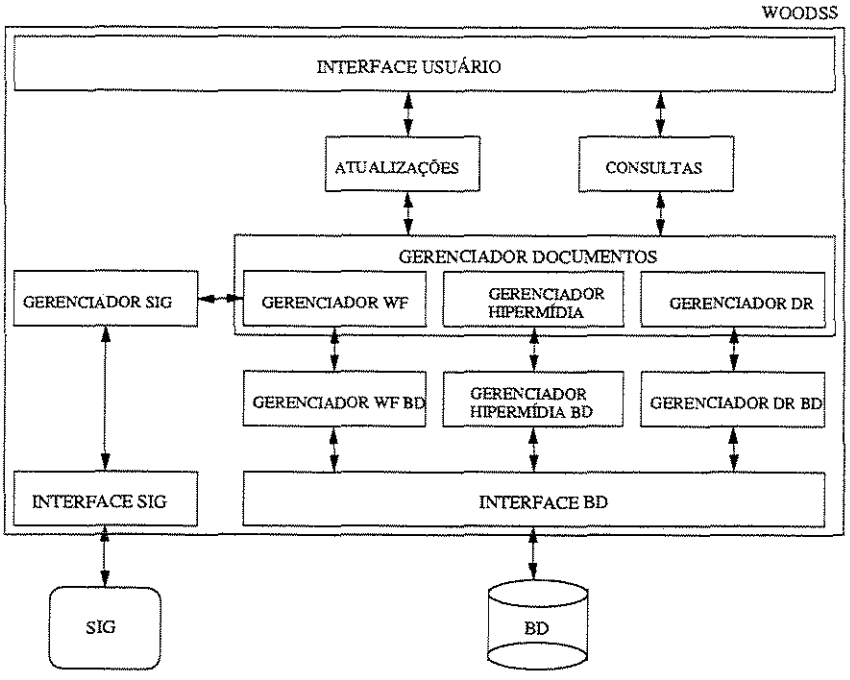


Figura 4.4: Nova arquitetura do WOODSS

A principal alteração feita no núcleo do WOODSS foi a reconsideração da estrutura utilizada para representar um *workflow*. O WOODSS representava um *workflow* de forma simplificada, considerando apenas seus componentes básicos: atividades e dependências. A figura 4.5 ilustra o diagrama entidade-relacionamento para aquela representação, obtido por aplicação de engenharia reversa no esquema relacional. A figura 4.6 ilustra as classes utilizadas para manipulá-lo. Este esquema simplificado é insuficiente para documentar o COMO, havendo sido substituído pelo esquema proposto na seção 3.2 (ilustrado na figura 3.4).

A utilização dessa nova representação de um *workflow* implicou na confecção de um novo esquema relacional para o banco de dados e na redefinição e reimplementação da maioria das classes utilizadas para manipulá-lo. A interface de usuário também foi modificada para suportar os novos elementos incorporados. A próxima seção descreve o novo diagrama de classes do WOODSS.

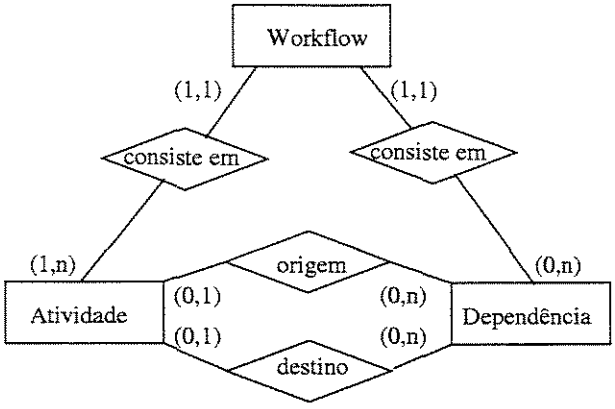


Figura 4.5: Representação de *workflow* utilizada na segunda versão do WOODSS

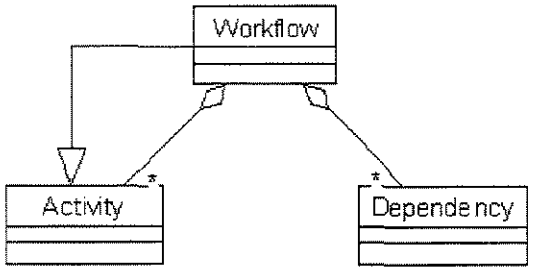


Figura 4.6: Classes implementadas para a manipulação de um *workflow* na segunda versão do WOODSS

4.2.2 Diagramas de Classes

Esta seção apresenta os novos diagramas de classes do WOODSS, em notação UML (*Unified Modeling Language*). Estes diagramas foram projetados de modo a incorporar todos os mecanismos de gerenciamento de documentação propostos no capítulo 3.

Os diagramas mostram diferentes visões do sistema, para maior clareza. A visão 1 (figura 4.7) mostra as classes da interface de usuário, interface SIG, gerenciador SIG, gerenciador de workflows, interface BD e demais classes responsáveis pela manipulação de um *workflow* (documentação COMO). Todas as classes desta visão foram implementadas. A descrição desta implementação está na seção 4.2.3.

A visão 2 (figura 4.8) mostra as classes responsáveis pela manipulação de hipermídia (documentação O QUE). A visão 3 (figura 4.9) mostra as classes responsáveis pela manipulação de *design rationale* (documentação PORQUE). A visão 4 (figura 4.10) mostra as classes responsáveis pela integração dos três tipos de documentos. As classes destas três

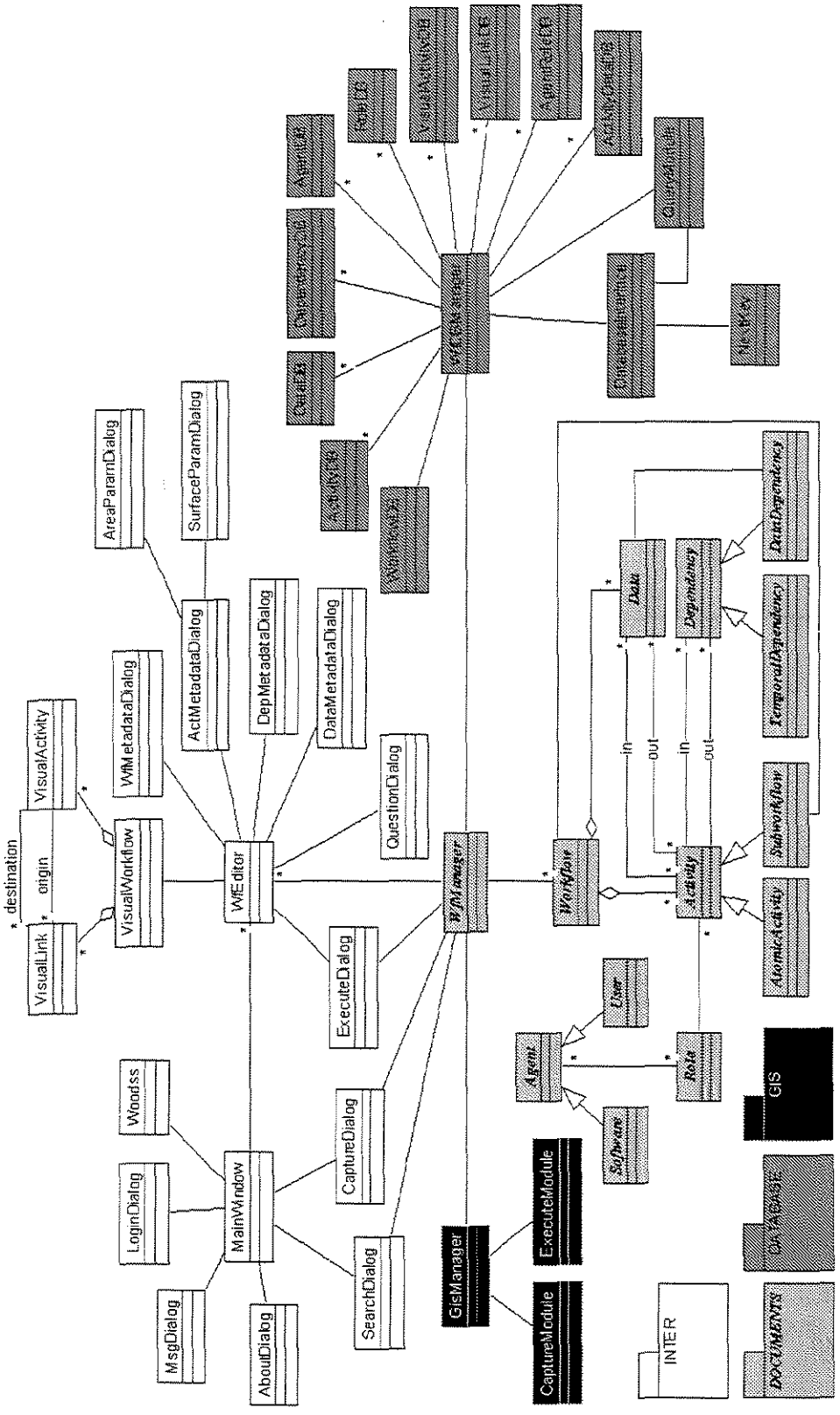


Figura 4.7: Visão 1: Interface Usuário, Interface SIG, Gerenciador SIG, Gerenciador Wf, Interface BD e manipulação de um *workflow*

visões não foram implementadas. Os diagramas indicam uma sugestão inicial para uma implementação futura.

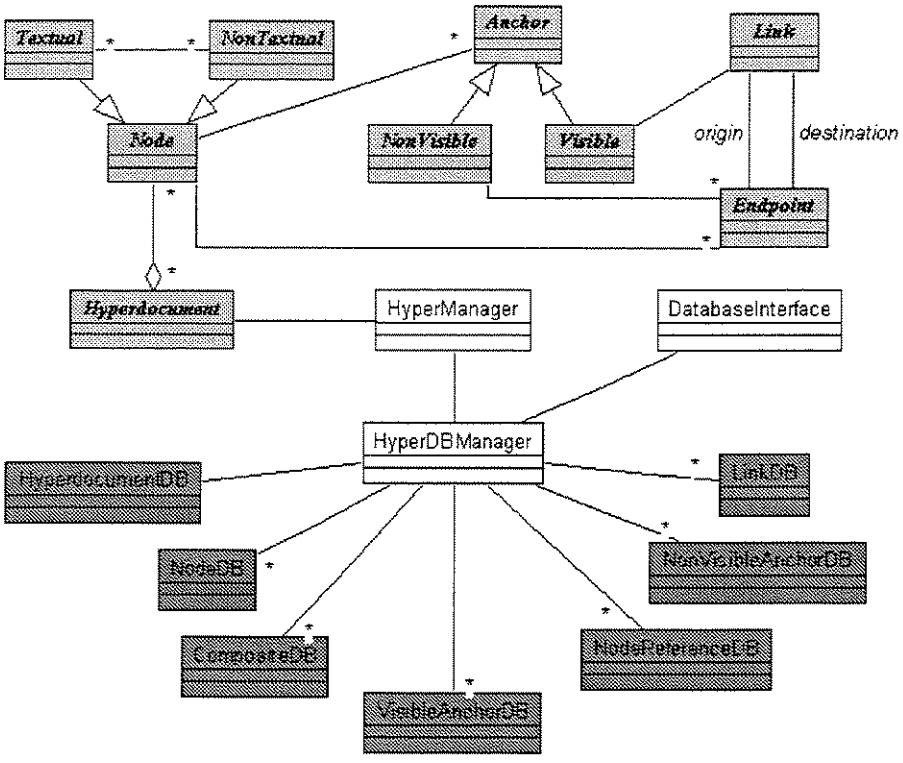


Figura 4.8: Visão 2: Manipulação de um hiperdocumento (O QUE)

As classes que aparecem repetidas nas diferentes visões estabelecem pontos de ligação entre elas. Nos diagramas, quando a cardinalidade de um relacionamento é omitida, representa o valor '1', e o símbolo '*' indica a cardinalidade 'n'.

