UFSC – Universidade Federal de Santa Catarina
CTC – Centro Tecnológico
INE – Departamento de Informática e Estatística
Curso de Ciências da Computação
Disciplina : Projeto em Ciências da Computação I

Título: FRH – Framework para Relatórios Hipermídia

Subtítulo: Validação do FRH: PCMAT Hipermídia

Autor: Thiago Linhares de Oliveira Matrícula: 9813248-2
Orientador: Roberto Willrich
Co-orientador: Juan W. Moore Espinoza
Banca Examinadora: Vitório Bruno Mazzola
Rosvelter Coelho da Costa

Florianópolis, 01 de fevereiro de 2003

Sinopse

Este trabalho implementa o FRH – Framework para Relatórios Hipermídia. O FRH visa auxiliar na implementação de aplicações que sejam capazes de prover sistemas hipermídia completamente autônomos.

O FRH segue os princípios das arquiteturas para sistemas hipermídia de três níveis e Dexter. O processo de utilização do FRH consiste basicamente no mapeamento do nível de base de dados para um modelo que descreve nodos e links no formato XML e na criação de especificações de apresentação dos nós feitas através de estilos XSLT (eXtensible Stylesheet Language Tranformation).

Abstract

This work implements the FHR – Framework for Hypermedia Reports. FHR is intended to help to create applications witch are able of supplying complete autonomous hypermedia reports.

The FHR follows principles of both three levels and Dexter hypermedia systems architectures. FHR's utilization process is basically constituted on the mapping between a database and the XML HAM model, witch describes nodes and links, and the creation of presentation specs for hypermedia nodes made through XSLT (eXtensible Stylesheet Language Tranformation) styles.

Lista de Abreviaturas

DOM	Document Object Model
DTD	Document Type Definition
PCMAT	Programa de Condições e Meio Ambiente do Trabalho na Indústria da Construção
SAX	Simple API for XML
XML	Extensible Markup Language
XSL	eXtensible Stylesheet Language
XSLT	XSL Transformation
W3C	Word Wide Web Consortium

Sumário

CAPÍTULO 1 - INTRODUÇÃO

A geração de relatórios deve levar em conta a impressão em papel e os inúmeros formatos de documentos eletrônicos. Além dos formatos comuns de documentos eletrônicos desejáveis para um relatório, pode-se incluir documentos em HTML. O HTML é um formato muito difundido desenvolvido para suportar as funções da Hipermídia, por exemplo os links.

Dependendo dos objetivos de um relatório, a hipermídia pode ser a melhor escolha de formato. Muitas facilidades como busca por conteúdos, navegação, interatividade e elementos multimídia podem ser acrescidas a um relatório através do uso consciente da Hipermídia. Desde que sigam padrões e sejam previamente planejados, é possível controlar o comportamento de um sistema hipermídia, podendo assim alterar links, elementos de interface, métodos de navegação e conteúdo, permitindo que o sistema seja apresentado na forma mais compatível com os desejos, necessidades e preferências dos usuários.

A XML é uma linguagem de metadados, onde a informação é estruturada e descrita através de tags, oferecendo assim um meio de desvincular a informação da apresentação, que é realizada através da XML Stylesheet Language (XSL). Além de possuir grande facilidade de uso conjunto com o HTML, a XML é uma tecnologia que se encaixa perfeitamente para a implementação da estrutura de Sistemas Hipermídia.

1.1 OBJETIVOS GERAIS

O presente trabalho propõe um framework para geração de relatórios sob a forma de Hipermídia. Sua implementação envolve estudo nas áreas de Hipermídia, Organização da Informação, Banco de Dados, Sistemas Aplicativos e Desenvolvimento para WEB.

A implementação do FRH deverá contemplar os seguintes requisitos gerais:

- A arquitetura do FRH deverá permanecer extensível e independente de aplicação.
- O FRH deverá agilizar o cumprimento das etapas da metodologias para construção de sistemas hipermídia.

1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos a serem alcançados neste projeto são os seguintes:

- Mapeamento de elementos comuns em sistemas hipermídia.

- Criação de uma arquitetura para o FRH, semelhante aos sistemas hipermídia:
  - Criação de um modelo da HAM no formato XML.
  - Definição de formas de mapeamento entre o nível de base de dados e a HAM.
  - Definição de formas de apresentação da HAM.

- Mapeamento e implementação dos requisitos do FRH:
  - Implementação de links contextuais implícitos através da geração automática de *hotwords*. Tanto o desenvolvedor como o usuário desenvolvedor não terão que se preocupar em especificar os locais de cada hotword.
  - Índices deverão ser construídos automaticamente;
  - Suporte multimídia através de tecnologias WEB;
  - Implementação de um sistema de navegação da hipermídia.

- Implementar meios de adaptação dos elementos hipermídia pelo usuário final, concentrando nos seguintes pontos: estrutura e layout dos relatórios; controle da geração de links;

- Especificação de como utilizar o framework.

1.3 DELIMITAÇÃO DO TEMA / LIMITAÇÕES

A pesquisa das áreas abordadas é realizada através de um estudo em profundidade para sustentação conceitual do núcleo do sistema que se propõe implementar. Com relação aos pontos não profundamente abordados, estes são apresentados através de uma visão geral mas com as referências para documentação mais profunda.

O FRH deve ser desenvolvido para servir como um módulo para aplicações que devem gerar relatórios hipermídia, sendo que estas aplicações devem prover os dados necessários para o

funcionamento do framework. A validação do FRH ocorre exatamente da maneira recém descrita, como um módulo de uma aplicação chamada Sistema PCMAT. Este aplicativo já possui a funcionalidade de cadastro da maior parte das informações das quais serão extraídos os relatórios, restando a implementação de um cadastro para acomodar o glossário e algumas modificações para acomodar tipos de mídia diferentes que não sejam imagens.

Por ocasião deste trabalho, somente requisitos referentes à aplicação no Sistema PCMAT serão suportados, implementando requisitos gerais e, se houverem, também requisitos específicos.

A área de Interface Homem Computador (IHC) tem forte relevância para o projeto. Porém, esta área é tratada de modo superficial, deixando espaço aberto para trabalhos futuros. Apenas considerações críticas são abordadas nesta área.

1.4 MOTIVAÇÃO / JUSTIFICATIVA

A realidade em que vivemos exige muitas vezes o uso de um mesmo conjunto de informações para diferentes finalidades. As informações necessitam, em muitos casos, devem estar disponíveis a uma grande quantidade de pessoas que possuem diferentes objetivos ao acessarem sistemas que disponibilizam essas informações. Objetivos diferentes requerem formas diferentes de apresentação da informação. Dependendo dos objetivos de um relatório, sua apresentação pode variar imensamente.

Outro fato é o grande volume de informação a ser gerenciada. A manipulação de grandes blocos de informação torna proibitiva a confecção manual de relatórios. A descentralização da informação fatalmente ocorre na maioria das organizações onde não se tem uma política de controle e gerência da informação. Várias versões da mesma informação são criadas, algumas vezes completamente desatualizadas, gerando relatórios inconsistentes entre si.

A organização das informações em bancos de dados resolve o problema da inconsistência dos relatórios. Sistemas e componentes de geração de relatórios podem gerar relatórios automaticamente em vários formatos. Existe também um nível de adaptação, geralmente limitada, dos relatórios em relação aos objetivos e preferências do usuário.

A hipermídia pode ser utilizada como forma de relatório. Os campos de atuação e as vantagens do uso da hipermídia são bem estudados. Várias experiências comprovaram que a localização de uma informação específica pode ser agilizada com o uso dessa tecnologia.

Apenas relatórios em papel ou em formatos de documento eletrônico para editores de texto podem não ser suficientes para alcançar os objetivos de certos relatórios. Assim, o uso da informação pode requerer desde relatórios impressos em papel até sistemas hipermídia adaptativos.

1.4.1 O Sistema PCMAT

O Sistema PCMAT foi o responsável pela motivação inicial para a criação do FRH. O PCMAT (Programa de Condições e Meio Ambiente do Trabalho na Indústria da Construção) consiste de um documento impresso em papel utilizado na área de construção civil para especificação e controle de aspectos ligados a segurança do trabalho e à caracterização de uma obra. Este documento é obrigatório para obras com mais de 20 funcionários, como mostra a norma NR 4 da Lei Nº 6.514 [Legislação, 2001].

De acordo com essa norma, o Engenheiro Técnico de Segurança do Trabalho deve incluir no PCMAT as seguintes informações:

- Caracterização da obra e das empresas: Dados das empresas participantes, empreiteiras, número de trabalhadores, descrição de canteiros de obra, etc.

- Segurança: atividades a serem desenvolvidas, riscos, medidas preventivas, equipamentos, etc.

A confecção manual deste relatório é trabalhosa e demorada, podendo até ser esquecidos pontos importantes devido a grande quantidade de atividades e detalhes do processo construtivo. Esta é, em parte a motivação deste trabalho.

A solução para o problema descrito acima é a utilização de uma base de dados com informações padrões (as quais se repetem em muitas obras) e informações a serem cadastradas pelo Engenheiro de Segurança do Trabalho, as quais serão em grande parte implementadas e portanto, poderão ser largamente reutilizadas.

O FRH fornece o suporte para a geração dos relatórios PCMAT no formato de Sistemas Hipermídia, uma vez que a geração de relatórios para impressão já é suportada pelo sistema. Mas por quê utilizar um sistema hipermídia para a apresentação do PCMAT, se apenas o relatório impresso é obrigatório? A pergunta é elucidada quando percebemos as vantagens que a hipermídia pode oferecer ao entendimento, aprendizado, navegabilidade, apresentação e funcionalidade para sistemas de recuperação de informação. Estas características são importantes e devem ser supridas pelo Sistema PCMAT devido a alguns requisitos desejáveis ao sistema:

- Apresentação do trabalho realizado pelo Engenheiro de Segurança do Trabalho à entidade contratante ou a terceiros;
- Melhor compreensão e treinamento mais eficiente dos profissionais encarregados da aplicação do PCMAT no processo de construção de uma obra;
- Maior facilidade na recuperação da informação pelos profissionais envolvidos;
- Atribuição de funcionalidades práticas ao sistema, como por exemplo a possibilidade de transferência do ambiente hipermídia gerado para a WWW;
- Diferencial em relação à possíveis sistemas similares.

Concluindo, o FRH vem contribuir muito na funcionalidade do Sistema PCMAT.

1.4.2 Contribuição

Não só o Sistema PCMAT ganha funcionalidade de hipermídia mas também outros sistemas de informação podem usufruir da mesma funcionalidade.

A criação do modelo da HAM em XML criado para o FRH incrementa a pesquisa sobre modelos para Sistemas Hipermídia e cria uma nova aplicação de XML. A pesquisa sobre a arquitetura Dexter indica possibilidades de uso das tecnologias XML/XSL.

A arquitetura do FRH cria um novo meio para documentação e armazenamento de informação que pode ser facilmente utilizada por humanos.

Este trabalho contribui também para a difusão do trabalho de padronização e definição de tecnologias para internet da W3C, principalmente a família de tecnologias XML.

CAPÍTULO 2 - SUSTENTAÇÃO CONCEITUAL E TÉCNICA

Este capítulo apresenta conceitos relevantes utilizados neste projeto. Serão apresentadas as bases para a geração do relatório Hipermídia, que terá sustentação através do estudo dos conceitos de Hipermídia: arquitetura, metodologia e elementos comuns. Após, serão abordadas as tecnologias utilizadas como base do FRH: XML e XSL.

2.1 DOCUMENTOS ELETRÔNICOS E RELATÓRIOS

O uso dos computadores e de sistemas de informação acarretou no surgimento dos documentos eletrônicos. Muitas informações passaram a ser armazenadas na forma eletrônica, deixando o papel em segundo plano. O uso de documentos eletrônicos oferece muitas facilidades em comparação ao papel.

Porém, o uso incorreto dessa tecnologia, acrescida do volume de dados, forma um caos de informação eletrônica.

2.1.1 Formatos

Aos documentos eletrônicos podem ser acrescentadas uma série de características em sua apresentação, incluindo texto, imagem e vídeo. Um DE contendo apenas texto plano (não formatado) possui enormes limitações para representação do conteúdo. Já um DE formatado inclui têm uma estrutura interna que descreve as várias estruturas e representações do conteúdo, tais como fontes, listas, imagens, tabelas, etc.

Ao conjunto de regras utilizadas para descrever a representação do conteúdo é chamado de *formatos*. Quanto a informação guardado pelo documento, os formatos podem ser:

- **Formatos de descrição de estrutura**: contém dados sobre a estrutura do conteúdo, mas não de sua representação. O SGML e o XML são exemplos dessa classe de formatos.

- **Formatos de descrição de Página**: são baseados em linguagens de programação para descrever páginas imprimíveis. O conteúdo é mapeado de forma a representar o layout do documento. Um exemplo dessa classe é o PostScript da Adobe.

Para que um documento eletrônico formatado seja utilizado, seja para visualização ou impressão, este precisa ser processado de acordo com o formato no qual foi produzido a fim de gerar o resultado visual esperado.

O formado de documento do Word (.doc) é um dos mais utilizados. Esforços foram feitos para a padronização de formatos, gerando alguns formatos que podem ser abertos e processados em uma grande variedade de plataformas e sistemas operacionais. Dentre os principais temos: RTF, PDF, Postscript, HTML, XML e SGML.

2.1.2 Geração de Relatórios

Documentos eletrônicos podem ser criados manualmente, através de editores variados, ou através ou automaticamente, através de geradores de relatórios. Entende-se por gerador de relatório, programas que processam informações e geram relatórios para visualização, impressos ou num formato de documento eletrônico.

Dentre as ferramentas utilizadas para a geração manual, encontra-se principalmente os editores de texto, planilhas eletrônicas e programas para confecção de apresentações. Essas ferramentas oferecem, alguns recursos de personalização e geração de conteúdo, normalmente implementadas através de *templates* (padrões reutilizáveis).

As ferramentas para geração automática de documentos eletrônicos surgem sob a forma de sistemas aplicativos especializados. Estes normalmente mapeiam as informações estruturadas e fornecem meios de recuperação através de relatórios. O poder de processamento é de fato muito maior, oferecendo a possibilidade de geração de novas informações a partir das iniciais e adaptação da apresentação dos relatórios de acordo com os objetivos do usuário para uma determinada aplicação.

Muitas aplicações Web possuem a funcionalidade de gerar vários formatos e vários layouts de um mesmo conjunto de informações. As seções 2.4 (XML) e 2.5 (XSL) fornecem os meios para a implementação desta funcionalidade.

2.1.3 Conclusão

A informação passou a ser manipulada eletronicamente, criando os documentos eletrônicos, os quais possuem diferentes formatos e que podem ser gerados manualmente ou por sistemas de informação. Uma melhor gerência da informação é notada, uma vez que ela é manipulada por programas com capacidade de gerar relatórios adaptáveis.

2.2 HIPERMÍDIA

Alguns autores se referem ao Hipertexto de maneira similar à Hipermídia. Já outros definem que a Hipermídia é a junção do Hipertexto com a Multimídia. Neste trabalho, o termo Hipermídia será utilizado sempre que possível. Quando for citado o termo "Hipertexto", este deve ser entendido como "Hipermídia".

> Hipermídia é "*uma base de dados textuais, visuais, gráficos e sonoros, onde cada ilha de informação é denominada de nó ou quadro*" [Rhéaume, 1993 apud Bulgay e Ulbricht, 2000].

Nielsen [1995] faz outra definição de hipermídia, enfatizando a existência de *links*. Segundo ele, o "*Hipertexto consiste de pedaços de texto (ou outra informação) interligados*".

2.2.1 Vantagens da Hipermídia

As mudanças ocorridas na sociedade pelo introduzidas pelo uso de sistemas de informação acarretaram num grande acréscimo de informação. Para que seja útil, a informação deve ser facilmente recuperável e assimilável. Por sua vez, a gerência de grandes lotes de informações tornou-se fator crítico no contexto atual dos negócios, sejam eles quais forem.

Segundo Martin [1990] a Hipermídia oferece:

- redução de Custos através da substituição do papel;
- evita a existência de grandes estoques para armazenamento de documentos em papel.
- facilidade para manutenção;

- outras funcionalidades que o papel não oferece, tais como: vídeo, interatividade, adaptabilidade, senhas para acesso à partes da informação, acesso simultâneo através de redes, etc;
- rápida assimilação da informação;
- rápida recuperação de informações específicas.

Dentre as facilidades citadas, o grande poder da Hipermídia se encontra nas duas últimas. A rápida assimilação e recuperação da informação, como já dito anteriormente são fatores críticos para alcançar objetivos no mundo atual.

2.2.3 Quando usar Hipermídia

Nem todas as aplicações podem se beneficiar das facilidades oferecidas pela hipermídia. O custo benefício da implantação de um sistema hipermídia pode não corresponder ao esperado. Nielsen [1995] expõe algumas desvantagens que devem ser levadas em consideração, tais como: maior dificuldade de leitura em dispositivos eletrônicos, custos extras de implementação, implantação e treinamento.

Ben Shneiderman [1998 apud Nielsen, 1995] propôs "três regras douradas" para que uma aplicação seja apropriada na forma de Hipertexto:

- Um grande corpo de informação é organizado em numerosos fragmentos;
- Os fragmentos relacionam-se uns com os outros;
- O usuário necessita de apenas uma pequena informação a qualquer momento.

2.2.4 Aplicações da Hipermídia

Nielsen [1995] cita as aplicações que podem tirar proveito da Hipermídia. As listas a seguir mostram as aplicações divididas em grupos. Algumas das aplicações podem ser encaixadas em mais de um grupo, porém elas estão distribuídas de acordo com o grau de similaridade com um grupo. São elas:

Aplicações orientadas ao Computador **Aplicações de Negócio**

- Documentação online;
- Orientação ao usuário;
- Engenharia de software;
- Sistemas operacionais.
- manuais de reparo e outros tipos;
- dicionários e livros de referência;
- auditoria;
- leis;
- marketing.

Aplicações de Suporte à Produção Intelectual

- Suporte à organização de idéias e "brainstorm";
- Jornalismo;
- Pesquisa.

Aplicações Educacionais

- ensino e treinamento de várias áreas de conhecimento;
- línguas estrangeiras;
- arte e cultura clássicas;
- museus;

Aplicações de Entretenimento e Lazer

- guias para turistas;
- bibliotecas;
- ficção interativa;
- noticiários, jornais e revistas;
- sexo;

Dentre as aplicações citadas, algumas aplicações são de interesse para este trabalho, especificamente, para o PCMAT Hipermídia. Os requisitos definidos para o sistema apontam funcionalidades largamente usadas pelas aplicações de:

- Documentação online;
- Ensino e treinamento de várias áreas de conhecimento;
- Manuais de reparo e outros tipos;
- Dicionários e livros de referência.

2.2.5 Arquitetura de Sistemas Hipermídia

Segundo Campbell e Goodman [1988 apud Nielsen, 1995], pode-se distinguir três níveis em um Sistema Hipermídia:

- Nível de apresentação: interface do usuário.
- Nível de HAM (Hipertext Abstract Machine - Máquina Abstrata de Hipertexto): gerência de nós e links.
- Nível de base de dados: armazenamento, compartilhamento da informação e disponibilização da informação.

Outra arquitetura muito semelhante à de três níveis foi proposta pelo grupo Dexter. A novidade consiste na especificação de interfaces entre os níveis:

- Ancoramento (anchoring): Este nível serve para a definição das ancoras dos links nos nós.
- Especificações de apresentação: Este nível guarda formas e estilos de apresentação dos nós.

A figura a 2.2.1 mostra as duas arquiteturas de forma comparativa [Nielsen 1995].

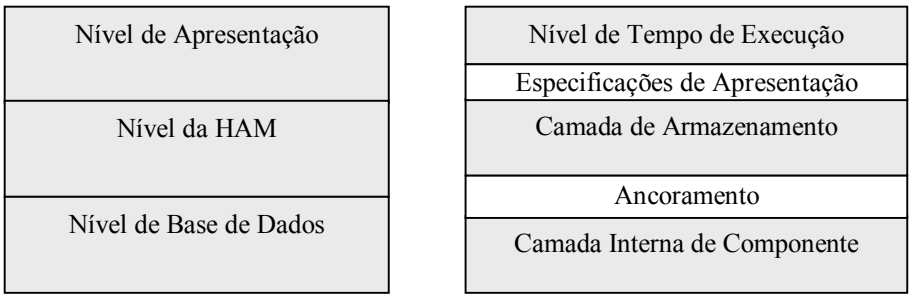


Figura 2.2.1-Arquitetura de Sistemas Hipermídia: os modelos de três níveis e Dexter

Apesar de o FRH utilizar a arquitetura Dexter, esta é tratada de modo superficial. Isto se deve ao fato de que somente as informações fornecidas sobre a arquitetura simples de três níveis, somadas à breve explanação sobre a arquitetura Dexter acima fornecida, são suficientes para para a contextualização do FRH nas arquiteturas.

2.2.5.1 Nós

Segundo Nielsen [1995], "cada unidade de informação é chamada de *nó*". Os nós são apresentados para o usuário através de telas. Para que o usuário tenha uma compreensão profunda sobre determinado assunto do conteúdo, Nielsen sugere que é melhor ter-se muitos nós pequenos, cada um abordando um assunto ou idéia específica, do que poucos nós grandes que abranjam uma maior área do conteúdo. Do contrário o usuário tende a ter uma noção do todo, em detrimento do conhecimento especializado.

> *"O sucesso na criação de hiperdocumentos, então, depende da divisão do assunto em fragmentos independentes." [Martin, 1990]*

Martin [1990] define elementos organizacionais para maior clareza e estruturação do conteúdo principal dos hiperdocumentos. Estes elementos são:

- **Blocos básicos de informação:** são elementos que contém uma idéia, como os parágrafos deste trabalho. Não é necessariamente relevante fora de contexto.
- **Unidades de diagramas:** contêm um diagrama, possivelmente com algum texto associado. Diagramas são geralmente figuras mas também podem ser gráficos, sons, animações e vídeos interativos. Neste trabalho, diagramas serão chamados por vezes de *elementos multimídia*, ou simplesmente de *mídias*.
- **Unidades de conceitos:** são módulos tutorias que contém explicações de conceitos. Uma unidade de conceito pode ser formada por um ou mais blocos básicos de organização e/ou uma ou mais unidades de conceitos. Pode-se também adicionar unidades de diagramas às unidades de conceitos, o que é muito comum nos Sistemas Hipermídia atuais.

Unidades de conceitos são, na verdade, nós com diferentes graus de granularidade, na medida que é possível ter-se unidades de conceitos dentro de unidades de conceitos. Tem-se ao mesmo tempo muitos nós tratando de assuntos específicos e a possibilidade de agregá-los em nós que abranjam uma área maior do conteúdo.

Na seção "Elementos Comuns em Sistemas Hipermídia" serão abordados diferentes tipos de nós.

2.2.5.2 Links

Os *links* (ou hiperlinks) são o poder de navegação dos Sistemas Hipermídia. Nós são conectados uns com os outros através de links. Os links são "conexões" que levam o usuário de um nó a outro, instantaneamente [Martin, 1990].

Uma sequência de nós conectados por links é chamada de *caminho* [Nielsen, 1995]. O usuário pode navegar por um caminho e voltar quando achar necessário. Este procedimento é conhecido como *backtrack*.

Um link pode ser implícito ou explícito, podendo estar definido no texto (hotwords), em um diagrama (imagem, animação), em elementos da interface (botões, menus) ou até mesmo no espaço temporal de um nó. Ele pode ainda definir vários destinos para o mesmo ponto de partida.

Um outro critério importante para a classificação de links é o conteúdo ao qual o link está relacionado. O conteúdo pode ser figuras, vídeos, textos, conteúdo com significado semântico especial (por exemplo, unidades de conceito e glossário), sons, animação, etc.

Vários tipos de links podem ser construídos automaticamente, por exemplo, links para glossário, para tabelas de conteúdo e para índices.

É interessante que os links apresentem de alguma forma o tipo ou um resumo dos nós destinos. Usuários procurando uma determinada figura podem encontrá-la mais rapidamente se o(s) link(s) existente(s) possui(em) uma imagem ou outro elemento de interface junto do texto do link que indique que o conteúdo do link é uma figura.

2.2.5.3 Nível de Base de Dados

O nível de base de dados está na base da arquitetura de Sistemas Hipermídia. Este nível é responsável pelo armazenamento da informação a ser apresentada (com ou sem processamento) pelo sistema.

Não importa como a informação é armazenada, mas deve ser possível recuperar rapidamente uma parte da informação. Existe vasto estudo e experiência na área de banco de dados. Ao

utilizá-los no nível de base de dados, Sistemas Hipermídia agregam, sem grande esforço, funcionalidades como controle de acesso multiusuário, backup e segurança.

Nós e links no nível de base de dados não apresentam significado algum, sendo apenas dados que podem ser alterados. Porém, é interessante que os dados sejam armazenados em estruturas que representem alguma semântica, facilitando o trabalho de interfaceamento com o nível da HAM.

2.2.5.4 Nível da HAM

A HAM se localiza entre o nível de base de dados e o nível de apresentação. Este nível é responsável pela gerência de nós e links.

É importante a existência de padrões para a especificação e implementação deste nível. Informações no nível de base de dados são fáceis de transferir de sistema para sistema. Já a transferência da informação na forma de nós e links (que são especificados no nível da HAM) é mais complexa. Uma solução é utilizar uma linguagem de metadados para a implementação da HAM.

Nielsen [1995] sugere que a HAM deve possuir conhecimento sobre a forma dos nós, links e atributos relativos a cada um.

O XML, como será visto no próximo capítulo, serve muito bem ao propósito da HAM. Este trabalho apresenta um padrão extensível para o nível da HAM utilizando XML.

2.2.5.5 Nível de Apresentação

A interface do usuário é responsável pela apresentação da informação da HAM, incluindo questões do tipo: que comandos devem estar disponíveis para o usuário, como mostrar nós e ligações e se deve ou não incluir diagramas [Nielsen, 1995].

Nielsen [1995] previu que a informação poderia ser composta de objetos e atributos, algo que hoje é possível com o XML.

"Eu espero que o computador processe objetos de informação com número significante maior de atributos que atualmente possui. Uma vez que o computador sabe mais sobre cada objeto, ele será capaz de mostrá-los em diversas formas, dependendo das necessidades específicas do usuário para sua tarefa atual."

2.2.6 Elementos Comuns em Sistemas Hipermídia

Esta sessão se destina ao mapeamento de elementos comuns em sistemas hipermídia feito pelo autor deste documento. O mapeamento em si é inédito, porém foi realizado com base nas referências bibliográficas relativas à hipermídia.

Elementos de Sistemas Hipermídia especificam a informação, as funcionalidades e as características de apresentação que o sistema fornece ao usuário. Vários destes elementos são frequentemente encontrados em Sistemas Hipermídia. Estes são citados a seguir:

- Coleção de Unidades de conceito: coleção de módulos tutoriais relacionados ao conteúdo do Sistema Hipermídia;
- Glossário, Lista de definição de acrônimos;
- Coleção de diagramas e elementos Multimídia;
- Coleção de referências;
- Coleção de informações gerais, como revendedores, endereços, pessoas a serem contatadas, etc;

O elemento *coleção de unidades de conceito* constitui a principal fonte de informação para o usuário, sendo que os outros elementos servem de apoio ao usuário para melhor entender ou navegar pelas unidades de conceito.

Os elementos citados são comuns a muitos sistemas, e portanto são de grande interesse neste trabalho, pois ao automatizar a geração dos mesmos, é poupado um grande esforço no desenvolvimento de um Sistema Hipermídia, reduzindo custos e tempo de projeto e desenvolvimento. Os elementos aqui descritos são todos passíveis de serem contemplados pelo FRH. É importante ressaltar que a Hipermídia, em sua totalidade, não é composta somente dos elementos apresentados.

Os elementos hipermídia possuem três propriedades: estrutura e funções, relativas ao nível da HAM, e apresentação, obviamente relativa ao nível de apresentação. Classificando de acordo com o nível da arquitetura tem-se:

Nível de Base de Dados	Nível de HAM	Nível de Apresentação
Nenhuma	**Estruturas**	**Apresentação**
	Funções	

2.2.6.1 Elementos Hipermídia: Estrutura

A estrutura de um elemento hipermídia refere-se a forma como os nós são organizados. Uma má estruturação acarreta em desorientação, dificuldade de navegação e assimilação do conteúdo por parte do usuário. A hierarquia de conceitos é uma forma de estrutura bem conhecida e que foi abordada na sessão sobre "Organização da Informação". Esta é a forma de organização adotada neste trabalho e deste ponto em diante, sempre que a palavra estrutura for mencionada ela estará relacionada a organização sob a forma de hierarquia de conceitos.

Quando se especifica a estrutura de um elemento hipermídia está-se, na verdade, especificando os nós e os componentes que os formam.

Um tipo de nó é um conjunto de nós com a mesma estrutura [Martin, 1990]. Cada elemento hipermídia possui conjuntos de tipos de nós específicos.

Os mesmos dados no nível de base de dados podem fazer parte da estrutura de mais de um nó. Por exemplo, a explicação do funcionamento de um equipamento pode ser simultaneamente um nó de glossário, ser uma unidade de conceito ou fazer parte de uma unidade de conceito mais abrangente.

2.2.6.2 Elementos de Hipermídia: Funções

As funções de elementos hipermídia descrevem principalmente as opções de navegação de um elemento hipermídia. Cada elemento hipermídia possui funções intrínsecas. A seguir são listadas as principais funções aplicáveis a cada elemento hipermídia.

Glossário, lista de definição de acrônimos, coleção de referências e coleção de informações gerais:

- Navegação sequencial pelos nós do elemento;
- Índice dos nós;
- Histórico de caminhos percorridos:
  - Navegação sequencial pelo caminho percorrido através de backtracking ou avanço para o próximo nó do caminho;
- Índice de nós do histórico;
- Links conectando palavras ou expressões a outros nós (hotwords);
- Busca por nós do elemento, podendo ser uma busca parametrizada ou busca por palavras;

Elementos Multimídia (Diagramas)

- Índice dos nós (fluxograma);
- Links conectando áreas da mídia a outros nós;
- Busca por nós do elemento, podendo ser uma busca parametrizada ou busca por palavras associadas a mídia;
- Funções de manipulação e interatividade com a mídia.

Unidades de conceito

- Navegação sequencial por nós do mesmo tipo de nó;
- Navegação para o nó imediatamente superior ao nó corrente (nó pai);
- Índice dos nós (fluxograma);
- Histórico de caminhos percorridos:
  - Navegação sequencial pelo caminho percorrido através de backtracking ou avanço para o próximo nó do caminho;
  - Índice de nós do histórico (fluxograma).
- Links conectando palavras ou expressões para suas respectivas unidades de conceito;
- Busca por nós do elemento, podendo ser uma busca parametrizada ou busca por palavras.

Anotações

- Fazer uma anotação relativa a um nó.

Bookmarks

- Adição, exclusão e manipulação da lista;
- Ir diretamente para um nó marcado por um Bookmark;
- Índice de Bookmarks.

Dentre as funções comuns citadas, as principais são: busca, índices, navegação hierárquica ou sequencial e histórico. Tem-se ainda a *função de exibição*, que ocorre em todos os elementos hipermídia e pode ser descrita como sendo o ato de exibir a informação. Esta função tem sua implementação no nível de apresentação. A próxima sessão comenta um pouco mais sobre esta função.

2.2.6.3 Elementos Hipermídia: Apresentação

Elementos de interface são elementos perceptíveis da interface do usuário, tais como botões, listas, janelas, tabelas, textos, imagens, etc. A propriedade de apresentação dos elementos hipermídia especifica quais e como os elementos de interface apresentam informações e disponibilizam funções dos elementos hipermídia ao usuário. Isto se dá através da especificação de uma série de parâmetros que mudam de acordo com o tipo de elemento de interface necessário, por exemplo botões e tabelas. Cor, fonte, borda e posição são exemplos de alguns dos parâmetros. O nível de apresentação então implementa a exibição das funções da HAM.

A apresentação dos elementos hipermídia variam muito de sistema para sistema. Elementos de interface dependem muito da metáfora utilizada na interface. Uma metáfora é uma diretriz para a apresentação dos elementos hipermídia. Uma metáfora muito utilizada e também adotada neste trabalho é o sistema de janelas e frames.

A maior parte dos elementos de interface existe em função dos elementos hipermídia. Existe porém uma pequena parcela que não está atrelado a nenhum elemento hipermídia. Estes elementos de interface fazem parte exclusivamente da metáfora do Sistema Hipermídia.

Sendo assim, para a completa definição do nível de apresentação devem ser especificadas as propriedades de apresentação dos elementos hipermídia e dos elementos de interface da metáfora. O preenchimento das propriedades de apresentação de um elemento hipermídia consiste na especificação dos elementos de interface para a função de visualização.

Muitos experimentos foram feitos com vários elementos de interface para testar as vantagens e desvantagem do uso de diferentes elementos com a mesma finalidade. Por exemplo, constatou-se que um índice que se expande e se contrai é mais eficiente do que um índice estático [Nielsen, 1995]. Não é o foco deste trabalho uma pesquisa sobre usabilidade de interface. Existem áreas como Interface Homem Computador e Engenharia de Usabilidade que abordam profundamente este e outros aspectos da interface.

2.2.7 Metodologia para Construção de Sistemas Hipermídia

> *"O procedimento para a criação de um hiperdocumento deve ser planejado antes de começar."* [Martin, 1990].

Martin [1990] descreve as etapas da metodologia para desenvolvimento de Sistemas Hipermídia. Esta metodologia é apresentada a seguir com alguns itens resumidos, por não apresentarem grande relevância para o escopo deste trabalho.

Fase 1: Preparação

- Determine quem irá ler o documento.
- Determine os objetivos do documento.
- Determine quem irá criar o documento.
- Determine que recursos serão usados para ajudar o(s) autor(es).
  - Recursos de software.
    - Programas de criação de hiperdocumentos.
    - Programas para confecção de mídias.
    - Recursos de apoio, como verificação ortográfica e gramatical.
    - Outros programas de apoio;
  - Recursos humanos.

        - Artista gráfico.
        - Especialista no assunto.
        - Outros.
    - Recursos de informação.
        - Glossário Central.
        - Coleção central de acrônimos.
        - Coleção central de unidades de conceito.
        - Coleção central de diagramas.
        - Módulos do documento.
- Determine que meio e padrões serão usados.
    - Estabeleça que facilidades de microinformática o leitor irá usar.
    - Estabeleça os padrões gráficos.
    - Estabeleça que dispositivos externos de vídeo serão usados, se houver algum.
    - Estabeleça as orientações de exibição.
    - Estabeleça as orientações de tipologia.
- Determine que itens do assunto devem estar no documento.
    - Defina quais os itens que devem estar no documento.
    - Determine como o documento pode atingir os problemas, necessidades e questões dos usuários.
    - Crie uma estrutura inicial de documento.
    - Revise o conteúdo e estrutura propostos com usuários potenciais.
    - Determine melhorias possíveis na estrutura.
- Determine quem irá revisar e testar na prática o documento.

Fase 2: Criação

- Projete o documentos.
- Crie o documento.
- Revise e melhore o documento.
- Teste na prática e refine o documento.

2.2.8 Automatização da Geração de Elementos Comuns

A geração de Sistemas Hipermídia não é totalmente automática, visto que os autores devem produzir todo o conteúdo do sistema, incluindo texto e mídias. Além disso deve ser criada a interface deste com o usuário.

Felizmente, alguns elementos muito comuns em Sistemas Hipermídia podem ser criados automaticamente. Os elementos que podem ser gerados automaticamente e que serão implementados pelo FRH em função do PCMAT Hipermídia são:

- Índices
- Adição de links de associação aos itens do glossário
- Adição de links de associação às unidades de conceito.
- Numeração de nós, figuras e outros elementos multimídia.
- Implementação dos elementos funcionais atribuídos aos elementos de interface.

2.2.9 Conclusão

Em certas aplicações, a Hipermídia permite a utilização da informação de maneira mais eficiente. Os Sistemas Hipermídia são ferramentas importantes para aplicações de recuperação de informação, de documentação online, de educação, entre outras.

A arquitetura de três níveis para Sistemas Hipermídia permite uma implementação modularizada. A metodologia apresentada aqui permite construir cada uma das partes da arquitetura, além de abordar aspectos de planejamento e instalação do sistema.

Existem elementos de Hipermídia muito comuns nos sistemas implementados. Uma vez implementados, estes elementos podem ser reutilizados em várias aplicações.

2.3 Extensible Markup Language (XML)

Esta seção objetiva descrever a tecnologia XML em seus aspectos conceituas e práticos relativos a este trabalho. Foge ao escopo reproduzir em detalhes os aspectos de baixo nível da especificação XML. De qualquer modo, o leitor poderá obter a especificação completa do XML no site do W3C [2000] (Word Wide Web Consortium).

2.3.1 Características e Vantagens

XML (e**X**tensible **M**arkup **L**anguage - linguagem de marcação extensível) é uma especificação desenvolvida pelo W3C [2000] a nível de recomendação.

A SGML (Structured Generalized Markup Language – linguagem estruturada generalizada de marcação) é o padrão internacional para definição de descrições da estrutura e conteúdo de documentos eletrônicos.

XML, assim como o HTML, são subconjuntos da SGML. Sendo assim, é natural que essas linguagens se pareçam, pois, assim como a SGML, são baseadas em *tags*. Não existem tags predefinidas. O desenvolvedor é quem deve criar suas próprias tags.

XML é um modo extensível, claro e formal para definir estruturas, descrever e compartilhar informações. Como uma linguagem de metadados, um documento XML não contém apenas dados, mas também outras informações. Estas informações adicionais servem para descrever os dados contidos no documento e representam as marcações da linguagem. Portanto, a principal característica do XML é a descrição da informação em uma estrutura processável.

O uso dos padrões ASCII e Unicode fornecem ao XML uma escalabilidade imensa, uma vez que estes padrões são suportados pela maioria dos dispositivos computacionais.

Sintetizando, as principais vantagens do uso do XML tem-se:

- **É facilmente legível por pessoas e máquinas;**
- **Permite a separação da estrutura da informação de sua apresentação;**
- **Permite o intercâmbio de informações;**
- **É extensível;**
- **É escalável;**
- **É orientado a objeto;**
- **É um padrão mundialmente reconhecido;**
- **Número cada vez maior de aplicações e ferramentas de desenvolvimento que utilizam XML;**

2.3.2 Componentes do XML

Elementos e *atributos* constituem a base do XML. Juntamente com eles, existe uma série de complementos que tornam o XML muito mais poderoso.

2.3.2.1 Elementos e Atributos

O nome dos elementos deve preferencialmente refletir o conteúdo neles contido. Quando a informação é descrita corretamente seu significado é melhor entendido.

Elementos são representados por *tags* (marcações). Existem tags para indicar o início dos elementos e seu fim.

Um elemento simples tem a seguinte sintaxe, onde a primeira parte é a tag de início (<nomeDoElemento>), a segunda é conteúdo que carrega (conteúdo) e a terceira parte é a tag indicando o fim do elemento(</nomeDoElemento>):

<**nomeDoElemento**>**conteúdo**</**nomeDoElemento**>

O conteúdo dos elementos pode ser um texto qualquer (desde que não possua caracteres de marcação do XML) ou então outros elementos. Por exemplo:

```
<listaObras>
   <obra>
      <nome>Residencial Praia II</nome>
      <dataInicio>29/04/2000</dataDeInicio>
      <dataPrevistaTermino>01/05/2003</dataPrevistaTermino>
   </obra>
    </listaObras>
```

Caso seja necessário adicionar os caracteres "**<**", "**>**", "**&**", "**'**" (aspas simples) ou "**"**" (aspas dupla) os mesmos devem ser substituídos, respectivamente, por entidades especiais ou então colocados em uma seção **CDATA.** Tudo o que estiver dentro de uma seção CDATA não será processado. Seções CDATA são úteis quando a troca de caracteres especiais se torna proibitiva. Uma seção CDATA tem a seguinte sintaxe:

<![CDATA[TEXTO **]]>**

Os elementos podem ainda conter **atributos.** Estes são especificados dentro da tag de início do documento da seguinte forma:

<nomeDoElemento **nomeDoAtributo**="**texto**" >

Não existe uma regra sobre quando deve-se utilizar elementos ou atributos para descrever uma informação. O exemplo anterior poderia ser escrito da seguinte maneira:

<obra **nome**="Residencial Praia II" dataInicio="29/04/2000" **dataPrevistaTermino** "01/05/2003" >
</obra>
</listaObras>

Normalmente atributos são usados para informações simples e que fornecem alguma dado extra sobre o conteúdo do elemento. Para informações compostas o ideal é utilizar elementos. Por exemplo, uma obra pode ter vários funcionários. O ideal seria utilizar elementos para representar a informação de cada funcionário, ao invés de crias algo do tipo: funcionario1="nome1" funcionario2="nome2" ... funcionárioN="nomeN".

2.3.2.2 Documentos XML

Um documento XML simples é composto da declaração XML e do *elemento raiz.* Pode existir apenas um elemento raiz em um documento XML. Este elemento é a raiz da árvore de elementos do documento. Um documento XML pode ser o seguinte:

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="yes"?>
<listaObras>
   <obra>
      <nome>Residencial Praia II</nome>
   </obra>
</listaObras>
```

Para o exemplo acima, o elemento raiz é o elemento **listaObras**.

Como visto no exemplo, a declaração XML é feita entre os caracteres "**<?**" e "**?>**". Este tipo de declaração é chamado de instrução processável. Instruções processáveis servem para

especificar algum tipo de processamento. A declaração também especifica alguns pseudo atributos, como a versão da linguagem, a codificação e o character set (conjunto de caracteres). Além destes existem também o atributo standalone, que especifica quando referencias externas devem ou não ser verificadas.

Um documento XML não precisa necessariamente ser um arquivo no computador. Ele pode ser uma string em alguma estrutura de dados armazenada na memória ou sendo transferida entre aplicações.

Se um documento XML foi construído de acordo com a especificação XML, esse documento é dito *bem formatado*. Documentos bem formatados podem ser processados tendo como guia a especificação XML. O último exemplo é um exemplo de um documento XML bem formatado.

Um documento XML pode ainda ser *validado* de acordo com um DTD ou XML Schema.

2.3.2.3 Namespaces

Namespaces (espaços de nomes) é também uma recomendação do W3C. Sua especificação mais recente é publicado no seguinte endereço: www.w3.org/TR/REC-xml-names.

Namespaces podem ser definidos como um mecanismo para identificar elementos XML. Uma vez que XML é extensível, um mesmo conjunto de nomes podem ser atribuídos por diferentes pessoas e organizações. Com o uso de namespaces pode-se identificar de forma única a origem dos elementos, obtendo assim sua correta interpretação e, consequentemente, sua correta utilização nas aplicações.

São definidos namespaces para importantes componentes do XML tais como: XSL (tanto para XSLT quanto para XSLFO), XML Schemas, XLink, etc. Um exemplo de declaração de um namespace pode ser a declaração do namespace para XSLT:

xmlns:xsl=”http://www.w3.org/1999/XSL/Transform/”

A declaração acima representa o seguinte:

- **A URL ”http://www.w3.org/1999/XSL/Transform/” especifica um domínio único, o qual representa o namespace do XSLT. Nenhuma outra pessoa ou organização poderá ter este domínio.**
- **A declaração se dá através de “xmlns:**xsl**”. Aqui, xsl é o prefixo o qual irá representar o namespace XSLT especificado pela URL.**

Uma vês declarado o namespace, os elementos nele contidos devem ser referenciados a partir do prefixo do namespace (sempre seguindo a sintaxe XML):

<prefixo**:**nomeDoElemento>

2.3.2.4 Entidades

Entidades podem ser um arquivo, registro de banco de dados, ou outro item que contém dados. O objetivo primário de uma entidade é manter conteúdo, ao invés de estrutura, regras ou gramática. Cada entidade é identificada um nome único e contém seu próprio conteúdo, que pode ser um simples caracter dentro do documento XML ou um grande arquivo que existe fora do documento. A função das entidades pode ser comparada a função de macros.

A declaração de uma entidade ocorre em um DTD. Existem entidades predefinidas para a representação de caracteres especiais.

2.3.2.5 DTD’s e XML Schema

Para algumas aplicações, apenas o fato de um documento ser bem formatado não é suficiente para que um documento XML seja corretamente utilizado. Normalmente o contexto (e as vezes até a ordem) em que os elementos aparecem faz diferença para as aplicações. No exemplo da lista de obras, se o elemento **nome** aparecer antes do elemento **obra,** este estará totalmente fora de contexto e a aplicação deverá reportar algum erro ao processar o documento.

Para verificar se um documento XML foi construído de acordo com os parâmetros da aplicação foram criados o DTD (Document Type Definition – Tipo de Definição de Documento) e o XML Schema. DTD’s e XML Schemas definem um conjunto de regras para testar se um documento XML atende as especificações da aplicação.

Um DTD é um documento que define basicamente quais elementos podem ser inseridos no documentos XML, quais elementos podem conter outros elementos, o número e a sequência dos elementos, os atributos que os elementos podem possui, e opcionalmente, os valores que dos atributos. Existem também regras para a definição de entidades e suas características.

O DTD é declarado no documento XML com uma declaração de tipo de documento (document type declaration). Um documento que foi testado e está de acordo com sua especificação DTD é dito um *documento válido.* Documentos válidos são também documentos bem formatados.

Um ponto fraco do DTD é que sua especificação exige que o desenvolvedor saiba a linguagem para especificação de uma gramática.

Nada mais natural do que descrever a estrutura de um documento XML com outro. O XML Schema tem esse objetivo e é a recomendação publicada em 02/05/2001 pelo W3C [2001] para substituir o DTD.

O XML Schema tem funcionalidade equivalente ao DTD, porém o uso de XML para descrever regras de validação de documentos XML trás novas funcionalidades como definição de tipos de dados, herança, regras de apresentação, além é claro da extensibilidade provida pelo XML. Portanto, XML Schema é uma linguagem mais poderosa e legível que o DTD.

O XML Schema define classes de documentos XML. Um documento validado via um XML Schema é também referenciado como sendo um *documento instanciado.* A especificação do XML Schema é na verdade um vocabulário XML definido com o seguinte namespace: *xmlns:xsd="http://www.w3.org/2001/XMLSchema"*>. Este vocabulário define elementos para definição das regras de validação de um documento XML.

A funcionalidade do provida pela definição de tipo que o XML Schema fornece permite o mapeamento preciso dos dados em outras fontes e formatos para um documento XML.

2.3.3 Campos de Atuação do XML

Existem basicamente duas situações que o XML pode ser usada eficientemente:

- Aplicações que necessitem de intercâmbio de informações.
- Aplicações que manipulem informações as quais devem ser apresentadas ao usuário.

Alguns campos que necessitam destas funcionalidades são:

- Automação na Web
- Integração de Bancos de Dados
- Tradução de Software
- Representações Intermediárias de Dados
- Publicação e geração de relatórios
- Comércio Eletrônico
- Computação Distribuída (Interoperabilidade)

Um grande número de aplicações já foram criadas, e novas aplicações surgem frequentemente. Dentre elas, algumas foram produzidas pelo próprio W3C.

2.3.4 Manipulação de XML através de API's

Para livrar o desenvolvedor das preocupações pertinentes a manipulação da sintaxe e de outros aspectos de baixo nível foram criadas duas especificações para manipulação de XML: Document Object Model (DOM) e Simple API for XML (SAX).

Estas especificações são implementadas por *parsers XML*. O W3C disponibiliza links para parsers implementados em várias linguagens. Nem todos os parsers implementam toda a especificação do W3C. Outros oferecem métodos não previstos na especificação. Deve-se analisar as características e funcionalidades que um parser oferece antes de adotá-lo em um projeto.

Um parser XML se situa entre o documento XML e a lógica do negócio implementada pelo desenvolvedor. Existem basicamente dois tipos de parsers: os que implementam o DOM e os que implementam o SAX. Algumas implementações suportam tanto o DOM quanto o SAX.

2.3.4.1 Document Objetct Model (DOM)

DOM é uma recomendação do W3C para processamento e manipulação das tecnologias padrões por eles publicadas, definida através de uma interface orientada a objetos. O DOM apenas define interfaces padrões, não implementando nenhuma delas. É trabalho dos fornecedores de aplicações implementarem a especificação. Para definição desta interface, o W3C utiliza o padrão IDL (definido na especificação do CORBA). O objetivo do uso da IDL é o provimento de uma interface de programação que pudesse ser utilizada em uma vasta variedade de ambientes e aplicações [W3C, 2000 b].

Sua especificação pode ser encontrada em: http://www.w3.org/DOM/DOMTR. No momento deste trabalho, o DOM está em sua segunda versão. A terceira versão está em desenvolvimento.

As aplicações podem optar por suportar módulos específicos do DOM. Assim sendo é sempre bom verificar quais objetos possuem suas funcionalidades de acordo com a especificação original.

A abordagem de modelagem do DOM utilizada para o XML é a de árvore. Esta árvore representa o documento XML e é inteiramente carregada na memória. O DOM oferece vários objetos que manipulam a árvore, permitindo criar documentos XML, navegar por sua estrutura, e adicionar, modificar, ou deletar elementos e conteúdo. Estes objetos são disponibilizados através das interfaces IDL. As interfaces mais importantes são:

- **Exception *DOMException***: define códigos para vários tipos de exceções.
- **Interface *DOMImplementation***: provê métodos para criação de objetos document, e um método para verificação de quais funcionalidades uma implementação suporta.
- **Interface *Document:*** representa o documento XML. Conceitualmente o objeto document é a raiz da árvore do documento. Esta interface provê o acesso às informações do documento e possui métodos para a criação de nós, comentários, entidades, atributos, etc.

- **Interface *Node:*** é a base do DOM. Esta interface define métodos para adicionar e remover uma série de tipos diferentes de objetos node (nó). Dentre os tipos de objeto node encontra-se os tipos ELEMENT_NODE, ATTRIBUTE_NODE e CDATA_SECTION_NODE. Além destes, existem vários outros tipos de nós.
- **Interfaces *CharacterData, Attr*, *Element*, *Text*, *Comment, etc:*** são especializações de node (os tipos de node citados acima), herdando assim suas propriedades e métodos. Estas interfaces especificam os atributos e métodos para lidar com os vários tipos de nós.

Existem outras interfaces que podem ser igualmente importantes para uma determinada aplicação. A especificação completa pode ser encontrada no site do W3C [2000 c].

Ao construir a árvore, o DOM verifica se o documento XML é bem formatado. Opcionalmente, pode-se também validar o documento de acordo com um DTD ou XML Schema.

2.3.4.2 Simple API for XML (SAX)

SAX, apesar de não ser origem em nenhuma organização é considerado um padrão para a implementação de parsers XML. O endereço da projeto na internet é: http://www.saxproject.org/

O SAX é proveniente de uma outra abordagem: orientação a eventos. A medida que o parser processa o documento XML são disparados eventos que são capturados por funções previamente definidas pelo desenvolvedor. Os principais eventos são [Megginson, 2002]:

- início do documento;
- fim do documento;
- início de elemento;
- fim de elemento;
- dados de caracter.

As interfaces e classes do SAX são divididas em cinco grupos. As interfaces em negrito são as mais utilizadas pelas aplicações:

- interfaces implementadas pelo parser: **Parser**, **AttributeList** e **Locator**.
- interfaces implementadas pela aplicação: **DocumentHandler**, **ErrorHandler**, **DTDHandler**, e **EntityResolver**.
- Classes SAX padrões: InputSource, SAXException, SAXParseException, HandlerBase.
- Classes opcionais de ajuda (específico para Java).
- Classes de demonstração (Java).

A medida que o SAX processa o documento XML eventos vão sendo disparados. Exemplificando, para o seguinte documento XML [Megginson, 2002]:

```
<?xml version="1.0"?>
<doc>
    <para>Hello, world!</para>
</doc>
```

O parser irá disparar os seguintes eventos:

- start document
- start element: doc
- start element: para
- characters: Hello, world!
- end element: para
- end element: doc
- end document

A abordagem de eventos provê rápido acesso ao conteúdo dos documentos XML, pois é uma implementação natural de parsers. Porém as aplicações tornam-se mais complexas e trabalhosas de serem criadas. Por exemplo, o parser SAX não armazena o contexto onde está processando um elemento. Esta é uma informação importante, pois um elemento em contextos diferentes normalmente deve ser processado de formas diferentes. Portanto, a aplicação deverá setar o contexto toda vez que ele for modificado. Além disso, o SAX não possui uma interface para alteração de conteúdo no documento XML, limitando-se basicamente na leitura de dados.

Comparando o SAX com o DOM, tem-se as seguintes observações [Marchal, 1999]:

- Opera em um nível mais baixo que a solução DOM;
- Fornece mais controle que o DOM;
- São mais rápidos que o DOM;
- Requer mais trabalho de implementação que o DOM;
- Ausência de interface para alteração de dados.

De fato, muitas implementações de DOM utilizam o SAX internamente. A escolha entre DOM e SAX deve ser feita considerando-se o seguinte:

Performance	**X**	Facilidade de implementação, manutenção e entendimento e legibilidade da aplicação.

2.3.5 Criando Documentos XML do Zero

De acordo com Marchal [1999], a criação de documentos XML a partir do zero é simples e pode ser otimizada sem muito esforço para uma aplicação específica, não necessitando, portanto, do DOM ou do SAX. A manipulação de XML através do DOM se justifica se quer alterar dados em um documento existente, e a do SAX quando performance na leitura é fundamental.

2. XSL

Esta seção objetiva descrever a tecnologia XSL em seus aspectos conceituas e práticos relativos a este trabalho. Assim como XML, foge ao escopo reproduzir em detalhes os aspectos de baixo nível da especificação XSL. A especificação completa pode ser obtida no site do W3C [2001 b].

2.1 Características e Vantagens

XML Stylesheet Language (Linguagem de estilo para XML) é uma aplicação de XML que especifica vocabulários para apresentação de informações contidas em documentos XML. O uso da XSL permite desvincular o conteúdo de um documento XML de sua apresentação. Em outras palavras, um mesmo documento XML pode ser apresentado de diferentes formas quando aplicados diferentes documentos XSL`s.

XSL não se limita somente com a apresentação. A transformação de um documento XML em outro é outro funcionalidade importante da especificação. Esta função é muito utilizada para a transformação de um documento XML em XHTML (HTML estruturado de forma a ser um documento XML bem formatado). A transformação de um documento XML especificado com um DTD ou Schema para outro documento XML especificado por outro DTD ou Schema é muito importante, principalmente no nível corporativo, onde o intercâmbio de informação de diferentes tipos é grande, e esta deve antes se adequar às especificações da corporação para depois ser utilizada pelas aplicações.

Portanto, XSL tem duas funcionalidades: transformação e apresentação. Estas duas funções representam, respectivamente, as duas partes da especificação XSL:

- XSLT – XSL Transformations (Transformações XSL)
- XML FO – XML Formating Objects (Objetos de Formatação XSL)

Ambas as linguagens podem ser utilizadas independentemente da outra, mas são complementares se utilizadas em conjunto.

2.2 XSLT

A XSLT é uma aplicação de XML que define um vocabulário para transformação de um documento XML em outro documento XML.

Por ocasião deste trabalho, a especificação já está em sua segunda versão. Porém, devido ao fato de que o processador XSLT utilizado neste trabalho suporta apenas a versão 1[1], esta será a versão utilizada neste trabalho, tanto neste documento como na implementação do FRH.

Foge aos objetivos deste trabalho a completa especificação da linguagem. Esta é apresentada de modo suficiente a fornecer embasamento para mostrar a importância e o uso da linguagem neste trabalho. A especificação completa pode ser obtida no site da W3C em: http://www.w3.org/TR/1999/REC-xslt-19991116

2.2.1 Conceitos Iniciais

[1]A XSLT Versão 1 foi publicada como recomendação pelo W3C em 16 de novembro de 1999.

A XSLT considera os seguintes elementos como sendo nós na árvore XML: Nó raiz, Elementos, Texto, Atributos, Namespaces, Instruções de processamento e Comentários.

A XSLT deve receber a como entrada uma árvore de elementos no formato XML. Esta árvore é então utilizada para construir uma nova árvore, gerada a partir na inicial e através das regras definidas pelo desenvolvedor. Existem operadores para seleção de nós da árvore inicial, ordenação de nós e inserção de nós na nova árvore.

Cada elemento representa uma árvore. Enquanto nós de texto, atributos, comentários, namespaces e instruções de processamento são folhas de uma árvore.

A saída do processamento da XSLT pode ser qualquer documento XML bem formatado, incluindo o formato XHTML. Existem outros dois métodos alternativos de saída: o HTML e o texto. Com o modo HTML pode-se produzir saídas em formato HTML padrão, com a vantagem de não terem caracteres dentro de scripts como < e > substituídos por entidades XML. O modo texto permite a produção de um texto plano sem seguir nenhum formato.

Os processadores XSLT são os responsáveis pela transformação da árvore XML. Estes devem ser implementados seguindo as especificações da linguagem publicadas pelo W3C. Atualmente alguns browser como o Internet Explorer possuem um processador XSLT embutido. Com isto é possível distribuir páginas em XML com os seus respectivos estilos, deixando o trabalho de processamento para o cliente.

2.2.2 Estrutura dos Estilos XSLT's

Um documento XSLT deve iniciar com as seguintes marcações XML:

<?xml version="1.0" encoding="ISO-8859-1"?>

<**xsl:stylesheet** version="1.0" xmlns:**xsl**="**http://www.w3.org/1999/XSL/Transform**">

A primeira linha é a declaração XML. Como a XSLT é uma aplicação de XML, a declaração deve ser inserida. A segunda linha se refere a declaração da versão da XSLT e seu namespace. O elemento xsl:stylesheet deve ser o elemento raiz do documento.

A partir das duas linhas iniciais o desenvolvedor deve especificar as regras de transformação da árvore XML. Estas regras são especificadas por uma série de elementos no namespace definido.

As regras de template, definidas por xsl:template, são a parte mais importante da XSLT. Estas associam um saída particular para uma entrada particular. Cada xsl:template tem o atributo match que especifica para quais nós do documento de entrada o template deve ser aplicado.
O conteúdo do template pode ser texto que irá literalmente aparecer no documento de saída, ou outras instruções XSLT que copiam dados do documento de entrada. O seguinte trecho é um exemplo de template:

```
<xsl:template match="/">
  <html>
  <head>
  </head>
  <body>
  </body>
  </html>
</xsl:template>
```

Este template é aplicado quando o nó de contexto é o elemento raiz do documento de entrada. O seguinte resultado é impresso:

```
  <html>
  <head>
  </head>
  <body>
  </body>
  </html>
```

Sempre que um template é aplicado, o nó especificado no atributo match se torna o nó de contexto. Para que possa aplicar as regras, o desenvolvedor deve sempre especificar um template que confronte o nó inicial de contexto, que normalmente é a raiz do documento.

Para aplicar as regras a outros elementos o desenvolvedor deve informar ao processador XSLT que este deve processar os elementos filho do nó raiz (ou o nó inicial de contexto). Isto é feito através do elemento xsl:apply-templates. Este elemento diz ao processador para comparar cada filho do nó de contexto com as regras definidas no documento XSLT. Se o processador encontra algum template com o atributo match igual ao nome de um nó filho, ele torna este nó filho o nó de contexto aplica o template especificado. Ao terminar de aplicar o

template, o processador volta ao ponto de onde apply-templates foi chamado, restaurando o nó de contexto para o seu valor antigo.

O elemento apply-templates pode ter um atribulo “select” especificado. Se for especificado este atributo define para quais nós relativos ao contexto atual devem ser verificadas as regras definidas no documento XSLT. Com esta abordagem, o desenvolvedor pode “caminhar” pela árvore e imprimindo os dados desejados. O exemplo a seguir ilustra essa situação:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

 <xsl:template match="/">
    <html>
    <xsl:apply-templates/>
    </html>
 </xsl:template>

 <xsl:template match="Tabela_periodica">
    <body>
    <xsl:apply-templates/>
    </body>
 </xsl:template>

 <xsl:template match="ATOMO">
      Um atomo
 </xsl:template>

 </xsl:stylesheet>
```

Considerando o XML de entrada:

```
<?xml version="1.0"?>
<Tabela_periodica>
    <Atomo>
        <nome>Cobre</nome>
    </Atomo>
    <Atomo>
        <nome>Carbono</nome>
    </Atomo>
</Tabela_periodica>
```

a seguinte saída seria impressa:

```
<html>
  <body>
      Um atomo
      Um atomo
  </body>
```

```
</html>
```

Para recuperar o nome do átomo, o seguinte elemento deve ser utilizado para recuperar o conteúdo do nó <nome>:

```
<xsl:value-of select="nome"/>
```

A linguagem XSLT fornece também a possibilidade de navegação procedural pela árvore, tal qual uma linguagem de programação comum. Estruturas como "para cada elemento em uma coleção faça:", "choose", "if" e até mesmo chamadas de funções, que no XSLT são representadas pelos elementos xsl:template, são suportadas através de elementos específicos.

2.2.3 XPath

No atributo select e em outras partes da linguagem, a XSLT utiliza uma outra especificação do W3C: o XPath. Esta especificação permite que seja expressado exatamente quais nós se quer ou não selecionar.

Por ocasião deste trabalho, o XPath está na versão 2. Porém a versão utilizada neste trabalho, por ter sido desenvolvida para trabalhar em conjunto com a versão 1 da XSLT, é também a versão 1 do XPath. A especificação completa desta versão pode ser encontrada em: http://www.w3.org/TR/1999/REC-xpath-19991116 (20 nov. 2002).

O XPath é baseado em paterns (padrões). Um padrão é um tipo de expressão que retorna um conjunto de nós baseados em um conjunto condições especificadas para um nó. Os nós que satisfazerem as condições são selecionados pela patern. A sintaxe para paterns é uma subclasse da sintaxe de expressões XPath, estas oferecendo um maior poder de expressão do que um patern. O atributo "match" do elemento xsl:template suporta apenas paterns. Já o "select" suporta qualquer tipo de expressão XPath. Exemplos de patern são mostrados na tabela a seguir:

Patern	**Nós que se encaixam no patern**
para	todos os elementos para
*	todos os elementos
lista/item	todos os elementos "item" que tem como pai o elemento "lista"
@tipo	todos os atributos "tipo"

Outro importante conjunto de expressões são os location paths (caminhos de localização). Estes têm a função de selecionar um conjunto de nós relativos ao nó de contexto. Cada location path é constituído de um ou mais location step (passo de localização) separados por "/". Um location step, por sua vez, tem três partes:

- **Axis (eixo):** Especifica a relação sobre a árvore XML entre os nós selecionados pelo location step e o nó de contexto.
- **Node test (teste de nó):** Especifica o tipo de nó e o nome expandido dos nós selecionados pelo location step.
- **Zero ou mais predicados (predicates):** Usa expressões para refinamentos adicionais do conjunto de nós selecionados pelo location step.

Por exemplo, o location path "child::node()" tem seu eixo igual a "child" e seu teste de nó igual a "node()". Não existem predicados. Este exemplo seleciona todos os nós filhos do nó de contexto.

Para o location path "parent::para[position()=1]" temos seu eixo igual a "parent", o teste de nó igual a "para" e um predicado "[position()=1]". O exemplo seleciona o primeiro elemento "para" do nó pai do nó de contexto.

A lista completa de eixos e funções pode ser obtida na especificação do XPath.

2.3 Conclusão

A utilização da linguagem de transformação do XSL (XSLT) sobre documentos XML permite uma real separação entre dados e apresentação. A XSLT é muito poderosa, permitindo muitas maneiras de selecionar nós e construir expressões. A aplicação de estilos XSLT para criação de HTML é um dos principais uso desta tecnologia, sendo que esta é a função da XSLT neste trabalho.

CAPÍTULO 3 – Framework para Relatórios Hipermídia

O FRH é um framework com uma arquitetura extensível que dá suporte a geração de relatórios hipermídia a partir de bases de dados quaisquer. Os relatórios hipermídia gerados são na verdade sistemas hipermídia estáticos sem qualquer dependência em relação a aplicação que o gerou usando o FRH, podendo, por exemplo, ser colocado em um cd para distribuição.

Este capítulo tem como objetivo documentar o trabalho de projeto, especificações e implementação resultante do FRH. Para informações mais detalhadas sobre a implementação pode-se ver o código do FRH nos anexos I e II, ou no disco que acompanha este trabalho.

Primeiramente é apresentada a arquitetura e o funcionamento do FRH, situando-o dentro dos conceitos apresentados e descrevendo tomadas de decisões relevantes. Em seguida é especificado os procedimentos para utilização do framework, tanto do ponto de vista do desenvolvedor como do usuário desenvolvedor.

3.1 VISÃO GERAL

Os componentes que fazem parte do FRH são os seguintes:

- **Templates FRH:** conjunto de arquivos contendo a "cópia original" da interface e dos scripts que irão constituir o sistema hipermídia final. Os templates FRH podem ser vistos como um sistema hipermídia sem os nós de conteúdo.
- **Componente TFRH:** este é um componente implementado na linguagem object pascal do Delphi. O TFRH é responsável pela geração de todo o conjunto de nós e da customização do sistema hipermídia embutido nos templates FRH.

Estes dois componentes são tratados mais a fundo nas respectivas sessões: Templates FRH e Compotente TFRH.

De modo geral o funcionamento do FRH ocorre a partir da interação de uma aplicação com o componente TFRH, da seguinte maneira:

- Uma aplicação utilizando o TFRH fornece uma string representando um documento XML estruturado de acordo com o modelo da HAM.
- O TFRH lê o documento XML, adicionando automaticamente links para palavras chaves de unidades de conceito e glossário.
- Estilos de apresentação previamente definidos pelo desenvolvedor são aplicados aos nós especificados no XML, gerando com isso todos os nós de conteúdo do sistema hipermídia em seu formato de apresentação final.
- O TFRH faz uma cópia dos templates FRH, customiza o sistema hipermídia embutido e coloca junto dos nós gerados.

3.2 ARQUITETURA

A arquitetura do FRH como um todo, segue a arquitetura de três níveis para Sistemas Hipermídia e, em um grau mais detalhado, segue também a arquitetura Dexter. Entretanto, algumas diferenças sutis existem e devem ser anotadas. O nível de tempo de execução do FRH (o relatório hipermídia) é na verdade outro sistema hipermídia, com seus próprios níveis da arquitetura simples de três níveis e totalmente independente da aplicação que o gera. Ao invés de apresentar um nó de cada vez, como fazem os sistemas hipermídia comuns, o FRH gera, não apresenta, todos os nós do conteúdo de uma só vez.

Analisando do ponto de vista do componente TFRH percebe-se que o que deveria ser seu nível de tempo de execução é na verdade outro sistema hipermídia. Assim sendo o TFRH não precisa implementar as funções de navegação da HAM e outras funções de controle de sistemas hipermídia. Estas funções estão implementadas nos templates FRH sob a forma de um sistema hipermídia "incompleto". Esse sistema é então suprido com as informações fornecidas pelo pré processamento dos nós realizado pelo TFRH. Este processamento define parâmetros para a navegação, busca e operações de gerência do sistema. Estes parâmetros são repassados para o sistema hipermídia final através do processamento dos templates FRH.

A figura 3.1 mostra a arquitetura e o processo de criação de um relatório Hipermídia, apontando a qual nível da arquitetura de Sistemas Hipermídia pertence cada componente.

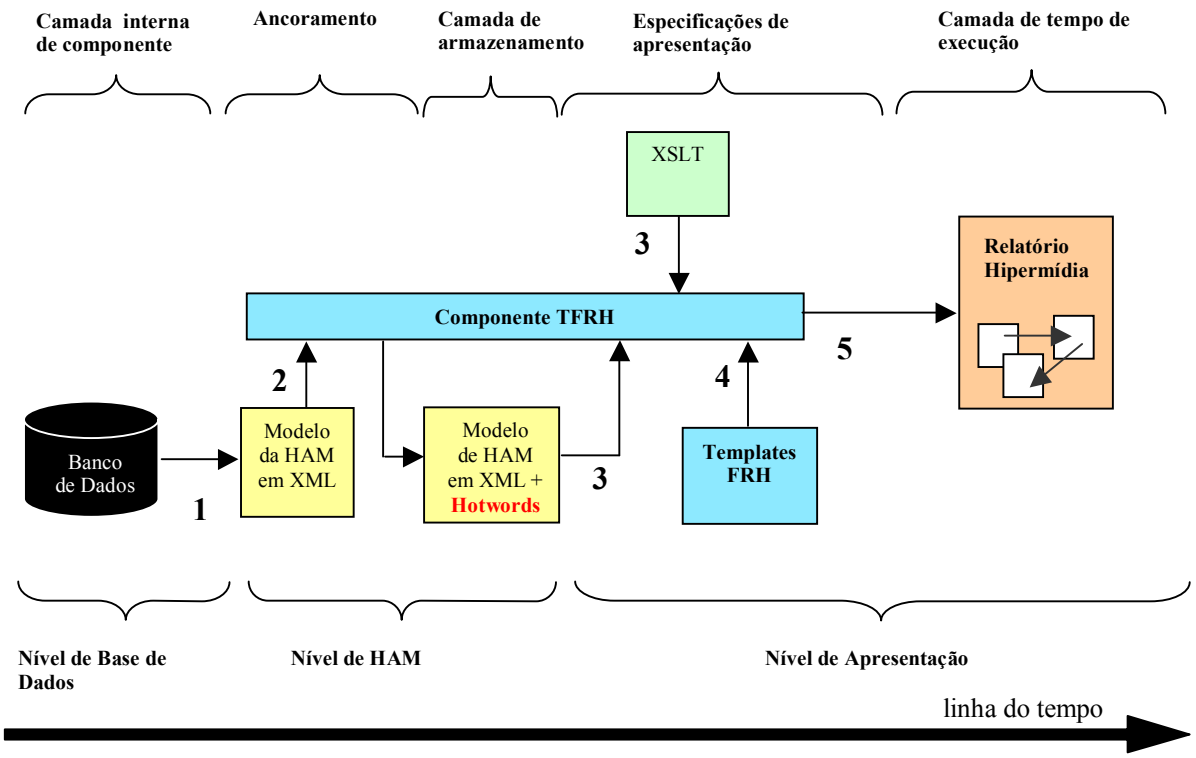


Figura 3.1-Processo de criação de um relatório Hipermídia. Abaixo da figura é mostrado os níveis da arquitetura simples de três níveis. Acima da figura é mostrado os níveis expandidos de acordo com a arquitetura Dexter.

Na figura 3.1, temos as seguintes transições:

1. **Mapeamento das informações armazenadas na base de dados para o modelo de HAM em XML:** a realização deste mapeamento não necessita de nenhuma das API's definidas para XML, por motivos já descritos na seção 2.4.5 (Criando Documentos XML do Zero). Uma vez conforme o modelo, a informação estará organizada em uma estrutura que representa os elementos hipermídia, seus nós e informações suficientes para criação de links implícitos.

2. **Criação de hotwords e mapa global:** É neste momento que são criados e ancorados todos os links implícitos contextuais (hotwords) para as unidades de conceito e para o glossário. O FRH usa os elementos XML que descrevem o título e as palavras chaves para cada nó (ver Modelo da HAM em XML) para criar hotwords.

3. **Formatação para apresentação dos nós:** formatação de acordo com as especificações de apresentação representada através da aplicação estilos XSLT a cada nó definido na HAM. Os estilos XSLT's definidos pelo desenvolvedor devem gerar XHTML como saída. A cada nó formatado é gerado um documento XHTML. O documento XHTML passa ainda por um pequeno processamento para indexação de palavras para a busca e para inserção de scripts. O formato final de um nó é o HTML.

4. **Processamento dos Templates FRH:** esta operação pode ser encarada como uma extensão das especificações de apresentação da arquitetura Dexter. Os templates FRH, bem como seu processamento, são tratados em maior detalhe na sessão Templates FRH deste capítulo.

5. **Criação do Sistema Hipermídia:** O Sistema Hipermídia resultante é a união dos nós gerados no processamento XSLT com a interface e o sistema de controle do sistema hipermídia obtidos através do processamento dos templates FRH.

3.2.1 Templates FRH

Os templates FRH são um conjunto de arquivos em vários formatos que representam o sistema hipermídia sem os nós de conteúdo. Nos templates estão definidos os elementos de interface, funções de gerência do sistema e funções de elementos hipermídia. Os templates FRH são utilizados toda vez em que o relatório hipermídia for solicitado. Estes arquivos são copiados, processados e inseridos no diretório definido para o relatório hipermídia.

As seções a seguir descrevem e explicam as características do sistema hipermídia implementado nos templates FRH.

3.2.1.1 Arquivos

Os arquivos originais dos templates FRH são os seguintes:

- Arquivos na raiz do diretório dos templates FRH. O código dos arquivos seguintes pode ser visto no anexo II:

- **index.html:** documento inicial para ativar o sistema hipermídia.

- **frame_gerente.html:** documento que representa o objeto gerente, o qual contém a maior parte das funções dos elementos hipermídia implementadas em javascript. Tem a função de gerência das janelas e frames do sistema, fornecendo uma interface comum para todo o sistema.
- **frame_ferramentas.html:** documento contendo a interface para as ações mais comuns em sistemas hipermídia. Estas ações podem ser estritamente funções de interface como abrir a janela do fluxograma, ou ser uma função de elemento hipermídia como, por exemplo, caminhar na estrutura do conteúdo.
- **glossario.html:** documento que serve de conteiner para o frame onde deve ser apresentados os nós do glossário.
- **glossario_ferramentas.html:** interface e funções para navegação pelos nós do glossário apenas.
- **fluxograma.html:** documento contendo a interface sobre a qual deve ser criada o fluxograma. Contem funções específicas para a criação e manipulação da árvore de conteúdo.
- **indice.html:** documento que contém a interface do índice de palavras chave. Existe uma instrução de processamento inserida em um determinado ponto do HTML, que deve ser resolvida pelo FRH no nível da HAM, inserindo assim os itens que compõem o índice.

- Diretórios relativos a raiz:

- **scripts/:** contém os arquivos javascript utilizados pelo fluxograma e pelos nós. Todos os outros scripts são incluídos no próprio documento html correspondente. Isto facilita se o desenvolvedor tiver de mudar os arquivos originais de diretórios: ele não precisa se preocupar em atualizar as referências aos scripts. Os scripts externos utilizados pelo fluxograma foram mantidos intactos por fazerem parte de um componente desenvolvido por terceiros, sendo negada a modificação de seu código fonte. Um novo componente pode perfeitamente ser introduzido no lugar do atual, desde que expresse seus nós em um array no padrão do componente atual.
- **busca/:** contém arquivos html e o componente em java para a funcionalidade de busca por palavras e frases. Esta funcionalidade está atualmente implementada através de um applet que busca expressões em uma lista previamente indexada.

- **css/:** contém os arquivos de css utilizados via link nos documentos html's dos nós e no documento html do índice do glossário. Os demais documentos têm seus estilos definidos no próprio documento. A razão disto é a mesma apresentada para os scripts: o desenvolvedor não precisa se preocupar em atualizar referências em caso de mudança nos templates.
- **img/:** este diretório contem todas as imagens e mídias utilizadas pelos templates padrões.

3.2.1.2 Sistema Hipermídia dos Templates

Como visto nas sessões anteriores, o FRH gera um sistema hipermídia como resultado. Excluindo-se os html's de conteúdo, todo o sistema gerado é proveniente dos templates FRH. O sistema hipermídia final se torna adaptável pela omissão de variáveis no sistema hipermídia dos templates FRH. Essas variáveis tem seus valores definidos no ato do processamento dos templates pelo TFRH, sendo atribuídos os valores especificados pelo desenvolvedor.

As funções dos elementos hipermídia estão distribuídas através dos objetos do sistema hipermídia. O sistema hipermídia utiliza os frames e janelas dos documentos html como objetos. Esta funcionalidade é conseguida através da adição de funções javascript nos documentos html que representam um objeto do sistema hipermídia. Assim sendo, o objeto *gerente* e o frame onde foi carregado o documento html que contém as funções relativas ao objeto gerente representam a mesma coisa. O mesmo vale para os outros objetos descritos nesta sessão.

As funções de elementos hipermídia implementadas podem ser visualizadas na documentação do código dos templates FRH originais anexados em meio magnético junto deste deste trabalho.

Existe um sistema de gerência central representado pelo objeto *gerente,* definido através do documento html "frame_gerente.html". Ele é o responsável por receber as mensagens relativas a um objeto e realizar as operações necessárias. Ao realizar tais operações, o gerente, por ser persistente durante toda a sessão, mantêm as variáveis necessárias para o controle do sistema. O gerente também tem a função de disponibilizar uma série de constantes definidas pelo desenvolvedor junto da instancia da classe TFRH no ambiente Delphi. Tais constantes

consistem em sua maioria da especificação dos diretórios dos html's em relação ao diretório raiz do sistema e nome dos html's.

Alguns objetos tem funções próprias que não precisam passar pelo gerente. Estas funções são relativas ao contexto do próprio objeto ou um contexto bem reduzido. Por exemplo, as funções de navegação sequencial e de histórico nos nós do glossário são implementadas pelo frame glossário_ferramentas.html. Estas funções são específicas do contexto do glossário e não influem no funcionamento do resto do sistema. Já a ativação de uma hotword em um nó do glossário precisa necessariamente ser controlada pelo gerente, pois o conteúdo do link pode ser aberto de várias maneiras dependendo do tipo do link, modificando assim o estado geral do sistema, e não só do glossário.