A figura 4.7 está dividida em cores segundo implementação dos pacotes Java (canto inferior esquerdo). O usuário começa a interagir com o sistema através da janela MainWindow, que por sua vez ativa outras funções – de consulta (por exemplo, SearchDialog); de interação com o SIG (por exemplo, CaptureDialog); de gerenciamento de *workflows*, através da classe WfEditor. Esta última, por sua vez, gerencia todas as atividades de criação, eliminação e modificação de *workflows* e seus componentes.

Na figura 4.7, a classes WfManager e WfDBManager correspondem ao gerenciamento de *workflows* em memória e no banco de dados, respectivamente. Ressalte-se que este projeto de classes corresponde à especificação, na arquitetura da figura 4.4, da separação entre o gerenciamento de dados em memória e no banco de dados. Essa separação foi feita para maior modularização de modo a facilitar expansões e manutenções futuras. As

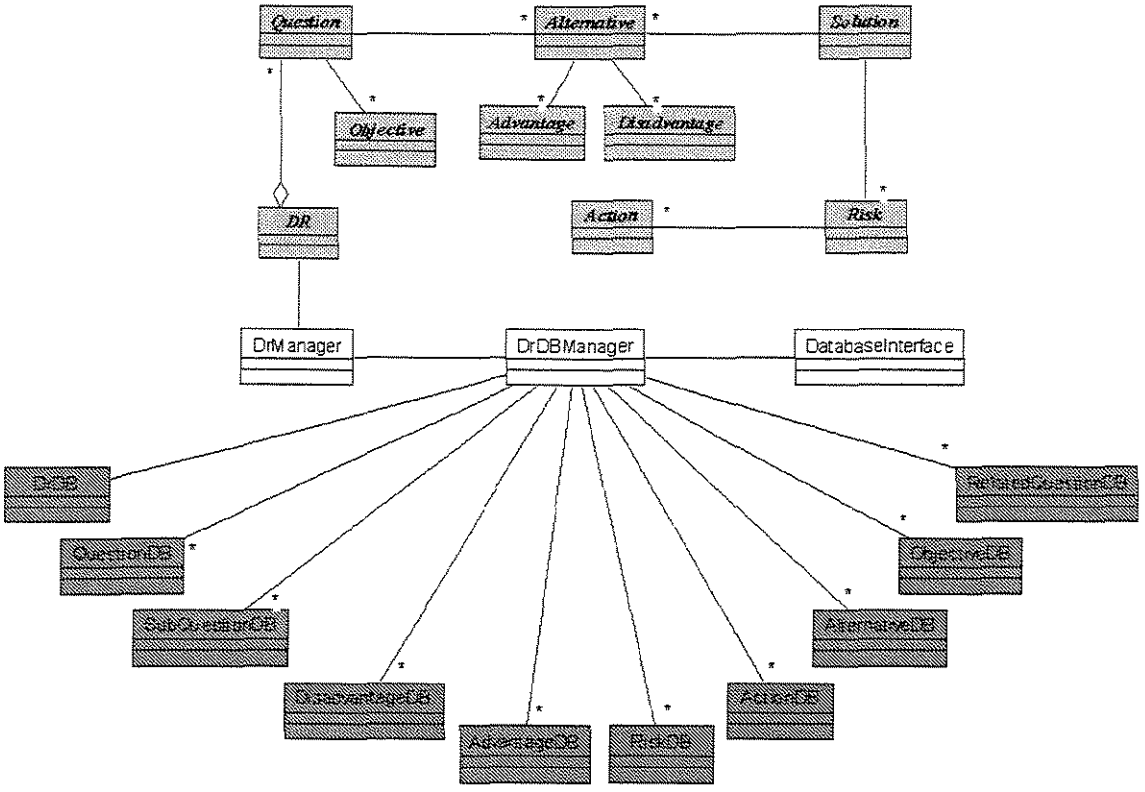


Figura 4.9: Visão 3: Manipulação do *design rationale* (PORQUE)

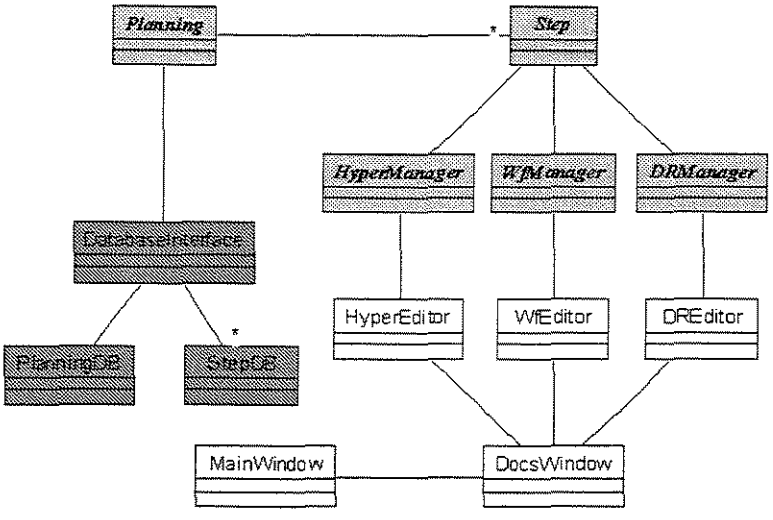


Figura 4.10: Visão 4: Integração dos documentos

demais classes em sombreado mais claro correspondem à especificação orientada a objetos do diagrama ER do capitulo 3 (figura 3.4), descrevendo um *workflow* em memória. As classes cujos nomes têm sufixo DB (classes em sombreado mais escuro ligadas à classe WfDBManager) correspondem a operações no banco de dados, de novo conforme o diagrama ER do capítulo 3 (figura 3.4). Estas classes executam operações de manutenção (inserção, atualização e exclusão) nas tabelas relacionais que armazenam os componentes de um *workflow* no banco de dados. Foi definida uma classe para cada relação de modo que uma relação é manipulada somente por sua classe correspondente. O principal benefício de se ter uma classe para cada relação é a localidade de mudança. Assim se, por exemplo, o tipo de um atributo de uma relação mudar somente a classe responsável por essa relação deverá ser alterada. Embora não ilustrado na figura 4.7, para não sobrecarregá-la, todas as classes DB se comunicam diretamente com a classe DatabaseInterface. Esta classe estabelece uma ponte ente o WOODSS e o SGBD.

A figura 4.8 mostra a proposta para implementação de documentos O QUE. De forma semelhante ao já descrito no parágrafo anterior para a figura 4.7, ela está dividida em :

- Classes em sombreado mais claro: definição orientada a objetos de documentos hipermídia em memória (vide especificação na figura 3.1 do capítulo 3);

- Classes HyperManager e HyperDBManager: separam gerenciamento de documentos hipermídia em memória e no banco de dados;

- Classes em sombreado mais escuro (nomes têm sufixo DB): correspondem a operações de manipulação, no banco de dados, das tabelas que irão armazenar documentos hipermídia. Todas essas classes se comunicam diretamente com a classe DatabaseInterface.

A figura 4.9 mostra a proposta para implementação de documentos PORQUE. Esta figura deve ser interpretada de forma análoga à figura 4.8.

Finalmente a figura 4.10 mostra a proposta de integração dos três tipos de documentos a qual é baseada na noção de documentação por etapas. A classe Planning representa um plano que possui várias etapas. A classe Step representa uma etapa e integra, em memória, os documentos associados a ela. As classes PlanningDB e StepDB são responsáveis pelas operações de manutenção nas tabelas relacionais correspondentes a plano e etapa, respectivamente. A classe DocsWindow é responsável pela apresentação integrada dos três tipos de documentos ao usuário, sendo que a apresentação e edição de um hiperdocumento, *workflow* e *design rationale* são gerenciadas pelas classes HyperEditor, WfEditor e DrEditor respectivamente.

4.2.3 Descrição dos Pacotes Java

A linguagem Java permite agrupar classes implementadas em pacotes, de modo a garantir maior modularização e legibilidade do código. Esta nova versão do WOODSS preserva a divisão em pacotes Java proposta em [Kas01] mas houve recodificação de grande parte do sistema. Como pode ser observado na figura 4.7 as classes são agrupadas em quatro pacotes: DATABASE, GIS, INTER e DOCUMENTS (este pacote era chamado de CORE na versão anterior do WOODSS). Estes pacotes serão descritos a seguir, destacando as principais mudanças feitas com relação à versão anterior do WOODSS.

Pacote DATABASE. Este é o pacote responsável pela comunicação do sistema com o banco de dados que armazena os documentos. Este pacote encapsula os módulos de atualizações, consultas, interface BD e gerenciadores BD. Além das classes em sombreado mais escuro na figura 4.7, fazem parte deste pacote as classes em sombreado mais escuro (nomes têm sufixo DB) nas figuras 4.8, 4.9 e 4.10 e as classes HyperDBManager e DrDBManager (figuras 4.8 e 4.9, respectivamente). A classe DatabaseInterface estabelece uma ponte entre o WOODSS e o SBGD, comunicando-se via JDBC (*Java Database Connectivity*), uma API Java padrão que fornece primitivas para execução de comandos SQL em um SGBD. A classe QueryModule implementa mecanismos de busca de *workflows*. A classe denominada NextKey foi criada para permitir a definição de identificadores sequenciais em relações, usados para gerar novas tuplas. Isto foi necessário porque a função de auto-incremento do MySQL não é totalmente confiável.