A seguir é apresentada uma tabela dos objetos existentes com seus respectivos documentos html's, nomes para os frames que os representam e as possíveis relações com o objeto gerente. Com exceção do gerente e da unidade de conceito que possuem seus frames fixos e sem possibilidade de mudança de nome, os nomes dos HTML's e dos frames podem ser modificados pelo desenvolvedor. Os nomes padrões são:

Nome Conceitual	**HTML original**	**Nome do frame**	**Relação do frame com o gerente**
Gerente	frame_gerente.html	gerente (fixo)	self (fixo)
Unidade de Conceito	(variável: nós de unidades de conceito)	unidade_conceito (fixo)	top.gerente (fixo)
Busca	busca.html	busca	top.gerente (fixo)
Glossário (nós)	(variável: nós do glossário)	glossario_nós	top.opener.top.gerente (fixo)
Fluxograma	fluxograma.html	fluxograma	top.gerente ou top.opener.top.gerente
Índice	indice.html	indice	top.gerente ou top.opener.top.gerente

A relação do gerente com a página principal deve ser a seguinte: top.

Dentre os objetos listados na tabela, dois deles possuem a possibilidade de serem abertos em um frame do "index.html" (a janela principal do sistema) ou em uma janela separada. Caso se queira abrir o objeto em uma nova janela, o nome do frame será usado como nome da janela, continuando sendo válido para referenciar o objeto em questão.

Uma janela conteiner é necessária quando o frame do objeto deve ser exibido dentro de um frameset em uma janela diferente da definida por “index.html”. As janelas conteiners representam a janela que contem o objeto, que no caso pode ser tanto o índice como o fluxograma. A especificação de tais janelas nesses casos é necessária para que o sistema hipermídia possa abri-las a partir de seus nomes e diretórios. A mesma janela conteiner pode ser especificada para mais de um objeto. Assim o sistema oferece a possibilidade de total controle dos objetos inseridos nos conteiners sem precisar que o desenvolvedor edite o código do sistema de gerência.

3.2.1.3 Processamento dos Templates FRH

O sistema hipermídia dos templates FRH não funciona corretamente até que os templates sejam processados pelo componente FRH implementado no ambiente Delphi. Como explanado na sessão Sistema Hipermídia dos Templates FRH, o sistema foi desenvolvido para suportar alterações de diretórios e relações entre janelas e frames dos html’s padrões.

O processamento dos templates FRH consiste na inclusão de scripts para customização do script do sistema hipermídia através da adequação a possíveis mudanças:

- de diretório dos html’s padrões.
- nos nomes dos html’s padrões.
- na relação entre as janelas e frames padrões do FRH.

Qualquer mudança deve ser feita diretamente nos templates e comunicada ao TFRH. É importante ressaltar que o processamento dos templates FRH corrige as variáveis que são inseridas nos scripts, e não as tags HTML.

3.2.2 Componente TFRH

O componente TFRH é desenvolvido na linguagem object pascal do Delphi. Suas funções já foram especificadas nas sessões Visão Geral e Arquitetura. Esta sessão se destina a documentar o resultado da implementação do componente. As units (módulos) do componente TFRH podem ser vistas no anexo I.

3.2.2.1 Arquitetura do TFRH

Para manipulação de XML, o TFRH utiliza um parser com a padronização DOM do W3C e um processador XSLT. Na verdade as duas funcionalidades são implementados no mesmo componente: o MSXML3. Este é o nome da implementação da Microsoft para o parser XML. Por conveniência, a Microsoft incluiu também um processador XSLT no mesmo pacote.

As funcionalidades do parser e do processador XSLT são obtidas através da dll "msxml3.dll". Esta é uma biblioteca de tipo que disponibiliza uma interface COM para acesso às suas funcionalidades. Esta interface foi importada para o ambiente Delphi, que gerou um arquivo .tlb, o qual contem as co-classes que podem ser utilizadas para instanciar e trabalhar com o objeto. As co-classes não foram utilizadas diretamente. Ao invés disso a foi criado um invólucro de componente ao redor das co-classes. O Delphi disponibiliza esta função automaticamente, permitindo instalar em sua paleta de componentes os objetos da dll.

Os objetos e métodos implementados no parser da Microsoft seguem, em sua maioria, o padrão especificado pelo DOM do W3C. O processador XSLT embutido é na verdade uma extensão ao DOM. A aplicação de um estilo a um documento XML é disponibilizada através da interface *Node*, que possui os seguintes métodos: *transformNode* e *transformNodeToObject*. O primeiro método permite transformar a árvore abaixo de um nó com um determinado estilo. O nó que chama o método é o nó corrente que deve ser confrontado no XSLT. O desenvolvedor pode acessar qualquer parte do documento XML enquanto desenvolvendo estilos para formatação dos nós.

É importante frisar que o primeiro nó corrente do XSLT é o nó cujo o qual o desenvolvedor definiu o atributo FRH_ESTILO, e não o elemento raiz do documento "/".

O componente TFRH é composto pela classe TFRH, pelo componente MSXML3 e outras classes de suporte, como a classe TInterfaceArquivos, que dá suporte para a manipulação de diretórios e arquivos.

A seguir é mostrada a classe principal do componente TFRH, a TFRH. Apenas são listados os métodos de mais alto nível:

```
TFRH = class
 private
```

```
    procedure InsCodNodo_CriaFluxo_CriaListaPalChave_CopiaArqs();
    procedure LoopXML_CriaLinks(nodo: IXMLDOMNode);
    procedure LoopXML_aplicaXSL(nodo: IXMLDOMNode);

protected
    destructor Destroy; override;
    procedure AplicaEstilos();

public
    constructor Create(PXML_str, PXSL_path, PRelatorio_path,
                PTemplatesFRH_path: string; PFuncoesUnidadeConceito:
                TFuncoesUnidadeConceito; PFuncoesGlossario:
                TFuncoesGlossario);
```

Os três métodos privados listados constituem nos três loops pela árvore XML necessários para a realização das tarefas. São eles:

- `InsCodNodo_CriaFluxo_CriaListaPalChave_CopiaArqs:` lpassa pelos elementos XML reconhecendo os nós hipermídia marcados de acordo com o modelo da HAM. No reconhecimento insere um código para cada nó hipermídia, constrói o arquivo contendo a hierarquia do fluxograma e cria uma lista com o mapeamento de cada palavra chave para o código do respectivo nó hipermídia.
- `LoopXML_CriaLinks:` passa por todos os nós XML do tipo texto e cria os links contextuais implícitos (hotwords) a partir do mapeamento das palavras chaves realizados no método anterior.
- `LoopXML_aplicaXSL:` passa pelos nós XML que são nós hipermídia, aplicando a esses nós os estilos XSLT especificados pelo desenvolvedor.

Os métodos `Create` e `AplicaEstilos` consistem na interface que o desenvolvedor deverá utilizar para implementar a geração de relatórios em sua aplicação. Para criar uma instância da classe TFRH o desenvolvedor precisa especificar uma string contendo o documento XML; os diretórios dos estilos XSLT, do relatório hipermídia e dos templates FRH; as funções do elemento hipermídia unidade de conceito e as funções do elemento hipermídia glossário.

A instância do TFRH é carregado com uma série de propriedades com valores padrões. Caso o desenvolvedor mude algum nome, diretório ou relação entre os frames e janelas dos objetos do sistema hipermídia embutido nos templates FRH listados na sessão Sistema Hipermídia

dos Templates FRH, a instância de TFRH deve ter suas propriedades atualizadas para que o processamento dos templates FRH seja correto.

3.2.3 Características do Relatório Hipermídia

O sistema hipermídia gerado pode seguir a metáfora de frames, janelas ou mista, sendo adaptável neste ponto de acordo com as sessões anteriores. A arquitetura do sistema hipermídia segue apenas a arquitetura simples de três níveis, não seguindo portanto o modelo Dexter.

Os elementos de interface e de script utilizados nos templates FRH originais são totalmente compatíveis com a WWW e desenvolvidos por meio de tecnologias padrões, possibilitando alta escalabilidade de plataformas e de browsers, desde que estes suportem os padrões comuns da internet.

O desenvolvedor pode customizar os templates FRH padrões para suas próprias necessidades e também permitir a inclusão de mídias mais complexas, como os famosos flash e shockwave da Macromedia. O HTML tornou-se um grande conteiner de tecnologias multimídia. Através de plug-ins e de activeX's pode-se tornar um browser num poderoso visualizador de mídias.

3.4 O PAPEL DO DESENVOLVEDOR E DO USUÁRIO

O desenvolvedor têm por objetivo implementar, com a ajuda do FRH, os três níveis da arquitetura de um Sistema Hipermídia, definindo, se necessário, meios para customização em cada nível. O usuário do sistema criado pelo desenvolvedor, chamado de usuário desenvolvedor, utiliza-se dos meios disponibilizados pelo desenvolvedor para gerar e customizar o relatório hipermídia.

3.4.1 Papel no Nível de base de dados

PAPEL DO USUÁRIO DESENVOLVEDOR: Os usuários devem alimentar o banco de dados com as informações a serem utilizadas no Sistema Hipermídia.

PAPEL DO DESENVOLVEDOR: A modelagem do banco de dados pode ser feita de forma independente do FRH, não devendo se preocupar com a estrutura dos nós na HAM.

3.4.2 Papel no Nível da HAM

PAPEL DO USUÁRIO DESENVOLVEDOR: O usuário deve definir quais elementos hipermídia estarão presentes em seu relatório e quais funções serão atribuídas a eles. Para cada elemento hipermídia, o usuário também deve definir quais nós farão parte da hipermídia, o que significa alterar a estrutura dos elementos hipermídia. Este procedimento é normalmente suprido através de interfaces para filtro de registros nas bases de dados.

A gama de funções disponíveis para cada elemento hipermídia fica reduzida ao mapeamento prévio realizado na sessão 2.2.6.2 (Elementos de Hipermídia: Funções) e às funções atualmente implementadas, sendo que o desenvolvedor pode restringir ainda mais.

PAPEL DO DESENVOLVEDOR: O desenvolvedor deve mapear os dados do banco de dados para o modelo da HAM em XML. A informação deve ser mapeada em unidades de conceitos, unidades básicas de informação e unidades de diagramas, em uma granularidade definida pelo desenvolvedor.

As principais funções de sistemas hipermídia já estão implementadas nos templates FRH. Todavia, as funções podem ser estendidas, modificadas ou novas funções podem ser criadas.

3.4.3 Papel no Nível de Apresentação

PAPEL DO USUÁRIO DESENVOLVEDOR: O usuário deve definir os valores dos parâmetros para apresentação das funções dos elementos hipermídia e dos elementos de interface restantes (relativos a metáfora). Os parâmetros dependem da função e do tipo de elemento de interface, podendo ser, por exemplo, cor, fonte, tipo de mídia, fundo, etc.

PAPEL DO DESENVOLVEDOR: O desenvolvedor deve criar estilos XSLT para cada tipo de nó definido. Cabe ao desenvolvedor definir parâmetros (caso seja requerido) sobre os XSLT`s criados para que o usuário tenha a possibilidade de personalizar seu relatório.

Os elementos de interface da metáfora presentes nos templates FRH podem também ser modificados pelo desenvolvedor, tando em tempo de design como em tempo de execução.

3.5 O Modelo da HAM em XML

O modelo da HAM em XML consiste em um conjunto de regras e definições em linguagem natural que especificam como criar um documento XML que represente os nós e links hipermídia no formato utilizado pelo componente TFRH. Estas regras devem ser utilizadas pelo desenvolvedor para o mapeamento das informações do banco de dados para o modelo.

Este modelo não depende de como o nível de base de dados está modelado, pois este nível deve preocupar-se somente em prover as informações a serem utilizadas na hipermídia, e não como a hipermídia será estruturada e gerada.

3.5.1 Especificação do Modelo da HAM

O modelo da HAM em XML representa os nós totalmente definidos, encaixados dentro de uma hierarquia de navegação e com todas as informações necessárias para apresentação.

Nós, unidades básicas de informação e mídias são representados por elementos XML. Um nó hipermídia é representado através de um elemento XML. As unidades básicas de informação e unidades de diagramas são também elementos XML. O que difere um nó é a presença dos atributos específicos para nós. Estes atributos são listados na próxima sessão.

A princípio, todas as informações do nível de base de dados que devem ser apresentadas em um nó devem estar contidas no fragmento XML definido abaixo do elemento XML que representa o nó. Se esta regra fosse imposta, o seguinte decorreria:

- **Um elemento XML pode definir apenas um nó e esse nó é definido somente por esse elemento XML. Se dois nós, por exemplo, são definidos pelo mesmo elemento XML, apesar de utilizarem o mesmo conjunto de informações, estas informações terão de ser duplicadas.**
- **Um nó não pode apresentar informações que estejam fora do fragmento XML definido para o nó.**

Para flexibilidade, a regra acima é revogada, permitindo assim que um nó utilize todo o documento XML para recuperar informações e apresentá-las ao usuário. Contudo, continua

valendo a regra de que um elemento XML pode somente representar um nó e este nó é representado somente por este elemento.

O modelo hipermídia aqui apresentado não impõe, recomenda, apenas um tipo de nó para o elemento glossário, representado pelo elemento XML “Termo”. Este tipo de nó é composto pelos seguintes elementos XML: “Nome”, “Definicao” e “Midia”.

A recomendação para o glossário utiliza uma outra recomendação, o elemento “Midia”, que pode ser utilizado para descrever as mídias no conteúdo XML. Este elemento tem por padrão dois atributos: “tipo” e “path”, representando respectivamente o tipo ou tecnologia da mídia e o caminho relativo de onde encontrar a mídia.

Para as Unidades de Conceito não se pode definir uma estrutura padrão, pois variam imensamente de caso para caso. Portanto, é tarefa do desenvolvedor definir a estrutura deste elemento. Nada impede que ele o faça também para o Glossário.

3.5.1.1 Links

O modelo da HAM não define elementos para representação direta de links. Todos os links gerados pelo TFRH são implícitos e são possíveis de serem gerados graças a estrutura do modelo da HAM, que define palavras chaves para cada nó hipermídia. A partir destas palavras o TFRH pode gerar os links para os nós onde estas foram definidas.

O controle de links é um ponto importante do modelo. Nem sempre é desejado que uma palavra seja ligada a todas as possíveis palavras chave do conteúdo. O modelo permite a exata escolha de quais nós devem ser verificados para a construção de links no texto de cada nó XML. Esta propriedade foi chamada de escopo de link.

O desenvolvedor não sabe quais palavras o usuário de seu sistema digitou no banco de dados. Porém ele sabe a estrutura da hipermídia e a semântica a ela associada. Isto permite que ele identifique os grupos de nós mais relevantes que podem ser ligados a outro nó.

Qualquer nó XML pode ter o escopo de seus links definidos. O atributo FRH_LINKS é o responsável por prover esta funcionalidade. Este atributo deve ser suprido com uma expressão

XPath, mais especificamente, um location path (caminho de localização) relativo ao nó XML cujo texto pode conter links. O location path deverá definir um conjunto de nós XML contendo as palavras chave que serão confrontadas com o texto do nó de origem.

Os links podem ser também completamente desativados através da atribuição de um valor nulo ao atribulo FRH_LINKS, da seguinte maneira: FRH_LINKS="".

Um nó XML pode ter somente um atributo FRH_LINKS. Isto significa que não é possível utilizar o texto de um elemento com diferentes escopos de links. Neste caso a informação deve ser duplicada para cada escopo existente.

Se o mesmo nó de texto deve ter diferentes escopos de link, e este é utilizado em dois ou mais nós hipermídia, é recomendado que o texto seja duplicado em cada um dos nós hipermídia.

Duplicar a mesma informação no mesmo nó XML, ao contrário do que acontece com os nós hipermídia, é no mínimo deselegante. É natural que um desenvolvedor necessite incluir, no mesmo nó hipermídia, a mesma palavra ou expressão com e sem links. Neste caso, o recomendado é a utilização de css para desativar a forma visual do link, se este existir, nos lugares onde a palavra não deve aparecer como um link, como por exemplo um título. Ainda assim o link estará ativo ao usuário, o que não traz problemas relevantes de usabilidade.

Por outro lado é improvável que os links de uma mesma palavra tenham diferentes escopos dentro de um mesmo nó. Se extremamente necessário, a informação pode ser duplicada no nó XML.

3.5.1.2 Especificação Completa dos Elementos e Atributos

A seguir é apresentada a especificação completa dos elementos e atributos XML que consistem no modelo da HAM:

- **Um único XML para todos os elementos hipermídia a serem contemplados;**
- **Vocabulário XML para o modelo da HAM:**

ELEMENTO	DESCRIÇÃO
<FRH_ROOT>	Elemento raiz do documento xml. (outros nomes podem ser usados)
<FRH_GLOSSARIO>	Elemento XML que representa o elemento hipermídia Glossário. (fixo)
<FRH_UNIDADE_CONCEITO>	Elemento XML que representa o elemento hipermídia Unidades de Conceito. (fixo)

- **Atributos específicos para caracterização de nós:**

<FRH_CODNODO>	Código gerado pelo TFRH para cada nó hipermídia.
FRH_TITULO	Atributo XML que contém o título de um nó. Esta informação aparecerá como título dos nós, além de ser utilizada para criação de fluxogramas e mapas conceituais. É obrigatório para todos os nós.
FRH_PALAVRASCH	Atributo XML que contém as palavras ou expressões (conceitos) chaves relevantes para um nó. As palavras ou expressões devem estar separadas pelo caracter "\" (barra invertida). Esta informação é utilizada para criação de hotwords para os nós, criação de índices de palavras chaves e para a busca. Este atributo é opcional.
FRH_GRUPO	Atributo que especifica se o elemento XML deve ser denido como um conteiner de nós. Esta informação é utilizada pelo fluxograma para agrupar um conjunto de nós. Nós com o atributo FRH_ESTILO definido são automaticamente tratados como grupos se contiverem elementos filhos, portanto este atributo não deve ser utilizado em conjunto com o FRH_ESTILO. Não importa o valor do atributo. Se este atributo existe então o nó é automaticamente tratado como grupo. Este atributo é opcional para os nós XML.
FRH_ESTILO_INDICE_GLOSSARIO	Atributo específico para o elemento XML FRH_GLOSSARIO. Este atributo deve especificar o estilo XSLT a ser utilizado para construção do índice do glossário. O valor padrão é "FRH_glossario_indice"

- **Atributo para definição do escopo dos links:**

FRH_LINKS	Define um conjunto de nós que devem ser verificados para criação de links. O valor deve ser uma expressão location path, para definir o conjunto de nós, ou vazio, para desativar links no nó em questão. Este atributo pode ser aplicado a qualquer nó XML. Se não definido, o valor padrão utilizado é o elemento do documento XML.

O desenvolvedor deve tomar cuidado para não utilizar os nomes dos elementos e atributos acima com outra finalidade senão a descrita.

Os estilos devem ser especificados sem a extensão ".xsl". Esta deve ser a extensão para todos os estilos XSLT.

3.6 Outras Especificações e Detalhes de Implementação

Os links no nível de ancoramento são criados inserindo-se a tag HTML:

> Texto do link

Onde:

- **elemento hipermídia** = nome do elemento hipermídia destino, por exemplo "glossario" ou "unidade_conceito". Se houver mais de um link o elemento hipermídia terá o valor "multiplo".
- **array de títulos** = array contendo uma lista com um título ou uma descrição breve para cada nó destino do link.
- **array de destino** = array contendo uma lista com as páginas destino de cada link, na mesma ordem que o array de descrição.
- **array de tipos** = tipo do link, representado pelo tipo do destino. Atualmente existem os tipos: "glossario" e "unidade_conceito".

Links não são feitos em palavras que sejam atributos de elementos XML.

A adaptação do mapa global pode ser feita através da edição do documento HTML fluxograma.html e seus estilo css correspondentes. Parâmetros extras da árvore de conteúdo

podem ser especificados utilizando-se o arquivo de configuração da árvore localizado em "scripts/fluxograma_tree_format.js".

3.7 METODOLOGIA

Esta sessão lista as etapas da metodologia utilizada para o desenvolvimento deste trabalho. A metodologia utilizada é uma fusão das seguintes áreas da Ciência da Computação:

- Sistemas Hipermídia
- Engenharia de Software

Tomadas de decisões críticas foram inseridas em determinados pontos da metodologia. A seguir tem-se as fases e etapas da metodologia utilizada:

Fase 1: Estudo de Viabilidade

- Levantamento de tecnologias utilizáveis para a solução do problema;
- Revisão bibliográfica e estudo introdutório das tecnologias afim de confirmar a viabilidade do sistema proposto;
- Levantamento do estado da arte;
- Mapeamento geral da solução e criação da proposta do projeto;

Fase 2: Pesquisa e estudo dos conceitos e tecnologias

- Hipermídia;
- Hipermídia Adaptativa**;**
- Documentos Eletrônicos;
- XML, XSL e tecnologias relacionadas;
- Estudo do potencial do par HTML / JavaScript;
- Banco de Dados Interbase e da interface deste com a aplicação, incluindo componentes de acesso via Ambiente Delphi;
- Ferramentas e API's a serem utilizadas com o Ambiente Delphi.

Fase 3: Elaboração da Metodologia

- Estudo da interação entre as metodologias de desenvolvimento de Sistemas Hipermídia e de Engenharia de Software;
- Adaptação e refinamento da metodologia inicial proposta.

Fase 4: Planejamento e Especificação

- Entrevista com possíveis usuários, definindo os requisitos do usuário;
- Mapeamento dos elementos comuns de sistemas hipermídia
  - Definição dos elementos estruturais;
  - Definição dos elementos funcionais;
  - Definição dos elementos de apresentação;
  - Validação da especificação dos elementos funcionais e de apresentação levando em conta as restrições impostas pelo HTML e JavaScript.
- Definição do Modelo da HAM através da especificação da estrutura dos documentos XML (pode ser usado DTD's ou Schemas);
- Planejamento e especificação da interface das classes do FRH. Se necessário pode-se usar UML;

Fase 5: Implementação

- Implementação do sistema hipermídia para os Templates FRH em Javascript;
- Implementação das classes do FRH;

- Aplicação do FRH para o Sistema PCMAT;
  - Definição de quais recursos serão usados para ajudar na construção da Hipermídia;
  - Determinação dos meios e padrões;
    - Criação do layout padrão do PCMAT-Hipermídia;
  - Inclusão das tabelas no banco de dados e modificação das existentes para suprir os requisitos do usuário. Utilização do Modelo Entidade-Relacionamento e técnicas de normalização;
  - Criação dos estilos (XSLT) para apresentação do PCMAT-Hipermídia;

- Criação de um módulo para o relatório hipermída;

Fase 6: Teste

- Aplicação do módulo FRH no Sistema PCMAT;
- Nesta fase, serão confeccionadas algumas mídias teste e cadastro de informações no banco, afim de gerar relatórios e validar o sistema.

Fase 7: Documentação do trabalho

CAPÍTULO 4 – Validação do FRH: PCMAT Hipermídia

A validação do Framework para Relatórios Hiperídia – FRH se dá através do Sistema PCMAT. O objetivo é criar um relatório hipermídia tendo como conteúdo o PCMAT. O Sistema PCMAT já possui a funcionalidade de cadastro das informações e filtro para consulta de uma parcela de informação.

A criação do PCMAT Hipermídia se deu através das etapas da metodologia para desenvolvimento de sistemas hipermídia, utilizando o FRH para agilizar a implementação do sistema. As sessões seguintes descrevem as etapas importantes para o estudo de caso em questão, descrevendo em cada uma as ações realizadas e como o FRH ajudou na solução dos problemas.

Os apêndices I,II e III mostram as principais telas do Sistema PCMAT e do PCMAT Hipermídia, enquanto os anexos III e IV mostram o código desenvolvido.

4.1 PREPARAÇÃO

4.1.1 Usuários e Objetivos

Etapas da metodologia:

- Determine quem irá ler o documento.
- Determine os objetivos do documento.
- Determine quem irá criar o documento.

Estas etapas foram resolvidas diretamente com o especialista no domínio e co-orientador deste trabalho, prof. Juan W. Moore Espinoza. Os usuários do sistema hipermídia poderão ser os seguintes profissionais:

- Engenheiro Civil;
- Médico do trabalho;
- Engenheiro de segurança do trabalho;
- Técnico de segurança;
- Auxiliar de enfermeiro do trabalho;
- Profissionais da área de Engenharia Civil

- Enfermeiro do trabalho

Os seguintes objetivos foram definidos:

Com o PCMAT Hipermídia deve ser possível:

- A apresentação do trabalho realizado pelo Engenheiro de Segurança do Trabalho à entidade contratante ou a terceiros.
- Melhor compreensão e treinamento mais eficiente dos profissionais encarregados da aplicação do PCMAT no processo de construção de uma obra.
- Maior facilidade na recuperação da informação pelos profissionais envolvidos;
- O relatório gerado deve ser portável;

O autor deste trabalho é o responsável pela criação do sistema hipermídia. Como o FRH é um sistema que gera um sistema hipermídia, deve-se também definir quais usuários terão a responsabilidade de gerar os relatórios hipermídia. No caso, somente o engenheiro de segurança do trabalho pode ter esta responsabilidade. Os usuários que têm essa responsabilidade, como visto na sessão "Papel do desenvolvedor e do usuário", são chamados de usuários desenvolvedores. O usuário final do sistema hipermídia gerado será chamado daqui em diante somente de usuário.

4.1.2 Recursos usados para ajudar o(s) autor(es)

Especialista no assunto: o autor do projeto contou com o apoio de seu co-orientador.
Em relação ao usuário desenvolvedor, os seguintes recursos são disponibilizados automaticamente pelo FRH, uma vez que este roda como um módulo do Sistema PCMAT:

- **Recursos de software.**
  - Sistema de Informação especializado para o PCMAT.
- **Recursos humanos.**
  - Artista gráfico: Design gráfico do sistema hipermídia não é necessário, pois já foi realizado pelo desenvolvedor.
  - Especialista no assunto: o próprio usuário desenvolvedor é o especialista.
- **Recursos de informação.**

- Glossário Central: termos técnicos da base de dados padrão do Sistema PCMAT.
- Coleção central de acrônimos: pode ser implementado através do glossário.
- Coleção central de unidades de conceito: conjunto de etapas do processo construtivo e outras informações padrões da base de dados do Sistema PCMAT

Coleção central de diagramas.

- Incorporação gradual de mídias nos formatos de imagem, animação 2D (flash e shockwave) e mídias 3D (shockwave 3D), interativas ou não. Vídeos também podem ser inseridos através do shockwave.

4.1.3 Meio e padrões

Uma vez estabelecidos os padrões gráficos, estes não poderiam ser facilmente modificados em um sistema hipermídia comum. A arquitetura do FRH porém, permite a especificação da apresentação totalmente desvinculada do conteúdo. Assim, modificações são muito fáceis de serem efetuadas.

O desenvolvedor, conta com o FRH para fornecer a interface e o sistema hipermídia padrão. Modificações não são necessárias, pois a interface padrão do FRH é a mesma do PCMAT. Assim sendo, o estabelecimento dos padrões gráficos fica reduzido à criação dos estilos XSLT.

As facilidades de microinformática disponibilizadas ao usuário desenvolvedor, e por conseguinte ao usuário final são: fluxograma, busca por palavras chave, busca por frases (full text search), ativação e desativação de links, impressão e navegação hierárquica.

Com relação ao usuário desenvolvedor, todos os itens abaixo são herdados do desenvolvedor e já foram resolvidos, não sendo permitido mudanças na interface:

- Estabeleça que facilidades de microinformática o leitor irá usar.
- Estabeleça os padrões gráficos.
- Estabeleça que dispositivos externos de vídeo serão usados, se houver algum.
- Estabeleça as orientações de exibição.
- Estabeleça as orientações de tipologia.

Se o software requisesse, o desenvolvedor poderia incluir parâmetros para a customização do sistema hipermídia pelo usuário desenvolvedor.

4.1.4 Determinação dos itens do assunto contemplados

Etapas da metodologia:

- Defina quais os itens que devem estar no documento.
- Determine como o documento pode atingir os problemas, necessidades e questões dos usuários.
- Crie uma estrutura inicial de documento.
- Revise o conteúdo e estrutura propostos com usuários potenciais.
- Determine melhorias possíveis na estrutura.

O item que deve ser implementado é, a priori, somente o relatório PCMAT relativo as etapas do processo construtivo, sendo este uma especificação das medidas de segurança e definição dos riscos de cada etapa de uma obra.

Não é possível cumprir todos os objetivos propostos sem antes alimentar o PCMAT com boa parte da informação necessária ao seu funcionamento, inclusive mídias. Espera-se que com a aplicação das funcionalidades do FRH, a maior parte dos objetivos seja alcançada.

Um trecho da estrutura inicial do documento hipermídia é apresentada a seguir. O documento está conforme o modelo da HAM em XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<FRH_ROOT>
<FRH_UNIDADE_CONCEITO FRH_GRUPO="sim" FRH_TITULO="Relatório de Medidas
Preventivas do Processo Construtivo">
<NListaEtapas FRH_ESTILO="lista_etapas" FRH_TITULO="Etapas do Processo
Construtivo" FRH_PALAVRASCH="Etapas do Processo Construtivo, Processo
Construtivo" FRH_ELEMENTO_CONTEUDO="parent::*" FRH_CODNODO="1">
      <etapa FRH_ESTILO="etapa" FRH_TITULO="CARPINTARIA"
FRH_PALAVRASCH="CARPINTARIA" FRH_CODNODO="2">
            <numero>1</numero>
            <nome>CARPINTARIA</nome>
```

```
            <atividade FRH_ESTILO="atividade" FRH_TITULO=" TRABALHOS COM
SERRA CIRCULAR" FRH_PALAVRASCH=" TRABALHOS COM SERRA CIRCULAR"
FRH_CODNODO="3">
                  <numero>1</numero>
                  <nome> TRABALHOS COM SERRA CIRCULAR</nome>
                  <risco FRH_ESTILO="risco" FRH_TITULO="Iluminação
inadequada" FRH_PALAVRASCH="Iluminação inadequada" FRH_CODNODO="6">
                        <numero>3</numero>
                        <nome>Iluminação inadequada</nome>
                        <Midia tipo="" path="O risco é um acidente de
trabalho que pode ocorrer em uma obra"/>
                  </risco>
            </atividade>
      </etapa>
      <etapa FRH_ESTILO="etapa" FRH_TITULO="ORDEM E LIMPEZA"
FRH_PALAVRASCH="ORDEM E LIMPEZA" FRH_CODNODO="7">
            <numero>2</numero>
            <nome>ORDEM E LIMPEZA</nome>
      </etapa>
</NListaEtapas>
</FRH_UNIDADE_CONCEITO>
<FRH_GLOSSARIO FRH_GRUPO="sim"
FRH_ESTILO_INDICE_GLOSSARIO="FRH_glossario_indice" FRH_TITULO="GLOSSÁRIO">
<Termo FRH_ESTILO="FRH_glossario_termo" FRH_TITULO="RISCO"
FRH_PALAVRASCH="RISCO" FRH_CODNODO="8">
      <Nome>RISCO</Nome>
      <Definicao>O risco é um acidente de trabalho que pode ocorrer em uma
obra</Definicao>
      <Midia tipo="imagem" path="midias\GLOSSARIOcoisa.jpg"/>
</Termo>
</FRH_GLOSSARIO>
</FRH_ROOT>
```

Como visto no capítulo 2, a estruturação de documentos hipermídia sob a forma hierárquica trás benefícios e é fácil de ser realizada.

Uma vez que a estrutura já foi previamente definida, o usuário desenvolvedor deve se preocupar apenas em definir quais os elementos que farão parte da hipermídia final.

4.2 Aplicação do FRH

Antes de proseguir com a metodologia de sistemas hipermídia é necessário incluir a etapa de aplicação do FRH relativa somente ao desenvolvedor, com as seguintes atividades:

- Inclusão da tabela GLOSSARIO no banco de dados e modificação das existentes para suprir os requisitos de múltiplos tipos de mídia.
- Criação de um módulo para o relatório hipermída;

A tabela GLOSSARIO foi inserido no banco de dados Interbase utilizado pelo PCMAT. O SLQ de criação desta tabela é o seguinte:

```
CREATE TABLE "GLOSSARIO"
(
  "ID_TERMO_GLOSSARIO"  INTEGER NOT NULL,
  "TX_TERMO"      VARCHAR(200) CHARACTER SET ISO8859_1 NOT NULL,
  "TX_DEFINICAO"  BLOB SUB_TYPE TEXT SEGMENT SIZE -1 CHARACTER SET
ISO8859_1 NOT NULL,
  "TX_PATH_MIDIA" VARCHAR(100) CHARACTER SET ISO8859_1,
CONSTRAINT "PK_GLOSSARIO" PRIMARY KEY ("ID_TERMO_GLOSSARIO")
);
```

O cadastro do glossário foi adicionado ao PCMAT utilizando para isso os componentes de acesso ao banco de dados Interbase do Delphi, como o IBQUERY.

Também foi criado um módulo para a construção do XML, também responsável por salvar as mídias dos nós no diretório do relatório e por acessar o componente TFRH. Este módulo pode ser visto no anexo III.

Uma tela de visualização de mídias foi também criada para permitir a visualização de mídias em flash e shockwave. Para tanto foram utilizados controles activeX.

4.3 CRIAÇÃO

Etapas da metodologia:

- Projete o documentos.
- Crie o documento.
- Revise e melhore o documento.
- Teste na prática e refine o documento.

Tudo o que o desenvolvedor deve fazer neste ponto é a criação dos estilos XSLT para apresentação dos nós do PCMAT Hipermídia. Os estilos produzidos podem são encontrados nos anexos ao final deste trabalho. Os testes e possíveis correções de interface são facílimas de serem realizadas. O desenvolvedor nem precisa gerar o executável de sua aplicação para gerar um sistema hipermídia com layout totalmente diferente, bastando para isso editar os estilos e até mesmo a interface dos templates FRH. Os estilos criados para o PCMAT Hipermídia podem ser vistos no anexo IV.

Foram também confeccionadas algumas mídias de teste e feito o cadastro de dados reais no banco, afim de gerar relatórios e validar o sistema.

Uma vez rodando, o FRH disponibiliza o sistema hipermídia padrão ao usuário desenvolvedor para que este o crie.

Glossário

Desenvolvedor: é o profissional responsável por integrar o FRH a um sistema aplicativo que deverá ser posteriormente utilizado por um usuário desenvolvedor. O termo desenvolvedor é utilizado de modo geral, representando todos os profissionais que deverão utilizar o FRH para construir um aplicativo. Dentre os profissionais pode-se citar o cientista da computação e o artista gráfico.

Usuário desenvolvedor: é o usuário que irá utilizar o aplicativo no qual o FRH foi embutido. Este usuário tem a responsabilidade de criar relatórios hipermídia utilizando-se das facilidades oferecidas pela aplicação.

Programa de Condições e Meio Ambiente do Trabalho na Indústria da Construção: É um documento que contém a especificação da política de segurança em uma obra de Construção Civil.

Sistema PCMAT: Software aplicativo para geração do PCMAT.

PCMAT-Hipermídia: Relatório Hipermídia para apresentação do PCMAT.

Conclusões

O FRH mostrou-se útil na agilização do processo de criação e manutenção de sistemas hipermídia de conteúdo estruturável em uma hierarquia de conceitos. Assim, sistemas hipermídia podem não apenas serem criados mais rapidamente, como também sua manutenção e modificação tornam-se muito mais simples.

As informações disponíveis em bancos de dados podem facilmente ser disponibilizadas através de um sistema de recuperação de informação (relatório hipermídia), aproveitando melhor o potencial de utilização desses dados.

A arquitetura utilizada no FRH e o modelo da HAM em XML podem ser utilizados para outras classes de sistemas hipermídia. A estruturação em XML permite também o fácil processamento do conteúdo por outras aplicações.

Agradecimentos

Agradeço a Deus por dar a oportunidade, todos os dias, de me superar.

Agradeço também minha família e amigos pelo grande apoio e demonstração de carinho que me deram, mesmo nas horas difíceis.

Aos professores da Universidade Federal de Santa Catarina que se preocupam com um ensino de qualidade para todos, meu muito obrigado.

Referências Bibliográficas

[Martin, 1992] Matin, James. **Hiper Documentos e Como Criá-los**; tradução de Marcelo Bernstein. Rio de Janeiro: Campus, 1992.

[Legislação, 2001] **Manual de Legislação: Lei N6.514 - 22 de dez. de 1977**. São Paulo: Atlas, 2001. 48ª Edição.

[Nielsen, 1995] Nielsen, Jakob. **Multimedia & Hipertext: the Internet and Beyond**. New York: Academic Press, Inc., 1995.

[Reed; Burton; Liu, 1994] Reed, W. Michael; Burton, John K.; Liu, Min. **Multimedia and Magachange**. New York: The Haworth Press, Inc., 1994.

[Willrich, 1999] Willrich, Roberto. **Apostila INE 5375 Multimídia** / Roberto Willrich - 1999.

[W3C, 1999] Word Wide Web Consortium. **XSL Transformations (XSLT) Version 1.0** Disponível por WWW em: http://www.w3.org/TR/1999/REC-xslt-19991116 (20 nov. 2002).

[W3C, 1999 a] Word Wide Web Consortium. **XML Path Language (XPath) Version 1.0** Disponível por WWW em: http://www.w3.org/TR/1999/REC-xpath-19991116 (20 nov. 2002).

[W3C, 2000] Word Wide Web Consortium. **Extensible Markup Language (XML) 1.0 (Second Edition) - W3C Recommendation**. Disponível por WWW em: http://www.w3.org/TR/REC-xml (7 set. 2002).

[W3C, 2000 b] Word Wide Web Consortium. **What is the Document Object Model?**. Disponível por WWW em: http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/introduction.html (08 set. 2002)

[W3C, 2000 c] Word Wide Web Consortium. **Document Object Model Core**. Disponível por WWW em: http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/core.html (08 set. 2002)

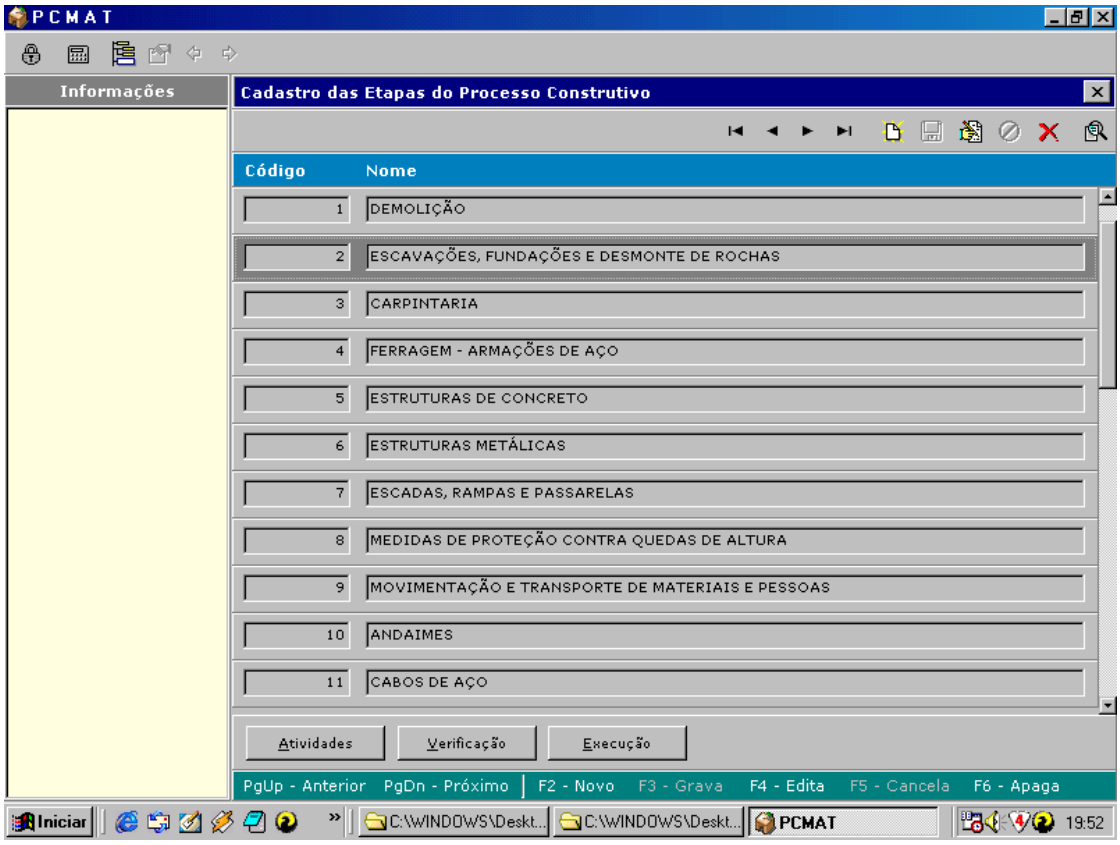
[W3C, 2001] Word Wide Web Consortium. **XML Schema Part 0: Primer - W3C Recommendation**. Disponível por WWW em: http://www.w3.org/TR/xmlschema-0/ (8 set. 2002).

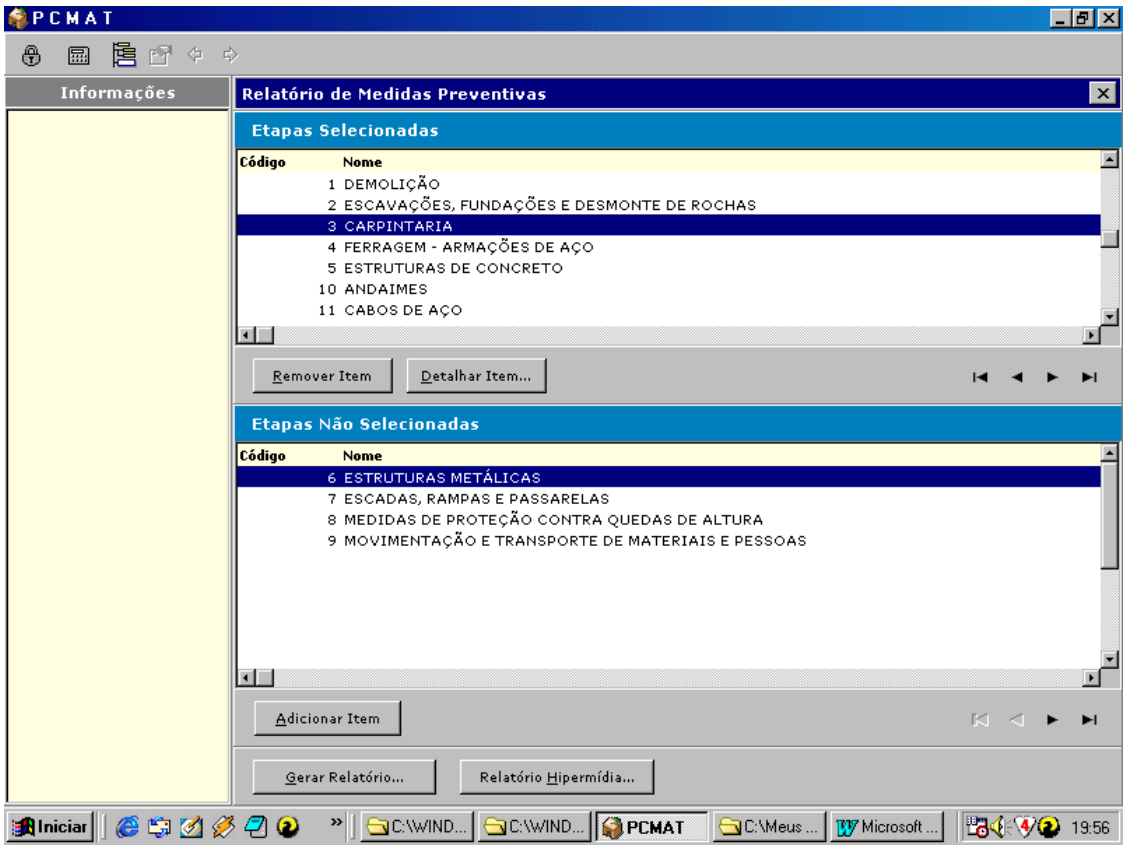
[Marchal, 1999] Marchal, Benoit. **XML BY EXAMPLE**. Indianapolis: QUE, 1999.

[Megginson, 2002] Megginson, David. **SAX Project**. Disponível por WWW em: http://www.saxproject.org (8 set. 2002).

Apêndices

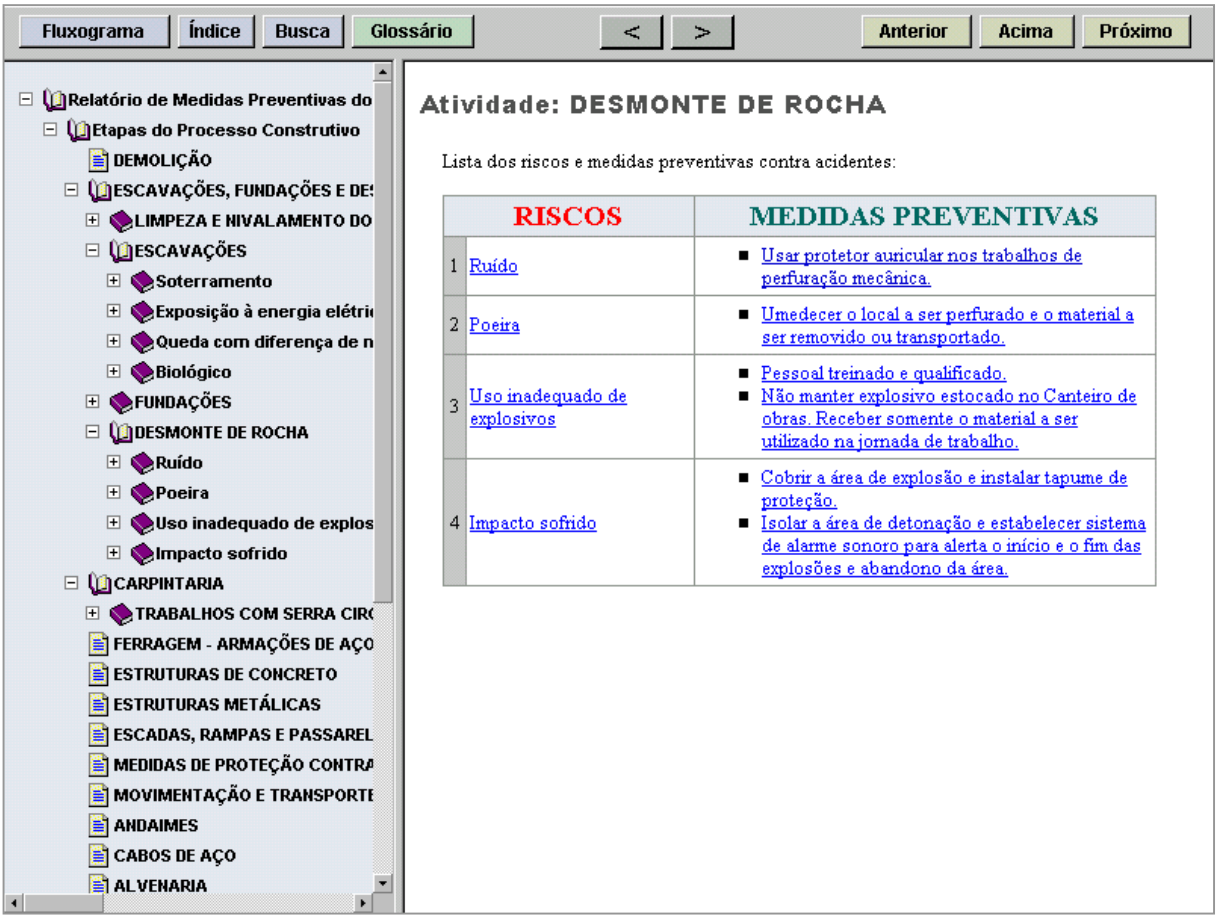
I. Telas de cadastro e geração de relatórios do Sistema PCMAT





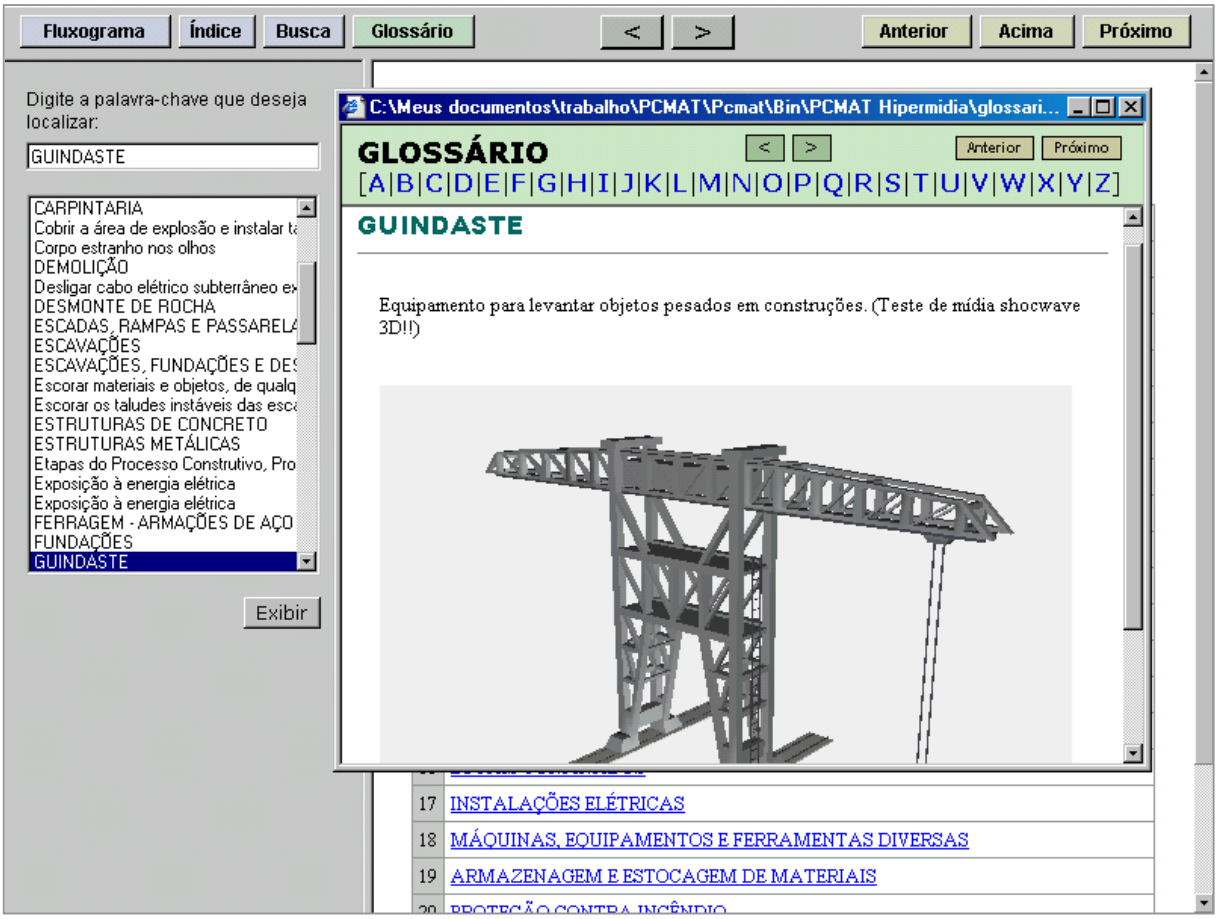
II. Tela Principal do PCMAT Hipermídia

Na tela aparece o Fluxograma (frame esquerdo) e um nó de Unidade de Conceito (frame direito).



III. Tela do PCMAT Hipermídia

Janela do Glossário exibindo um termo com uma mídia 3D interativa. Ao fundo (frame esquerdo) encontra-se o índice de palavras, o qual foi utilizado, nesta exemplo, para abrir o termo “guindaste”.



IV. Artigo do trabalho

Um Framework para Geração de Relatórios Hipermídia

Thiago Linhares de Oliveira, Roberto Willrich

Depto de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brazil

{linhares,willrich}@inf.ufsc.br

Abstract. *This work proposes a Framework for Generation of Hypermedia Reports (FHR) from documents stored in databases. The FHR follows principles of hypermedia systems architecture proposed by Campbell and Goodman and also the reference model proposed by Dexter group. FHR's utilization process is basically constituted on the mapping between non hypermedia documents stored in a database and a XML model, and the creation of presentation specifications through XSLT styles (eXtensible Stylesheet Language Transformation). FHR is intended to help to implement applications which are capable of supplying completely autonomous hypermedia reports.*

Resumo. *Este trabalho propõe um Framework para geração de Relatórios Hipermídia (FHR) a partir de documentos armazenados em banco de dados. O FRH segue os princípios da arquitetura para sistemas hipermídia proposto por Campbell e Goodman e também o modelo de referência proposto pelo grupo Dexter. O processo de utilização do FRH consiste basicamente no mapeamento de documentos não hipermídia armazenados em uma base de dados para um modelo no formato XML e na criação de especificações de apresentação feitas através de estilos XSLT (eXtensible Stylesheet Language Transformation). O FRH visa auxiliar na implementação de aplicações que sejam capazes de prover sistemas hipermídia completamente autônomos.*

1. Introdução

Um relatório é constituído de informações em bases de dados que são processadas e inseridas em documentos eletrônicos ou impressas em papel para desempenho de funções diversas, tais como documentação, treinamento, suporte a tomada de decisão, controle, manuais, etc. O PCMAT – Programa de Condições e Meio Ambiente do Trabalho na Indústria da Construção – é o relatório utilizado para estudo de caso.

Relatórios em papel ou em formatos de documento eletrônico para editores de texto podem não ser suficiente para alcançar os objetivos de certos relatórios. Dependendo dos objetivos de um relatório, a hipermídia, ou um sistema hipermídia, pode ser a melhor escolha de formato, daí o nome relatório hipermídia. Muitas facilidades como busca por conteúdos, navegação, interatividade e elementos multimídia podem ser acrescidas a um relatório através do uso consciente da Hipermídia. Várias experiências comprovaram que a localização de uma informação específica pode ser agilizada com o uso dessa tecnologia. Neste trabalho o termo *sistema hipermídia* é utilizado de forma similar a *relatório hipermídia* e vice-versa.

O presente trabalho propõe um framework para geração de relatórios Hipermídia, chamado de FRH – Framework para Relatórios Hipermídia. O processo de utilização do FRH consiste basicamente no mapeamento de documentos não hipermídia armazenados em uma base de dados para um modelo no formato XML [W3C, 2000] e na criação de especificações de apresentação feitas através de estilos XSLT (eXtensible Stylesheet Language Transformation) [W3C, 1999].

A implementação do FRH contempla os seguintes requisitos gerais:

- A arquitetura do FRH deverá permanecer extensível e independente de aplicação.
- O FRH deverá agilizar o cumprimento das etapas das metodologias para construção de documentos hipermídia.
- O FRH foi desenvolvido para servir como um módulo para aplicações que devem gerar relatórios hipermídia, sendo que estas aplicações devem prover os dados necessários para o funcionamento do framework.

O restante deste artigo está organizado na forma que segue. A seção 2 apresenta as duas arquiteturas nas quais o FRH se baseia. Em seguida, a seção 3 apresenta os elementos básicos encontrados em sistemas hipermídia. A seção 4 apresenta o FRH, sua arquitetura, modelos e funcionamento. Finalmente, a seção 5 apresenta um estudo de caso que acaba por gerar o PCMAT Hipermídia.

2. Arquiteturas de Sistemas Hipermídia

Sistema hipermídia podem ser estruturados em níveis. Existem diversas arquiteturas, as quais definem diferentes níveis e funções para cada nível. Esta sessão apresenta os níveis e as funções das duas arquiteturas utilizadas neste trabalho.

2.1. Arquitetura de Campbell e Goodman (1988)

Segundo Campbell e Goodman (1988) pode-se distinguir três níveis em um Sistema Hipermídia: nível de base de dados, nível da HAM e nível de apresentação.

Nível de base de dados

Este nível é responsável pelo armazenamento da informação a ser apresentada pelo sistema. Não importa como a informação é armazenada, mas deve ser possível recuperar rapidamente uma parte da informação. Ao utilizar um banco de dados, relacional por exemplo, no nível de base de dados, funcionalidades como controle de acesso multiusuário, backup e segurança são incorporadas sem grande esforço a sistemas hipermídia.

Nós e links no nível de base de dados não apresentam significado algum, sendo este nível apenas um conjunto de dados. Porém, é interessante que os dados sejam armazenados em estruturas que representem alguma semântica, facilitando o trabalho de interfaceamento com o nível da HAM.

Nível de HAM (*Hipertext Abstract Machine* - Máquina Abstrata de Hipertexto)

A HAM se localiza entre o nível de base de dados e o nível de apresentação. Este nível é responsável pela gerência de nós e links. Nós são pedaços do conteúdo total da hipermídia apresentados ao usuário em um dado momento. Links conectam nós com alguma semântica em comum.

Informações no nível de base de dados são fáceis de transferir de sistema para sistema. Já a transferência de nós e links é mais complexa. Assim é importante a existência de padrões

para a especificação e implementação deste nível, permitindo maior reusabilidade. Uma solução é utilizar uma linguagem de metadados e um modelo que represente a rede de nodos mantida pela HAM. Nielsen [1995] sugere que a HAM deve possuir conhecimento sobre a forma dos nós e links. O XML serve muito bem ao propósito da HAM. O modelo da HAM desenvolvido para o FRH é apresentado na sessão 3.5.

Nível de apresentação

A interface do usuário é responsável pela apresentação da informação da HAM, incluindo questões do tipo: quais comandos devem estar disponíveis para o usuário, como mostrar nós e ligações e se deve ou não incluir diagramas [Nielsen, 1995]. Neste nível controla tanto a apresentação dos nodos e links como também da interface do sistema hipermídia.

2.2 Arquitetura de Dexter

Outra arquitetura muito semelhante à de três níveis foi proposta pelo grupo Dexter [Halasz e Schwartz, 1994]. A principal diferença consiste na especificação de duas interfaces entre os níveis: ancoramento e especificações de apresentação.

Ancoramento (anchoring)

Este nível serve para a definição das ancoras dos links nos nós. A arquitetura simples de três níveis não permite a gerência do ancoramento dos links pela HAM, uma vez que este depende do formato de armazenamento da mídia no nível de base de dados. O nível de ancoramento devolve a gerência do ancoramento dos links à HAM, através do mapeamento de identificadores de ancoras na HAM para seus reais valores nos nós.

Especificações de apresentação

Este nível guarda formas e estilos para apresentação de nós e links. Através das especificações de apresentação, um conjunto de dados de um nó pode, por exemplo, ser apresentado através de um gráfico, tabela ou texto plano. Também é possível controlar como os links deverão ser apresentados ao usuário.

A Figura 1 mostra as duas arquiteturas de forma comparativa [Nielsen 1995].

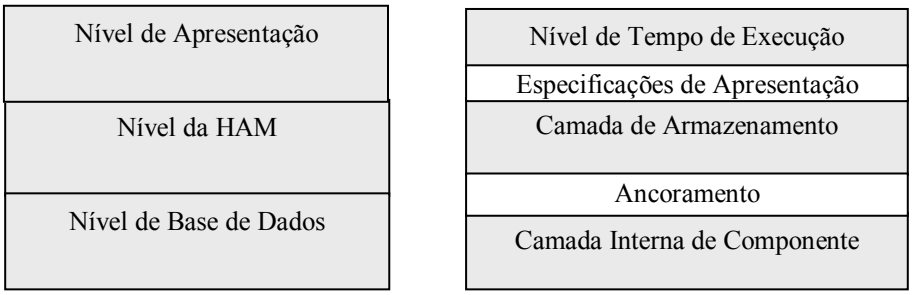


Figure 1. Arquitetura de Sistemas Hipermídia: três níveis (à esquerda) e Dexter (à direita)

3. Elementos Essenciais em Relatórios Hipermídia

Relatórios hipermídia possuem elementos essenciais que são freqüentemente encontrados na maioria dos sistemas hipermídia.

Elementos de Sistemas Hipermídia, ou elementos hipermídia, especificam a informação, as funcionalidades e as características de apresentação que o sistema fornece ao usuário. Os elementos mais encontrados são citados a seguir:

- Coleção de Unidades de conceito: coleção de módulos tutoriais relacionados ao conteúdo do Sistema Hipermídia;
- Glossário, Lista de definição de acrônimos;
- Coleção de diagramas e elementos Multimídia;
- Coleção de referências;
- Coleção de informações gerais, como revendedores, endereços, pessoas a serem contatadas, etc;

Os elementos hipermídia possuem três propriedades: estrutura e funções, relativas ao nível da HAM, e apresentação, obviamente relativa ao nível de apresentação.

A estrutura de um elemento hipermídia refere-se a forma como os nós são organizados. Neste trabalho, a hierarquia de conceitos é utilizada como base para a definição da estrutura.

Cada elemento hipermídia possui funções intrínsecas. A seguir são listadas as funções mais comuns que podem ser aplicadas aos dois principais elementos hipermídia: unidades de conceito e glossário. São elas:

- Navegação seqüencial pelos nós do elemento;
- Navegação para o nó imediatamente superior ao nó corrente (nó pai);
- Índice dos nós;
- Histórico de caminhos percorridos:
- Navegação seqüencial pelo caminho percorrido através de backtracking ou avanço para o próximo nó do caminho;
- Índice de nós do histórico;
- Índice de nós do histórico;
- Links conectando palavras ou expressões a outros nós (hotwords);
- Busca por nós do elemento, podendo ser uma busca parametrizada ou busca por palavras;

3. Framework para Relatórios Hipermídia

O FRH é um framework com uma arquitetura extensível que dá suporte a geração de relatórios hipermídia a partir de bases de dados quaisquer. Esta seção apresenta o FRH.

3.1. Arquitetura e Funcionamento

A arquitetura do FRH, mostrada na figura 2, segue a arquitetura de três níveis para Sistemas Hipermídia e, em um grau mais detalhado, segue também a arquitetura Dexter. Entretanto, algumas diferenças sutis existem e devem ser anotadas. O nível de tempo de execução do FRH (o relatório hipermídia) é na verdade outro sistema hipermídia, com seus próprios níveis da arquitetura simples de três níveis e totalmente independente da aplicação que o gera, podendo, por exemplo, ser colocado em um cd para distribuição. Ao invés de apresentar um nó de cada vez, como fazem os sistemas hipermídia comuns, o FRH gera, não apresenta, todos os nós do conteúdo de uma só vez.

O TFRH não implementa nem as funções de elementos hipermídia nem outras funções de controle do sistema hipermídia final. Estas funções estão implementadas nos templates FRH sob a forma de um sistema hipermídia "incompleto". Esse sistema é então suprido com as informações fornecidas pelo processamento dos nós realizado pelo TFRH.

A Figura 2 mostra a arquitetura e o processo de criação de um relatório Hipermídia, apontando a qual nível da arquitetura de Sistemas Hipermídia pertence cada componente.

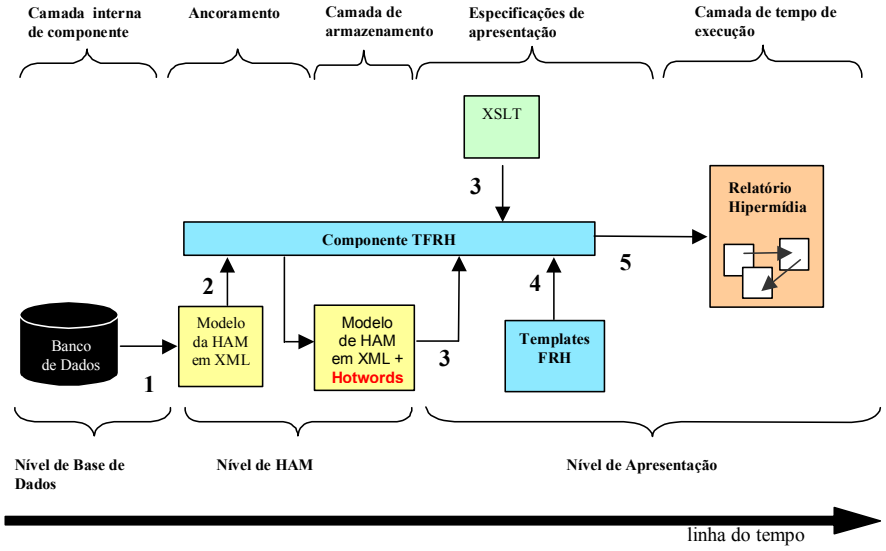


Figure 2. Processo de criação de um relatório Hipermídia

3.1. Visão Geral

Os componentes que fazem parte do FRH são os seguintes:

- **Templates FRH:** conjunto de arquivos contendo a "cópia original" da interface e dos scripts que irão constituir o sistema hipermídia final. Os templates FRH podem ser vistos como um sistema hipermídia sem os nós de conteúdo.
- **Componente TFRH:** este é um componente implementado na linguagem object pascal do Delphi. O TFRH é responsável pela geração de todo o conjunto de nós e da customização do sistema hipermídia embutido nos templates FRH.

De modo geral o funcionamento do FRH ocorre a partir da interação de uma aplicação com o componente TFRH, da seguinte maneira:

Uma aplicação utilizando o TFRH fornece uma string representando um documento XML estruturado de acordo com o modelo da HAM.

O TFRH lê o documento XML, adicionando automaticamente links para palavras chaves de unidades de conceito e glossário.

Estilos de apresentação previamente definidos pelo desenvolvedor são aplicados aos nós especificados no XML, gerando com isso todos os nós de conteúdo do sistema hipermídia em seu formato de apresentação final.

O TFRH faz uma cópia dos templates FRH, customiza o sistema hipermídia embutido e coloca junto dos nós gerados.

Na figura 2, temos as seguintes transições:

- **Mapeamento das informações armazenadas na base de dados para o modelo de HAM em XML:** a realização deste mapeamento não necessita de nenhuma das API's

definidas para XML, por motivos já descritos na seção 2.4.5 (Criando Documentos XML do Zero). Uma vez conforme o modelo, a informação estará organizada em uma estrutura que representa os elementos hipermídia, seus nós e informações suficientes para criação de links implícitos.

- **Criação de hotwords e mapa global:** É neste momento que são criados e ancorados todos os links implícitos contextuais (hotwords) para as unidades de conceito e para o glossário. O FRH usa os elementos XML que descrevem o título e as palavras chaves para cada nó (ver Modelo da HAM em XML) para criar hotwords. Os links são criados inserindo-se a tag HTML <a> nos nós tipo texto do XML. Esta tag possui código javascript que dispara uma função implementada nos templates FRH.
- **Formatação para apresentação dos nós:** formatação de acordo com as especificações de apresentação representada através da aplicação estilos XSLT a cada nó definido na HAM. Os estilos XSLT's definidos pelo desenvolvedor devem gerar XHTML como saída. A cada nó formatado é gerado um documento XHTML. O documento XHTML passa ainda por um pequeno processamento para indexação de palavras para a busca e para inserção de scripts. O formato final de um nó é o HTML.
- **Processamento dos Templates FRH:** esta operação pode ser encarada como uma extensão das especificações de apresentação da arquitetura Dexter. Os templates FRH, bem como seu processamento, são tratados em maior detalhe na sessão Templates FRH deste capítulo.
- **Criação do Sistema Hipermídia:** O Sistema Hipermídia resultante é a união dos nós gerados no processamento XSLT com a interface e o sistema de controle do sistema hipermídia obtidos através do processamento dos templates FRH.

3.3. Templates FRH

Os templates FRH são um conjunto de arquivos em vários formatos que representam o sistema hipermídia sem os nós de conteúdo. Nos templates estão definidos os elementos de interface, funções de gerência do sistema e funções de elementos hipermídia. Os templates FRH são utilizados toda vez que o relatório hipermídia for solicitado. Estes arquivos são copiados, processados e inseridos no diretório definido para o relatório hipermídia.

Sistema Hipermídia dos Templates

Excluindo-se os html's de conteúdo, todo o sistema hipermídia gerado é proveniente dos templates FRH. O sistema hipermídia final se torna adaptável pela omissão de variáveis no sistema hipermídia dos templates FRH. Essas variáveis tem seus valores definidos no ato do processamento dos templates pelo TFRH, sendo atribuídos os valores especificados pelo desenvolvedor. De fato, o sistema hipermídia dos templates FRH não funciona corretamente até que os templates sejam processados pelo componente TFRH.

O sistema hipermídia utiliza os frames e janelas dos documentos html como objetos. Esta funcionalidade é conseguida através da adição de funções javascript nos documentos html que representam um objeto do sistema hipermídia. As funções dos elementos hipermídia estão distribuídas através dos objetos do sistema hipermídia

Existe um sistema de gerência central representado pelo objeto gerente, definido através do documento html "frame_gerente.html". Ele é o responsável por receber as mensagens relativas a um objeto e realizar as operações necessárias. Ao realizar tais operações, o gerente, por ser persistente durante toda a sessão, mantêm as variáveis necessárias para o controle do sistema.

3.4. Componente TFRH

O componente TFRH é composto pela classe TFRH, pelo componente MSXML3 e outras classes de suporte, como a classe TInterfaceArquivos, que dá suporte para a manipulação de diretórios e arquivos.

A manipulação de XML, é feita em sua maioria através do MSXML3, que consiste num parser XML DOM [W3C] e um processador XSLT, disponibilizados através da dll "msxml3.dll".

A seguir é mostrada a interface da classe principal do componente TFRH, a TFRH. Apenas são listados os métodos de alto nível:

```
TFRH = class
 private
    procedure InsCodNodo_CriaFluxo_CriaListaPalChave_CopiaArqs();
    procedure LoopXML_CriaLinks(nodo: IXMLDOMNode);
    procedure LoopXML_aplicaXSL(nodo: IXMLDOMNode);

 protected
    destructor Destroy; override;
    procedure AplicaEstilos();

 public
constructor Create(PXML_str, PXSL_path, PRelatorio_path,
    PTemplatesFRH_path: string; PFuncoesUnidadeConceito:
    TFuncoesUnidadeConceito; PFuncoesGlossario: TFuncoesGlossario);
end;
```

Os três métodos privados listados constituem nos três loops pela árvore XML necessários para a realização das tarefas. São eles:

- `InsCodNodo_CriaFluxo_CriaListaPalChave_CopiaArqs:` passa pelos elementos XML reconhecendo os nós hipermídia marcados de acordo com o modelo da HAM. No reconhecimento insere um código para cada nó, constrói o arquivo contendo a hierarquia do fluxograma e cria uma lista com o mapeamento de cada palavra chave para o código do respectivo nó hipermídia.
- `LoopXML_CriaLinks:` passa por todos os nós XML do tipo texto e cria os links contextuais implícitos (hotwords) a partir do mapeamento das palavras chaves realizados no método anterior.
- `LoopXML_aplicaXSL:` passa pelos nós XML que são nós hipermídia, aplicando a esses nós os estilos XSLT especificados pelo desenvolvedor.
- Os métodos `Create` e `AplicaEstilos` consistem na interface que o desenvolvedor deverá utilizar para implementar a geração de relatórios em sua aplicação. Para criar uma instância da classe TFRH o desenvolvedor precisa especificar uma string contendo o documento XML; os diretórios dos estilos XSLT, do relatório hipermídia e dos templates FRH; as funções do elemento hipermídia unidade de conceito e as funções do elemento hipermídia glossário.

3.5. Modelo da HAM

O modelo da HAM em XML define regras para implementação do método de organização da informação no fomato XML. As linhas gerais do modelo são:

Nós (unidades de conceitos), unidades básicas de informação e mídias são representados por elementos XML. O que difere um nó dos outros tipos de informação é a presença dos atributos específicos para nós.

- Um único documento XML para todos os elementos hipermídia a serem contemplados.
- Um nó pode utilizar todo o documento XML para recuperar informações e apresentá-las ao usuário.
- Um elemento XML pode somente representar um nó e este nó é pode ser representado somente por este elemento XML.
- A estrutura interna dos elementos hipermídia deve ser definida pelo desenvolvedor.

Links

O modelo da HAM não define elementos para representação direta de links. Todos os links gerados pelo TFRH são implícitos e são possíveis de serem gerados automaticamente graças a estrutura e atributos do modelo da HAM. As hotwords (links contextuais) são geradas a partir das palavras chaves especificadas nos atributos dos nós hipermídia.

O escopo dos links, uma propriedade que define quais nós devem ser verificados para a construção de links no texto de cada nó XML é controlado através do atributo FRH_LINKS. Este atributo deve ser suprido com uma expressão XPath [W3C, 1999 a] que deverá definir um conjunto de nós XML contendo as palavras chave que serão confrontadas com o texto para criação dos links. A atribuição de um valor nulo a FRH_LINKS desabilita todos os links no elemento XML onde foi definido.

Especificação dos elementos e atributos

A tabela 1 apresenta os elementos e atributos utilizados pelo modelo.

Tabela 1. Elementos e Atributos do Modelo

ELEMENTO	DESCRIÇÃO
<FRH_ROOT>	Elemento raiz do documento xml. (outros nomes podem ser usados)
<FRH_GLOSSARIO>	Elemento XML que representa o elemento hipermídia Glossário. (fixo)
<FRH_UNIDADE_CONCEITO>	Elemento XML que representa o elemento hipermídia Unidades de Conceito. (fixo)
ATRIBUTO	**DESCRIÇÃO**
FRH_CODNODO	Código gerado pelo TFRH para cada nó hipermídia.
FRH_TITULO	Atributo XML que contém o título de um nó. É obrigatório para todos os nós hipermídia.
FRH_PALAVRASCH	Atributo XML que contém as palavras ou expressões (conceitos) chaves relevantes para um nó. As palavras ou expressões devem estar separadas pelo caracter "\" (barra invertida). (opcional)
FRH_GRUPO	Atributo que especifica se o elemento XML deve ser definido como um conteiner de nós. Esta informação é utilizada pelo fluxograma para agrupar um conjunto de nós. (opcional)
FRH_ESTILO_INDICE_GLOSSARIO	Atributo específico para o elemento XML FRH_GLOSSARIO. Este atributo deve especificar o estilo XSLT a ser utilizado para construção do índice do glossário.
FRH_LINKS	Define o escopo dos links para um nó XML tipo texto.

O desenvolvedor deve tomar cuidado para não utilizar os nomes dos elementos e atributos acima com outra finalidade senão a descrita.

4. Estudo de Caso: PCMAT Hipermídia

O PCMAT é um documento que serve para a especificação de programa contra acidentes de trabalho e caracterização de uma obra. A validação do FRH se deu através do Sistema PCMAT, um aplicativo que dá suporte a criação de relatórios PCMAT. O objetivo era criar uma versão hipermídia do PCMAT, gerando assim o PCMAT Hipermídia.

A criação do PCMAT Hipermídia se deu seguindo as etapas da metodologia para desenvolvimento de sistemas hipermídia, utilizando o FRH para agilizar a implementação do sistema.

Um trecho da estrutura do documento XML resultante do mapeamento do banco de dados Interbase para o modelo da HAM em XML é apresentado a seguir:

```
      <?xml version="1.0" encoding="ISO-8859-1"?>
      <FRH_ROOT>
      <FRH_UNIDADE_CONCEITO FRH_GRUPO="sim" FRH_TITULO="Relatório de
Medidas Preventivas do Processo Construtivo">
      <NListaEtapas FRH_ESTILO="lista_etapas" FRH_TITULO="Etapas do
Processo Construtivo" FRH_PALAVRASCH="Etapas do Processo Construtivo,
Processo Construtivo" FRH_ELEMENTO_CONTEUDO="parent::*" FRH_CODNODO="1">
           <etapa FRH_ESTILO="etapa" FRH_TITULO="CARPINTARIA"
FRH_PALAVRASCH="CARPINTARIA" FRH_CODNODO="2">
                 <numero>1</numero>
                 <nome>CARPINTARIA</nome>
                 <atividade FRH_ESTILO="atividade" FRH_TITULO=" TRABALHOS
COM SERRA CIRCULAR" FRH_PALAVRASCH=" TRABALHOS COM SERRA CIRCULAR"
FRH_CODNODO="3">
                       <numero>1</numero>
                       <nome> TRABALHOS COM SERRA CIRCULAR</nome>
                 </atividade>
           </etapa>
      </NListaEtapas>
      </FRH_UNIDADE_CONCEITO>
      </FRH_ROOT>
```

Um dos estilos XSLT's utilizados para formatar a estrutura acima é apresentado a seguir. No caso o estilo apresentado é referente ao tipo de nodo <etapa>.