Pacote GIS. O pacote GIS realiza a interação do WOODSS com o SIG acoplado, traduzindo operações realizadas pelo usuário no SIG em *workflows* (captura) e vice-versa (execução). A classe CaptureModule é responsável pela captura da interação do usuário com o SIG. Ela implementa um conjunto de métodos necessários para ler um arquivo de *log* do Idrisi e gerar o *workflow* científico correspondente. A classe ExecuteModule gera um arquivo de *macro* que reflete a estrutura do *workflow* que se deseja executar, ou seja, gera um conjunto de comandos executáveis a partir de um *workflow*. A classe GisManager foi acrescentada para maior modularização de modo a facilitar a futura integração do WOODSS com outros SIGs. Esta versão do WOODSS interage somente com o SIG Idrisi. Para acoplamento a outros SIGs, será necessário acrescentar novos métodos à classe GisManager, ou especializá-la.

Pacote INTER. Este pacote é composto pelas classes da interface com o usuário do sistema. Estas classes têm duas funções: apresentar os *workflows* e dados e documentos associados; e receber e responder a eventos do usuário. A classe MainWindow representa a janela principal do sistema e estabelece o ponto de ligação entre os diferentes módulos.

Todas as tarefas de apresentação e edição de documentos são gerenciadas pelas classes WfEditor (responsável pela edição de *workflows*), HyperEditor (responsável pela edição de estruturas de hipermídia) e DrEditor (responsável pela edição do *design rationale*). Essas classes são agregadas pela classe DocsEditor, responsável pela apresentação integrada dos três documentos. As classes HyperEditor, DrEditor e DocsEditor (figura 4.10) não foram implementadas tendo em vista que esta versão do WOODSS implementa apenas o gerenciamento de *workflows*.

A classe WfEditor possui um conjunto de subjanelas associadas necessárias para a manipulação de um *workflow*. As classes SearchDialog, CaptureDialog e ExecuteDialog gerenciam, respectivamente, as solicitações de consulta ao banco de dados, captura e execução de *workflows*. A classe Woodss é a classe de inicialização do sistema. Ela realiza alguns testes iniciais e mostra a janela principal do WOODSS.

A nova versão implementada tem uma série de novas classes de interface, por exemplo, as classes das janelas de edição de parâmetros de atividades (AreaDialog e SurfaceDialog na figura 4.7). As janelas de edição de parâmetros representam os parâmetros de cada função do Idrisi que o WOODSS pode capturar. A figura 4.11 ilustra a edição dos parâmetros da função OVERLAY.

Nesta nova implementação do WOODSS, uma atividade cujos parâmetros não foram informados é dita "incompleta" e é exibida no grafo do *workflow* com uma cor diferente do resto do diagrama (atualmente vermelho). Se um *workflow* possuir atividades incompletas a *macro* para sua execução não poderá ser gerada.

Trabalhos futuros de implementação dos documentos O QUE (hipermídia) e PORQUE (*design rationale*) deverão associar novos conjuntos de funções às classes HyperEditor e DrEditor para permitir a manipulação dos respectivos documentos. De novo, este projeto de software está mais modular, permitindo incrementar paulatinamente as funções do sistema.

Pacote DOCUMENTS. Este pacote visa o gerenciamento dos documentos em memória. Ele foi totalmente reestruturado e recodificado. As classes deste pacote são responsáveis por manter em memória os documentos buscados no banco de dados e por definir novos documentos a serem armazenados no banco de dados.

O pacote é formado pelas classes em sombreado mais claro nas figuras 4.7, 4.8, 4.9 e 4.10 e pelas classes HyperManager e DrManager (figuras 4.8 e 4.9, respectivamente). As classes WfManager, HyperManager e DrManager gerenciam os documentos O QUE, COMO e PORQUE, respectivamente.

Na figura 4.7, a classe Workflow encapsula as funcionalidades e componentes de um *workflow* (COMO). A classe Activity representa uma atividade, a classe Dependency representa uma dependência e assim por diante, com cada classe representando uma

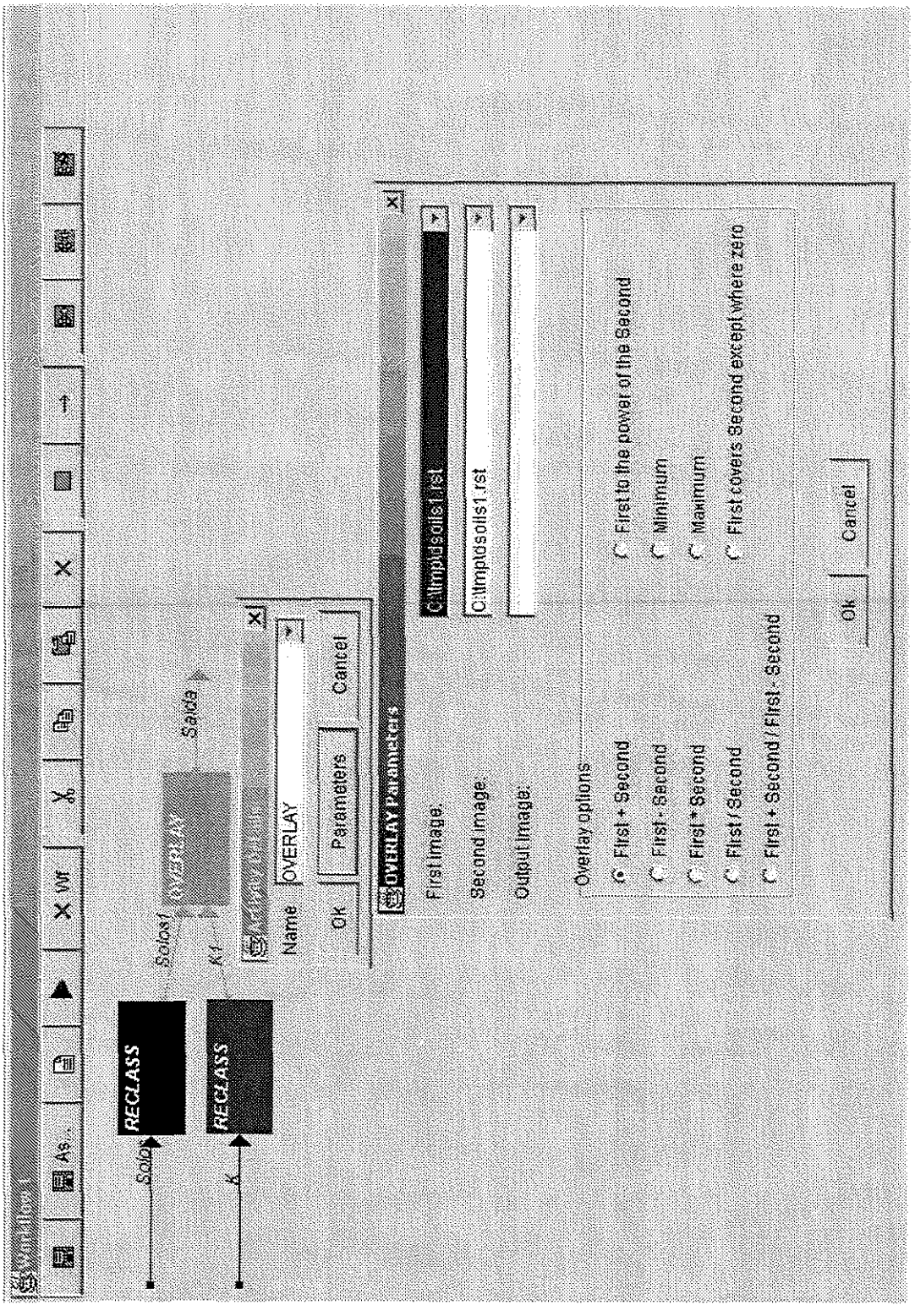


Figura 4.11: Edição dos parâmetros de uma atividade no WOODSS

entidade definida no esquema conceitual de um *workflow*, ilustrado na figura 3.4. A mesma idéia foi utilizada para os demais tipos de documentos (figuras 4.8 e 4.9).

4.3 Resumo

Este capítulo apresentou uma visão geral do sistema WOODSS e em seguida destacou os principais aspectos de reengenharia e reimplementação deste sistema, para torná-lo mais modular e adaptá-lo à incorporação dos novos mecanismos de gerenciamento de documentação. A nova versão do WOODSS gerencia os documentos do tipo COMO, representados através de *workflows* científicos. O capítulo descreveu o projeto de um conjunto de classes que direciona a implementação futura dos documentos O QUE e PORQUE representados respectivamente por estruturas de hipermídia e *design rationale*.

O próximo capítulo mostra a aplicação dos mecanismos de gerenciamento de documentação propostos nesta dissertação em um problema de planejamento agro-ambiental real.

Capítulo 5

Exemplo de Documentação de um Problema de Planejamento Ambiental

Este capítulo mostra a aplicação dos mecanismos de gerenciamento de documentação propostos na dissertação em um problema de planejamento agro-ambiental. Este exemplo de documentação contribui para a validação da proposta da dissertação. A seção 5.1 descreve o problema e a solução adotada com o apoio do SIG Idrisi. As seções 5.2, 5.3 e 5.4 apresentam, respectivamente, os documentos O QUE, COMO e PORQUE relativos a este problema.

5.1 Descrição do Problema e Solução

O problema estudado é voltado ao planejamento agro-ambiental e tem como objetivo a confecção de um mapa de aptidão agrícola. Este é um problema comum em planejamento, que visa o aproveitamento das melhores terras de uma determinada região para o desenvolvimento de atividades agrícolas, levando em conta a necessidade de preservação do meio ambiente. A área de estudo é a microbacia de Iracemápolis localizada no município de Iracemápolis–SP.

Trata-se de um problema de decisão que deve considerar um conjunto de critérios para se obter uma solução. A partir de discussões, os especialistas decidiram que os seguintes critérios deveriam ser considerados para a solução do problema:

- O mapa de aptidão agrícola deve restringir-se às áreas que são ocupadas por **pastagem** e **cultura**;
- Para a escolha das melhores terras os seguintes fatores devem ser considerados:

- a **proximidade da água**, visando irrigação;
- a **classificação das terras** pelo sistema de capacidade de uso; e
- a **declividade**. Uma vez que esse fator já foi considerado na classificação das terras, seu peso/influência como fator não deve ser muito grande.

Com isso, as áreas que devem ser consideradas neste problema são as que possuem alguma atividade agrícola (pastagem e cultura) e as melhores áreas são aquelas localizadas mais próximas da água, que possuem as melhores classificações no sistema de capacidade de uso e que possuem os menores declives.

Os mapas iniciais disponíveis para a solução do problema são: mapa de uso da terra (*uso96*), mapa de capacidade de uso (*capacov*) e mapa de declividades (*declive*). Estes mapas são ilustrados nas figuras 5.1, 5.2 e 5.3, respectivamente.

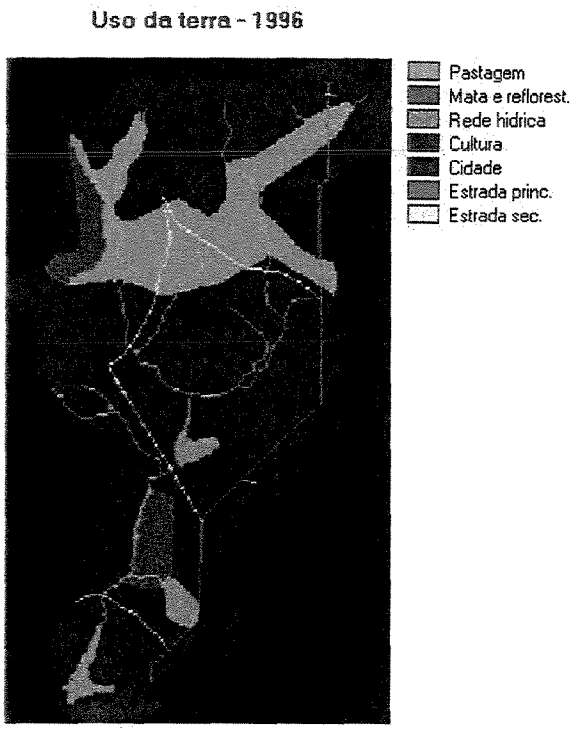


Figura 5.1: Microbacia de Iracemápolis – mapa de uso da terra

O mapa de uso da terra (figura 5.1) mostra que a região de estudo possui áreas de pastagem, mata e reflorestamento, rede hídrica, cultura, cidade, estrada principal e estrada secundária.

O mapa de capacidade de uso (figura 5.2) classifica a região em oito classes. As terras mais aptas à agricultura possuem as menores classes, segundo o sistema de classificação

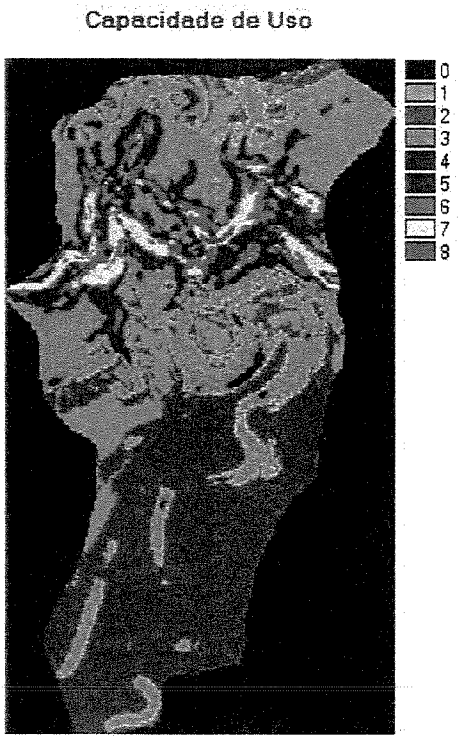


Figura 5.2: Microbacia de Iracemápolis – mapa de capacidade de uso

utilizado. Duas informações são essenciais para a classificação de uma área quanto à capacidade de uso: os tipos de solos e as classes de declividade.

O mapa de declividades (figura 5.3) mostra as escalas de declividade da área. Os maiores declives apresentam os maiores valores da escala.

Este problema pode ser resolvido com o apoio do SIG Idrisi através do seguinte procedimento:

1. Confeccionar o mapa de restrições (*terres*), selecionando somente áreas ocupadas por pastagem e cultura;

   (a) Atribuir o valor 1 às áreas ocupadas por pastagem e cultura no mapa de uso da terra (*uso96*), através da função ASSIGN, produzindo o mapa *terres*.

2. Confeccionar o mapa relativo ao fator proximidade da água (*fatagua*);

   (a) Atribuir o valor 1 às áreas classificadas como rede hídrica no mapa *uso96*, através da função ASSIGN, produzindo o mapa *redehidr*;

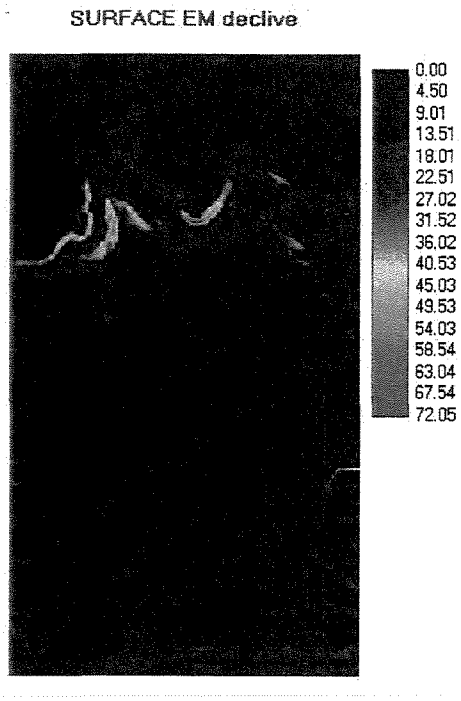


Figura 5.3: Microbacia de Iracemápolis – mapa de declividades

(b) Produzir a imagem de distância para o mapa *redehidr*, usando a função DISTANCE. Este comando calcula a distância a partir de um grupo de pixels alvo, que neste caso é a rede hídrica. Denomine a imagem de saída de *tmpagua*;

(c) Padronizar a escala de valores do mapa *tmpagua* para o tipo linear, escala de 0 a 255 (256 níveis), usando o comando STRETCH. O resultado é o mapa *tmpagstr*;

(d) Inverter a ordem de valores do mapa *tmpagstr*, pois os valores aumentam à medida que aumenta a distância em relação a rede hídrica e a melhores terras para uso agrícola são as mais próximas da água. Como não existe uma imagem de inversão, esta deverá ser criada através da função INITIAL. A imagem de inversão (*invert*) deve utilizar os mesmos parâmetros da imagem *uso96* e deve ter tipo de dado byte, tipo de arquivo binário e valor inicial 255. Utilize a função OVERLAY para subtrair o mapa *invert* do mapa *tmpagstr*, produzindo mapa *fatagua* (fator proximidade da água).

3. Confeccionar o mapa relativo ao fator declividade (*fatdecl*);

(a) Padronizar o mapa de declividades (*declive*), utilizando a função STRETCH,

tipo linear, escala de 0 a 255, produzindo o mapa *tmpdecl*;

(b) Inverter a ordem de valores da imagem *tmpdecl*, pois os maiores declives apresentam o maiores valores da escala e as áreas de interesse devem ter os menores declives. Como a imagem de inversão já foi gerada para o fator anterior, basta usar a função OVERLAY para subtrair as imagens *invert* e tmpdecl. O resultado é o mapa *fatdecl* (fator declividade).

4. Confeccionar o mapa relativo ao fator classificação das terras (*fatsolo*);

   (a) Classificar o mapa de capacidade de uso (*capacov*) de modo que as classes mais aptas possuam os mais altos valores de classificação. As oito classes do mapa *capacov* estão classificadas de maneira inversa. A inversão de valores pode ser feita atribuindo valores de 8 – 1 aos valores de 1 – 8 da imagem *capacov*, através da função ASSIGN. O resultado é o mapa *tmpsolo*;

   (b) Padronizar os valores de *tmpsolo*, usando a função STRETCH, tipo linear, escala de 0 a 255, produzindo o mapa *fatsolo* (fator classificação das terras).

5. Atribuir pesos relativos aos fatores envolvidos. No Idrisi, este passo é apoiado pela função WEIGHT que recebe como entrada uma matriz de comparação de pares de fatores e produz um relatório de pesos;

6. Avaliar os múltiplos critérios usando a função MCE. Fornecer como restrição a imagem *terres* e como fatores as imagens *fatagua*, *fatsolo* e *fatdecl*. Os pesos de cada fator são os apresentados no relatório gerado no passo anterior pela aplicação da função WEIGHT. O resultado deste passo é o mapa *agriapt* (aptidão agrícola) que constitui a solução do problema.

O mapa de aptidão agrícola está ilustrado na figura 5.4. As melhores áreas para a agricultura são as que apresentam os maiores valores na escala de valores (áreas em tom de verde). As áreas em cor preta não podem ser utilizadas.

A figura 5.5 ilustra o *workflow* correspondente ao procedimento anterior, gerado pelo sistema WOODSS a partir do *log* do Idrisi. Neste *workflow* (documento COMO), as atividades são funções do Idrisi e os dados são os arquivos processados por essas funções. As seções seguintes mostram a documentação que deveria ser associada ao procedimento.

5.2 Documentação O QUE

A figura 5.6 mostra parte da documentação O QUE correspondente ao problema, representada por uma rede hipermídia de nós e ligações. Note que o nó principal do documento

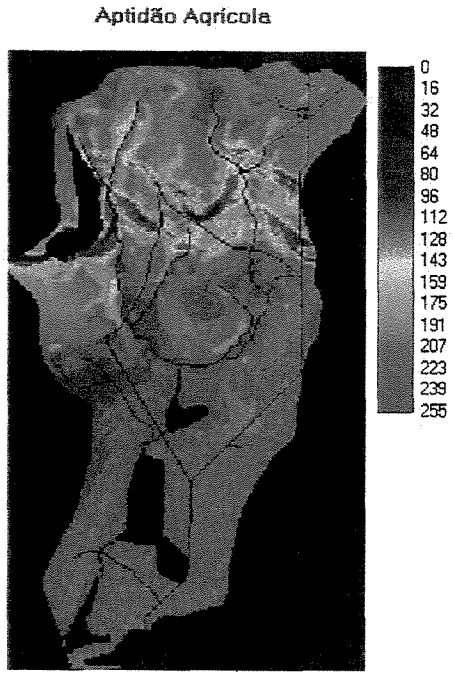


Figura 5.4: Microbacia de Iracemápolis – mapa de aptidão agrícola

(canto superior esquerdo) descreve o problema geral. A seguir, este nó está ligado a outro nó que descreve textualmente a metodologia utilizada para resolver o problema. Este segundo nó contém três âncoras visíveis:

- *próximas da água*, que aponta para outro nó que descreve como foi feito o cálculo de distâncias a partir da rede hídrica;
- *classificações no sistema de capacidade de uso*, que aponta para um nó descrevendo a classificação das terras quanto à capacidade de uso;
- *menores declives*, que aponta para um nó onde o usuário terá colocado, por exemplo, uma descrição textual indicando como foi feito o cálculo de declividades.