```
      <?xml version="1.0" encoding="ISO-8859-1"?>
      <xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
      <xsl:output method="xml" encoding="ISO-8859-1"/>
      <xsl:template match="etapa">
      <HTML><HEAD><TITLE><xsl:value-of select="@FRH_TITULO"/></TITLE>
      <META http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1"/>
         <link rel="stylesheet" href="../css/padrao.css"/>
         <link ID="estilo_links" rel="stylesheet"
href="../css/estilo_links.css"/>
      </HEAD><BODY class="FontePadrao">
      <h2 style="font-size=13pt"> <xsl:value-of select="@FRH_TITULO"/>
</h2>

      <table align="center" width="94%" cellpadding="0" cellspacing="0"
border="0"><tr><td>
      Lista das atividades:<br/><br/>
        <table style=" font-size: 10pt;" width="100%" class="Tabela"
cellpadding="4" cellspacing="1" border="0">
          <xsl:apply-templates select="child::atividade"/>
        </table><br/><br/></td></tr></table>
      </BODY></HTML>
      </xsl:template>
      <xsl:template match="atividade">
```

```
          <tr><td width="3%" class="CelulaCabecalho"><xsl:value-of
select="numero"/></td>
            <td width="95%"><xsl:value-of select="nome"/></td>
          </tr>
      </xsl:template>
      </xsl:stylesheet>
```

O resultado da aplicação do FRH no Sistema PCMAT é um sistema hipermídia portável, podendo ser publicado na web sem nenhuma mudança, com mídias podendo variar de figuras até mídias interativas em 3D. A Figura 4 apresenta a tela principal do PCMAT Hipermídia.

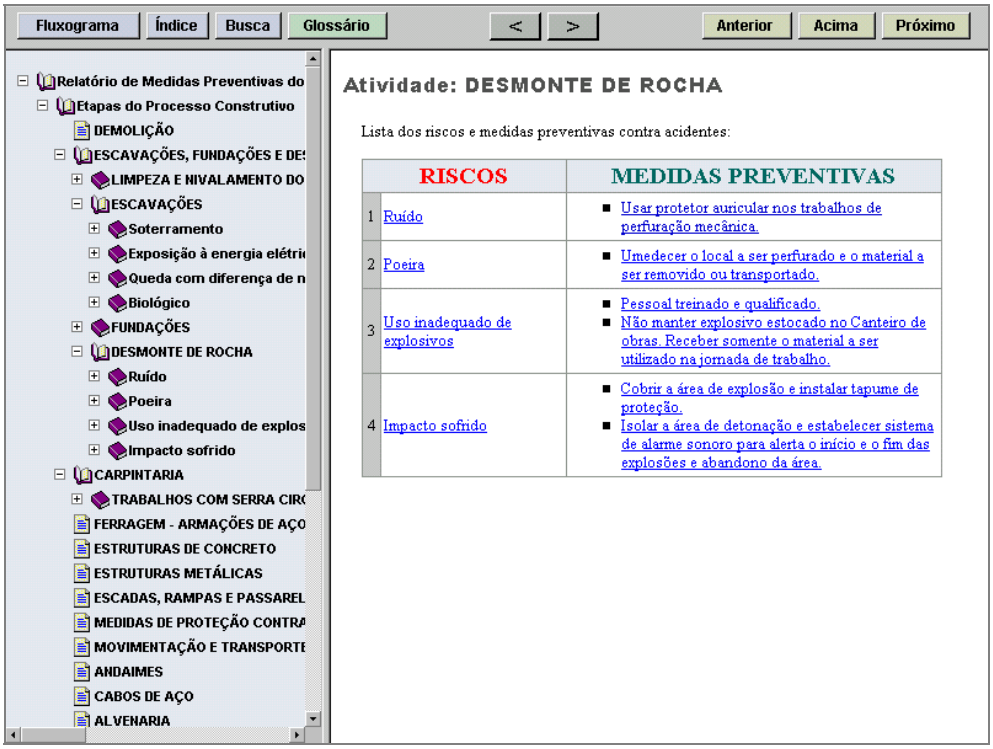


Figure 3. Tela Principal do PCMAT Hipermídia. Na tela aparece o Fluxograma (frame esquerdo) e um nó de Unidade de Conceito (frame direito)

5. Conclusões

O FRH mostrou-se útil na agilização do processo de criação e manutenção de sistemas hipermídia de conteúdo estruturável em uma hierarquia de conceitos. Assim, sistemas hipermídia podem não apenas serem criados mais rapidamente, como também sua manutenção e modificação tornam-se muito mais simples.

As informações disponíveis em bancos de dados podem facilmente ser disponibilizadas através de um sistema de recuperação de informação (relatório hipermídia), aproveitando melhor o potencial de utilização desses dados.

A arquitetura utilizada no FRH e o modelo da HAM em XML podem ser utilizados para outras classes de sistemas hipermídia. A estruturação em XML permite também o fácil processamento do conteúdo por outras aplicações.

Referências

Halasz, F., e Schwartz, M. (1994). The Dexter hypertext reference model. Proc. NIST Hypertext Standardization Workshop (Gaithersburg, MD, 16-18 January), 95-133.

Martin, James (1992). Hiper Documentos e Como Criá-los. Campus, 1992.

Nielsen, Jakob (1995). Multimedia & Hipertext: the Internet and Beyond. Academic Press Inc.

Reed, W. Michael; Burton, John K.; Liu, Min (1994). Multimedia and Magachange. New York: The Haworth Press, Inc., 1994.

Word Wide Web Consortium (1999). XSL Transformations (XSLT) Version 1.0 Disponível por WWW em: http://www.w3.org/TR/1999/REC-xslt-19991116 (20 nov. 2002).

Word Wide Web Consortium (1999 a). XML Path Language (XPath) Version 1.0 Disponível por WWW em: http://www.w3.org/TR/1999/REC-xpath-19991116 (20 nov. 2002).

Word Wide Web Consortium (2000). Extensible Markup Language (XML) 1.0 (Second Edition) - W3C Recommendation. Disponível por WWW em: http://www.w3.org/TR/REC-xml (7 set. 2002).

Word Wide Web Consortium (2000 b). What is the Document Object Model?. Disponível por WWW em: http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/introduction.html (08 set. 2002)

Word Wide Web Consortium (2000 c). Document Object Model Core. Disponível por WWW em: http://www.w3.org/TR/2000/REC-DOM-Level-2-Core-20001113/core.html (08 set. 2002)

Anexos

I. Units (módulos) do Componente TFRH

unit uFRH;

```
interface

uses
  uFRH_XML, SysUtils, Classes, xmldom, XMLIntf, msxmldom, XMLDoc, Contnrs,
StrUtils, MSXML2_TLB, uFRHArquivos;

const

  CtNodoTipoElemento = 1;
  CtNodoTipoTexto = 3;
  CtNodoTipoCData = 4;

  { - O nome das janelas ou frames seguintes nao devem ser modificados.
Eles são:
          NomeFrameGerente = 'gerente';
          NomeFrameUnidadeConceito = 'unidade_conceito';
          NomeFrameFerramentas = 'ferramentas';
          NomeFrameFluxograma = 'fluxograma';
          NomeFrameBusca = 'busca';
          NomeFrameIndice = 'indice';
          NomeFrameConteinerGlossarioNodos = 'glossario_nodos';
          NomeFrameGlossarioNodos = 'glossario_nodos';
          NomeFrameGlossarioFerramentas = 'glossario_ferramentas';
*pode ser modificado sem problemas no HTML
  }

  //nomes dos TemplatesFRH. Se desejar o desenvolvedor pode modificar os
templates, desde que atualize os nomes.
  //O relatório será gerado com os nomes das propriedades do obj TFRH.
  ConstNomeIndex = 'index.html';
  ConstNomeFrameGerente = 'frame_gerente.html';
  ConstNomeFerramentas = 'frame_ferramentas.html';
  ConstNomeFluxograma = 'fluxograma.html';
```

```
  ConstNomeBusca = 'busca.html';
  ConstNomeIndice = 'indice.html';
  ConstNomeGlossarioFerramentas = 'glossario_ferramentas.html';
  ConstNomeGlossarioIndice = 'glossario_indice.html';
  ConstNomeXML = 'HAM.xml';

  //diretórios padroes relativos a raiz do relatorio hipermidia.
Desenvolvedor pode modificar.
  ConstDirScripts = 'scripts/';     //diretorio que deve conter os scripts.
No HTML para incluir todas as constantes basta adicionar o js constantes.
  ConstDirCSS = 'css/';
  ConstDirImg = 'img/';
  ConstDirUnidadeConceito = 'conteudo/';      //diretório onde devem ser
colocados os nodos de unidades de conceito
  ConstDirNodosGlossario = 'conteudo/';       //diretório onde devem ser
colocados os nodos de unidades de glossario
  ConstDirIndex = '';
  ConstDirFrameGerente = '';
  ConstDirFerramentas = '';
  ConstDirFluxograma = '';
  ConstDirBusca = 'busca/';
  ConstDirIndice = '';
  ConstDirGlossarioFerramentas = '';
  ConstDirGlossarioIndice = '';
  ConstDirXML = 'xml/';


  //codigo JS contendo a relacao padrão entre os documentos HTML com suas
janelas/frames e o GERENTE.
  //lembrar que o nome das janelas não é modificável. Se uma documento for
aberta em uma nova janela
  //entao usar "null" (minusculo).

  //isto pode ser construído automaticamente no futuro
  ConstGerenteParaUnidadeConceito = 'self.top.unidade_conceito';
  ConstGerenteParaFluxograma = 'self.top.frame_esquerda';
  ConstGerenteParaGlossarioNodos = 'null';
  ConstGerenteParaBusca = 'null';
//*ATENÇÃO: PARÂMETRO FIXO*
  ConstGerenteParaIndice = 'self.top.frame_esquerda';
```

```
  ConstGerenteParaFerramentas = 'self.top.ferramentas';
//*ATENÇÃO: PARÂMETRO FIXO*

  ConstUnidadeConceitoParaGerente = 'self.top.gerente';
  ConstFluxogramaParaGerente = 'self.top.gerente';
  ConstGlossarioNodosParaGerente = 'self.top.opener.top.gerente';
  ConstBuscaParaGerente = 'self.opener.top.gerente';
//*ATENÇÃO: PARÂMETRO FIXO*
  ConstIndiceParaGerente = 'self.top.gerente';
  ConstFerramentasParaGerente = 'self.top.gerente';
//*ATENÇÃO: PARÂMETRO FIXO*


  {-------------------------------------------------------------------------
-----------------
  *                                ESPECIFICAÇÃO DE CONTEINERS
*

   Especifica se as janelas acima são carregadas dentro de conteiners
(dentro de frames em
   janelas diferentes que o index.html). Se forem deve ser especificado seu
HTML e a relação
   conteiner->conteudoDoConteiner, conteiner->gerente, diretório e opcões
de janela.
   Um mesmo conteiner pode ser especificado para mais de um documento HTML,
bastando
   atribuir os mesmos dados.
   -------------------------------------------------------------------------
-----------------}

  //Nomes dos HTML's
//  ConstNomeConteinerUnidadeConceito = '';                  //NAO
SUPORTADO!!!!!
  ConstNomeConteinerGlossarioNodos = 'glossario.html';       //conteiner
padrão para o glossário
  ConstNomeConteinerFluxograma = '';
  ConstNomeConteinerBusca = '';
  ConstNomeConteinerIndice = '';

  //Para o GERENTE
//  ConstConteinerUnidadeConceitoParaGerente = '';
```

```
  ConstConteinerGlossarioNodosParaGerente = 'self.opener.top.gerente';
//conteiner padrão para o glossário
  ConstConteinerFluxogramaParaGerente = '';
  ConstConteinerBuscaParaGerente = '';
  ConstConteinerIndiceParaGerente = '';

  //Para o CONTEUDO
//  ConstConteinerParaUnidadeConceito = '';
  ConstConteinerParaGlossarioNodos = 'glossario_nodos';        //conteiner
padrão para o glossário
  ConstConteinerParaFluxograma = '';
  ConstConteinerParaBusca = '';
  ConstConteinerParaIndice = '';

  //Diretórios dos conteiners
//  ConstDirConteinerUnidadeConceito = '';
  ConstDirConteinerGlossarioNodos = '';        //conteiner padrão para o
glossário
  ConstDirConteinerFluxograma = '';
  ConstDirConteinerBusca = '';
  ConstDirConteinerIndice = '';

  //Parâmetros para abertura de janelas conteiners.
//  ConstOpcoesConteinerUnidadeConceito =
'toolbar=0,location=0,directories=0,status=0,scrollbars=yes,menubar=0,resiz
able=1,width=600,height=400';
  ConstOpcoesConteinerGlossarioNodos =
'toolbar=0,location=0,directories=0,status=0,scrollbars=yes,menubar=0,resiz
able=1,width=485,height=350';
  ConstOpcoesConteinerFluxograma =
'toolbar=0,location=0,directories=0,status=0,scrollbars=yes,menubar=0,resiz
able=1,width=400,height=350';
  ConstOpcoesConteinerBusca =
'toolbar=0,location=0,directories=0,status=0,scrollbars=yes,menubar=0,resiz
able=1,width=400,height=350';
  ConstOpcoesConteinerIndice =
'toolbar=0,location=0,directories=0,status=0,scrollbars=yes,menubar=0,resiz
able=1,width=400,height=350';

  //codigos para Elementos Hipermídia suportados mapeados para suas
respectivas janelas em JS
```

```
  CtGlossario = 'glossario';
  CtUnidadeConceito = 'unidade_conceito';


type
  // Tipos que descrevem as funções possíveis para cada Elemento Hipermídia
  TFuncaoUnidadeConceito = ( ucNavegacaoHierarquica, ucFluxograma,
ucLinksAutomaticos, ucBusca, ucIndice );
  TFuncoesUnidadeConceito = set of TFuncaoUnidadeConceito;

  TFuncaoGlossario = ( gLinksAutomaticos, gIndice, gFluxograma,
gIncluiNoIndicePrincipal);
  TFuncoesGlossario = set of TFuncaoGlossario;

  //classe principal
  TFRH = class
  private
    InterfaceArquivos: TInterfaceArquivos;
    docXML:TXMLDocument;
    XSL_str: TStringList;
    DomDocFull: TDomDocument;
    DomDocXML: TDomDocument;
    DomDocXSL: TDomDocument;


    CodNodo: integer;
    CodPrimeiroNodoGlossario: integer;
    CodUltimoNodoGlossario: integer;
    CodPrimeiroNodoUnConceito: integer;
    CodUltimoNodoUnConceito: integer;
    CodUltimoNodo: integer;

    ArqIndiceBusca: TStringList;
    FluxogramaStrNodes: string;
    NivelNodoHiper: integer;    //utilizado para a criação do fluxograma
    ListaInfoNodos: TStringList;
    ContextoNodo: string;

    procedure InsCodNodos_CriaFluxo_CriaListaPalChave_CopiaArqs();
    procedure LoopXML_criaTabelaNomes(nodo: IXMLNode);
    procedure FluxogramaInsereNodoConteudo(nodo: IXMLNode);
```

```
    procedure FluxogramaInsereNodoGrupo(nodo: IXMLNode);
    procedure FluxogramaInsereFecha();
    procedure CopiaArquivosFRH();
    procedure AplicaXSLIndiceGlossario();

    procedure LoopXML_CriaLinks(nodo: IXMLDOMNode);
    procedure CriaLinks(nodo: IXMLDOMNode);
 {   procedure CopiaElementoListaInfoNodos(Palavra: string; PCodNodo:
integer; var Lista: TStringList);}
    procedure AdicionaInformacoesLinks(Palavra: string; PCodNodo: integer;
var ListaLinks: TStringList);

    procedure LoopXML_aplicaXSL(nodo: IXMLDOMNode);
    procedure FormataNodo(NodoHiper: IXMLDOMNode; PathArquivoXSL: string);
    function  InsereScriptsNodosHiper(textoHTML: string; nodo:
IXMLDOMNode): string;
    function  AchaNodoHiperPai(NodoHiper: IXMLDOMNode): IXMLDOMnode;
    function  AchaNodoHiperFilho(NodoHiper: IXMLDOMNode): IXMLDOMnode;
    function  AchaNodoHiperSeguinte(NodoHiper: IXMLDOMNode): IXMLDOMnode;
    function  AchaNodoHiperAnterior(NodoHiper: IXMLDOMNode): IXMLDOMnode;
    function  PegaCodHTMLNodoHiperPai(NodoHiper: IXMLDOMnode): string;
    function  PegaCodHTMLNodoHiperSeguinte(NodoHiper: IXMLDOMnode): string;
    function  PegaCodHTMLNodoHiperAnterior(NodoHiper: IXMLDOMnode): string;

    procedure ResolveProcessingInstructions();
    procedure CriaScriptsApoio();
    procedure InsereVariaveisFrameGerente();
    procedure ConstroiSelectOptionsIndice(PathIndice: string);

    function  EhNodoHipermidia(nodo: IXMLDOMNode):boolean;
    function  TemAtributo(Nodo: IXMLDOMNode; NomeAtributo: string):boolean;
    function  PegaCaminhoParaRaiz(path: string): string;
    function  InsereTagNoInicioDoHEAD(HtmlStr, Tag: string): string;
    procedure InsereVariavelJS(PathHtml, NomeVariavel, Valor : string);
    function  RetiraEntidadesEspeciaisDosLinks(TextoHTML: string): string;

   protected
    XSL_path: string;
    TemplatesFRH_path: string;
    Relatorio_path: string;
    FuncoesUnidadeConceito: TFuncoesUnidadeConceito;
```

```
FuncoesGlossario: TFuncoesGlossario;

IncluiXML: boolean;

NomeIndex: string;
NomeFrameGerente: string;
NomeFerramentas: string;
NomeFluxograma: string;
NomeBusca: string;
NomeIndice: string;
NomeGlossarioFerramentas: string;
NomeGlossarioIndice: string;
NomeXML: string;

DirUnidadeConceito: string;
DirNodosGlossario: string;
DirScripts: string;
DirCSS: string;
DirImg: string;
DirIndex: string;
DirFrameGerente: string;
DirFerramentas: string;
DirFluxograma: string;
DirBusca: string;
DirIndice: string;
DirGlossarioFerramentas: string;
DirGlossarioIndice: string;
DirXML: string;

GerenteParaUnidadeConceito: string;
GerenteParaGlossarioNodos: string;
GerenteParaFluxograma: string;
GerenteParaBusca: string;
GerenteParaIndice: string;
GerenteParaFerramentas: string;

UnidadeConceitoParaGerente: string;
GlossarioNodosParaGerente: string;
FluxogramaParaGerente: string;
BuscaParaGerente: string;
IndiceParaGerente: string;
```

```
    FerramentasParaGerente: string;

    //CONTEINERS
//    NomeConteinerUnidadeConceito: string;
    NomeConteinerGlossarioNodos: string;
    NomeConteinerFluxograma: string;
    NomeConteinerBusca: string;
    NomeConteinerIndice: string;

//  ConteinerUnidadeConceitoParaGerente: string;
    ConteinerGlossarioNodosParaGerente: string;
    ConteinerFluxogramaParaGerente: string;
    ConteinerBuscaParaGerente: string;
    ConteinerIndiceParaGerente: string;

//    ConteinerParaUnidadeConceito: string;
    ConteinerParaGlossarioNodos: string;
    ConteinerParaFluxograma: string;
    ConteinerParaBusca: string;
    ConteinerParaIndice: string;

//    DirConteinerUnidadeConceito: string;
    DirConteinerGlossarioNodos: string;
    DirConteinerFluxograma: string;
    DirConteinerBusca: string;
    DirConteinerIndice: string;

//    OpcoesConteinerUnidadeConceito: string;
    OpcoesConteinerGlossarioNodos: string;
    OpcoesConteinerFluxograma: string;
    OpcoesConteinerBusca: string;
    OpcoesConteinerIndice: string;
    destructor Destroy; override;
  public
    procedure AplicaEstilos();
    constructor Create(PXML_str, PXSL_path, PRelatorio_path,
PTemplatesFRH_path: string; PFuncoesUnidadeConceito:
TFuncoesUnidadeConceito; PFuncoesGlossario: TFuncoesGlossario);
  end;

{-----------------------------------------}
```

```
  EErroFRH = class(Exception);

{-------------------------------------------}

  TNodoHiper = class
  protected
   CodNodo: integer;
   ContextoNodo: string;
   Titulo: string;
  public
     constructor Create(PCodNodo: integer; PContextoNodo, PTitulo: string);
  end;

  {-------------------------------------------}

  TParametrosLink = class
  protected
   StrArrayTitulo: string;
   StrArrayDestino: string;
   StrArrayContexto: string;
  public
     constructor Create(PStrArrayTitulo, PStrArrayDestino,
PStrArrayContexto: string);
  end;


implementation

{-------------------------------------------------------------}
constructor TNodoHiper.Create(PCodNodo: integer; PContextoNodo, PTitulo:
string);
begin
   self.CodNodo := PCodNodo;
   self.ContextoNodo := PContextoNodo;
   self.Titulo := PTitulo;

end;

{-------------------------------------------------------------}
```

```
constructor TParametrosLink.Create(PStrArrayTitulo, PStrArrayDestino,
PStrArrayContexto: string);
begin
   self.StrArrayTitulo:= PStrArrayTitulo;
   self.StrArrayDestino:= PStrArrayDestino;
   self.StrArrayContexto:= PStrArrayContexto;

end;

{-------------------------------------------------------------}
constructor TFRH.create(PXML_str, PXSL_path, PRelatorio_path,
PTemplatesFRH_path: string; PFuncoesUnidadeConceito:
TFuncoesUnidadeConceito; PFuncoesGlossario: TFuncoesGlossario);
begin
//   if PFuncoesUnidadeConceito = [] then
//      raise EFRHErro.Create('O elemento hipermídia Unidades de Conceito
deve conter pelo menos uma função definida na propriedade
"FuncoesUnidadeConceito".');
//   if (not(gIndice in PFuncoesGlossario))and(gLinksAutomaticos in
PFuncoesGlossario) then
//      raise EFRHErro.Create('Se o elemento hipermídia Glossário tiver
funções definidas no parâmetro "FuncoesGlossario", a função "gIndice" deve
ser incluída.');
   self.FuncoesGlossario := PFuncoesGlossario;
   self.FuncoesUnidadeConceito := PFuncoesUnidadeConceito;

   InterfaceArquivos := TInterfaceArquivos.Create();

   self.IncluiXML := True;
   self.DirXML := ConstDirXML;
   self.XSL_path := PXSL_path;
   self.Relatorio_path := PRelatorio_path;
   self.TemplatesFRH_path := PTemplatesFRH_path;
   self.XSL_str:= TStringList.Create;
   self.DomDocFull := ComponentesXML.DOMDocFull;
   //seta a linguagem utilizada pelo selectnodes para XPath
   self.DomDocFull.setProperty('SelectionLanguage', 'XPath');

   self.docXML := ComponentesXML.docXML;
   self.docXML.XML.Text := PXML_str;
   self.docXML.Active := true;
```

```
   self.CodNodo := 0;            //zera a sequencia de codigos para os
nodos
   self.ContextoNodo := '';      //zera o contexto
   self.ListaInfoNodos := TStringlist.Create;
   self.ListaInfoNodos.Sorted := true;
   self.ListaInfoNodos.Duplicates := dupAccept;
   self.NivelNodoHiper := 0;
   self.DomDocXML := ComponentesXML.DomDocXML;
   self.DomDocXSL := ComponentesXML.DomDocXSL;
   self.ArqIndiceBusca:= TStringList.Create();
   //carrega definicoes padroes de diretorios e outros.
   self.NomeIndex := ConstNomeIndex;
   self.NomeFrameGerente := ConstNomeFrameGerente;
   self.NomeFerramentas := ConstNomeFerramentas;
   self.NomeFluxograma := ConstNomeFluxograma;
   self.NomeBusca := ConstNomeBusca;
   self.NomeIndice := ConstNomeIndice;
   self.NomeGlossarioFerramentas := ConstNomeGlossarioFerramentas;
   self.NomeGlossarioIndice := ConstNomeGlossarioIndice;
   self.NomeXML := ConstNomeXML;

   self.DirUnidadeConceito := ConstDirUnidadeConceito;
   self.DirNodosGlossario := ConstDirNodosGlossario;
   self.DirScripts := ConstDirScripts;
   self.DirCSS := ConstDirCSS;
   self.DirImg := ConstDirImg;
   self.DirIndex := ConstDirIndex;
   self.DirFrameGerente := ConstDirFrameGerente;
   self.DirFerramentas := ConstDirFerramentas;
   self.DirFluxograma := ConstDirFluxograma;
   self.DirBusca := ConstDirBusca;
   self.DirIndice := ConstDirIndice;
   self.DirXML := ConstDirXML;

   self.GerenteParaUnidadeConceito := ConstGerenteParaUnidadeConceito;
   self.GerenteParaGlossarioNodos := ConstGerenteParaGlossarioNodos;
   self.GerenteParaFluxograma := ConstGerenteParaFluxograma;
   self.GerenteParaBusca := ConstGerenteParaBusca;
   self.GerenteParaIndice := ConstGerenteParaIndice;
   self.GerenteParaFerramentas := ConstGerenteParaFerramentas;
```

```
   self.UnidadeConceitoParaGerente := ConstUnidadeConceitoParaGerente;
   self.FluxogramaParaGerente := ConstFluxogramaParaGerente;
   self.GlossarioNodosParaGerente := ConstGlossarioNodosParaGerente;
   self.BuscaParaGerente := ConstBuscaParaGerente;
   self.IndiceParaGerente := ConstIndiceParaGerente;
   self.FerramentasParaGerente := ConstFerramentasParaGerente;

//    self.NomeConteinerUnidadeConceito :=
ConstNomeConteinerUnidadeConceito;
    self.NomeConteinerGlossarioNodos := ConstNomeConteinerGlossarioNodos;
    self.NomeConteinerFluxograma := ConstNomeConteinerFluxograma;
    self.NomeConteinerBusca := ConstNomeConteinerBusca;
    self.NomeConteinerIndice := ConstNomeConteinerIndice;

//  self.ConteinerUnidadeConceitoParaGerente :=
ConstConteinerUnidadeConceitoParaGerente;
    self.ConteinerGlossarioNodosParaGerente :=
ConstConteinerGlossarioNodosParaGerente;
    self.ConteinerFluxogramaParaGerente :=
ConstConteinerFluxogramaParaGerente;
    self.ConteinerBuscaParaGerente := ConstConteinerBuscaParaGerente;
    self.ConteinerIndiceParaGerente := ConstConteinerIndiceParaGerente;


//    self.ConteinerParaUnidadeConceito :=
ConstConteinerParaUnidadeConceito;
    self.ConteinerParaGlossarioNodos := ConstConteinerParaGlossarioNodos;
    self.ConteinerParaFluxograma := ConstConteinerParaFluxograma;
    self.ConteinerParaBusca := ConstConteinerParaBusca;
    self.ConteinerParaIndice := ConstConteinerParaIndice;

//    self.DirConteinerUnidadeConceito := ConstDirConteinerUnidadeConceito;
    self.DirConteinerGlossarioNodos := ConstDirConteinerGlossarioNodos;
    self.DirConteinerFluxograma := ConstDirConteinerFluxograma;
    self.DirConteinerBusca := ConstDirConteinerBusca;
    self.DirConteinerIndice := ConstDirConteinerIndice;

//    self.OpcoesConteinerUnidadeConceito :=
ConstOpcoesConteinerUnidadeConceito;
    self.OpcoesConteinerGlossarioNodos :=
ConstOpcoesConteinerGlossarioNodos;
```

```
    self.OpcoesConteinerFluxograma := ConstOpcoesConteinerFluxograma;
    self.OpcoesConteinerBusca := ConstOpcoesConteinerBusca;
    self.OpcoesConteinerIndice := ConstOpcoesConteinerIndice;


end;

{-------------------------------------------------------------}
destructor TFRH.Destroy;
var i: integer;
begin
    self.XSL_str.Free;
    for i:= 0 to self.ListaInfoNodos.Count-1 do
        self.ListaInfoNodos.Objects[i].Free;
    self.ListaInfoNodos.Free;
    self.InterfaceArquivos.Free;
    self.ArqIndiceBusca.Free();
end;

{-------------------------------------------------------------}
procedure TFRH.AplicaEstilos();
var s: string;
begin
    //Copia os arquivos Templates FRH e executa o Primeiro Loop pelos nodos
xml: Insere código nos Nodos Hipermídia, Cria o script para o Fluxograma e
cria a lista de palavras chaves dos nodos.
    self.InsCodNodos_CriaFluxo_CriaListaPalChave_CopiaArqs();

    if gIndice in FuncoesGlossario then
       self.AplicaXSLIndiceGlossario();   //aplica o estilo para o indice
do glossario

    //começa o loop de aplicação dos XSLT para os nodos. Adiciona links.
Também vai salvando e  os HTML's e outros arquivos resultantes.
    self.ContextoNodo := '';      //zera o contexto

    //transfere todo o documento do objeto TXMLDocument (MSXML) para o
TDOMXMLDocument (MSXML3)
    self.DomDocFull.DefaultInterface.loadXML(self.docXML.XML.Text);

    //Loop de criação de links
```

```
self.LoopXML_CriaLinks(self.DomDocFull.DefaultInterface.documentElement);

    //Loop de aplicação dos estilos

self.LoopXML_aplicaXSL(self.DomDocFull.DefaultInterface.documentElement);

    s:= self.DomDocFull.DefaultInterface.xml;

    //cria os arquivos restantes
    self.ResolveProcessingInstructions();
    self.CriaScriptsApoio();
    //salva o arquivo do INDICE da BUSCA
    self.ArqIndiceBusca.SaveToFile(self.Relatorio_path + self.DirBusca +
'\indice_busca.txt');


     //XML com os codigos mas sem os links
     if IncluiXML then begin
        //CRIA DIRETÓRIOS
        InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirXML);
        //doc xml
        self.docXML.SaveToFile(self.Relatorio_path + self.DirXML +
self.NomeXML);
     end;


end;

{-------------------------------------------------------------}
procedure TFRH.InsCodNodos_CriaFluxo_CriaListaPalChave_CopiaArqs();
var FluxogramaStrNodesArq: TStringList;
    FluxogramaStrNodesArqInicio: string;
    ExecutaCodFluxograma: boolean;
begin

  FluxogramaStrNodesArq := TStringList.Create();
  try
     //habilida ou nao a execução de códigos para o fluxograma neste
escopo.
```

```
     ExecutaCodFluxograma := (gFluxograma in
self.FuncoesGlossario)or(ucFluxograma in FuncoesUnidadeConceito);

     //inicializa o FluxogramaStrNodes, que continua sendo criado dentro da
funcao criaTabelaNomes.
     if ExecutaCodFluxograma then begin
        FluxogramaStrNodesArqInicio := '//   FRH: Array de nodos gerado
automaticamente para criaçao do fluxograma pelo componente "nostree.js"' +
#13#10;
        FluxogramaStrNodesArqInicio := self.FluxogramaStrNodes + 'var
TREE_NODES = [' + #13#10;
     end;

     //Cria uma tabela de nomes contendo o mapeamento de expressoes para
nodos.
     //Nodos são codificados por números inteiros crescentes.
     self.LoopXML_criaTabelaNomes(self.docXML.DocumentElement);
     self.CodUltimoNodo := self.CodNodo;
     if self.CodUltimoNodoGlossario = 0 then
        self.CodUltimoNodoGlossario := self.CodUltimoNodo
     else
        self.CodUltimoNodoUnConceito := self.CodUltimoNodo;

     self.CopiaArquivosFRH();    //temque ser depois do
LooPXML_criaTabelaNomes

     if ExecutaCodFluxograma then begin
        Delete(self.FluxogramaStrNodes, 6, 1);        //retira a primeira
","
        self.FluxogramaStrNodes := FluxogramaStrNodesArqInicio +
self.FluxogramaStrNodes + '];';      //completa a informação do array JS
        FluxogramaStrNodesArq.Text := self.FluxogramaStrNodes;
        FluxogramaStrNodesArq.SaveToFile(self.Relatorio_path +
self.DirScripts + 'fluxograma_tree_nodes.js');   //salva o arquivo js
     end;
  finally
     //libera memoria
     FluxogramaStrNodesArq.Free();
  end;

end;
```

```
{-------------------------------------------------------------}
procedure TFRH.CopiaArquivosFRH();
begin
//copia os arquivos necessários. O diretório TemplatesFRH_path não pode ter
a estrutura existente alterada, porém novos arquivos e diretórios podem ser
adicionados.

    //cria os diretorios se necessario. Os primeiros são obrigatórios
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path);
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path + self.DirImg);
//este diretório é fixo
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirScripts);
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirUnidadeConceito);
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirNodosGlossario);
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path + self.DirIndex);
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirFrameGerente);
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirFerramentas);
    InterfaceArquivos.CriaDiretorios(self.Relatorio_path + self.DirCSS);

    //IMAGENS E OUTRAS MIDIAS DA PASTA "img"
    InterfaceArquivos.CopiaDiretorio(self.TemplatesFRH_path + self.DirImg,
self.Relatorio_path + self.DirImg, True);
    //SCRIPS
    InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirScripts +
'script_nodos.js', self.Relatorio_path + self.DirScripts +
'script_nodos.js', True);
    //INDEX
    InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.NomeIndex,
self.Relatorio_path + self.NomeIndex, True);
    InterfaceArquivos.CopiaArq(self.TemplatesFRH_path +
self.DirFrameGerente + self.NomeFrameGerente, self.Relatorio_path +
self.DirFrameGerente + self.NomeFrameGerente, True);
    InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirFerramentas
+ self.NomeFerramentas, self.Relatorio_path + self.DirFerramentas +
self.NomeFerramentas, True);
```

```
    //CSS's
    InterfaceArquivos.CopiaDiretorio(self.TemplatesFRH_path + self.DirCSS,
self.Relatorio_path + self.DirCSS, True);

    //GLOSSARIO
    if (gIndice in self.FuncoesGlossario) then begin
       //CRIA DIRETÓRIOS
       InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirGlossarioFerramentas);
       InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirGlossarioIndice);
       //Conteiner
       if self.NomeConteinerGlossarioNodos <> '' then begin
          InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirConteinerGlossarioNodos);
          InterfaceArquivos.CopiaArq(self.TemplatesFRH_path +
self.DirConteinerGlossarioNodos + self.NomeConteinerGlossarioNodos,
self.Relatorio_path + self.DirConteinerGlossarioNodos +
self.NomeConteinerGlossarioNodos, True);
       end;
       //html's interface
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path +
self.DirGlossarioIndice + self.NomeGlossarioIndice, self.Relatorio_path +
self.DirGlossarioIndice + self.NomeGlossarioIndice, True);
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path +
self.DirGlossarioFerramentas + self.NomeGlossarioFerramentas,
self.Relatorio_path + self.DirGlossarioFerramentas +
self.NomeGlossarioFerramentas, True);
    end;

    //FLUXOGRAMA
    if (ucFluxograma in self.FuncoesUnidadeConceito)or( (gFluxograma in
self.FuncoesGlossario)and(gIndice in self.FuncoesGlossario) ) then begin
       //CRIA DIRETÓRIOS
       InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirFluxograma);
       //Conteiner
       if self.NomeConteinerFluxograma <> '' then begin
          InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirConteinerFluxograma);
```

```
          InterfaceArquivos.CopiaArq(self.TemplatesFRH_path +
self.DirConteinerFluxograma + self.NomeConteinerFluxograma,
self.Relatorio_path + self.DirConteinerFluxograma +
self.NomeConteinerFluxograma, True);
       end;
       //html's interface
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path +
self.DirFluxograma + self.NomeFluxograma, self.Relatorio_path +
self.DirFluxograma + self.NomeFluxograma, True);
       //scripts de apoio do fluxograma
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirScripts
+ 'fluxograma_nostree.js', self.Relatorio_path + self.DirScripts +
'fluxograma_nostree.js', True);
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirScripts
+ 'fluxograma_tree_format.js', self.Relatorio_path + self.DirScripts +
'fluxograma_tree_format.js', True);
    end;

    //BUSCA. A busca, por enquanto, só é feita nas unidades de conceito.
todos os arquivos do diretório "busca" deverão ficar num mesmo diretório
    if ucBusca in self.FuncoesUnidadeConceito then begin
       //CRIA DIRETÓRIOS
       InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirBusca);
       //Conteiner
       if self.NomeConteinerBusca <> '' then begin
          InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirConteinerBusca);
          InterfaceArquivos.CopiaArq(self.TemplatesFRH_path +
self.DirConteinerBusca + self.NomeConteinerBusca, self.Relatorio_path +
self.DirConteinerBusca + self.NomeConteinerBusca, True);
       end;
       //html's interface
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirBusca +
self.NomeBusca, self.Relatorio_path + self.DirBusca + self.NomeBusca,
True);
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirBusca +
'vazio.html', self.Relatorio_path + self.DirBusca + 'vazio.html', True);
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirBusca +
'semBusca.html', self.Relatorio_path + self.DirBusca + 'semBusca.html',
True);
```

```
       //biblioteca .jar
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirBusca +
'search.jar', self.Relatorio_path + self.DirBusca + 'search.jar', True);
    end;

    //INDICE
    if ucIndice in self.FuncoesUnidadeConceito then begin
       //CRIA DIRETÓRIOS
       InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirIndice);
       //Conteiner
       if self.NomeConteinerIndice <> '' then begin
          InterfaceArquivos.CriaDiretorios(self.Relatorio_path +
self.DirConteinerIndice);
          InterfaceArquivos.CopiaArq(self.TemplatesFRH_path +
self.DirConteinerIndice + self.NomeConteinerIndice, self.Relatorio_path +
self.DirConteinerIndice + self.NomeConteinerIndice, True);
       end;
       //html's interface
       InterfaceArquivos.CopiaArq(self.TemplatesFRH_path + self.DirIndice +
self.NomeIndice, self.Relatorio_path + self.DirIndice + self.NomeIndice,
True);
    end;
end;

{-------------------------------------------------------------}
Procedure TFRH.loopXML_criaTabelaNomes(nodo: IXMLNode);
var i,j, nrInseridos: integer;
    nodoAtual: IXMLNode;
    strAux, PalavraChave: string;
    ExecutaCodFluxograma: boolean;
    ListaAuxPalavrasChave: TStringList;
begin
   if nodo.NodeType = ntElement then begin
      //seta o contexto do nodo e marca o cod de inicio e fim de cada tipo
de nodo
      if nodo.NodeName = 'FRH_GLOSSARIO' then begin
         if self.ContextoNodo <> CtGlossario then begin
            if self.ContextoNodo = '' then
               self.CodPrimeiroNodoGlossario := 1
            else begin
```

```
               self.CodPrimeiroNodoGlossario := self.CodNodo + 1;
               self.CodUltimoNodoUnConceito := self.CodNodo;
            end;
            self.ContextoNodo := CtGlossario;
         end;
      end
      else if nodo.NodeName = 'FRH_UNIDADE_CONCEITO' then begin
         if self.ContextoNodo <> CtUnidadeConceito then begin
            if self.ContextoNodo = '' then
               self.CodPrimeiroNodoUnConceito := 1
            else begin
               self.CodPrimeiroNodoUnConceito := self.CodNodo + 1;
               self.CodUltimoNodoGlossario := self.CodNodo;
            end;
            self.ContextoNodo := CtUnidadeConceito;
         end;
      end;