5.3 Documentação COMO

A figura 5.7 mostra o documento COMO correspondente ao problema. Este documento representa o procedimento utilizado para resolver o problema a partir de um conjunto de mapas iniciais. O processo de obtenção desses mapas não foi documentado. Na figura 5.7 cada atividade do *workflow* está rotulada com o passo do procedimento, descrito na seção

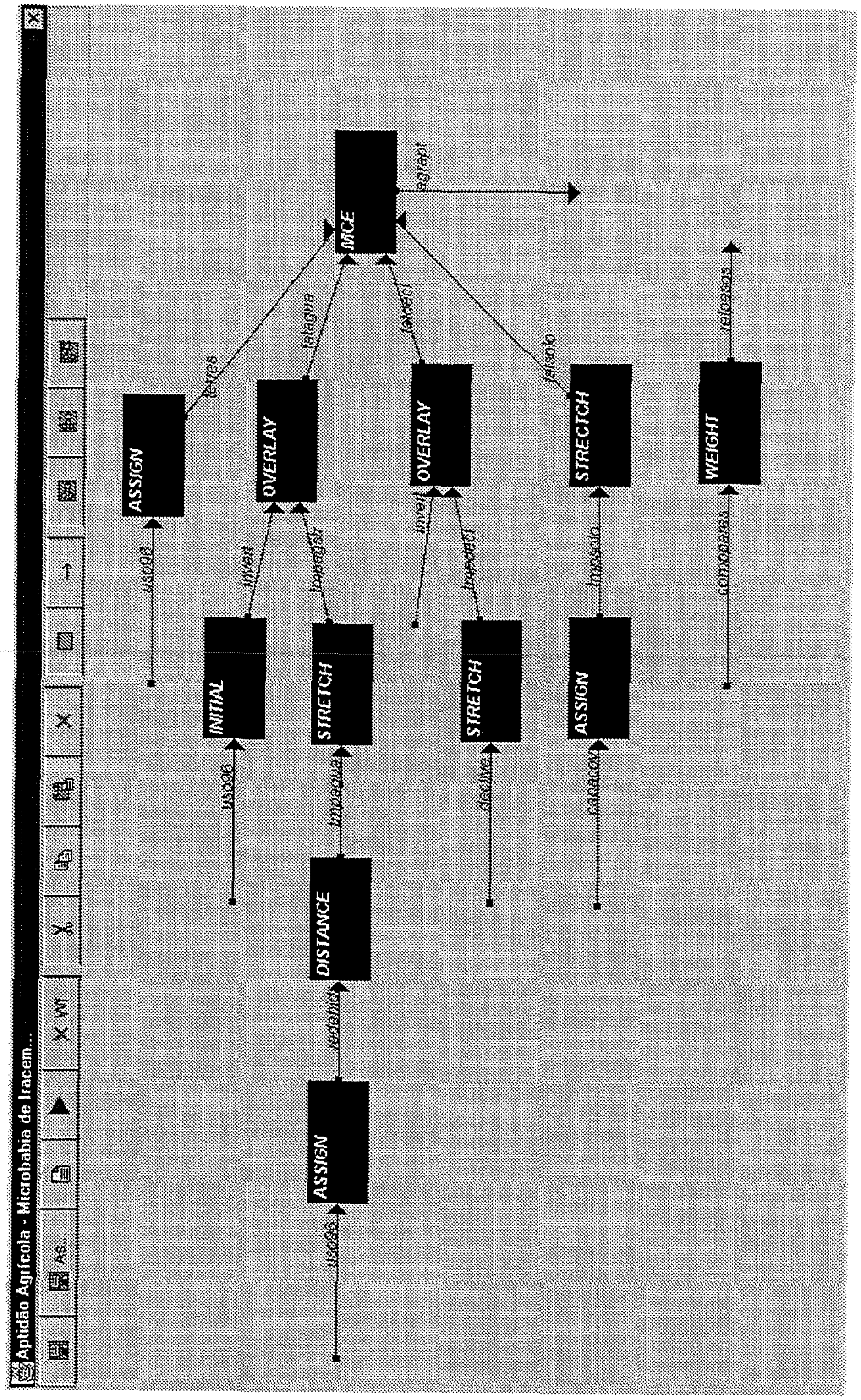


Figura 5.5: WOODSS – *workflow* para documentação do problema de aptidão agrícola da microbacia de Iracemápolis

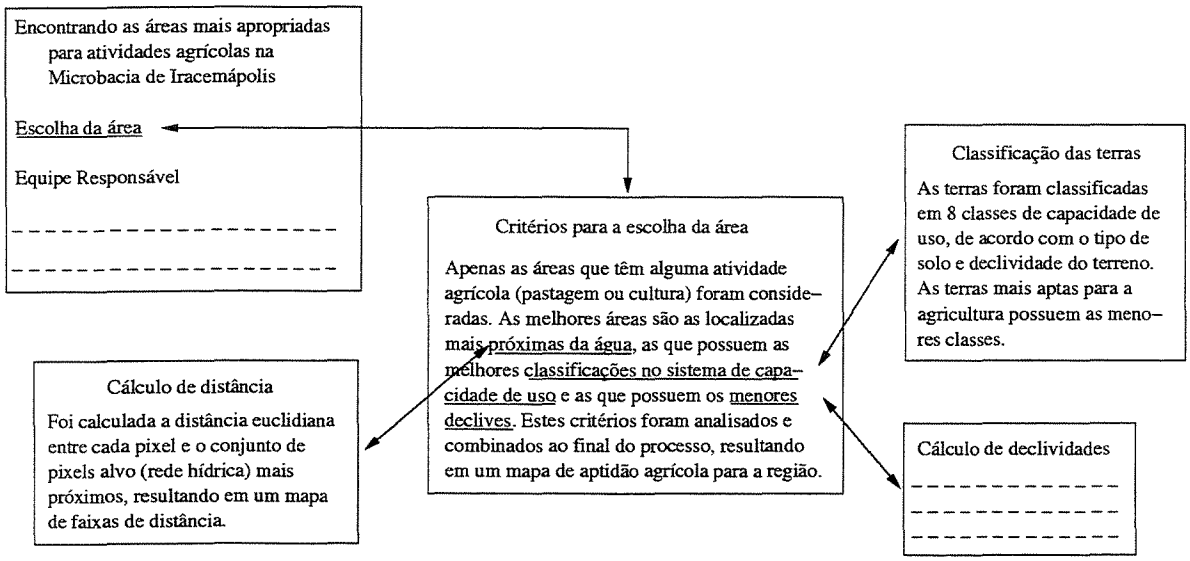


Figura 5.6: Exemplo de documentação O QUE para a confecção de um mapa de aptidão agrícola para a microbacia de Iracemápolis

5.1, a que corresponde. Os rótulos não fazem parte do documento. Eles foram colocados apenas para destacar a correspondência entre o *workflow* e o procedimento.

Note que o *workflow* da figura 5.7 é idêntico ao *workflow* gerado pelo WOODSS (figura 5.5) exceto que cada componente (atividades, dados e dependências) está anotado textualmente com indicações que facilitam o entendimento do COMO. Na verdade, o WOODSS permite que o usuário edite os *workflows* por ele gerados e documente essas anotações.

Assim, por exemplo, a cadeia mais longa de atividades de *workflow* (terceira de cima para baixo na figura 5.7) indica que:

- O dado de entrada é o mapa de uso da terra;
- A primeira atividade (ASSIGN no *workflow* gerado pelo WOODSS, figura 5.5) separa a rede hídrica dos demais dados do mapa de uso da terra. O mapa da rede hídrica é então passado à etapa seguinte;
- A segunda atividade (DISTANCE no Idrisi – veja figura 5.5) tem por objetivo calcular faixas de distância de cada ponto da região em relação à rede hídrica;
- A terceira atividade (STRETCH no Idrisi – veja figura 5.5) visa padronizar escalas de valores do mapa de entrada (mapa de distâncias da rede hídrica) para permitir a comparação posterior de todos os fatores que estão sendo considerados.

Os demais componentes do documento podem ser interpretados de forma semelhante.

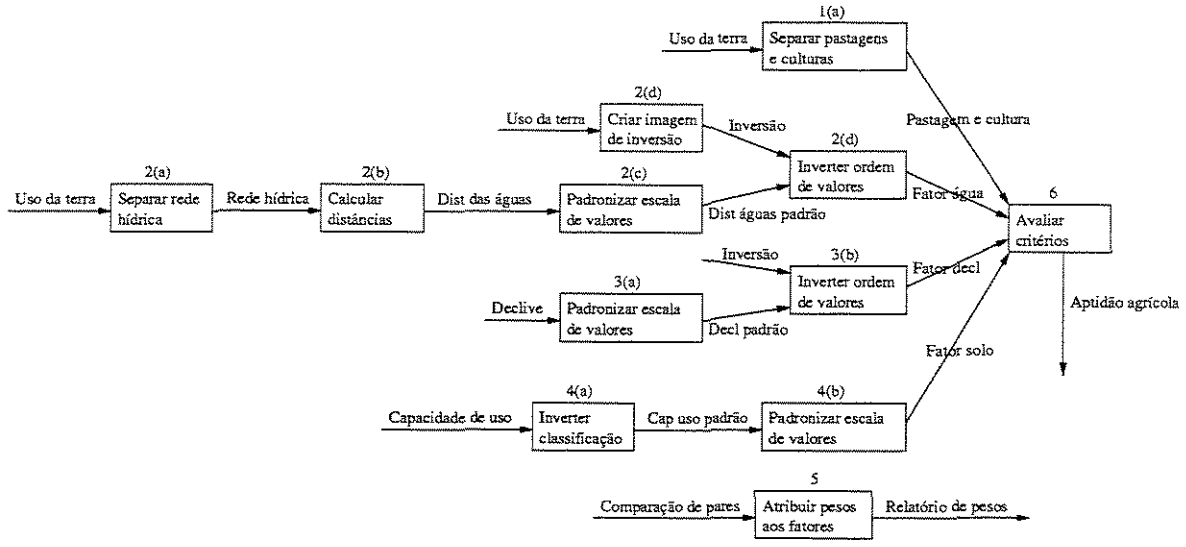


Figura 5.7: Exemplo de documentação COMO para a confecção de um mapa de aptidão agrícola para a microbacia de Iracemápolis.
Os rótulos acima de cada atividade não fazem parte do documento e foram colocados para destacar a correspondência do *workflow* com o procedimento proposto para resolver o problema (seção 5.1).

5.4 Documentação PORQUE

A figura 5.8 mostra uma parte da documentação PORQUE associada ao problema. Este documento mostra as discussões e decisões relacionadas à escolha das restrições e fatores a serem considerados na solução do problema de decisão – encontrar áreas mais aptas para a agricultura. O documento é exibido como um grafo direcionado. As letras em maíusculo indicam os elementos do modelo de *design rationale* proposto no capítulo 3. (**Q** = questão, **SQ** = subquestão, **O** = objetivo, **A** = alternativa, **V** = vantagem, **D** = desvantagem e **R** = risco). As alternativas envolvidas por caixas são as escolhidas para a solução da questão a que se referem.

A discussão começa com uma questão geral:

- Como classificar as áreas de acordo com a aptidão agrícola?

Esta questão tem como objetivo *encontrar as melhores áreas para a agricultura.* Trata-se de uma questão complexa, que é dividida em duas subquestões:

- Quais áreas podem ser utilizadas para a agricultura?
- Quais fatores devem ser considerados para determinar as melhores áreas?

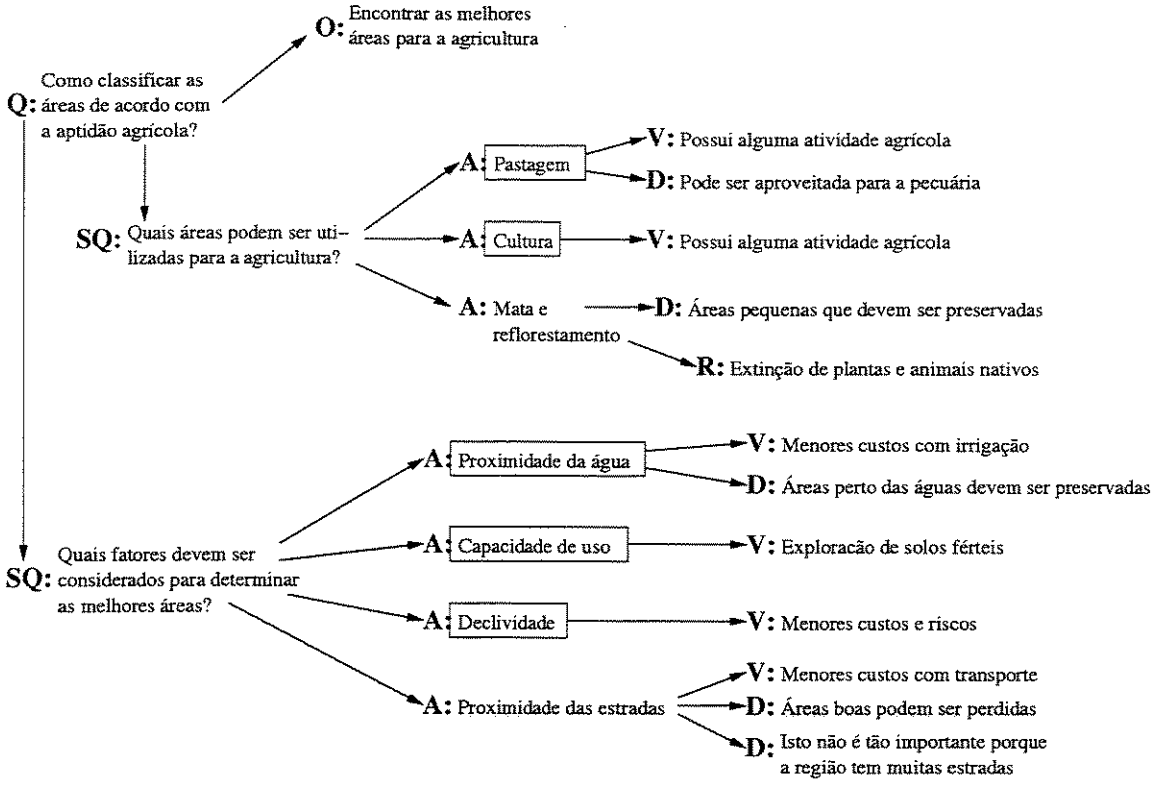


Figura 5.8: Exemplo de documentação PORQUE para a confecção de um mapa de aptidão agrícola para a microbacia de Iracemápolis

A primeira subquestão pode ser respondida por três alternativas:

- Pastagem. Esta alternativa tem uma vantagem – *possui alguma atividade agrícola*, e uma desvantagem – *pode ser aproveitada para a pecuária*;
- Cultura. Esta alternativa tem uma vantagem – *possui alguma atividade agrícola*;
- Mata e reflorestamento. Esta alternativa possui uma desvantagem – *áreas pequenas que devem ser preservadas* e um risco – *extinção de plantas e animais nativos.*

As duas primeiras alternativas (pastagem e cultura) foram escolhidas para a solução da questão e estão envolvidas por caixas. A terceira alternativa não foi escolhida.

O restante da figura pode ser descrito de forma semelhante.

5.5 Resumo

Este capítulo deu um breve exemplo da aplicação da metodologia de documentação proposta nesta dissertação. Ele detalhou os documentos O QUE, COMO e PORQUE para um estudo de caso na área agro-ambiental. O principal problema para criar os documentos reside na grande dependência que se tem dos especialistas que desenvolvem o plano ambiental. Este exemplo foi fornecido pelo Prof. Dr. Jansle Vieira Rocha da Faculdade de Engenharia Agrícola – FEAGRI-UNICAMP. Faltam, no entanto, vários detalhes de documentação que exigiriam uma interação muito demorada com o pesquisador.

O próximo capítulo apresenta as conclusões e contribuições desta dissertação e propõe algumas extensões aos mecanismos de gerenciamento de documentação propostos, sob os pontos de vista teórico e de implementação.

Capítulo 6

Conclusões e Extensões

6.1 Contribuições

Esta dissertação concentrou-se na especificação e implementação parcial de um ambiente para auxiliar a documentação de atividades de planejamento ambiental. A proposta considerou três tipos de documentos: descrição do produto final do planejamento (documentos O QUE), descrição do processo usado para obter o produto final (documentos COMO) e descrição das razões que estão por trás das decisões para se chegar a este produto (documentos PORQUE). Documentos O QUE foram definidos através de estruturas de hipermídia, documentos COMO através de *workflows* científicos e documentos PORQUE através de *design rationale*. Estes documentos são gerenciados, de forma unificada, através de um único banco de dados.

Inicialmente foi feito um levantamento de modelos existentes na literatura para representar cada tipo de documento. Com base nesses modelos foi definida uma especificação para cada tipo de documento tendo em vista que o alvo de documentação eram atividades de planejamento ambiental. A especificação dos documentos O QUE (estruturas de hipermídia) foi baseada no modelo de referência Dexter [HS90, HS94] e em algumas de suas extensões propostas no modelo DHM [GT94]. A especificação de documentos COMO (*workflows* científicos) seguiu o meta-modelo padrão de representação de *workflows* proposto pela WFMC [Coa99]. A especificação de documentos PORQUE (*design rationale*) combinou os conceitos básicos presentes em todos os modelos de *design rationale* analisados (praticamente a base do IBIS [KR70]) com parte do modelo proposto pelo *Proteus* [MSPD95].

Em seguida os mecanismos de gerenciamento dos documentos COMO foram projetados e implementados como uma extensão do sistema WOODSS [SMRY99]. Esta implementação envolveu a reengenharia e reimplementação de parte deste software para torná-lo mais modular e adaptá-lo à incorporação das novas necessidades de documentação.

Também foi apresentada uma proposta de implementação dos documentos O QUE e PORQUE neste sistema.

Finalmente os mecanismos de gerenciamento de documentação propostos na dissertação foram aplicados em um problema de planejamento ambiental real. Esta etapa do trabalho exemplificou a utilização dos três tipos de documentos, contribuindo para a validação da proposta.

Uma grande dificuldade deste trabalho reside na abrangência do domínio de planejamento ambiental. Atividades de planejamento ambiental não são estruturadas e variam com o tipo de especialista e objetivos. A literatura correlata é bastante especializada e dirigida a especialistas de cada área (por exemplo, estudo sobre erosão e seus fatores determinantes). Assim, fica difícil caracterizar de forma adequada o processo de planejamento ambiental e documentá-lo eficientemente.

As principais contribuições desta dissertação foram:

- Especificação centrada em bancos de dados das estruturas dos documentos O QUE, COMO e PORQUE;
- Especificação do ambiente para gerenciá-los, visando facilitar trabalho cooperativo na área de planejamento ambiental. Esta especificação servirá igualmente de base para outros ambientes que visem documentação de trabalho cooperativo;
- Implementação parcial deste ambiente;
- Reengenharia do sistema WOODSS, tornando-o mais modular de modo a facilitar manutenções e extensões futuras.

6.2 Extensões

As extensões para esta dissertação são de dois tipos: teóricas e de implementação.

Do ponto de vista teórico, um grande problema associado é que a completude da documentação depende do especialista envolvido no planejamento. Isto significa que talvez a documentação descrita nunca atinja o desejado porque o especialista não terá disponibilidade (ou paciência) para fornecer todos os detalhes necessários. Assim, uma extensão imediata é definir um subconjunto de campos de documentos imprescindíveis a um primeiro nível de documentação.

Outra extensão é determinar formas adicionais de documentação - por exemplo, usando gravação de voz ou vídeo de reuniões. Estes documentos podem, por sua vez, também ser anotados e armazenados no banco de dados.

Uma terceira extensão seria projetar mecanismos de geração automática de documentos estruturados através de dados obtidos pelo processamento de documentos não

estruturados. Este tipo de mecanismo já existe no sistema WOODSS, que gera *workflows* científicos (documentos COMO) a partir do *log* do SIG IDRISI. Documentos de som ou vídeo, por exemplo, poderiam ser utilizados para gerar documentos PORQUE.

Outras extensões incluem ampliação dos modelos de documentos propostos, a partir de sua implementação e uso por especialistas. Por exemplo, o modelo de documentos O QUE foi projetado considerando que esses documentos envolvem conjuntos de dados bem definidos, com gerenciamento centrado em um banco de dados. Assim, aspectos implementados em vários sistemas hipermídia tais como suporte a ligações dinâmicas não são suportados pelo modelo proposto. No caso de gerenciamento centrado na WEB ou incorporação de novos mecanismos de documentação (por exemplo, som e vídeo) seria importante suportar esses aspectos.

Finalmente será necessário propor estruturas para permitir o gerenciamento integrado de todos os documentos. Uma integração possível é permitir que nós de hiperdocumentos estabeleçam ligações diretas para os demais tipos de documentos (*workflows* e *design rationale*). Atualmente, isso só é possível se esses documentos forem salvos como conteúdo de outros nós. Outra possibilidade é integrar o *design rationale* com componentes de um *workflow* tais como atividades, dependências e dados.

A extensão de implementação mais óbvia é o término da implementação do sistema proposto. A seguir, o sistema deverá ser testado e utilizado em grandes problemas de planejamento ambiental. Isso permitirá validar sua efetividade como ferramenta de apoio ao processo de planejamento. Outra questão a ser considerada é a melhoria da usabilidade do sistema através do desenvolvimento de uma interface amigável e facilmente utilizada pelos diversos especialistas. O ideal é que a interface permita aos usuários elaborar os documentos sem que eles tenham conhecimento das estruturas utilizadas para gerenciar cada tipo de documento. Uma possibilidade seria a implementação de uma interface com um conjunto de campos de modo a facilitar a inserção de documentos.