      ExecutaCodFluxograma := ( (self.ContextoNodo =
CtGlossario)and(gFluxograma in self.FuncoesGlossario) )or(
(self.ContextoNodo = CtUnidadeConceito)and(ucFluxograma in
FuncoesUnidadeConceito) );

      //se for um nodo de conteudo deve possuir os atributos FRH_ESTILO e
FRH_TITULO. É atribuido um cod para o nodo. Também forma a arvore para o
fluxograma.
      if
(nodo.HasAttribute('FRH_ESTILO'))and(nodo.HasAttribute('FRH_TITULO')) then
begin
         CodNodo := CodNodo + 1;
         //insere o código do nodo hipermídia no nodo XML
         nodo.Attributes['FRH_CODNODO'] := CodNodo;

         //forma a árvore para o fluxograma
         if ExecutaCodFluxograma then begin
            inc(self.NivelNodoHiper);
            self.FluxogramaInsereNodoConteudo(nodo);
         end;

         //se tem palavras chaves entao e feito o mapeamento delas para o
cod do nodo.
```

```
         if ( (gLinksAutomaticos in
Self.FuncoesGlossario)and(self.ContextoNodo = CtGlossario) )or(
(ucLinksAutomaticos in FuncoesUnidadeConceito)and(self.ContextoNodo =
CtUnidadeConceito) ) then begin
            if nodo.HasAttribute('FRH_PALAVRASCH') then begin
               ListaAuxPalavrasChave := TStringList.Create();
               try
                 PalavraChave := Trim(nodo.Attributes['FRH_PALAVRASCH']);
                 nrInseridos := ExtractStrings(['\'], [],
PChar(PalavraChave), ListaAuxPalavrasChave);
                 for j:= 0 to nrInseridos-1 do begin

ListaInfoNodos.AddObject(ListaAuxPalavrasChave.Strings[j],
TNodoHiper.create(CodNodo,ContextoNodo, PalavraChave));
                 end;
               finally
                 ListaAuxPalavrasChave.Free();
               end;
            end;
         end;
      end
      //se o nodo for um agrupador de nodos (sem conteúdo, apenas junta uma
serie de nodos) forma a arvore para o fluxograma.
      else if ExecutaCodFluxograma then begin
         if
(nodo.HasAttribute('FRH_GRUPO'))and(nodo.HasAttribute('FRH_TITULO')) then
begin
           inc(self.NivelNodoHiper);
           self.FluxogramaInsereNodoGrupo(nodo);
        end;
      end;

      //chama a esta função para cada filho do nodo atual
      nodoAtual := nodo.ChildNodes.First;
      for i:=0 to nodo.ChildNodes.Count-1 do begin
          if nodoAtual.NodeType = ntElement then begin
             loopXML_criaTabelaNomes(nodoAtual);      //soh continua na
arvore se o nodo eh um elemento XML
          end;
          nodoAtual := nodoAtual.NextSibling;
      end;
```

```
      //se o nodo e um nodo hipermidia (de conteudo ou agrupamento)
decrementa o nivel na saida da função.
      if ExecutaCodFluxograma then begin
         if
((nodo.HasAttribute('FRH_ESTILO'))or(nodo.HasAttribute('FRH_TITULO'))) and
(nodo.HasAttribute('FRH_TITULO')) then begin
            FluxogramaInsereFecha();
            Dec(self.NivelNodoHiper);
         end;
      end;
   end;

end;

{-------------------------------------------------------------}
Procedure TFRH.FluxogramaInsereFecha();
var strBranco: string;
    i: integer;
begin
    strBranco := '';
    for i:= 1 to self.NivelNodoHiper do
        strBranco := strBranco + '   ';
    self.FluxogramaStrNodes := self.FluxogramaStrNodes + strBranco + ']';
end;

{-------------------------------------------------------------}
Procedure TFRH.FluxogramaInsereNodoConteudo(nodo: IXMLNode);
var StrBranco: string;
    i: integer;
begin
    StrBranco := '';
    for i:= 1 to self.NivelNodoHiper do
        StrBranco := strBranco + '   ';
    self.FluxogramaStrNodes := self.FluxogramaStrNodes +  #13#10 +
strBranco + ',['''' + Trim(nodo.Attributes['FRH_TITULO']) +
'''","Javascript:gerente.SegueLinkSimples(''' +
nodo.Attributes['FRH_CODNODO'] + '.html'',''' + self.ContextoNodo + ''','''
+ self.PegaCaminhoParaRaiz(self.DirFluxograma) + ''');",null';
end;
```

```
{-------------------------------------------------------------}
Procedure TFRH.FluxogramaInsereNodoGrupo(nodo: IXMLNode);
var strBranco: string;
    i: integer;
begin
    strBranco := '';
    for i:= 1 to self.NivelNodoHiper do
        strBranco := strBranco + '   ';
    self.FluxogramaStrNodes := self.FluxogramaStrNodes +  #13#10 +
strBranco + ',['''' + Trim(nodo.Attributes['FRH_TITULO']) + ''', null,
null';
end;

{-------------------------------------------------------------}
procedure TFRH.AplicaXSLIndiceGlossario();
var saida: WideString;
    arq_HTML: TextFile;
    XML_str: string;
    i: integer;
    NodoHiperGlossario: IXMLNode;
begin
   //acha o nodo FRH_GLOSSARIO
   for i:=0 to self.docXML.DocumentElement.ChildNodes.Count-1 do begin
       if self.docXML.DocumentElement.ChildNodes.Nodes[i].NodeName =
'FRH_GLOSSARIO' then begin
          NodoHiperGlossario :=
self.docXML.DocumentElement.ChildNodes.Nodes[i];
          break;
       end;
   end;

   if NodoHiperGlossario = nil then
      raise EErroFRH.Create('O elemento XML "FRH_GLOSSARIO" não existe ou
não é decendente direto do documento XML');
   if NodoHiperGlossario.HasAttribute('FRH_ESTILO_INDICE_GLOSSARIO') then
begin
      if Trim(NodoHiperGlossario.Attributes['FRH_ESTILO_INDICE_GLOSSARIO'])
= '' then
         raise EErroFRH.Create('O elemento XML "FRH_GLOSSARIO" deve ter um
valor especificado');
   end
```

```
   else
      raise EErroFRH.Create('O elemento XML "FRH_GLOSSARIO" deve ter o
atributo "FRH_ESTILO_INDICE_GLOSSARIO" especificado');
   //Insere linha de declaracao para o fragmento, acertando assim a
codificacao
   XML_str := '<?xml version="1.0" encoding="ISO-8859-1"?>' +
NodoHiperGlossario.XML;
   self.DomDocXML.DefaultInterface.loadXML(XML_str);
   self.DomDocXSL.DefaultInterface.load(XSL_path +
NodoHiperGlossario.Attributes['FRH_ESTILO_INDICE_GLOSSARIO'] + '.xsl');

   saida :=
self.DomDocXML.DefaultInterface.transformNode(self.DomDocXSL.DefaultInterfa
ce.documentElement);

   //Conserta os estragos que o XSLT fez na representação de caracteres
especiais
   saida := self.RetiraEntidadesEspeciaisDosLinks(saida);

   //grava o HTML
   assignFile(arq_HTML, self.Relatorio_path + self.DirGlossarioIndice +
self.NomeGlossarioIndice);
   rewrite(arq_HTML);
   writeln(arq_HTML, saida);
   close(arq_HTML);

end;

{-------------------------------------------------------------}
Procedure TFRH.LoopXML_CriaLinks(nodo: IXMLDOMNode);
var i: integer;
    nodoAtual: IXMLDOMNode;
    PathArquivoXSL: string;
    Atrs, NodoElementoConteudo: IXMLDOMNode;
begin
   case nodo.nodeType of
      CtNodoTipoElemento: begin
                     //seta o contexto do nodo
                     if nodo.NodeName = 'FRH_GLOSSARIO' then
                        self.ContextoNodo := CtGlossario
                     else if nodo.NodeName = 'FRH_UNIDADE_CONCEITO' then
```

```
                        self.ContextoNodo := CtUnidadeConceito;
                    Atrs := nodo.attributes.getNamedItem('FRH_CODNODO');
                    if Atrs <> nil then begin     //caso seja um Nodo
Hipermídia, tem que estar com seu codigo.
                        self.CodNodo:=
StrToInt(nodo.attributes.getNamedItem('FRH_CODNODO').nodeValue);
                    end;
                    nodoAtual := nodo.FirstChild;
                    for i:=0 to nodo.ChildNodes.length-1 do begin
                        self.LoopXML_CriaLinks(nodoAtual);
                        nodoAtual := nodoAtual.NextSibling;
                    end;
                end;
      CtNodoTipoTexto, CtNodoTipoCData:
                begin
                    if ( (gLinksAutomaticos in
Self.FuncoesGlossario)and(self.ContextoNodo = CtGlossario) )or(
(ucLinksAutomaticos in FuncoesUnidadeConceito)and(self.ContextoNodo =
CtUnidadeConceito) ) then
                        Atrs :=
nodo.parentNode.attributes.getNamedItem('FRH_LINKS');
                        if (Atrs = nil) then begin     //verifica se o nodo
deve ter links
                           self.CriaLinks(nodo);
                        end
                        else if ( Trim(Atrs.nodeValue) <> '') then
                              self.CriaLinks(nodo);
                end;
   end;

end;

{-------------------------------------------------------------}

//LEMBAR DE NAO MUDAR O CASE DAS PALAVRAS ORIGINAIS

procedure TFRH.CriaLinks(nodo: IXMLDOMNode);
var NAux, n, i, j, k, NrInseridos, Numero: integer;
    VetorIndices: array of integer;
    TextoNodo, Depois, StrArrayTitulo, StrArrayDestino, StrArrayTipo,
Antes, style: string;
```

```
    NodoAtual, Atrs, old, novoNodo: IXMLDOMNode;
    ListaNodos: IXMLDOMNodeList;
    ListaLinks, AuxPalavrasChave: TStringList;
    ListaPLinkAtual: TObjectList;
    ParaA, ParaB: boolean;
    indicesParaDeletar: array [0..500] of string;
begin
   AuxPalavrasChave := TStringList.Create();
   //esta lista contem as listas de cada link
   ListaLinks := TStringList.Create();
   ListaLinks.Sorted := true;
   ListaLinks.Duplicates := dupError;
//   self.ListaLinks.CaseSensitive := false;

   try
      //cria a lista de nodos especificada pelo usuário que podem ser
destino para algum link deste texto
      if self.TemAtributo(nodo.parentNode,'FRH_LINKS') then
         ListaNodos :=
nodo.parentNode.selectNodes(nodo.ParentNode.attributes.getNamedItem('FRH_LI
NKS').nodeValue)
      else
         ListaNodos := nodo.parentNode.selectNodes('/descendant-or-
self::*');
      //verifica strings que batem.
      for i:=0 to ListaNodos.length-1 do begin
          NodoAtual := ListaNodos.item[i];
          //verifica se o nodoAtual é um nodo Hipermídia e se tem palavras
chave.
          if (
(self.EhNodoHipermidia(NodoAtual))and(self.TemAtributo(NodoAtual,'FRH_PALAV
RASCH')) ) then begin

//             if
(NodoAtual.attributes.getNamedItem('FRH_CODNODO').nodeValue <> CodNodo)
then begin  //não verifica links para o proprio nodo
               NrInseridos := ExtractStrings(['\'], [], PChar(
Trim(NodoAtual.attributes.getNamedItem('FRH_PALAVRASCH').nodeValue) ),
AuxPalavrasChave);
               for j:= 0 to nrInseridos-1 do begin
```

```
                    if AnsiContainsText(nodo.nodeValue,
AuxPalavrasChave.Strings[j]) then begin
//
self.CopiaElementoListaInfoNodos(AuxPalavrasChave.Strings[j],
NodoAtual.attributes.getNamedItem('FRH_CODNODO').nodeValue,
ListaPalavrasChave);
                       //manda a palavra chave e o codigo do nodo. esta
funcão ira adicionar um ítem nas strings de titulos, destinos e contexto

self.AdicionaInformacoesLinks(AuxPalavrasChave.Strings[j],
NodoAtual.attributes.getNamedItem('FRH_CODNODO').nodeValue, ListaLinks);
                    end;
                end;
                AuxPalavrasChave.Clear();
//              end;
            end;
        end;

        numero := 0;
        for i:=0 to ListaLinks.Count-1 do begin
            for j:=0 to ListaLinks.Count-1 do begin
                if i<>j then begin
                   if AnsiContainsText(ListaLinks.Strings[i],
ListaLinks.Strings[j]) then begin
                      indicesParaDeletar[numero] := ListaLinks.Strings[j];
                      numero := numero+1;
                   end
                end;
            end;
        end;

        if numero <> 0 then begin
            for i:=0 to numero-1 do begin
                ListaLinks.Find( indicesParaDeletar[i], j);
                ListaLinks.Delete(j);
            end;
        end;
```

```
      //passa pelos objetos da ListaLinks e cria os links para JS: 3
parametros: titulos,destinos,tipo
      //IMPORTANTE: o Processador XSLT insere entidades no lugar de
caracteres especiais como '<' e '>'. isto eh corrigido logo após a formacao
do HTML.
      for n:= 0 to ListaLinks.Count-1 do begin
          ListaPLinkAtual := TObjectList(ListaLinks.Objects[n]);
          if ListaPLinkAtual.Count > 1 then
             Style := '"FRH_Multiplo"'
          else begin
              if TParametrosLink(ListaPLinkAtual.Items[0]).StrArrayContexto
= CtGlossario then
                 style := '"FRH_Glossario"'
              else
                 style := '"FRH_UnidadeConceito"';
          end;
          //Pega o valor correto de maiúsculas e minúsculas do texto do
nodo
           Antes :=  ListaLinks.Strings[n];  //Esta é a palavra chave a ser
substituida
{          Antes :=  '';
          TextoNodo := nodo.nodeValue;
          NAux := AnsiPos(ListaLinks.Strings[n], TextoNodo);
          for k:=NAux to Length(Antes) do
              Antes := Antes + TextoNodo[k];
 }
          //cria os parâmetros da função JS.
          StrArrayTitulo := '''' +
TParametrosLink(ListaPLinkAtual.Items[0]).StrArrayTitulo + '''';
          StrArrayDestino := '''' +
TParametrosLink(ListaPLinkAtual.Items[0]).StrArrayDestino + '.html''';
          StrArrayTipo := '''' +
TParametrosLink(ListaPLinkAtual.Items[0]).StrArrayContexto + '''';
          for i:=1 to ListaPLinkAtual.Count-1 do begin
              //faz os dados para o restante dos links, se houver
              StrArrayTitulo := StrArrayTitulo + ',''' +
TParametrosLink(ListaPLinkAtual.Items[0]).StrArrayTitulo + '''';
              StrArrayDestino := StrArrayDestino + ',''' +
TParametrosLink(ListaPLinkAtual.Items[0]).StrArrayDestino + '.html''';
              StrArrayTipo := StrArrayTipo + ',''' +
TParametrosLink(ListaPLinkAtual.Items[0]).StrArrayContexto + '''';
```

```
          end;
          Depois := '<a class=' + style + ' href="javascript: segueLink(new
Array(' + StrArrayTitulo + '),new Array(' + StrArrayDestino + '),new
Array(' + StrArrayTipo + '));">' + Antes + '</a>';
          //substitui todas as ocorrencias da palavra chave pelo link
construido.
          //o link deve ser posto numa CDATA!!!
          if nodo.NodeType = CtNodoTipoCData then
             nodo.NodeValue := AnsiReplaceText(nodo.NodeValue, Antes,
Depois)
          else begin
          //se o nodo atual não é uma CDATA, então cria uma no lugar do
nodo texto, aproveitando para incluir o link
             old := nodo.parentnode;
             novoNodo := nodo.ownerDocument.createCDATASection(
AnsiReplaceText(nodo.NodeValue, Antes, Depois) );
             nodo.ParentNode.replaceChild(novoNodo, nodo);
             nodo := novoNodo;
          end;
      end;
   finally
      for i:=0 to ListaLinks.Count-1 do
          ListaLinks.Objects[i].Free();
      ListaLinks.Free();
      AuxPalavrasChave.Free();
   end;

end;



{-------------------------------------------------------------}
procedure TFRH.AdicionaInformacoesLinks(Palavra: string; PCodNodo: integer;
var ListaLinks: TStringList);
var k, i, m: integer;
    Para: boolean;
    ListaObjetosParametrosLink: TObjectList;
    ObjNodoHiper: TNodoHiper;
    ObjParametros: TParametrosLink;
begin
    Para := false;
```

```
     //recupera o objeto NodoHiper da ListaInfoNodos. Procura ate achar a
palavra e o nodo iguais aos dos parametros.
     self.ListaInfoNodos.Find(Palavra, m);
     while ((Para=False)and(m<=self.ListaInfoNodos.Count-1)) do begin
        if ((Palavra =
self.ListaInfoNodos.Strings[m])and(TNodoHiper(self.ListaInfoNodos.Objects[m
]).CodNodo = PCodNodo)) then begin
           ObjNodoHiper := TNodoHiper(self.ListaInfoNodos.Objects[m]);
           Para := True;
        end
        else
           m := m+1;
     end;

     //agora insere os parâmetros no link correto da listaLinks
     //primeiro constroi o objeto parametros;
     ObjParametros :=
TParametrosLink.Create(ObjNodoHiper.Titulo,IntToStr(ObjNodoHiper.CodNodo),O
bjNodoHiper.ContextoNodo);
     if ListaLinks.Find(Palavra, k) then begin
     //O link ja existe. Adiciona um conjunto de parametros no objeto que
representa a lista de parametros
        TObjectList(ListaLinks.Objects[k]).Add(ObjParametros);
     end
     else begin
     //Não existe este link ainda, então cria e insere um conjunto de
parametros no objeto que representa a lista de parametros, recem criado.
        ListaObjetosParametrosLink := TObjectList.Create(True);
        ListaObjetosParametrosLink.Add(ObjParametros);
        ListaLinks.AddObject(Palavra, ListaObjetosParametrosLink);
     end;

end;

{-------------------------------------------------------------}
function TFRH.EhNodoHipermidia(nodo: IXMLDOMNode):boolean;
var Atrs1, Atrs2: IXMLDOMNode;
begin
   Atrs1 := nodo.attributes.getNamedItem('FRH_CODNODO');
   Atrs2 := nodo.attributes.getNamedItem('FRH_TITULO');
```

```
   if ( (Atrs1 <> nil)and(Atrs2 <> nil) ) then begin     //caso seja um
Nodo Hipermídia, tem que estar com seu codigo e com o título
      result := true;
   end
   else
      result := false;
end;

{-------------------------------------------------------------}
function TFRH.TemAtributo(Nodo: IXMLDOMNode; NomeAtributo: string):boolean;
var Atrs: IXMLDOMNode;
begin
   Atrs := nodo.attributes.getNamedItem(NomeAtributo);
   if Atrs <> nil then begin
      result := true;
   end
   else
      result := false;
end;

{-------------------------------------------------------------}
Procedure TFRH.loopXML_aplicaXSL(nodo: IXMLDOMNode);
var i: integer;
    nodoAtual: IXMLDOMNode;
    PathArquivoXSL: string;
    Atrs, NodoElementoConteudo: IXMLDOMNode;
begin
   if nodo.nodeType = CtNodoTipoElemento then begin
       //seta o contexto do nodo
       if nodo.NodeName = 'FRH_GLOSSARIO' then
          self.ContextoNodo := CtGlossario
       else if nodo.NodeName = 'FRH_UNIDADE_CONCEITO' then
          self.ContextoNodo := CtUnidadeConceito;
       Atrs := nodo.attributes.getNamedItem('FRH_CODNODO');
       if Atrs <> nil then begin     //caso seja um Nodo Hipermídia, tem
que estar com seu codigo.
          self.CodNodo:=
StrToInt(nodo.attributes.getNamedItem('FRH_CODNODO').nodeValue);
       end;
       nodoAtual := nodo.FirstChild;
       for i:=0 to nodo.ChildNodes.length-1 do begin
```

```
            self.LoopXML_aplicaXSL(nodoAtual);
            nodoAtual := nodoAtual.NextSibling;
        end;
        //só formata os nodos codificados anteriormente
        if
(nodo.attributes.getNamedItem('FRH_ESTILO')<>nil)and(nodo.attributes.getNam
edItem('FRH_CODNODO')<>nil) then begin
           PathArquivoXSL := self.XSL_path +
nodo.attributes.getNamedItem('FRH_ESTILO').nodeValue + '.xsl';
           self.FormataNodo(nodo, PathArquivoXSL);
        end;
    end;

end;

{-------------------------------------------------------------}
procedure TFRH.FormataNodo(NodoHiper: IXMLDOMNode; PathArquivoXSL: string);
var saida: WideString;
    arq_HTML: TextFile;
    XML_str: string;
    XHTML : IXMLDOMNode;
    ListNodoHead: IXMLDOMNodeList;
    NomeHTML, PalavrasChave, TextoXHTML: string;
    Para: boolean;
    i, TamanhoA, IndiceInicioA: integer;
begin
   self.DomDocXSL.DefaultInterface.load(PathArquivoXSL);
//   ComponentesXML.DOMDocXHTML.DefaultInterface.a :=
nodoHiper.transformNode(self.DomDocXSL.DefaultInterface.documentElement);
   saida :=
NodoHiper.transformNode(self.DomDocXSL.DefaultInterface.documentElement);
   ComponentesXML.DOMDocXHTML.DefaultInterface.loadXML(saida);
   saida :=
ComponentesXML.DOMDocXHTML.DefaultInterface.documentElement.xml;

   //faz o arquivo de indice para busca de FULL TEXT SEARCH.
      //prepara os parametros
      //deleta o HEAD do xhtml

    if self.ContextoNodo = CtUnidadeConceito then begin   //soh inclui a
busca para unidades de conceito!
```

```
       NomeHTML := self.PegaCaminhoParaRaiz(self.DirBusca) +
self.DirUnidadeConceito +
NodoHiper.attributes.getNamedItem('FRH_CODNODO').nodeValue + '.html';

       ListNodoHead :=
ComponentesXML.DOMDocXHTML.DefaultInterface.documentElement.getElementsByTa
gName('HEAD');
       if ListNodoHead.item[0] = nil then
          ListNodoHead :=
ComponentesXML.DOMDocXHTML.DefaultInterface.documentElement.getElementsByTa
gName('head');
       if ListNodoHead.item[0] <> nil then

ComponentesXML.DOMDocXHTML.DefaultInterface.documentElement.removeChild(Lis
tNodoHead.item[0]);

       self.ArqIndiceBusca.Add('<url>' + NomeHTML + '</url>');
       TextoXHTML :=
Trim(ComponentesXML.DOMDocXHTML.DefaultInterface.text);

       para:= false;
       while Para=false do begin
           IndiceInicioA := Pos('<a class="',TextoXHTML);
           if IndiceInicioA > 0 then begin
               TamanhoA := 0;
               for i:= IndiceInicioA to Length(TextoXHTML)-1 do begin
                   if TextoXHTML[i] = '"' then begin
                      if i = Length(TextoXHTML)-1 then
                         break
                      else if TextoXHTML[i+1] = '>' then begin
                         TamanhoA := TamanhoA + 2;
                         break;
                      end;
                   end;
                   TamanhoA := TamanhoA + 1;
               end;
               Delete(TextoXHTML,IndiceInicioA,TamanhoA);
           end
           else
              Para := True;
       end;
```

```
       TextoXHTML := AnsiReplaceText(TextoXHTML, '</a>', '');

       self.ArqIndiceBusca.Add('<title>' +
NodoHiper.attributes.getNamedItem('FRH_TITULO').nodeValue + '</title>' +
TextoXHTML );
       self.ArqIndiceBusca.Add('');
   end;
   //Conserta os estragos que o XSLT fez na representação de caracteres
especiais
   saida := self.RetiraEntidadesEspeciaisDosLinks(saida);
   saida := InsereScriptsNodosHiper(saida, NodoHiper);

   //grava o HTML
   assignFile(arq_HTML, self.Relatorio_path + self.DirUnidadeConceito +
nodoHiper.Attributes.getNamedItem('FRH_CODNODO').nodeValue + '.html');
   rewrite(arq_HTML);
   writeln(arq_HTML, saida);
   close(arq_HTML);

end;

{-------------------------------------------------------------}
function TFRH.RetiraEntidadesEspeciaisDosLinks(TextoHTML: string): string;
begin
   TextoHTML := AnsiReplaceText(TextoHTML, '&lt;a class="', '<a class="');
   TextoHTML := AnsiReplaceText(TextoHTML, '''));"&gt;', '''));">');
   TextoHTML := AnsiReplaceText(TextoHTML, '&lt;/a&gt;', '</a>');
   result := TextoHTML;
end;

{-------------------------------------------------------------}
function TFRH.InsereScriptsNodosHiper(textoHTML: string; nodo:
IXMLDOMNode): string;
var n: integer;
    script, StrVarsJs: string;
begin
    //seta a relacao do nodo com o gerente
    if self.ContextoNodo = CtGlossario then begin
       StrVarsJs := 'var gerente=' + Self.GlossarioNodosParaGerente + ';' +
#13#10;
```

```
       StrVarsJs := StrVarsJs + 'var FRH_dir_raiz=''' +
self.PegaCaminhoParaRaiz(Self.DirNodosGlossario) + ''';' + #13#10;
       StrVarsJs := StrVarsJs + 'var FRH_tipo_nodo=''' + CtGlossario +
''';' + #13#10;
    end
    else if self.ContextoNodo = CtUnidadeConceito then begin
       StrVarsJs := 'var gerente=' + Self.UnidadeConceitoParaGerente + ';'
+ #13#10;
       StrVarsJs := StrVarsJs + 'var FRH_dir_raiz=''' +
self.PegaCaminhoParaRaiz(Self.DirUnidadeConceito) + ''';' + #13#10;
       StrVarsJs := StrVarsJs + 'var FRH_tipo_nodo=''' + CtUnidadeConceito
+ ''';' + #13#10;
       //encontra os nodos para navegacao na estrutura
       StrVarsJs := StrVarsJS + 'var FRH_nodo_pai=''' +
self.PegaCodHTMLNodoHiperPai(nodo) + ''';' + #13#10;
    end;

    //encontra os nodos para navegacao na estrutura
    StrVarsJs := StrVarsJs + 'var FRH_nodo_seguinte=''' +
self.PegaCodHTMLNodoHiperSeguinte(nodo) + ''';' + #13#10;
    StrVarsJs := StrVarsJs + 'var FRH_nodo_anterior=''' +
self.PegaCodHTMLNodoHiperAnterior(nodo) + ''';' + #13#10;
    StrVarsJs := StrVarsJs + 'var FRH_nodo_corrente=''' +
nodo.Attributes.getNamedItem('FRH_CODNODO').nodeValue + '.html'';' +
#13#10;
    //codigo JS a ser inserido
    script := #13#10 + '<SCRIPT LANGUAGE="JavaScript">' + #13#10;
    script := script + '<!--' + #13#10;
    script := script + '/*------------  FRH: Variáveis de navegação  ------
------------*/' + #13#10 + #13#10;
    script := script + StrVarsJs + #13#10;
    script := script + '//-->' + #13#10;
    script := script + '</SCRIPT>' + #13#10;
    script := script + #13#10 + '<SCRIPT LANGUAGE="JavaScript" SRC="' +
self.PegaCaminhoParaRaiz(self.DirUnidadeConceito) + self.DirScripts +
'script_nodos.js"></SCRIPT>' + #13#10;

    //procura pelos elementos HEAD e HTML para inserir na posicao correta
    n := Pos('<HEAD>',textoHTML);
    if n = 0 then   //tenta denovo
       n := Pos('<head>',textoHTML);
```

```
    //adiciona <HEAD>
    if n = 0 then begin
       n := Pos('<HTML>',textoHTML);
       if n = 0 then
          n := Pos('<html>',textoHTML);
       script := '<HEAD>' + #13#10 + script + #13#10'</HEAD>' + #13#10;
       Insert(script, textoHTML, n+6);
    end
    //Ja existe o <HEAD>
    else begin
      Insert(script, textoHTML, n+6);
    end;
    result := textoHTML;

end;

{-------------------------------------------------------------}
function TFRH.PegaCodHTMLNodoHiperPai(NodoHiper: IXMLDOMNode): string;
var nodo: IXMLDOMNode;
begin
    nodo := AchaNodoHiperPai(NodoHiper);
    if nodo = nil then
       result := ''
    else
       result := nodo.Attributes.getNamedItem('FRH_CODNODO').nodeValue +
'.html';
end;

{-------------------------------------------------------------}
function TFRH.PegaCodHTMLNodoHiperSeguinte(NodoHiper: IXMLDOMNode): string;
var codNodo: integer;
begin
    codNodo :=
StrToInt(NodoHiper.Attributes.getNamedItem('FRH_CODNODO').nodeValue);
    if self.ContextoNodo = CtGlossario then begin
       if (codNodo = self.CodUltimoNodo)or(codNodo =
self.CodUltimoNodoGlossario) then
          result := ''
       else
          result := IntToStr(codNodo + 1) + '.html';
```

```
    end
    else if self.ContextoNodo = CtUnidadeConceito then begin
       if (codNodo = self.CodUltimoNodo)or(codNodo =
self.CodUltimoNodoUnConceito) then
          result := ''
       else
          result := IntToStr(codNodo + 1) + '.html';
    end;
end;

{-------------------------------------------------------------}
function TFRH.PegaCodHTMLNodoHiperAnterior(NodoHiper: IXMLDOMNode): string;
var codNodo: integer;
begin
    codNodo :=
StrToInt(NodoHiper.Attributes.getNamedItem('FRH_CODNODO').nodeValue);

    if self.ContextoNodo = CtGlossario then begin
       if (codNodo = 1)or(codNodo = self.CodPrimeiroNodoGlossario) then
          result := ''
       else
          result := IntToStr(codNodo - 1) + '.html';
    end
    else if self.ContextoNodo = CtUnidadeConceito then begin
       if (codNodo = 1)or(codNodo = self.CodPrimeiroNodoUnConceito) then
          result := ''
       else
          result := IntToStr(codNodo - 1) + '.html';
    end;

end;

{-------------------------------------------------------------}
function TFRH.AchaNodoHiperPai(NodoHiper: IXMLDOMNode): IXMLDOMNode;
var Achou, NaoExiste: boolean;
    NodoAtual: IXMLDOMNode;
begin
    Achou := False;
    NaoExiste := False;
    if NodoHiper.ParentNode = nil then
       NaoExiste := true
```

```
    else
       NodoAtual := NodoHiper.ParentNode;

    while (not Achou) and (not NaoExiste) do begin
        if NodoAtual.attributes.getNamedItem('FRH_CODNODO')<>nil then
           Achou := true
        else begin
           if (NodoAtual.ParentNode =
nil)or(NodoAtual.ParentNode.nodeTypeString = 'document') then
              NaoExiste := true
           else
              NodoAtual := NodoAtual.ParentNode;
        end;
    end;
    if NaoExiste then
       result := nil
    else
       result := NodoAtual;
end;

{-------------------------------------------------------------}
function TFRH.AchaNodoHiperFilho(NodoHiper: IXMLDOMNode): IXMLDOMNode;
var NaoExiste, Achou: boolean;
    NodoAtual, NodoFilho: IXMLDOMNode;
begin
    NaoExiste := false;
    Achou := false;
    NodoAtual := NodoHiper.firstChild;
    if NodoAtual = nil then begin
       result := nil;
       NaoExiste := True;
    end;

    while (not NaoExiste) and (not Achou) do begin
        if NodoAtual.attributes.getNamedItem('FRH_CODNODO')<>nil then begin
           Achou := true;
           result := NodoAtual;
        end
        else begin
           NodoFilho := self.AchaNodoHiperFilho(NodoAtual);
           if NodoFilho <> nil then begin
```

```
            Achou := True;
            result := NodoFilho;
         end
         else begin
            NodoAtual := NodoAtual.nextSibling;
            if NodoAtual = nil then begin
               result := nil;
               NaoExiste := True;
            end
         end;
      end;
   end;

end;

{-------------------------------------------------------------}
function TFRH.AchaNodoHiperSeguinte(NodoHiper: IXMLDOMNode): IXMLDOMNode;
var NaoExiste, Achou, AchouCandidato: boolean;
   NodoPai, NodoFilho, NodoAtual: IXMLDOMNode;
begin
   if (NodoHiper.NextSibling = nil) then begin
      NodoPai := NodoHiper;
      Achou := false;
      NaoExiste := false;
      while (not NaoExiste) and (not Achou) do begin
         //se nao tem mais seguintes do nodo pai, pega o pai do pai (um
novo candidato)
         AchouCandidato := false;
         while (not NaoExiste) and (not AchouCandidato) do begin
            NodoPai := NodoPai.ParentNode;
            if NodoPai = nil then begin
               NaoExiste := true;
               result := nil;
            end
            else begin
               if NodoPai.NextSibling <> nil then
                  AchouCandidato := true;
                  NodoAtual := NodoPai.NextSibling;   //seta o candidato
para o proximo loop
            end;
         end;
```

```
            //para os seguintes do nodo pai, faz a procura
            while ((not NaoExiste) and (not Achou) and (NodoAtual <> nil))
do begin
               if NodoAtual.HasChildNodes then begin
                  NodoFilho := self.AchaNodoHiperFilho(NodoAtual);
                  if NodoFilho <> nil then begin
                     Achou := True;
                     result := NodoFilho;
                  end
                  else
                     NodoAtual := NodoAtual.NextSibling;
               end
               else
                  NodoAtual := NodoAtual.NextSibling;
            end;
         end;
      end


      else begin
         Achou := false;
         NaoExiste := false;
         NodoAtual := NodoHiper;
         if NodoHiper.HasChildNodes then begin
            NodoFilho := AchaNodoHiperFilho(NodoHiper);
            if NodoFilho <> nil then
               Achou := True;
               result := NodoFilho;
         end;
         while (not NaoExiste) and (not Achou) do begin
             NodoAtual := NodoAtual.NextSibling;
             if (NodoAtual.NodeType = CtNodoTipoTexto)or(NodoAtual.NodeType =
CtNodoTipoCData )then
                 Achou := Achou;
            if NodoAtual = nil then begin
               result := nil;
               NaoExiste := true;
            end
            else begin
               if NodoAtual.attributes.getNamedItem('FRH_CODNODO')<>nil then
begin
```

```
                Achou := true;
                result := NodoAtual;
             end
             else begin
                if NodoAtual.HasChildNodes then begin
                   NodoFilho := AchaNodoHiperFilho(NodoAtual);
                   if NodoFilho <> nil then
                      Achou := true;
                      result := NodoFilho;
                end;
             end;
          end;
          if NaoExiste then
             result := nil
       end;
    end;

end;

{-------------------------------------------------------------}
function  TFRH.AchaNodoHiperAnterior(NodoHiper: IXMLDOMNode): IXMLDOMNode;
begin
//similar ao achaNodoHiperSeguinte, mudando basicamente a direcao.
end;

{-------------------------------------------------------------}
procedure TFRH.CriaScriptsApoio();
var arq_JS: TextFile;
    str: string;
begin

    str := '/*----------------------   FRH: Constantes    -----------------
----*/' + #13#10 + #13#10;
    str := str + '/*------- Caminhos relativos dos frames e componentes   -
-------*/' + #13#10;
    str := str + 'DirScripts = "' + self.DirScripts + '";' + #13#10;

    str := str + 'DirUnidadeConceito = "' + self.DirUnidadeConceito + '";'
+ #13#10;
    str := str + 'DirUnidadeConceito_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirUnidadeConceito) + '";' + #13#10 + #13#10;
```

```
    str := str + 'DirNodosGlossario = "' + self.DirNodosGlossario + '";' +
#13#10;
    str := str + 'DirNodosGlossario_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirNodosGlossario) + '";' + #13#10 + #13#10;

    str := str + 'DirImg = "' + self.DirImg + '";' + #13#10;

    str := str + 'NomeIndex = "' + self.NomeIndex + '";' + #13#10;
    str := str + 'DirIndex = "' + self.DirIndex + '";' + #13#10;
    str := str + 'DirIndex_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirIndex) + '";' + #13#10 + #13#10;

    str := str + 'NomeFrameGerente = "' + self.NomeFrameGerente + '";' +
#13#10;
    str := str + 'DirFrameGerente = "' + self.DirFrameGerente + '";' +
#13#10;
    str := str + 'DirFrameGerente_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirFrameGerente) + '";' + #13#10 + #13#10;

    str := str + 'NomeFerramentas = "' + self.NomeFerramentas + '";' +
#13#10;
    str := str + 'DirFerramentas = "' + self.DirFerramentas + '";' +
#13#10;
    str := str + 'DirFerramentas_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirFerramentas) + '";' + #13#10 + #13#10;

    str := str + 'NomeFluxograma = "' + self.NomeFluxograma + '";' +
#13#10;
    str := str + 'DirFluxograma = "' + self.DirFluxograma + '";' + #13#10;
    str := str + 'DirFluxograma_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirFluxograma) + '";' + #13#10 + #13#10;

    str := str + 'NomeBusca = "' + self.NomeBusca + '";' + #13#10;
    str := str + 'DirBusca = "' + self.DirBusca + '";' + #13#10;
    str := str + 'DirBusca_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirBusca) + '";' + #13#10 + #13#10;

    str := str + 'NomeIndice = "' + self.NomeIndice + '";' + #13#10;
    str := str + 'DirIndice = "' + self.DirIndice + '";' + #13#10;
    str := str + 'DirIndice_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirIndice) + '";' + #13#10 + #13#10;
```

```
    str := str + 'NomeGlossarioIndice = "' + self.NomeGlossarioIndice +
'";' + #13#10;
    str := str + 'DirGlossarioIndice = "' + self.DirGlossarioIndice + '";'
+ #13#10;
    str := str + 'DirGlossarioIndice_raiz = "' +
self.PegaCaminhoParaRaiz(self.DirGlossarioIndice) + '";' + #13#10 + #13#10;

    str := str + 'NomeXML = "' + self.NomeXML + '";' + #13#10;
    str := str + 'DirXML = "' + self.DirXML + '";' + #13#10;

    //CONTEINERS

//    str := str + 'NomeConteinerUnidadeConceito = "' +
self.NomeConteinerUnidadeConceito + '";' + #13#10;
    str := str + 'NomeConteinerFluxograma = "' +
self.NomeConteinerFluxograma + '";' + #13#10;
    str := str + 'NomeConteinerGlossarioNodos = "' +
self.NomeConteinerGlossarioNodos + '";' + #13#10;
    str := str + 'NomeConteinerBusca = "' + self.NomeConteinerBusca + '";'
+ #13#10;
    str := str + 'NomeConteinerIndice = "' + self.NomeConteinerIndice +
'";' + #13#10;

//    str := str + 'ConteinerParaUnidadeConceito = "' +
self.ConteinerParaUnidadeConceito + '";' + #13#10;
    str := str + 'ConteinerParaGlossarioNodos = "' +
self.ConteinerParaGlossarioNodos + '";' + #13#10;
    str := str + 'ConteinerParaFluxograma = "' +
self.ConteinerParaFluxograma + '";' + #13#10;
    str := str + 'ConteinerParaBusca = "' + self.ConteinerParaBusca + '";'
+ #13#10;
    str := str + 'ConteinerParaIndice = "' + self.ConteinerParaIndice +
'";' + #13#10 + #13#10;

//    str := str + 'OpcoesConteinerUnidadeConceito = "' +
self.OpcoesConteinerUnidadeConceito + '";' + #13#10;
    str := str + 'OpcoesConteinerGlossarioNodos = "' +
self.OpcoesConteinerGlossarioNodos + '";' + #13#10;
    str := str + 'OpcoesConteinerFluxograma = "' +
self.OpcoesConteinerFluxograma + '";' + #13#10;
```

```
    str := str + 'OpcoesConteinerBusca = "' + self.OpcoesConteinerBusca +
'";' + #13#10;
    str := str + 'OpcoesConteinerIndice = "' + self.OpcoesConteinerIndice +
'";' + #13#10 + #13#10;

//    str := str + 'DirConteinerUnidadeConceito = "' +
self.DirConteinerUnidadeConceito + '";' + #13#10;
    str := str + 'DirConteinerGlossarioNodos = "' +
self.DirConteinerGlossarioNodos + '";' + #13#10;
    str := str + 'DirConteinerFluxograma = "' + self.DirConteinerFluxograma
+ '";' + #13#10;
    str := str + 'DirConteinerBusca = "' + self.DirConteinerBusca + '";' +
#13#10;
    str := str + 'DirConteinerIndice = "' + self.DirConteinerIndice + '";'
+ #13#10 + #13#10;

    str := str + 'DirConteinerGlossarioNodos_raiz = "' +
Self.PegaCaminhoParaRaiz(DirConteinerGlossarioNodos) + '";' + #13#10;
    str := str + 'DirConteinerFluxograma_raiz = "' +
Self.PegaCaminhoParaRaiz(self.DirConteinerFluxograma) + '";' + #13#10;
    str := str + 'DirConteinerBusca_raiz = "' +
Self.PegaCaminhoParaRaiz(self.DirConteinerBusca) + '";' + #13#10;
    str := str + 'DirConteinerIndice_raiz = "' +
Self.PegaCaminhoParaRaiz(self.DirConteinerIndice) + '";' + #13#10 + #13#10;

    assignFile(arq_JS, self.Relatorio_path + self.DirScripts +
'constantes.js');     //mudar
    rewrite(arq_JS);
    writeln(arq_JS, str);
    close(arq_JS);


end;

{-------------------------------------------------------------}
function TFRH.PegaCaminhoParaRaiz(path: string): string;
var i,n:integer;
    CaminhoInverso: string;
begin
    n := Length(path);
    CaminhoInverso := '';
```

```
    for i:= 1 to n do begin
        if path[i] = '/' then
           CaminhoInverso := CaminhoInverso + '../';
    end;
    result := CaminhoInverso;
end;

{-------------------------------------------------------------}
procedure TFRH.ResolveProcessingInstructions();
begin

   //INCLUI AS REFERENCIAS AO GERENTE EM CADA ARQUIVO. Também completa o
índice.