Apêndice A

Script para Criação do Banco de Dados em MySQL

```
/*************************************************************************/
/************************ Esquema de um workflow ************************/
/*************************************************************************/

CREATE TABLE IF NOT EXISTS Workflow (
w_id INT NOT NULL,
w_name VARCHAR(80),
w_description VARCHAR(255),
w_creation_date DATE NOT NULL,
w_state INT NOT NULL, /* 0: incompleto, 1: completo */
PRIMARY KEY (w_id));

CREATE TABLE IF NOT EXISTS Activity (
a_id INT NOT NULL,
a_name VARCHAR(80),
a_description VARCHAR(255),
a_parameters VARCHAR(60),
a_state INT NOT NULL, /* 0: incompleta, 1: completa */
a_category ENUM ('intern', 'join', 'split'),
a_type ENUM ('simple', 'composite'),
w_id INT NOT NULL,
r_id INT,
visual_a_id INT NOT NULL,
subwf_id INT,
```

```
PRIMARY KEY (a_id),
FOREIGN KEY (w_id) REFERENCES Workflow,
FOREIGN KEY (r_id) REFERENCES Role,
FOREIGN KEY (visual_a_id) REFERENCES VisualActivity,
FOREIGN KEY (subwf_id) REFERENCES Workflow);

/* Representação visual de uma atividade */
CREATE TABLE IF NOT EXISTS VisualActivity (
visual_a_id INT NOT NULL,
a_xposition INT NOT NULL,
a_yposition INT NOT NULL,
PRIMARY KEY (visual_a_id));

CREATE TABLE IF NOT EXISTS Dependency (
d_id INT NOT NULL,
d_label VARCHAR(100) NOT NULL,
d_type ENUM ('data', 'temporal'),
d_in_order INT, /* 1, 2, 3,... */
d_out_order INT, /* 1, 2, 3,... */
a_origin INT NOT NULL,
a_destination INT NOT NULL,
visual_l_id INT NOT NULL,
data_id INT,
PRIMARY KEY (d_id),
FOREIGN KEY (a_origin) REFERENCES Activity,
FOREIGN KEY (a_destination) REFERENCES Activity,
FOREIGN KEY (data_id) REFERENCES Data,
FOREIGN KEY (visual_l) REFERENCES VisualLink);

CREATE TABLE IF NOT EXISTS Data (
data_id INT NOT NULL,
description VARCHAR(100) NOT NULL,
path VARCHAR(100),
PRIMARY KEY (data_id));

CREATE TABLE IF NOT EXISTS ActivityData (
a_id INT NOT NULL,
data_id INT NOT NULL,
```

```
type ENUM ('in', 'out') NOT NULL,
data_order INT, /* 1, 2, 3,... */
visual_l_id INT NOT NULL,
PRIMARY KEY (a_id, data_id, type),
FOREIGN KEY (a_id) REFERENCES Activity,
FOREIGN KEY (data_id) REFERENCES Data,
FOREIGN KEY (visual_l_id) REFERENCES VisualLink);

/* Representação visual de dependências e de dados de entrada e saída de atividades
*/
CREATE TABLE IF NOT EXISTS VisualLink (
visual_l_id INT NOT NULL,
l_xorigin INT NOT NULL,
l_yorigin INT NOT NULL,
l_xdestination INT NOT NULL,
l_ydestination INT NOT NULL,
PRIMARY KEY (visual_l_id));

CREATE TABLE IF NOT EXISTS Role (
r_id INT NOT NULL,
r_description VARCHAR(100) NOT NULL,
PRIMARY KEY (r_id));

CREATE TABLE IF NOT EXISTS Agent (
agent_id INT NOT NULL,
name VARCHAR(80),
description VARCHAR(255),
type ENUM ('software', 'user'),
PRIMARY KEY (agent_id));

/* Relacionamento <desempenha> entre agente e papel */
CREATE TABLE IF NOT EXISTS AgentRole (
agent_id INT NOT NULL,
r_id INT NOT NULL,
PRIMARY KEY (agent_id, r_id),
FOREIGN KEY (agent_id) REFERENCES Agent,
FOREIGN KEY (r_id) REFERENCES Role);
```

```
/*************************************************************************/
/************************ Esquema de Hipermídia ************************/
/*************************************************************************/

CREATE TABLE IF NOT EXISTS Hyperdocument (
h_id INT NOT NULL,
PRIMARY KEY (h_id));

CREATE TABLE IF NOT EXISTS Node (
n_name VARCHAR(80) NOT NULL,
n_content TEXT NOT NULL,
n_type ENUM ('textual', 'nontextual'),
PRIMARY KEY (n_name));

/* Relacionamento <faz referência a> entre nó textual e nó não textual */
CREATE TABLE IF NOT EXISTS NodeReference (
n_from VARCHAR(80) NOT NULL,
n_to VARCHAR(80) NOT NULL,
PRIMARY KEY (n_from, n_to),
FOREIGN KEY (n_from) REFERENCES Node,
FOREIGN KEY (n_to) REFERENCES Node);

CREATE TABLE IF NOT EXISTS Composite (
h_id INT NOT NULL,
n_name VARCHAR(80) NOT NULL,
main_node ENUM ('yes', 'no'),
PRIMARY KEY (h_id, n_name),
FOREIGN KEY (h_id) REFERENCES Hyperdocument,
FOREIGN KEY (n_name) REFERENCES Node);

CREATE TABLE IF NOT EXISTS VisibleAnchor (
an_name VARCHAR(80) NOT NULL,
an_value VARCHAR(255) NOT NULL,
n_father VARCHAR(80) NOT NULL,
PRIMARY KEY (an_name),
FOREIGN KEY (n_father) REFERENCES Node);

CREATE TABLE IF NOT EXISTS NonVisibleAnchor (
```

```
an_name VARCHAR(80) NOT NULL,
an_value VARCHAR(255) NOT NULL,
n_father VARCHAR(80) NOT NULL,
PRIMARY KEY (an_name),
FOREIGN KEY (n_father) REFERENCES Node);

CREATE TABLE IF NOT EXISTS Link (
anchor VARCHAR(80) NOT NULL,
n_origin VARCHAR(80) NOT NULL,
an_origin VARCHAR(80),
n_dest VARCHAR(80) NOT NULL,
an_dest VARCHAR(80),
FOREIGN KEY (anchor) REFERENCES VisibleAnchor,
FOREIGN KEY (n_origin) REFERENCES Node,
FOREIGN KEY (an_origin) REFERENCES NonVisibleAnchor,
FOREIGN KEY (n_dest) REFERENCES Node,
FOREIGN KEY (an_dest) REFERENCES NonVisibleAnchor);

/*************************************************************************/
/********************** Esquema de design rationale **********************/
/*************************************************************************/

CREATE TABLE IF NOT EXISTS DR (
dr_id INT NOT NULL,
PRIMARY KEY (dr_id));

CREATE TABLE IF NOT EXISTS Question (
q_id INT NOT NULL,
question VARCHAR(255) NOT NULL,
PRIMARY KEY (q_id));

/* Autorelacionamento <subdivide-se em> */
CREATE TABLE IF NOT EXISTS SubQuestion (
sq_id INT NOT NULL,
q_id INT NOT NULL,
PRIMARY KEY (sq_id, q_id),
FOREIGN KEY (q_id) REFERENCES Question,
FOREIGN KEY (sq_id) REFERENCES Question);
```

```
/* Relacionamento <está relacionada com> */
CREATE TABLE IF NOT EXISTS RelatedQuestion (
q_id1 INT NOT NULL,
q_id2 INT NOT NULL,
PRIMARY KEY (q_id1, q_id2),
FOREIGN KEY (q_id1) REFERENCES Question,
FOREIGN KEY (sq_id2) REFERENCES Question);

CREATE TABLE IF NOT EXISTS Objective (
o_id INT NOT NULL,
objective VARCHAR(255) NOT NULL,
q_id INT NOT NULL,
PRIMARY KEY (o_id),
FOREIGN KEY (q_id) REFERENCES Question);

CREATE TABLE IF NOT EXISTS Alternative (
al_id INT NOT NULL,
alternative VARCHAR(255) NOT NULL,
solution ENUM ('yes', 'no'),
q_id INT NOT NULL,
PRIMARY KEY (al_id),
FOREIGN KEY (q_id) REFERENCES Question);

CREATE TABLE IF NOT EXISTS Advantage (
ad_id INT NOT NULL,
advantage VARCHAR(255) NOT NULL,
al_id INT NOT NULL,
PRIMARY KEY (ad_id),
FOREIGN KEY (al_id) REFERENCES Alternative);

CREATE TABLE IF NOT EXISTS Disadvantage (
dis_id INT NOT NULL,
disadvantage VARCHAR(255) NOT NULL,
al_id INT NOT NULL,
PRIMARY KEY (dis_id),
FOREIGN KEY (al_id) REFERENCES Alternative);
```

```
CREATE TABLE IF NOT EXISTS Risk (
risk_id INT NOT NULL,
risk VARCHAR(255) NOT NULL,
al_id INT NOT NULL,
PRIMARY KEY (risk_id),
FOREIGN KEY (al_id) REFERENCES Alternative);

CREATE TABLE IF NOT EXISTS Action (
act_id INT NOT NULL,
action VARCHAR(255) NOT NULL,
risk_id INT NOT NULL,
PRIMARY KEY (act_id),
FOREIGN KEY (risk_id) REFERENCES Risk);

/*************************************************************************/
/********************** Integração dos documentos **********************/
/*************************************************************************/

CREATE TABLE IF NOT EXISTS Planning (
p_id INT NOT NULL,
p_name VARCHAR(80),
p_description VARCHAR(255),
p_contact VARCHAR(255),
PRIMARY KEY (p_id));

CREATE TABLE IF NOT EXISTS Step (
seq_number INT NOT NULL,
p_id INT NOT NULL,
s_name VARCHAR(80) NOT NULL,
s_description VARCHAR(255),
w_id INT,
dr_id INT,
h_id INT,
PRIMARY KEY (seq_number, p_id),
FOREIGN KEY (p_id) REFERENCES Planning,
FOREIGN KEY (w_id) REFERENCES Workflow,
FOREIGN KEY (dr_id) REFERENCES DR,
FOREIGN KEY (h_id) REFERENCES Hyperdocument);
```

Referências Bibliográficas

[AMY88] R. M. Akscyn, D. L. McCracken, e E. A. Yoder. KMS: a distributed hypermedia system for managing knowledge in organizations. *Communications of the ACM*, 31(7):820–835, 1988.

[ATW00] K. M. Anderson, R. N. Taylor, e E. J. Whitehead, Jr. Chimera: hypermedia for heterogeneous software development enviroments. *ACM Transactions on Information Systems (TOIS)*, 18(3):211–245, 2000.

[AYM89] A. MacLean, R. M. Young, e T. P. Moran. Design rationale: The argument behind the artifact. Em *Proc. of the SICCHI conference on Wings for the mind*, páginas 247–252, 1989.

[Bar96] P. Barthelmess. Sistemas de workflow: Análise da Área e proposta de modelo. Dissertação de Mestrado, IMECC-UNICAMP, Campinas–SP, 1996.

[BB00] J. E. Burge e D. C. Brown. Reasoning with design rationale. Em Jonh Gero, editor, *Artificial Intelligence in Design'00*, páginas 611–629. Kluwer Academic Publishers, 2000.

[Ber02] M. Bernstein. Storyspace 1. Em *Proceedings of the thirteenth conference on Hypertext and hypermedia*, páginas 172–181. ACM Press, 2002.