   //GERENTE
   InsereVariaveisFrameGerente();
   //Frame Ferramentas
   InsereVariavelJS(self.Relatorio_path + self.DirFerramentas +
self.NomeFerramentas, 'VInicialGerente', '''' + self.FerramentasParaGerente
+ '''');
   InsereVariavelJS(self.Relatorio_path + self.DirFerramentas +
self.NomeFerramentas, 'FRH_dir_raiz', '''' +
self.PegaCaminhoParaRaiz(self.DirFerramentas) + '''');

    //GLOSSARIO
    if (gIndice in self.FuncoesGlossario) then begin
       //Conteiner
       if self.NomeConteinerGlossarioNodos <> '' then begin
          InsereVariavelJS(self.Relatorio_path +
self.DirConteinerGlossarioNodos + self.NomeConteinerGlossarioNodos,
'VInicialGerente', '''' + self.ConteinerGlossarioNodosParaGerente + '''');
          InsereVariavelJS(self.Relatorio_path +
self.DirConteinerGlossarioNodos + self.NomeConteinerGlossarioNodos,
'FRH_dir_raiz', '''' +
self.PegaCaminhoParaRaiz(self.DirConteinerGlossarioNodos) + '''' );
       end;
    end;

    //FLUXOGRAMA
    if (ucFluxograma in self.FuncoesUnidadeConceito)or( (gFluxograma in
self.FuncoesGlossario)and(gIndice in self.FuncoesGlossario) ) then begin
```

```
       InsereVariavelJS(self.Relatorio_path + self.DirFluxograma +
self.NomeFluxograma, 'VInicialGerente', '''' + self.FluxogramaParaGerente +
'''');
       InsereVariavelJS(self.Relatorio_path + self.DirFluxograma +
self.NomeFluxograma, 'FRH_dir_raiz', '''' +
self.PegaCaminhoParaRaiz(self.DirFluxograma) + '''');
       //Conteiner
       if self.NomeConteinerFluxograma <> '' then begin
          InsereVariavelJS(self.Relatorio_path +
self.DirConteinerFluxograma + self.NomeConteinerFluxograma,
'VInicialGerente', '''' + self.ConteinerFluxogramaParaGerente + '''');
          InsereVariavelJS(self.Relatorio_path +
self.DirConteinerFluxograma + self.NomeConteinerFluxograma, 'FRH_dir_raiz',
'''' + self.PegaCaminhoParaRaiz(self.DirConteinerFluxograma) + '''');
       end;
    end;

    //INDICE
    if ucIndice in self.FuncoesUnidadeConceito then begin
       InsereVariavelJS(self.Relatorio_path + self.DirIndice +
self.NomeIndice, 'VInicialGerente', '''' + Self.IndiceParaGerente + '''');
       InsereVariavelJS(self.Relatorio_path + self.DirIndice +
self.NomeIndice, 'FRH_dir_raiz', '''' +
self.PegaCaminhoParaRaiz(self.DirIndice) + '''');
       //Conteiner
       if self.NomeConteinerIndice <> '' then begin
          InsereVariavelJS(self.Relatorio_path + self.DirConteinerIndice +
self.NomeConteinerIndice, 'VInicialGerente', '''' +
self.ConteinerIndiceParaGerente + '''');
          InsereVariavelJS(self.Relatorio_path + self.DirConteinerIndice +
self.NomeConteinerIndice, 'FRH_dir_raiz', '''' +
self.PegaCaminhoParaRaiz(self.DirConteinerIndice) + '''');
       end;
       ConstroiSelectOptionsIndice(self.Relatorio_path + self.DirIndice +
self.NomeIndice);
    end;

    //BUSCA: nao precisa se usar o java em nova janela. MUDAR!
end;

{-------------------------------------------------------------}
```

```
procedure TFRH.InsereVariaveisFrameGerente();
var Str, Script, PathGerente: string;
    Arq: TStringList;
begin
    PathGerente := self.Relatorio_path + self.DirFrameGerente +
self.NomeFrameGerente;
    Arq := TStringList.Create();
    Arq.LoadFromFile(PathGerente);

    Script := '<SCRIPT LANGUAGE="JavaScript">' + #13#10;
    Script := Script + '<!--' + #13#10;
    Script := Script + '/*-------------  FRH: Variáveis Geradas
Automaticamente ---------------*/' + #13#10;
    Script := Script + '/*----------- Caminho via Frames para os vários
documentos --------------*/' + #13#10;
    Script := Script + 'var VInicialGerenteParaUnidadeConceito = ''' +
self.GerenteParaUnidadeConceito + ''';' + #13#10;
    Script := Script + 'var VInicialGerenteParaGlossarioNodos = ''' +
self.GerenteParaGlossarioNodos + ''';' + #13#10;
    Script := Script + 'var VInicialGerenteParaFluxograma = ''' +
self.GerenteParaFluxograma + ''';' + #13#10;
    Script := Script + 'var VInicialGerenteParaBusca = ''' +
self.GerenteParaBusca + ''';' + #13#10;
    Script := Script + 'var VInicialGerenteParaIndice = ''' +
self.GerenteParaIndice + ''';' + #13#10;
    Script := Script + 'var VInicialGerenteParaFerramentas = ''' +
self.GerenteParaFerramentas + ''';' + #13#10;
    Script := Script + '/*-----------------------------------------------
----------------------*/' + #13#10;
    Script := Script + '//-->' + #13#10;
    Script := Script + '</SCRIPT>' + #13#10;

    Str := InsereTagNoInicioDoHEAD(Arq.Text, Script);
    Arq.Text := Str;
    Arq.SaveToFile(PathGerente);
    Arq.Free();

end;

{-------------------------------------------------------------}
procedure TFRH.InsereVariavelJS(PathHtml, NomeVariavel, Valor : string);
```

```
var Str, Script: string;
    Arq: TStringList;
begin
    Arq := TStringList.Create();
    Arq.LoadFromFile(PathHtml);

    Script := '<SCRIPT LANGUAGE="JavaScript">' + #13#10;
    Script := Script + '<!--' + #13#10;
    Script := Script + '/*------  FRH: Variáveis Gerada Automaticamente ---
----*/' + #13#10;
    Script := Script + 'var ' + NomeVariavel + ' = ' + Valor + ';' +
#13#10;
    Script := Script + '/*-----------------------------------------------
----*/' + #13#10;
    Script := Script + '//-->' + #13#10;
    Script := Script + '</SCRIPT>' + #13#10;

    Str := InsereTagNoInicioDoHEAD(Arq.Text, Script);
    Arq.Text := Str;
    Arq.SaveToFile(PathHtml);
    Arq.Free();
end;

{-------------------------------------------------------------}
function TFRH.InsereTagNoInicioDoHEAD(HtmlStr, Tag: string): string;
var n: integer;
begin
    //procura pelos elementos HEAD e HTML para inserir na posicao correta
    n := AnsiPos('<HEAD>',HtmlStr);
    if n = 0 then   //tenta denovo
       n := AnsiPos('<head>',HtmlStr);

    //adiciona <HEAD>
    if n = 0 then begin
       n := AnsiPos('<HTML>',HtmlStr);
       if n = 0 then
          n := AnsiPos('<html>',HtmlStr);
       Tag := '<HEAD>' + #13#10 + Tag + #13#10'</HEAD>' + #13#10;
       Insert(Tag, HtmlStr, n+7);
    end
    //Ja existe o <HEAD>
```

```
    else begin
      Insert(Tag, HtmlStr, n+7);
    end;
    result := HtmlStr;
end;

{-------------------------------------------------------------}
procedure TFRH.ConstroiSelectOptionsIndice(PathIndice: string);
var Str, PalavraChaveAnterior, Valor, StrArrayTitulo, StrArrayDestino,
StrArrayTipo: string;
    Arq: TStringList;
    n,i: integer;
begin
    Str := '';
    Arq := TStringList.Create();
    Arq.LoadFromFile(PathIndice);
//    n := AnsiPos('    <?frh ConstroiSelectDoIndice(name="lista" size="15"
class="Controles" onDblClick="ExibeConteudo()")?>',textoHTML);
    n := AnsiPos('<?frh ConstroiSelectOptionsIndice ?>', Arq.Text);
    if n <> 0 then begin
       self.ListaInfoNodos.Sort();
       PalavraChaveAnterior := self.ListaInfoNodos.Strings[0];
       StrArrayTitulo := '''' +
TNodoHiper(ListaInfoNodos.Objects[0]).Titulo + '''';
       StrArrayDestino := '''' +
IntToStr(TNodoHiper(ListaInfoNodos.Objects[0]).CodNodo) + '.html''';
       StrArrayTipo := '''' +
TNodoHiper(ListaInfoNodos.Objects[0]).ContextoNodo + '''';
       if self.ListaInfoNodos.Count < 1 then begin
          if self.ListaInfoNodos.Count <> 0 then begin
             Valor := 'new Array(' + StrArrayTitulo + '),new Array(' +
StrArrayDestino + '),new Array(' + StrArrayTipo + ')';
             Str := Str + #13#10 + '            <option value="' + Valor +
'">' + self.ListaInfoNodos.Strings[0] + '</option>';
          end;
       end
       else begin
           if self.ListaInfoNodos.Strings[0] <>
self.ListaInfoNodos.Strings[1] then begin
              Valor := 'new Array(' + StrArrayTitulo + '),new Array(' +
StrArrayDestino + '),new Array(' + StrArrayTipo + ')';
```

```
              Str := Str + #13#10 + '            <option value="' + Valor +
'">' + self.ListaInfoNodos.Strings[0] + '</option>';
           end;
           for i:=1 to self.ListaInfoNodos.Count-1 do begin
               //se é o ultimo elemento deve inserir o OPTION
               if self.ListaInfoNodos.Strings[i] = PalavraChaveAnterior
then begin
                  StrArrayTitulo := StrArrayTitulo + ',''' +
TNodoHiper(ListaInfoNodos.Objects[i]).Titulo + '''';
                  StrArrayDestino := StrArrayDestino + ',''' +
IntToStr(TNodoHiper(ListaInfoNodos.Objects[i]).CodNodo) + '.html''';
                  StrArrayTipo := StrArrayTipo + ',''' +
TNodoHiper(ListaInfoNodos.Objects[i]).ContextoNodo + '''';
                  //se a proxima for <> da anterior entao insere mais um
OPTION
                  if i = self.ListaInfoNodos.Count then begin
                     Valor := 'new Array(' + StrArrayTitulo + '),new
Array(' + StrArrayDestino + '),new Array(' + StrArrayTipo + ')';
                     Str := Str + #13#10 + '            <option value="' +
Valor + '">' + self.ListaInfoNodos.Strings[i] + '</option>';
                  end
                  else begin
                      if self.ListaInfoNodos.Strings[i] <>
self.ListaInfoNodos.Strings[i+1] then begin
                         Valor := 'new Array(' + StrArrayTitulo + '),new
Array(' + StrArrayDestino + '),new Array(' + StrArrayTipo + ')';
                         Str := Str + #13#10 + '            <option
value="' + Valor + '">' + self.ListaInfoNodos.Strings[i] + '</option>';
                      end;
                  end;
               end
               else begin
                  StrArrayTitulo := '''' +
TNodoHiper(ListaInfoNodos.Objects[i]).Titulo + '''';
                  StrArrayDestino := '''' +
IntToStr(TNodoHiper(ListaInfoNodos.Objects[i]).CodNodo) + '.html''';
                  StrArrayTipo := '''' +
TNodoHiper(ListaInfoNodos.Objects[i]).ContextoNodo + '''';

                  Valor := 'new Array(' + StrArrayTitulo + '),new Array(' +
StrArrayDestino + '),new Array(' + StrArrayTipo + ')';
```

```
                Str := Str + #13#10 + '            <option value="' +
Valor + '">' + self.ListaInfoNodos.Strings[i] + '</option>';
                PalavraChaveAnterior := self.ListaInfoNodos.Strings[i];
              end;
           end;
       end;
       Str := AnsiReplaceText(Arq.Text, '<?frh ConstroiSelectOptionsIndice
?>', Str);
       Arq.Text := Str;
       Arq.SaveToFile(PathIndice);
       Arq.Free();
    end;
end;

{-------------------------------------------------------------}

end.
```

unit uFRH_XML;

```
interface

uses
  SysUtils, Classes, xmldom, XMLIntf, msxmldom, XMLDoc, XSLProd, OleServer,
  MSXML2_TLB;

type
  TComponentesXML = class(TDataModule)
    DomDocXML: TDOMDocument;
    DomDocXSL: TDOMDocument;
    DocXML: TXMLDocument;
    DOMDocFull: TDOMDocument;
    DOMDocXHTML: TDOMDocument;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
```

```
  ComponentesXML: TComponentesXML;

implementation

{$R *.dfm}

end.
```

unit uFRHArquivos;

```
interface

uses Classes,Sysutils, FileCtrl, uFRHInterfaceWin;

type
  TInterfaceArquivos = class
  private
  public
     procedure CopiaArq(NomeOriginal, NomeCopia: string;
Sobrescrever:boolean);
     procedure CopiaDiretorio(NomeOriginal, NomeCopia: string;
Sobrescrever:boolean);
     procedure CriaDiretorios(Path: string);
  end;

{------------------------------}

implementation

uses
   StrUtils;

{-------------------------------------------------------------}
procedure TInterfaceArquivos.CopiaArq(NomeOriginal, NomeCopia: string;
Sobrescrever:boolean);
begin
    NomeOriginal := AnsiReplaceText(NomeOriginal, '/', '\');
    NomeCopia := AnsiReplaceText(NomeCopia, '/', '\');
    winCopyFile(Pchar(NomeOriginal),Pchar(NomeCopia),not Sobrescrever);
end;
```

```
{-------------------------------------------------------------}
procedure TInterfaceArquivos.CopiaDiretorio(NomeOriginal, NomeCopia:
string; Sobrescrever:boolean);
var  I: Integer;
     SearchRec: TSearchRec;
     NovoDir: string;
begin

    // passa pelos arquivos.
    I := FindFirst(NomeOriginal + '\*', faArchive, SearchRec);
    while I = 0 do begin
        self.CopiaArq(NomeOriginal + SearchRec.Name, NomeCopia +
SearchRec.Name, True);
        I := FindNext(SearchRec);
    end;
    FindClose(SearchRec);

    // passa pelos diretórios.
    I := FindFirst(NomeOriginal + '\*', faDirectory, SearchRec);
    while I = 0 do begin
       if SearchRec.Attr = faDirectory then begin
          if (SearchRec.Name <> '.') and (SearchRec.Name <> '..') then
begin
              NovoDir := NomeCopia + SearchRec.Name + '\';
              self.CriaDiretorios(NovoDir);
              self.CopiaDiretorio(NomeOriginal + SearchRec.Name + '\',
NovoDir, Sobrescrever);
          end;
       end;
       I := FindNext(SearchRec);
    end;
    FindClose(SearchRec);

end;

{-------------------------------------------------------------}
procedure TInterfaceArquivos.CriaDiretorios(Path: string);
begin
    if path <> '' then begin
       Path := AnsiReplaceText(Path, '/', '\');
```

```
       if not ForceDirectories(Path) then
          raise Exception.Create('Não foi possível criar o diretório ' +
Path);
    end;
end;

{-------------------------------------------------------------}

end.
```

II. Documentos HTML dos Templates FRH

index.html

```
<html>
<head>
<title>FRH - Framework para Relatórios Hipermídia</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<!----------------------------------------------------------------------->
<!--    FRH - ATENÇÃO: O FRAME GERENTE DEVE ESTAR SOZINHO NO PRIMEIRO
        FRAMESET                                                       -->
<!----------------------------------------------------------------------->
<frameset rows="1,1*" framespacing="0" frameborder="NO" border="0">
  <frame src="frame_gerente.html" name="gerente" scrolling="NO" noresize >

<!------------   INÍCIO DOS FRAMES DO DESENVOLVEDOR    ---------------->

  <frameset rows="0,34,1*" frameborder="NO" border="0" framespacing="0">
    <frame src="busca/vazio.html" name="busca" scrolling="NO" noresize
frameborder="NO" marginwidth="0" marginheight="0">
    <frame src="frame_ferramentas.html" name="ferramentas" scrolling="NO"
frameborder="NO" marginwidth="0" marginheight="0">
    <frameset cols="220,1*" framespacing="4" frameborder="YES" border="4"
bordercolor="#CCCCCC">
      <frame src="fluxograma.html" name="frame_esquerda" frameborder="YES"
bordercolor="#CCCCCC">
      <frame src="abertura.html" name="unidade_conceito" frameborder="YES">
     </frameset>
   </frameset>
<!--------------------   FIM DOS FRAMES DO DESENVOLVEDOR    ---------------
---------->
</frameset>
<noframes><body>
</body></noframes>
</html>
```

frame_gerente.html

```
<HTML>
```

```
<HEAD>
<SCRIPT LANGUAGE="JavaScript" SRC="scripts/constantes.js"></SCRIPT>
<SCRIPT LANGUAGE="JavaScript">
<!--
/*-------------------------------------------------------------------------*/
/* FRH - Framework para Relatórios Hipermidia (variaveis e funções do
Gerente) */
/*-------------------------------------------------------------------------*/

/*-------------------------------------------------*/
/* CÓDIGO PARA CORRETA INICIALIZAÇÃO DAS VARIAVEIS  */
/*-------------------------------------------------*/

var InicialGerenteParaUnidadeConceito;
var InicialGerenteParaGlossarioNodos;
var InicialGerenteParaFluxograma;
var InicialGerenteParaBusca;
var InicialGerenteParaIndice;
var InicialGerenteParaFerramentas;

var GerenteParaUnidadeConceito;
var GerenteParaGlossarioNodos;
var GerenteParaFluxograma;
var GerenteParaBusca;
var GerenteParaIndice;
var GerenteParaFerramentas;

//as janelas que existiram irao atualizar estas variaveis
var janela_unidade_conceito = true;  //Fixo!!
var janela_glossario_nodos = false;
var janela_ferramentas = true;  //fixo!
var janela_fluxograma = false;
var janela_busca = false;
var janela_indice = false;


function EsperaVariavelUnidadeConceito(){
   if  (typeof(eval(VInicialGerenteParaUnidadeConceito)) == 'undefined'){
        window.setTimeout('EsperaVariavelUnidadeConceito()',800);
   }else{
```

```
       GerenteParaUnidadeConceito =
eval(VInicialGerenteParaUnidadeConceito);
       InicialGerenteParaUnidadeConceito = GerenteParaUnidadeConceito;
   }
}

function EsperaVariavelFluxograma(){
   if  (typeof(eval(VInicialGerenteParaFluxograma)) == 'undefined'){
        window.setTimeout('EsperaVariavelFluxograma()',1000);
   }else{
       GerenteParaFluxograma = eval(VInicialGerenteParaFluxograma);
       InicialGerenteParaFluxograma = GerenteParaFluxograma;
   }
}

function EsperaVariavelIndice(){
   if  (typeof(eval(VInicialGerenteParaIndice)) == 'undefined'){
        window.setTimeout('EsperaVariavelIndice()',1000);
   }else{
       GerenteParaIndice = eval(VInicialGerenteParaIndice);
       InicialGerenteParaIndice = GerenteParaIndice;
   }
}

function EsperaVariavelGlossarioNodos(){
   if  (typeof(eval(VInicialGerenteParaGlossarioNodos)) == 'undefined'){
        window.setTimeout('EsperaVariavelGlossarioNodos()',1000);
   }else{
       GerenteParaGlossarioNodos = eval(VInicialGlossarioNodos);
       InicialGerenteParaGlossarioNodos = GerenteParaGlossario;
   }
}
//-------------------------------------------------------------------------
----------

//UNIDADE CONCEITO
var GerenteParaUnidadeConceito;
if  (typeof(eval(VInicialGerenteParaUnidadeConceito)) == 'undefined'){
    window.setTimeout('EsperaVariavelUnidadeConceito()',500);
}else{
   GerenteParaUnidadeConceito = eval(VInicialGerenteParaUnidadeConceito);
```

```
   InicialGerenteParaUnidadeConceito = GerenteParaUnidadeConceito;
}

//GLOSSARIO NODOS (DIFERENTE DOS OUTROS FIXO!!!)
var GerenteParaGlossarioNodos;
if  (typeof(eval(VInicialGerenteParaGlossarioNodos)) == 'undefined'){
    window.setTimeout('EsperaVariavelGlossarioNodos()',500);
}else{
   GerenteParaGlossarioNodos = eval(VInicialGerenteParaGlossarioNodos);
   InicialGerenteParaGlossarioNodos = GerenteParaGlossarioNodos;
}

//FLUXOGRAMA
var GerenteParaFluxograma;
if  (typeof(eval(VInicialGerenteParaFluxograma)) == 'undefined'){
    window.setTimeout('EsperaVariavelFluxograma()',500);
}else{
   GerenteParaFluxograma = eval(VInicialGerenteParaFluxograma);
   InicialGerenteParaFluxograma = GerenteParaFluxograma;
}

//INDICE
var GerenteParaIndice;
if  (typeof(eval(VInicialGerenteParaIndice)) == 'undefined'){
    window.setTimeout('EsperaVariavelIndice()',500);
alert('COMO I');
}else{
   GerenteParaIndice = eval(VInicialGerenteParaIndice);
   InicialGerenteParaIndice = GerenteParaIndice;
}

//BUSCA (diferente dos outros. busca abre em um applet.
GerenteParaBuscaDeve ser um frame de index.html)
var GerenteParaBusca =  eval(VInicialGerenteParaBusca);
InicialGerenteParaBusca = GerenteParaBusca;

//FERRAMENTAS (diferente dos outros. FIXO!!)
var GerenteParaFerramentas =  eval(VInicialGerenteParaFerramentas);
InicialGerenteParaFerramentas = GerenteParaFerramentas;

/*------------------------------------------------------ */
```

```
//Janelas de conteiners. Não são permitidos conteiners para os frames de
unidade de conceito e ferramentas
var janela_conteiner_glossario_nodos = null;
var janela_conteiner_fluxograma = null;
var janela_conteiner_indice = null;
var janela_conteiner_busca = null;


var gerente=self;
var termo_glossario="";
var janela_links = null;


/*-------------         FIM DAS INICIALIZAÇÕES       ------------------*/
/*---------------------------------------------------------------------*/

function AtualizaFluxograma(html){
   if (janela_fluxograma)
      GerenteParaFluxograma.AtualizaFluxograma(html);
}

/*---------------------------------------------------------------------*/
function AbreFluxograma(dir_raiz){
   if(NomeConteinerFluxograma != ""){
      if(janela_conteiner_fluxograma != null){
         if(janela_fluxograma){
            janela_conteiner_fluxograma.focus();
         }else{
            var caminho_todo = dir_raiz + DirFluxograma + NomeFluxograma;
            eval('janela_conteiner_fluxograma.'+ ConteinerParaFluxograma +
'.location="' + caminho_todo + '"');
            janela_fluxograma = true;
    //focus
         }
      }else{
         var caminho_todo = dir_raiz + DirFluxograma +
NomeConteinerFluxograma;
         janela_conteiner_fluxograma = window.open (caminho_todo,
"conteiner_fluxograma", OpcoesConteinerFluxograma);
```

```
         //verifica se o fluxograma é a primeira janela a abrir no frame
"ConteinerParaFluxograma". Espera dar o LOAD primeiro COMO??? Só funciona
se a janela abre rápido o suficiente. Se não der a janela é aberta
novamente no else
         if (janela_fluxograma){
            GerenteParaFluxograma.focus();
            janela_fluxograma = true;
         }else{
            caminho_todo = dir_raiz + DirFluxograma + NomeFluxograma;
            eval('janela_conteiner_fluxograma.'+ ConteinerParaFluxograma +
'.location="' + caminho_todo + '"');
            janela_fluxograma = true;
    //focus
         }
      }
   }else{
      if(janela_fluxograma){
         GerenteParaFluxograma.focus();
      }else{
         var caminho_todo = dir_raiz + DirFluxograma + NomeFluxograma;
         if (InicialGerenteParaFluxograma != null){
            //É um frame
            InicialGerenteParaFluxograma.location = caminho_todo;
            GerenteParaFluxograma = InicialGerenteParaFluxograma;
            janela_fluxograma = true;
         }else{
            //É uma janela nova com um único frame
            GerenteParaFluxograma = window.open (caminho_todo,
"fluxograma",
"toolbar=0,location=0,directories=0,status=0,scrollbars=yes,menubar=0,resiz
able=1,width=400,height=350");
            janela_fluxograma = true;
         }
      }
   }
}

/*-------------------------------------------------------------------*/
function AbreIndice(dir_raiz){
   if(NomeConteinerIndice != ""){
      if(janela_conteiner_indice != null){
```

```
         if(janela_indice){
            GerenteParaIndice.focus();
         }else{
            var caminho_todo = dir_raiz + DirIndice + NomeIndice;
            eval('janela_conteiner_indice.'+ ConteinerParaindice +
'.location="' + caminho_todo + '"');
            janela_indice = true;
         }
      }else{
         var caminho_todo = dir_raiz + DirIndice + NomeConteinerIndice;
         janela_conteiner_indice = window.open (caminho_todo,
ConteinerParaIndice, OpcoesConteinerIndice);

         //verifica se o Indice é a primeira janela a abrir no frame
"ConteinerParaIndice". Espera dar o LOAD primeiro COMO??? Só funciona se a
janela abre rápido o suficiente. Se não der a janela é aberta novamente no
else
         if (janela_indice){
            GerenteParaIndice.focus();
            janela_indice = true;
         }else{
            caminho_todo = dir_raiz + DirIndice + NomeIndice;
            eval('janela_conteiner_indice.'+ ConteinerParaindice +
'.location="' + caminho_todo + '"');
            janela_indice = true;
         }
      }
   }else{
      if(janela_indice){
         GerenteParaIndice.focus();
      }else{
         var caminho_todo = dir_raiz + DirIndice + NomeIndice;
         if (InicialGerenteParaIndice != null){
            //É um frame
            InicialGerenteParaIndice.location = caminho_todo;
            GerenteParaIndice = InicialGerenteParaIndice;
            janela_indice = true;
         }else{
            //É uma janela nova com um único frame
```

```
            GerenteParaIndice = window.open (caminho_todo, "indice",
"toolbar=0,location=0,directories=0,status=0,scrollbars=yes,menubar=0,resiz
able=1,width=390,height=380");
            janela_indice = true;
         }
      }
   }
}

/*-------------------------------------------------------------------*/
function AbreBusca(dir_raiz) {
   if (parseInt(navigator.appVersion) < 4){
      window.open("busca/semBusca.html",'');
   }else{
      gerente.top.busca.location = "busca/busca.html";
   }
}

/*-------------------------------------------------------------------*/
function AbreGlossario(termo_html, dir_raiz){
   termo_glossario = termo_html;
   if(janela_glossario_nodos == true){
      janela_conteiner_glossario_nodos.focus();
      if (termo_html != ''){
         var caminho_todo = dir_raiz + DirNodosGlossario + termo_html;
         eval('janela_conteiner_glossario_nodos.'+
ConteinerParaGlossarioNodos + '.location="' + caminho_todo + '"');
      }
   }else{
      var caminho_todo = dir_raiz + DirConteinerGlossarioNodos +
NomeConteinerGlossarioNodos;
//      gerente.janela_conteiner_glossario_nodos = window.open
(caminho_todo, "conteiner_glossario_nodos", "","" );
      gerente.janela_conteiner_glossario_nodos = window.open (caminho_todo,
"conteiner_glossario_nodos", OpcoesConteinerGlossarioNodos );
   }
}

/*-------------------------------------------------------------------*/
function AbreJanelaUnidadeConceito() {
```

```
}

/*-----------------------------------------------------------------*/
function NodoSeguinte() {
  if (janela_unidade_conceito){
     if (GerenteParaUnidadeConceito.FRH_nodo_seguinte != ""){
        GerenteParaUnidadeConceito.location = DirUnidadeConceito +
GerenteParaUnidadeConceito.FRH_nodo_seguinte;
        if (janela_fluxograma){

AtualizaFluxograma(GerenteParaUnidadeConceito.FRH_nodo_seguinte);
        }
     }
  }
}

/*-----------------------------------------------------------------*/
function NodoAnterior() {
  if (janela_unidade_conceito){
     if (GerenteParaUnidadeConceito.FRH_nodo_anterior != ""){
        GerenteParaUnidadeConceito.location = DirUnidadeConceito +
GerenteParaUnidadeConceito.FRH_nodo_anterior;
        if (janela_fluxograma){

AtualizaFluxograma(GerenteParaUnidadeConceito.FRH_nodo_anterior);
        }
     }
  }
}

/*-----------------------------------------------------------------*/
function NodoPai() {
  if (janela_unidade_conceito){
     if (GerenteParaUnidadeConceito.FRH_nodo_pai != ""){
        GerenteParaUnidadeConceito.location = DirUnidadeConceito +
GerenteParaUnidadeConceito.FRH_nodo_pai;
        if (janela_fluxograma){
           AtualizaFluxograma(GerenteParaUnidadeConceito.FRH_nodo_pai);
        }
     }
  }
```

```
}

/*---------------------------------------------------------------------*/
function AbreConteudoURL(html){

   alert("USANDO FUNCAO QUE NAO EH PRA USAR!!!");
   gerente.janela_unidade_conceito.location = html;
}

/*---------------------------------------------------------------------*/
function SegueLinkSimples(html, tipo_elemento, dir_raiz){
   switch (tipo_elemento) {
      case 'unidade_conceito': {
         GerenteParaUnidadeConceito.location = dir_raiz +
DirUnidadeConceito + html;
         AtualizaFluxograma(html);
         break;
      }
      case 'glossario': {
         AbreGlossario(html, dir_raiz);
         break;
      }
   }
}

/*---------------------------------------------------------------------*/
function SegueLink(titulos, destinos, tipos, janela_origem, dir_raiz){
   if (destinos.length == 1){
      gerente.SegueLinkSimples(destinos[0], tipos[0], dir_raiz);
   }else{
      if(gerente.janela_links != null){
         gerente.janela_links.focus();
      }else{
         gerente.janela_links = window.open ("", "links",
"toolbar=0,location=0,directories=0,status=1,scrollbars=yes,menubar=0,resiz
able=1,width=360,height=250");
         var str = '<HTML>';
         str += '<TITLE>Links Disponíveis</TITLE>' + '\n';
         str += '<BODY OnUnLoad="self.opener.gerente.janela_links=null;"
bgcolor="#CCCCCC">' + '\n';
```

```
         str += '   <font size="2" face="Arial, Helvetica, sans-
serif"><ul>' + '\n';
         for(i=0; i<=destinos.length-1; i++){
            str += '     <br><li><a href="Javascript:
self.opener.gerente.SegueLinkSimples(' + "'" + destinos[i] + "','" +
tipos[i] + "','" + dir_raiz +"');" + '">' + titulos[i] + '</a></li>' +
'\n';
         }
         str += '   </ul><br><br>' + '\n';

         // apenas bota a volta para origem para unidade de conceito e
glossario.
         if (typeof(janela_origem) != 'undefined') {
            if ((janela_origem == 'glossario')||(janela_origem ==
'unidade_conceito')) {
               str += '   <center><a href="Javascript:
self.opener.gerente.SegueLinkSimples(' + "'" +
janela_origem.FRH_nodo_corrente + "','" + janela_origem.FRH_tipo_nodo +
"','" + dir_raiz + "');" + '">Voltar à origem do link</a></center>' + '\n';
            }
         }
         str += '</font></BODY></HTML>';
         janela_links.document.write(str);
         janela_links.document.close();
      }
   }
}