[BLCL+94] T. Berners-Lee, R. Cailliau, A. Luotonen, H. F. Nielsen, e A. Secret. The world-wide web. *Communications of the ACM*, 37(8):76–82, 1994.

[CB88] J. Conklin e M. Begeman. gIBIS: A hypertext tool for exploratory policy discussion. *ACM Transactions on Office Information Systems*, 6(4):303–331, 1988.

[CNF+00] F. Cattaneo, E. Di Nitto, A. Fuggetta, L. Lavazza, e G. Valetto. Managing software artifacts on the web with labyrinth. Em *Proceedings of the 22nd international conference on Software engineering*, páginas 746–749. ACM Press, 2000.

[Coa96] Workflow Management Coalition. Terminology and glossary. Relatório Técnico, Workflow Management Coalition, 1996. Document Number WFMC-TC-1011.

[Coa99] Workflow Management Coalition. Interface 1: Process definition interchange–process model. Relatório Técnico, Workflow Management Coalition, 1999. Document Number WFMC TC-1016-P.

[CWL00] T. Chiueh, W. Wu, e L. Lam. Variorum: A multimedia-based program documentation system. Em *IEEE International Conference on Multimedia and Expo*, páginas 155–158, 2000.

[CWP95] M. Crossland, B. Wynne, e W. Perkins. Spatial decision support systems: an overview of technology and a test of efficacy. *Decision Support Systems*, 14:219–235, 1995.

[DH98] L. Dempsey e R. Heery. Metadata: a current view of practice and issues. *The Journal of Documentation 54*, páginas 145–172, 1998.

[Eas01] J. R. Eastman. *Idrisi32 Release 2 Tutorial.* Clark Labs, 2001. Manual Version 32.20.

[Fag99] A. S. Fagundes. Projeto e implementação de um banco de metadados para o sistema de informação de biodiversidade do estado de São Paulo. Dissertação de Mestrado, IC-UNICAMP, Campinas–SP, 1999.

[Feda] Federal Geographic Data Committee. Content Standard for Digital Geospatial Metadata (CSDGM). http://www.fgdc.gov/metadata/contstan.html. (consulta em 18/12/2002).

[Fedb] Federal Geographic Data Committee. FGDC - Federal Geographic Data Committee. http://www.fgdc.gov. (consulta em 18/12/2002).

[FHS+92] J. C. Ferrans, D. W. Hurst, M. A. Sennett, B. M. Covnot, W. Ji, P. Kajka, e W. Ouyang. Hyperweb: a framework for hypermedia-based environments. Em *Proceedings of the Fifth ACM SIGSOFT Symposium on Software Development Environments*, páginas 1–10. ACM Press, 1992.

[Fra00] M. A. R. Franco. *Planejamento Ambiental para a Cidade Sustentável.* São Paulo: Annablume: Fapesp, 2000.

[Gon97] M. A. Gonçalves. Uso de modelos de hipermídia em bibliotecas digitais para dados geográficos. Dissertação de Mestrado, IC-UNICAMP, Campinas–SP, 1997.

[GT94] K. Grønbæk e R. H. Trigg. Design issues for a dexter-based hypermedia system. *Communications of the ACM*, 37(2):40–49, 1994.

[HE00] G. P. Heliades e E. A. Edmonds. Notation and nature of task in comprehending design rationale. *Knowledge Based Systems*, 13(4):215–224, 2000.

[Hel01] D. Helic. *Aspects of Semantic Data Modeling in Hypermedia Systems.* Tese de Doutorado, Institute for Information Processing and Computer Supported New Media (IICM), Graz University of Technology, Austria, 2001.

[Hol95] D. Hollingsworth. The workflow reference model. Relatório Técnico, Workflow Management Coalition, 1995. Document Number TC00-1003.

[HPP+00] X. Hu, J. Pang, Y. Pang, M. Atwood, W. Sun, e W. C. Regli. A survey on design rationale: Representation, capture and retrieval. Em *ASME Design Engineering Technical Confs., 5th Design for Manufacturing Conf (DETC 2000/DFM-14008)*, 2000.

[HS90] F. Halasz e M. Schwartz. The dexter hypertext reference model. Em *Proceedings of the Hypertext Standardization Workshop*, páginas 95–133, 1990.

[HS94] F. Halasz e M. Schwartz. The dexter hypertext reference model. *Communications of the ACM*, 37(2):30–39, 1994.

[Kas01] D. S. Kaster. Combinando bancos de dados e raciocínio baseado em casos para apoio à decisão em planejamento ambiental. Dissertação de Mestrado, IC-UNICAMP, Campinas–SP, 2001.

[KR70] W. Kunz e H. W. J. Rittel. Issues as elements of information systems. Working paper, No. 131, Institute of Urban and Regional Development, Univ. of California, Berkeley, Calif., 1970.

[Lab] Clark Labs. Geographic analysis and image processing software. http://www.idrisi.com. (consulta em 18/12/2002).

[Lau01] R. Laurini. *Information Systems for Urban Planning: A Hypermidia Cooperative Approach.* Taylor & Francis, 2001.

[MC96a] T. P. Moran e J. M. Carroll, editores. *Design Rationale: Concepts, Techniques and Use.* Laurence Erlbaum Associates, 1996.

[MC96b] T. P. Moran e J. M. Carroll. *Overview of Design Rationale*, capítulo 1, páginas 1–20. In [MC96a], 1996.

[McC91] R. I. McCall. PHI: A conceptual foundation for design hypermedia. *Design Studies*, 12(1):30–41, 1991.

[MSPD95] S. R. Monk, I. Sommerville, J. M. Pendaries, e B. Durin. Supporting design rationale for system evolution. Em *European Software Engineering Conference (ESEC95)*, páginas 307–323, 1995.

[MYBM96] A. MacLean, R. Young, V. Bellotti, e T. Moran. *Questions, Options and Criteria: Elements of Design Space Analysis*, capítulo 3, páginas 53–107. In [MC96a], 1996.

[OPM97] J. L. Oliveira, F. Pires, e C. B. Medeiros. An environment for modeling and design of geographic applications. *GeoInformática*, 1(1):29–58, 1997.

[Ort84] L. Ortolano. *Environmental Planning and Decision Making.* j. Wiley, 1984.

[OW96] K. Osterbye e U. K. Wiil. The flag taxonomy of open hypermedia systems. Em *Proceedings of the Seventh ACM conference on Hypertext*, páginas 129–139. ACM Press, 1996.

[Pir97] F. Pires. *Um Ambiente Computacional para Modelagem de Aplicações Ambientais.* Tese de Doutorado, IC-UNICAMP, Campinas – SP, 1997.

[PMJV01] M. A. Peerbocus, C. B. Medeiros, G. Jomier, e A. Voisard. Documenting changes in a spatiotemporal database. Em *Anais do XVI Simpósio Brasileiro de Banco de Dados*, páginas 10–24, 2001.

[Pol98] S. Polyak. Applying design space analysis to planning. Em *Workshop on Knowledge Engineering and Acquisition for Planning: Bridging Theory and Practice*, páginas 40–47, 1998.

[PT98] S. Polyak e A. Tate. Rationale in planning: Causality, dependencies and decisions. *Knowledge Engineering Review*, 13(3):247–262, 1998.

[PV96] F. Peña-Mora e S. Vadhavkar. Design rationale and design patterns in reusable software design. Em J. S. Gero e F. Sudweeks, editores, *Artificial Intelligence in Design 96*, páginas 251–268. Kluwer Academic Press, 1996.

[Roc03] H. A. Rocha. Metadados para workflows científicos no apoio ao planejamento ambiental. Dissertação de Mestrado, IC-UNICAMP, Campinas–SP, 2003.

[Rod94] J. M. Mateo Rodriguez. Planejamento ambiental como campo de ação da geografia. Em *Anais do V Congresso Brasileiro de Geográfos*, volume 1, páginas 582–594, 1994.

[RS92] A. Rizk e L. Sauter. Multicard: an open hypermedia system. Em *Proceedings of the ACM conference on Hypertext*, páginas 4–10. ACM Press, 1992.

[RS95] M. Rusinkiewicz e A. Sheth. Specification and execution of transactional workflows. Em W. Kim, editor, *Modern Database Systems. The Objet Model, Interoperability and Beyond*, páginas 592–620. ACM Press, 1995.

[SCP97] R. F. Santos, H. B. Carvalhais, e F. Pires. Planejamento ambiental e sistemas de informações geográficas. *Caderno de Informações Georreferenciadas - CIG*, 1(2), 1997. Revista eletrônica, http://orion.cpa.unicamp.br/revista/cig.html.

[Sef98] L. Seffino. WOODSS - sistema espacial de apoio ao processo decisório baseado em workflows. Dissertação de Mestrado, IC-UNICAMP, Campinas–SP, 1998.

[SHH+92] N. Streitz, J. Haake, J. Hannemann, A. Lemke, W. Schuler, H. Schütt, e M. Thüring. Sepia: a cooperative hypermedia authoring environment. Em *Proceedings of the Fourth ACM conference on Hypertext*, páginas 11–22. ACM Press, 1992.

[Shu91] S. J. Shum. *A Cognitive Analysis of Design Rationale Representation.* Tese de Doutorado, University of York, Department of Psychology, 1991.

[SMRY99] L. Seffino, C. B. Medeiros, J. Rocha, e B. Yi. WOODSS - A Spatial Decision Support System based on Workflows. *Decision Support Systems*, 27(1–2):125–123, 1999.

[VMJ00] A. Voisard, C. B. Medeiros, e G. Jomier. Database support for cooperative work documentation. Em *Proc. of COOP2000 - Fourth International Conference on the Design of Cooperative Systems*, páginas 65–79, 2000.

[Whi02] E. J. Whitehead, Jr. Uniform comparison of data models using containment modeling. Em *Proceedings of the Thirteenth Conference on Hypertext and Hypermedia*, páginas 182–191. ACM Press, 2002.

[WL96] U. K. Wiil e J. J. Leggett. The hyperdisco approach to open hypermedia systems. Em *Proceedings of the seventh ACM conference on Hypertext*, páginas 140–148. ACM Press, 1996.

[WL97] U. K. Wiil e J. J. Leggett. Hyperform: a hypermedia system development environment. *ACM Transactions on Information Systems*, 15(1):1–31, 1997.

[Wor] Workflow Management Coalition. The Workflow Management Coalition. http://www.wfmc.org/. (consulta em 18/12/2002).

[WVM96] M. Weske, G. Vossen, e C. B. Medeiros. Scientific workflow management: Wasa architecture and applications. *Fachbericht Angewandte Mathematik und Informatik*, 3/96-I, 1996.

[WWVM96] J. Wainer, M. Weske, G. Vossen, e C. B. Medeiros. Scientific workflow systems. Em *Proc. of the NSF Workshop on Workflow and Process Automation Information Systems*, 1996.