/*--------------------------------------------------------------------*/
/*--------------------------------------------------------------------*/
/*--------------------------------------------------------------------*/
//-->
</SCRIPT>
</HEAD>
<BODY></BODY>
</html>
```

frame_ferramentas.html

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<SCRIPT LANGUAGE="JavaScript">
<!--
/*-------------------------------------------------------------------------*/
/*     FRH - Framework para Relatórios Hipermidia (ferramentas)        */
/*-------------------------------------------------------------------------*/


//INICIALIZAÇÕES CRÍTICAS: verificar se o gerente ja foi carregado...
var gerente;

function EsperaVariavelGerente(){
   if (typeof(eval(VInicialGerente)) == 'undefined'){
      window.setTimeout('EsperaVariavelGerente()',300);
   }else{
      gerente = eval(VInicialGerente);
   }
}
EsperaVariavelGerente();

//FIM INICIALIZAÇÕES CRÍTICAS


function Fluxograma() {
   gerente.AbreFluxograma(FRH_dir_raiz);
}

/*-------------------------------------------------------------------------*/
function Vai(){
   gerente.GerenteParaUnidadeConceito.history.go(+1);
}

/*-------------------------------------------------------------------------*/
function Volta(){
   gerente.GerenteParaUnidadeConceito.history.go(-1);
}
```

```
/*-----------------------------------------------------------------------*/
function Glossario() {
   gerente.AbreGlossario('',FRH_dir_raiz);
}

/*-----------------------------------------------------------------------*/

function Busca() {
   gerente.AbreBusca(FRH_dir_raiz);
}

/*-----------------------------------------------------------------------*/

function Indice() {
   gerente.AbreIndice(FRH_dir_raiz);
}

/*-----------------------------------------------------------------------*/
function NodoSeguinte() {
   gerente.NodoSeguinte(FRH_dir_raiz);
}

/*-----------------------------------------------------------------------*/
function NodoAnterior() {
   gerente.NodoAnterior(FRH_dir_raiz);
}

/*-----------------------------------------------------------------------*/
function NodoPai() {
   gerente.NodoPai(FRH_dir_raiz);
}

/*-----------------------------------------------------------------------*/
/*-----------------------------------------------------------------------*/
/*-----------------------------------------------------------------------*/
//-->
</SCRIPT>
<style type="text/css">
<!--
.BotaoFluxogramaEIndice {  font-family: Arial, Helvetica, sans-serif; font-
size: 9pt; font-weight: bold; color: #000000; background-color: #CDD1E0}
```

```
.BotaoGlossario {  font-family: Arial, Helvetica, sans-serif; font-size:
9pt; font-weight: bold; color: #000000; background-color: #BFDDC6}
.BotoesNavegacao {  font-family: Arial, Helvetica, sans-serif; font-size:
9pt; font-weight: bold; background-color: #D9D8BD}
.BotoesHistoria {  font-family: Arial, Helvetica, sans-serif; font-size:
15pt; background-color: #B9B9B9; font-weight: normal; border-color: #FFFFFF
#000000 #000000 #FFFFFF; line-height: 11pt; height: 23px; width: 42px}
-->
</style>
</head>

<body bgcolor="#CCCCCC" leftmargin="0" topmargin="0" marginwidth="0"
marginheight="0">
<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr valign="middle">
    <td nowrap height="34" width="36%">  
      <input type="button" name="fluxograma" value="Fluxograma"
onClick="Fluxograma()" class="BotaoFluxogramaEIndice">
      <input type="button" name="indice" value="&Iacute;ndice"
onClick="Indice()" class="BotaoFluxogramaEIndice">
      <input type="button" name="busca" value="Busca" onClick="Busca()"
class="BotaoFluxogramaEIndice">
      <input type="button" name="glossario" value="Gloss&aacute;rio"
onClick="Glossario()" class="BotaoGlossario">
    </td>
    <td nowrap height="34" width="50%" align="center" valign="middle">
      <table width="199" border="0" cellspacing="0" cellpadding="0">
        <tr>
          <td align="center">
            <input type="button" name="hvolta" value="&lt;"
onClick="Volta()" class="BotoesHistoria">
            <input type="button" name="hvai" value="&gt;" onClick="Vai()"
class="BotoesHistoria">
          </td>
        </tr>
      </table>

    </td>
    <td nowrap height="34" width="14%" align="right">
      <input type="button" name="Anterior" value="Anterior "
onClick="NodoAnterior()" class="BotoesNavegacao">
```

```
      <input type="button" name="acima" value=" Acima " onClick="NodoPai()"
class="BotoesNavegacao">
      <input type="button" name="proximo" value="Pr&oacute;ximo"
onClick="NodoSeguinte()" class="BotoesNavegacao">
        </td>
  </tr>
</table>
</body>
</html>
```

fluxograma.html

```
<html>
<head>
<TITLE>Fluxograma</TITLE>
<!------ CAMINHO PARA OS SCRIPS DO COMPONENTE USADO NO FLUXOGRAMA ------>
<script language="JavaScript"
src="scripts/fluxograma_tree_nodes.js"></script>
<script language="JavaScript"
src="scripts/fluxograma_tree_format.js"></script>
<script language="JavaScript" src="scripts/fluxograma_nostree.js"></script>
<!-------------------------------------------------------------------->
<script language="JavaScript">
/*--------------------------------------------------------------------*/
/*   FRH - Framework para Relatórios Hipermidia (Modulo Fluxograma)    */
/*--------------------------------------------------------------------*/

//INICIALIZAÇÕES CRÍTICAS: verificar se o gerente ja foi carregado...
var gerente;

function EsperaVariavelGerente(){
   if (typeof(eval(VInicialGerente)) == 'undefined'){
      window.setTimeout('EsperaVariavelGerente()',300);
   }else{
      gerente = eval(VInicialGerente);
   }
}
EsperaVariavelGerente();

//FIM INICIALIZAÇÕES CRÍTICAS
```

```
var html_nodo = "";
var array_indice_nodo = new Array();
var indice_atual;

/*-----------------------------------------------------------------------*/
function Inicializa(){
   if  (typeof(gerente) == 'undefined'){
       window.setTimeout('EsperaVariavelGerente()',100);
       window.setTimeout('Inicializa()',300);
   }else{
      gerente.janela_fluxograma = true;
      if (typeof(gerente.GerenteParaUnidadeConceito) != 'undefined') {
         if (gerente.GerenteParaUnidadeConceito.FRH_nodo_corrente != '') {

AtualizaFluxograma(gerente.GerenteParaUnidadeConceito.FRH_nodo_corrente);
         }
      }else{
         window.setTimeout('Inicializa()',300);
      }
   }
}

/*-----------------------------------------------------------------------*/
function Finaliza(){
   gerente.janela_fluxograma = false;
   gerente.GerenteParaFluxograma = null;
}

/*-----------------------------------------------------------------------*/

function AtualizaFluxograma(html_str) {
var i;
   html_nodo = html_str;
   indice_atual = -1;

   for (i = 0; i < TREE_NODES.length; i++){
       if (LoopProcuraNodo(TREE_NODES[i]))
          break;
   }
```

```
   for (i = 0; i < array_indice_nodo.length; i++){
       NTrees[treeName].Nodes[array_indice_nodo[i]].expanded = true;
   }
   array_indice_nodo.length = 0;    //destroi o array
   NTrees[treeName].draw();
}

/*-------------------------------------------------------------------------*/
function LoopProcuraNodo(array_nodos) {
var i, indice_local;
   indice_atual++;
   indice_local = indice_atual;
   if ( ProcuraString(array_nodos[1]) ){
      array_indice_nodo[array_indice_nodo.length] = indice_atual;
      return(true);
   }
   for (i = 3; i < array_nodos.length; i++){
       if ( LoopProcuraNodo(array_nodos[i]) ) {
          array_indice_nodo[array_indice_nodo.length] = indice_local;
          return(true);
       }
   }
   return(false);
}

/*-------------------------------------------------------------------------*/
function ProcuraString(texto){
   if (texto == null){
      return(false);
   }else{
      if (texto.indexOf(html_nodo) == -1)
         return(false);
      else{
         return(true);
      }
   }
}
/*-------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------*/
```

```
</script>
<!------------------------------------------------------------------------>
<!                                CSS                                     >
<!------------------------------------------------------------------------>
<style type="text/css">
<!--
BODY, TD, TH, A, P,H1,H2,H3{
      font-family: Arial, Helvetica, sans-serif;
      font-size: 10px;
}

.clsNode:hover{
      color: blue;
      font-weight : bold;;
      font-size : 8pt;
      text-decoration : underline;
}

.clsNode:active{
        color: #CC0000;
      font-weight : bold;;
      font-size : 8pt;
}

.clsNode {  color: #000000;
      font-weight : bold;;
      font-size : 8pt;
      text-decoration : none;
 } -->
</style>
<!------------------------------------------------------------------------>
<!------------------------------------------------------------------------>
<!------------------------------------------------------------------------>
</head>
<body OnLoad="Inicializa()" OnUnLoad="Finaliza()" bgcolor="#EAECF2">
<script language="JavaScript">
/*------------------------------------------------------------------------*/
/*                 SCRIPT PARA DESENHAR O FLUXOGRAMA                      */
/* This script featured on JavaScript Kit (http://www.javascriptkit.com) */
/*------------------------------------------------------------------------*/
```

```
var treeName = "Tree";
new NosTree (treeName, TREE_NODES, TREE_FORMAT);

//UNCOMMENT 2 lines below to expand first node of tree and redraw it
//NTrees[treeName].Nodes[0].expanded = true;
//NTrees[treeName].draw();

/*--------------------------------------------------------------------------*/
/*--------------------------------------------------------------------------*/
/*--------------------------------------------------------------------------*/
</script>
</body>
</html>
```

indice.html

```
<html>
<head>
<title>Índice</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<SCRIPT LANGUAGE="JavaScript">
<!--
/*--------------------------------------------------------------------------*/
/*  FRH - Framework para Relatórios Hipermidia (Localização por índice) */
/*--------------------------------------------------------------------------*/


//INICIALIZAÇÕES CRÍTICAS: verificar se o gerente ja foi carregado...
var gerente;

function EsperaVariavelGerente(){
   if (typeof(eval(VInicialGerente)) == 'undefined'){
      window.setTimeout('EsperaVariavelGerente()',300);
   }else{
      gerente = eval(VInicialGerente);
   }
}
EsperaVariavelGerente();

//FIM INICIALIZAÇÕES CRÍTICAS
```

```
function Inicializa() {
   if (typeof(eval(VInicialGerente)) == 'undefined'){
      window.setTimeout('EsperaVariavelGerente()',100);
      window.setTimeout('Inicializa()',300);
   }else{
      gerente.janela_indice = true;
   }
}

/*-------------------------------------------------------------------------*/
function Finaliza() {
   gerente.janela_indice = false;
   gerente.GerenteParaIndice = null;
}

/*-------------------------------------------------------------------------*/
function LocalizaPalavraChaveRapido(palavra){
var achou, nao_existe = false;
var indice;

    indice_atual = round(lista.length/2);
    while ( (!achou)&&(!nao_existe) ) {
        if (lista.options[indice_atual].text.toLowerCase() ==
palavra.toLowerCase( ) ) {
           achou = true;
        }else{
           indice_atual = round(lista.length/2);
        }
    }
}

/*-------------------------------------------------------------------------*/
function LocalizaPalavraChave(palavra){
var i, indiceMatch;
    for(i=0; i<=lista.length-1; i++){
        indiceMatch =
lista.options[i].text.toLowerCase().indexOf(palavra.toLowerCase());
        if (indiceMatch == 0 ) {
```

```
            lista.selectedIndex = i;
            break;
        }
    }
}

/*-------------------------------------------------------------------------*/
function ExibeConteudo() {
   var tres_primeiros_parametros =
lista.options[lista.selectedIndex].value;
//alert('gerente.SegueLink(' + tres_primeiros_parametros + ', self,
gerente.DirIndice');
   eval('gerente.SegueLink(' + tres_primeiros_parametros + ', self,
gerente.DirIndice)');
}

/*-------------------------------------------------------------------------*/
function AtualizaTextBox() {
   expressao.value = lista.options[lista.selectedIndex].text;

}

/*-------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------*/
//-->
</SCRIPT>
<!--------------------------------------------------------------------------->
<!                              CSS                                        -->
<!--------------------------------------------------------------------------->
<style type="text/css">
<!--
.Controles {  width: 100%; margin-top: 5px; margin-bottom: 5px; font-size:
8pt}
.Body {  font-family: Arial, Helvetica, sans-serif; font-size: 10pt;
background-color: #CCCCCC}
.BotaoExibir {  font-family: Verdana, Arial, Helvetica, sans-serif; font-
size: 9pt; margin-top: 5px; margin-right: 0px; margin-bottom: 0px; margin-
left: 0px}
.Tabela {  font-family: Arial, Helvetica, sans-serif; font-size: 9pt;
color: #000000}
```

```
-->
</style>
<!------------------------------------------------------------------------>
<!------------------------------------------------------------------------>
<!------------------------------------------------------------------------>
</head>

<body OnLoad="Inicializa()" OnUnLoad="Finaliza()" class="Body">
<table border="0" cellspacing="0" cellpadding="2" class="Tabela"
width="98%" align="center">
  <tr>
    <td> Digite a palavra-chave que deseja localizar:<br>
      <input OnKeyUp="LocalizaPalavraChave(expressao.value);" type="text"
name="expressao" size="60" class="Controles">
    </td>
  </tr>
  <tr>
    <td>
      <select name="lista" size="19" class="Controles"
onDblClick="ExibeConteudo()" onClick="AtualizaTextBox();">
        <!------------------------------------------------------------->
        <!-- A seguinte instrução de processamento do FRH  irá construir
             as opções do <SELECT> com os atributos especificados no
             parâmetro. Este objeto irá conter a lista de palavras do
             índice.                                                  -->
        <!------------------------------------------------------------->

        <?frh ConstroiSelectOptionsIndice ?>

        <!------------------------------------------------------------->
        <!--                        FIM OPTIONS                      -->
        <!------------------------------------------------------------->
      </select>
    </td>
  </tr>
  <tr align="right">
    <td>
      <input type="submit" name="exibir" value="Exibir" class="BotaoExibir"
onClick="ExibeConteudo()">
    </td>
  </tr>
```

```
</table>
<div align="center"></div>
</body>
</html>
```

glossario.html

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title> </title>
<SCRIPT LANGUAGE="JavaScript">
<!--
/*-------------------------------------------------------------------------*/
/*   FRH - Framework para Relatórios Hipermidia (Módulo GLOSSÁRIO)       */
/*-------------------------------------------------------------------------*/

//INICIALIZAÇÕES CRÍTICAS: verificar se o gerente ja foi carregado...
var gerente;

function EsperaVariavelGerente(){
   if (typeof(eval(VInicialGerente)) == 'undefined'){
      window.setTimeout('EsperaVariavelGerente()',300);
   }else{
      gerente = eval(VInicialGerente);
   }
}
EsperaVariavelGerente();

//FIM INICIALIZAÇÕES CRÍTICAS

function Inicializa(){
   if (typeof(eval(VInicialGerente)) == 'undefined'){
      window.setTimeout('EsperaVariavelGerente()',100);
      window.setTimeout('Inicializa()',300);
   }else{
      gerente.janela_conteiner_glossario_nodos = self;
      gerente.janela_glossario_nodos = true;
      AbreTermo();
   }
```

```
}

/*-------------------------------------------------------------------------*/
function Finaliza(){
   gerente.janela_conteiner_glossario_nodos = null;
   gerente.janela_glossario_nodos = false;
}

/*-------------------------------------------------------------------------*/
function AbreTermo(){
   if (gerente.termo_glossario != ""){
      self.glossario_nodos.location = gerente.DirNodosGlossario +
gerente.termo_glossario;
      gerente.termo_glossario = '';
   }
}

/*-------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------*/
-->
</SCRIPT>

</head>

<!--------------------------------------------------------------------------->
<!-- FRH: Este é um conteiner para o frame que apresentará os NODOS DO
GLOSSÁRIO -->
<!--------------------------------------------------------------------------->
<frameset OnLoad="Inicializa()" OnUnLoad="Finaliza()" rows="52,354*"
framespacing="2" frameborder="YES" border="2" bordercolor="#666666"
cols="*">
  <frame src="glossario_ferramentas.html" name="glossario_ferramentas"
scrolling="NO" noresize marginwidth="10" marginheight="6" frameborder="NO"
>
  <frame src="glossario_indice.html" name="glossario_nodos"
frameborder="NO" >
</frameset>
<noframes><body>
</body></noframes>
</html>
```

glossario_ferramentas.html

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!------------------------------------------------------------------------>
<!--  FRH - Framework para Relatórios Hipermidia (Módulo Ferramentas
Glossario)-->
<!------------------------------------------------------------------------>
<SCRIPT LANGUAGE="JavaScript">
<!--

//INICIALIZAÇÕES CRÍTICAS: verificar se o frame GlossarioNodos e o gerente
ja foram carregados...
//                          inicializar as variaveis do gerente e do
glossarioNodos.
var VInicialGerente = 'top.gerente';
var VInicialGlossarioNodos = 'top.glossario_nodos';

var gerente;
var GlossarioNodos;

function EsperaVariavelGlossarioNodos(){
   if (typeof(eval(VInicialGlossarioNodos)) == 'undefined'){
      window.setTimeout('EsperaVariavelGlossarioNodos()',300);
   }else{
      GlossarioNodos = eval(VInicialGlossarioNodos);
   }
}

function EsperaVariavelGerente(){
   if (typeof(eval(VInicialGerente)) == 'undefined'){
      window.setTimeout('EsperaVariavelGerente()',300);
   }else{
      gerente = eval(VInicialGerente);
      if (typeof(eval(VInicialGlossarioNodos)) == 'undefined'){
         window.setTimeout('EsperaVariavelGlossarioNodos()',300);
      }else{
         GlossarioNodos = eval(VInicialGlossarioNodos);
      }
```

```
   }
}
EsperaVariavelGerente();

//FIM INICIALIZAÇÕES CRÍTICAS


/*-------------------------------------------------------------------------*/
function AbreSeguinte(){
  if ( (typeof(GlossarioNodos.FRH_nodo_seguinte) !=
'undefined')&&(GlossarioNodos.FRH_nodo_seguinte != '') ) {
     GlossarioNodos.location = gerente.DirConteinerGlossarioNodos_raiz +
gerente.DirNodosGlossario + GlossarioNodos.FRH_nodo_seguinte;
  }
}

/*-------------------------------------------------------------------------*/
function AbreAnterior(){
  if ( (typeof(GlossarioNodos.FRH_nodo_anterior) !=
'undefined')&&(GlossarioNodos.FRH_nodo_anterior != '') ) {
     GlossarioNodos.location = gerente.DirConteinerGlossarioNodos_raiz +
gerente.DirNodosGlossario + GlossarioNodos.FRH_nodo_anterior;
  }
}

/*-------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------*/
/*-------------------------------------------------------------------------*/
-->
</SCRIPT>
<!--------------------------------------------------------------------------->
<!--                              CSS                                    -->
<!--------------------------------------------------------------------------->
<style type="text/css">
<!--
  a:visited {  color: #0000CC; text-decoration: none;font-weight: bold}
  a:link {  color: #0000CC; text-decoration: none;font-weight: bold}
  a:hover {  color: #FF0000; text-decoration: underline;font-weight: bold}
.BotoesHistoriaGl {  font-family: Arial, Helvetica, sans-serif; font-size:
11pt; font-weight: normal; background-color: #A4C09C; border-color: #000000
#000000 #333333; margin-bottom: 4px; line-height: 10pt; width: 26px;
```

```
height: 18px; border-top-width: 1px; border-right-width: 1px; border-
bottom-width: 1px; border-left-width: 1px}
.BotoesNavegaGl {  font-family: Arial, Helvetica, sans-serif; font-size:
7pt; font-weight: bold; background-color: #CECDA6; margin-bottom: 4px;
border: 1px #000000 solid}
-->
</style>
<!-------------------------------------------------------------------------->
<!-------------------------------------------------------------------------->
<!-------------------------------------------------------------------------->
</head>

<body bgcolor="#CEEBCB">
<b><font face="Verdana, Arial, Helvetica, sans-serif" size="3"> </font></b>
<table border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td><b><font face="Verdana, Arial, Helvetica, sans-serif"
size="3">GLOSSÁRIO</font></b></td>
    <td>
      <input type="button" name="hvolta2" value="&lt;"
onClick="GlossarioNodos.history.go(-1);" class="BotoesHistoriaGl">
      <input type="button" name="hvai2" value="&gt;"
onClick="GlossarioNodos.history.go(+1);" class="BotoesHistoriaGl">
    </td>
    <td align="right">
      <input type="button" name="anterior" value="Anterior"
onClick="AbreAnterior();" class="BotoesNavegaGl">
      <input type="button" name="seguinte" value="Pr&oacute;ximo"
onClick="AbreSeguinte();" class="BotoesNavegaGl">
    </td>
  </tr>
  <tr>
    <td colspan="3"><font face="Verdana, Arial, Helvetica, sans-serif"
size="2">[<a href="glossario_indice.html#a"
target="glossario_nodos">A</a>|<a href="glossario_indice.html#b"
target="glossario_nodos">B</a>|<a href="glossario_indice.html#c"
target="glossario_nodos">C</a>|<a href="glossario_indice.html#d"
target="glossario_nodos">D</a>|<a href="glossario_indice.html#e"
target="glossario_nodos">E</a>|<a href="glossario_indice.html#f"
target="glossario_nodos">F</a>|<a href="glossario_indice.html#g"
target="glossario_nodos">G</a>|<a href="glossario_indice.html#h"
```

```
target="glossario_nodos">H</a>|<a href="glossario_indice.html#i"
target="glossario_nodos">I</a>|<a href="glossario_indice.html#j"
target="glossario_nodos">J</a>|<a href="glossario_indice.html#k"
target="glossario_nodos">K</a>|<a href="glossario_indice.html#l"
target="glossario_nodos">L</a>|<a href="glossario_indice.html#m"
target="glossario_nodos">M</a>|<a href="glossario_indice.html#n"
target="glossario_nodos">N</a>|<a href="glossario_indice.html#o"
target="glossario_nodos">O</a>|<a href="glossario_indice.html#p"
target="glossario_nodos">P</a>|<a href="glossario_indice.html#q"
target="glossario_nodos">Q</a>|<a href="glossario_indice.html#r"
target="glossario_nodos">R</a>|<a href="glossario_indice.html#s"
target="glossario_nodos">S</a>|<a href="glossario_indice.html#t"
target="glossario_nodos">T</a>|<a href="glossario_indice.html#u"
target="glossario_nodos">U</a>|<a href="glossario_indice.html#v"
target="glossario_nodos">V</a>|<a href="glossario_indice.html#w"
target="glossario_nodos">W</a>|<a href="glossario_indice.html#x"
target="glossario_nodos">X</a>|<a href="glossario_indice.html#y"
target="glossario_nodos">Y</a>|<a href="glossario_indice.html#z"
target="glossario_nodos">Z</a>]
      </font> </td>
  </tr>
</table>
</body>
</html>
```

III. Unit (módulo) do Sistema PCMAT responsável pela criação do documento XML e uso do componente TFRH

```
unit uRelatorioHipermidia;

interface

 Uses Sysutils,RelatorioProcs, Contnrs, dmRelatorioHipermidia, Classes,
Forms, uFRH, ufMostraRelatorioHipermidia, UFRHArquivos;

type

//-------------------------------------------------------------//
  TAtributo = record
                Nome: string;
                Valor: string;
              end;
//-------------------------//
  TListaAtributos = array of TAtributo;

//-------------------------------------------------------------//
  TGeradorCodigoXML = class
  private
     XML: TStringList;
     function GetTextoXML: string;
  protected
     property TextoXML: string read GetTextoXML;
     procedure InsereDeclaracaoXMLPadrao();
     procedure InsereTexto(Texto: string);
     procedure InsereElementoVazio(NomeElemento: string; Atributos:
TListaAtributos);
     procedure InsereElementoSimples(NomeElemento: string; Atributos:
TListaAtributos; Texto: string); overload;
     procedure InsereElementoSimples(NomeElemento: string; Texto: string);
overload;
     procedure InsereInicioElemento(NomeElemento: string; Atributos:
TListaAtributos); overload;
     procedure InsereInicioElemento(NomeElemento: string); overload;
```

```
      procedure InsereFimElemento(NomeElemento: string); overload;
  public
      constructor Create();
      destructor Destroy();
  end;

//-------------------------------------------------------------//
  TRecordProficao = class
  private
       MESTRES: integer;
       ENCARREGADOS: integer;
       PEDREIROS: integer;
       SERVENTES: integer;
       FERREIROS: integer;
       ELETRICISTAS: integer;
       CARPINTEIROS: integer;
       VIGIAS: integer;
  end;

//-------------------------------------------------------------//
  TGeradorHipermidia = class
  private
      //diretorios
      ExePach: string;
      Midias_path: string;
      TemplatesFRH_path: string;
      XSL_path: string;
      Relatorio_path: string;
      MidiasHiper_path: string;
      MidiasHiper_pathRelativo: string;  //path relativo das midias
hipermidia em relacao ao diretorio de conteudo

      RelatProcs: TRelatorioProcs;
      XMLMedPrevent: TstringList;
      ANrEtapas: integer;
      ANrAtividades: integer;  //para cada Etapa
      ANrRiscos: integer;   //para cada Atividade
      ANrPrecaucoes: integer;  //para cada Risco
      AEtapa: string;
      AAtividade: string;
      ARisco: string;
```

```
      APrecaucao: string;
      function TipoDeMidia(StrArqMidia: string): string;
      procedure ExibeRelatorioHipermidia(IndexPath: string);
      function GeraXMLGlossario(): string;
      function decodifica01(n: integer):string;
  public
      procedure geraHTMLMedPreventHiper(Exibe: boolean);
      constructor Create();
      destructor Destroy();
  end;

//-------------------------------------------------------------//

var
  GeradorHipermidia: TGeradorHipermidia;

//-------------------------------------------------------------//
implementation

uses
    StringProcs;


(*-------------------------------------------------------------*)
constructor TGeradorCodigoXML.Create();
begin
    self.XML := TstringList.Create();
end;

(*-------------------------------------------------------------*)
destructor TGeradorCodigoXML.Destroy();
begin
    self.XML.Free();
end;

(*-------------------------------------------------------------*)
procedure TGeradorCodigoXML.InsereDeclaracaoXMLPadrao();
begin
    self.XML.Add('<?xml version="1.0" encoding="ISO-8859-1"?>');
end;
```

```
(*-----------------------------------------------------------------*)
function TGeradorCodigoXML.GetTextoXML: string;
begin
    result := self.XML.Text;
end;

(*-----------------------------------------------------------------*)
procedure TGeradorCodigoXML.InsereTexto(Texto: string);
begin
    Self.XML.Add(Texto);
end;

(*-----------------------------------------------------------------*)
procedure TGeradorCodigoXML.InsereElementoVazio(NomeElemento: string;
Atributos: TListaAtributos);
var strAtributos: string;
    i: integer;
begin
    StrAtributos := '';
    for i := 0 to Length(Atributos)-1 do begin
        if Atributos[i].Nome = '' then
           break;
        StrAtributos := StrAtributos + Atributos[i].Nome + '="' +
Atributos[i].Valor + '" ';
    end;
    Self.XML.Add('<' + NomeElemento + ' ' + StrAtributos + '/>');
end;

(*-----------------------------------------------------------------*)
procedure TGeradorCodigoXML.InsereElementoSimples(NomeElemento: string;
Atributos: TListaAtributos; Texto: string);
var strAtributos: string;
    i: integer;
begin
    StrAtributos := '';
    for i := 0 to Length(Atributos)-1 do begin
        if Atributos[i].Nome = '' then
           break;
        StrAtributos := StrAtributos + Atributos[i].Nome + '="' +
Atributos[i].Valor + '" ';
    end;
```

```
    Self.XML.Add('<' + NomeElemento + ' ' + StrAtributos + '>' + texto +
'</' + NomeElemento + '>');
end;

(*----------------------------------------------------------------*)
procedure TGeradorCodigoXML.InsereElementoSimples(NomeElemento: string;
Texto: string);
begin
    Self.XML.Add('<' + NomeElemento + '>' + texto + '</' + NomeElemento +
'>');
end;

(*----------------------------------------------------------------*)
procedure TGeradorCodigoXML.InsereInicioElemento(NomeElemento: string;
Atributos: TListaAtributos);
var strAtributos: string;
    i: integer;
begin
    StrAtributos := '';
    for i := 0 to Length(Atributos)-1 do begin
        if Atributos[i].Nome = '' then
           break;
        StrAtributos := StrAtributos + Atributos[i].Nome + '="' +
Atributos[i].Valor + '" ';
    end;
    Self.XML.Add('<' + NomeElemento + ' ' + StrAtributos + '>');
end;

(*----------------------------------------------------------------*)
procedure TGeradorCodigoXML.InsereInicioElemento(NomeElemento: string);
begin
    Self.XML.Add('<' + NomeElemento + '>');
end;

(*----------------------------------------------------------------*)
procedure TGeradorCodigoXML.InsereFimElemento(NomeElemento: string);
begin
    Self.XML.Add('</' + NomeElemento + '>');
end;
(*----------------------------------------------------------------*)
(*----------------------------------------------------------------*)
```

```
(*-----------------------------------------------------------------*)

constructor TGeradorHipermidia.Create();
begin
    self.RelatProcs := TRelatorioProcs.Create();
    self.ExePach := ExtractFilePath(Application.ExeName);
    self.TemplatesFRH_path :=  self.ExePach + 'DocsFRH\TemplatesFRH\';
    self.XSL_path := self.ExePach + 'DocsFRH\xsl\';
    self.Relatorio_path := self.ExePach + 'PCMAT Hipermidia\';
    self.Midias_path := self.ExePach + 'Relatorio\Imagens\';
    self.MidiasHiper_pathRelativo := 'midias\';
    self.MidiasHiper_path := self.Relatorio_path + 'conteudo\' +
self.MidiasHiper_pathRelativo;

end;

(*-----------------------------------------------------------------*)
destructor TGeradorHipermidia.Destroy();
begin
    self.RelatProcs.Free();
end;

(*-----------------------------------------------------------------*)
function TGeradorHipermidia.decodifica01(n: integer):string;
begin
    if n = 1 then
       result := 'x'
    else
       result := 'o';
end;

(*-----------------------------------------------------------------*)
function TGeradorHipermidia.TipoDeMidia(StrArqMidia: string): string;
var ExtMidia: string;
begin
   ExtMidia := LowerCase(ExtractFileExt(StrArqMidia));
   if (ExtMidia='.gif') or (ExtMidia='.jpeg') or (ExtMidia='.jpg') or
(ExtMidia='.tif') or (ExtMidia='.png') then
       result := 'imagem'
   else if ExtMidia='.swf' then
       result := 'flash'
```

```
   else if ExtMidia='.dcr' then
       result := 'shockwave'
   else
       result := '';

end;

(*-----------------------------------------------------------------*)
function TGeradorHipermidia.GeraXMLGlossario(): string;
var
    XMLGlossario, vlPathMidia: string;
    i: integer;
    GeradorCodigoXML: TGeradorCodigoXML;
    Atributos: TListaAtributos;
    InterfaceArquivos: TInterfaceArquivos;
begin
    //cria objetos a serem utilizados
    GeradorCodigoXML := TGeradorCodigoXML.Create();
    self.RelatProcs.InicializaGlossario();
    try
        SetLength(Atributos,3);
        Atributos[0].Nome := 'FRH_GRUPO';
        Atributos[0].Valor := 'sim';
        Atributos[1].Nome := 'FRH_ESTILO_INDICE_GLOSSARIO';
        Atributos[1].Valor := 'FRH_glossario_indice';
        Atributos[2].Nome := 'FRH_TITULO';
        Atributos[2].Valor := 'GLOSSÁRIO';
        GeradorCodigoXML.InsereInicioElemento('FRH_GLOSSARIO', Atributos);

        for i:=0 to self.RelatProcs.Glossario_NrRegistros-1 do begin
            SetLength(Atributos,3);
            Atributos[0].Nome := 'FRH_ESTILO';
            Atributos[0].Valor := 'FRH_glossario_termo';
            Atributos[1].Nome := 'FRH_TITULO';
            Atributos[1].Valor := self.RelatProcs.Glossario_Termo;
            Atributos[2].Nome := 'FRH_PALAVRASCH';
            Atributos[2].Valor := self.RelatProcs.Glossario_Termo;
            GeradorCodigoXML.InsereInicioElemento('Termo', Atributos);
              GeradorCodigoXML.InsereElementoSimples('Nome',
self.RelatProcs.Glossario_Termo);
```

```
              GeradorCodigoXML.InsereElementoSimples('Definicao',
self.RelatProcs.Glossario_Definicao);
              //se existe midia, insere o elemento e copia a midia para o
diretorio padrao das midias do PCMATHipermidia
              vlPathMidia := self.RelatProcs.Glossario_PathMidia;
              if vlPathMidia <> '' then begin
                 SetLength(Atributos,2);
                 Atributos[0].Nome := 'tipo';
                 Atributos[0].Valor := self.TipoDeMidia(vlPathMidia);
                 Atributos[1].Nome := 'path';
                 Atributos[1].Valor := self.MidiasHiper_pathRelativo +
vlPathMidia;
                 GeradorCodigoXML.InsereElementoVazio('Midia', Atributos);
                 InterfaceArquivos.CopiaArq(self.Midias_path + vlPathMidia,
self.MidiasHiper_path + vlPathMidia, True);
              end;
            GeradorCodigoXML.InsereFimElemento('Termo');
            self.RelatProcs.Glossario_TermoSeguinte();
        end;
        GeradorCodigoXML.InsereFimElemento('FRH_GLOSSARIO');
        XMLGlossario := GeradorCodigoXML.TextoXML;
    finally
       SetLength(Atributos,0);
       GeradorCodigoXML.Free();
    end;
    result := XMLGlossario;
end;

(*----------------------------------------------------------------*)
procedure TGeradorHipermidia.geraHTMLMedPreventHiper(Exibe: boolean);
var
    i,j,k,z,codEtapa,codAtividade,codRisco: integer;
    vlPathMidia: string;
    GeradorCodigoXML: TGeradorCodigoXML;
    FRH : TFRH;
    Atributos: TListaAtributos;
    InterfaceArquivos: TInterfaceArquivos;
begin
    ExePach := ExtractFilePath(Application.ExeName);
    InterfaceArquivos := TInterfaceArquivos.Create();
    //cria o diretorio das midias
```

```
    InterfaceArquivos.CriaDiretorios(self.MidiasHiper_path);

    GeradorCodigoXML := TGeradorCodigoXML.Create();
    GeradorCodigoXML.InsereDeclaracaoXMLPadrao();
    GeradorCodigoXML.InsereInicioElemento('FRH_ROOT');

    SetLength(Atributos,2);
    Atributos[0].Nome := 'FRH_GRUPO';
    Atributos[0].Valor := 'sim';
    Atributos[1].Nome := 'FRH_TITULO';
    Atributos[1].Valor := 'Relatório de Medidas Preventivas do Processo
Construtivo';
    GeradorCodigoXML.InsereInicioElemento('FRH_UNIDADE_CONCEITO',
Atributos);

  try
(*************************************************************************)
(*                      LISTA ETAPAS                                     *)
(*************************************************************************)
    SetLength(Atributos,4);
    Atributos[0].Nome := 'FRH_ESTILO';
    Atributos[0].Valor := 'lista_etapas';
    Atributos[1].Nome := 'FRH_TITULO';
    Atributos[1].Valor := 'Etapas do Processo Construtivo';
    Atributos[2].Nome := 'FRH_PALAVRASCH';
    Atributos[2].Valor := 'Etapas do Processo Construtivo, Processo
Construtivo';
    GeradorCodigoXML.InsereInicioElemento('ListaEtapas', Atributos);

(*-----------------------------------------------------------------------*)
(*                       ETAPAS                                          *)
(*-----------------------------------------------------------------------*)
    //GERA RELATORIO MEDIDAS PREVENTIVAS
    RelatProcs.InicializaEtapas();
    ANrEtapas := RelatProcs.NumeroEtapas();
    RelatProcs.PrimeiraEtapa();
    //laco principal onde percorre todas Etapas e constroi o HTML
    for i:=1 to ANrEtapas do
    begin
        AEtapa := RelatProcs.DescricaoEtapa();
        codEtapa := RelatProcs.CodigoEtapa();
```

```
SetLength(Atributos,3);
Atributos[0].Nome := 'FRH_ESTILO';
Atributos[0].Valor := 'etapa';
Atributos[1].Nome := 'FRH_TITULO';
Atributos[1].Valor := AEtapa;
Atributos[2].Nome := 'FRH_PALAVRASCH';
Atributos[2].Valor := AEtapa;
GeradorCodigoXML.InsereInicioElemento('etapa', Atributos);
GeradorCodigoXML.InsereElementoSimples('numero',IntToStr(i));
GeradorCodigoXML.InsereElementoSimples('nome',AEtapa);

//laco de Atividades
RelatProcs.SetaAtividades(codEtapa);
ANrAtividades := RelatProcs.NumeroAtividades();
RelatProcs.PrimeiraAtividade();
for j:=1 to ANrAtividades do
begin
    AAtividade := RelatProcs.DescricaoAtividade();
    codAtividade := RelatProcs.CodigoAtividade();

    SetLength(Atributos,3);
    Atributos[0].Nome := 'FRH_ESTILO';
    Atributos[0].Valor := 'atividade';
    Atributos[1].Nome := 'FRH_TITULO';
    Atributos[1].Valor := AAtividade;
    Atributos[2].Nome := 'FRH_PALAVRASCH';
    Atributos[2].Valor := AAtividade;
    GeradorCodigoXML.InsereInicioElemento('atividade', Atributos);
    GeradorCodigoXML.InsereElementoSimples('numero',IntToStr(j));
    GeradorCodigoXML.InsereElementoSimples('nome',AAtividade);

    RelatProcs.SetaRiscos(codAtividade);
    RelatProcs.PrimeiroRisco();
    ANrRiscos := RelatProcs.NumeroRiscos();
    //laco de riscos
    for k:=1 to ANrRiscos do
    begin
        ARisco := RelatProcs.DescricaoRisco();
        codRisco := RelatProcs.CodigoRisco();
        vlPathMidia := RelatProcs.FiguraRisco();
```

```
                SetLength(Atributos,3);
                Atributos[0].Nome := 'FRH_ESTILO';
                Atributos[0].Valor := 'risco';
                Atributos[1].Nome := 'FRH_TITULO';
                Atributos[1].Valor := ARisco;
                Atributos[2].Nome := 'FRH_PALAVRASCH';
                Atributos[2].Valor := ARisco;
                GeradorCodigoXML.InsereInicioElemento('risco', Atributos);

GeradorCodigoXML.InsereElementoSimples('numero',IntToStr(k));
                SetLength(Atributos,1);
                Atributos[0].Nome := 'FRH_LINKS';
                Atributos[0].Valor := 'parent::risco';
                GeradorCodigoXML.InsereElementoSimples('nome', Atributos,
ARisco);
                //Insere uma midia e copia para o diretorio padrao (se
exitir uma midia)
                if vlPathMidia <> '' then begin
                   SetLength(Atributos,2);
                   Atributos[0].Nome := 'tipo';
                   Atributos[0].Valor := self.TipoDeMidia(vlPathMidia);
                   Atributos[1].Nome := 'path';
                   Atributos[1].Valor := self.MidiasHiper_pathRelativo +
vlPathMidia;
                   GeradorCodigoXML.InsereElementoVazio('Midia',
Atributos);
                   InterfaceArquivos.CopiaArq(self.Midias_path +
vlPathMidia, self.MidiasHiper_path + vlPathMidia, True);
                end;
                //laco das Precaucoes
                RelatProcs.SetaMedidasPreventivas(codRisco);
                RelatProcs.PrimeiraMedidaPreventiva();
                ANrPrecaucoes := RelatProcs.NumeroMedidasPreventivas();
                for z:=1 to ANrPrecaucoes do begin
                    APrecaucao := RelatProcs.DescricaoMedidaPreventiva();
                    vlPathMidia := RelatProcs.FiguraMedidaPreventiva();

                    SetLength(Atributos,3);
                    Atributos[0].Nome := 'FRH_ESTILO';
                    Atributos[0].Valor := 'medida_preventiva';
```

```
                    Atributos[1].Nome := 'FRH_TITULO';
                    Atributos[1].Valor := APrecaucao;
                    Atributos[2].Nome := 'FRH_PALAVRASCH';
                    Atributos[2].Valor := APrecaucao;

GeradorCodigoXML.InsereInicioElemento('medida_preventiva', Atributos);

GeradorCodigoXML.InsereElementoSimples('nome',APrecaucao);
                    //Insere uma midia e copia para o diretorio padrao (se
exitir uma midia)
                    if vlPathMidia <> '' then begin
                       SetLength(Atributos,2);
                       Atributos[0].Nome := 'tipo';
                       Atributos[0].Valor := self.TipoDeMidia(vlPathMidia);
                       Atributos[1].Nome := 'path';
                       Atributos[1].Valor := self.MidiasHiper_pathRelativo
+ vlPathMidia;
                       GeradorCodigoXML.InsereElementoVazio('Midia',
Atributos);
                       InterfaceArquivos.CopiaArq(self.Midias_path +
vlPathMidia, self.MidiasHiper_path + vlPathMidia, True);
                    end;
                   GeradorCodigoXML.InsereFimElemento('medida_preventiva');
(******************* FECHA Medida preventiva ***************************)
                   RelatProcs.ProximaMedidaPreventiva();
                end;
                GeradorCodigoXML.InsereFimElemento('risco');
(********************** FECHA Risco ************************************)
                RelatProcs.ProximoRisco();
            end;
            GeradorCodigoXML.InsereFimElemento('atividade');
(************************ FECHA Atividade ******************************)
            RelatProcs.ProximaAtividade();
        end;
        GeradorCodigoXML.InsereFimElemento('etapa');
(************************** FECHA Etapa ********************************)
        RelatProcs.ProximaEtapa();
    end;
    GeradorCodigoXML.InsereFimElemento('ListaEtapas');
(************************* FECHA Lista de Etapas ***********************)
    GeradorCodigoXML.InsereFimElemento('FRH_UNIDADE_CONCEITO');
```

```
(********************** FECHA UNIDADE CONCEITO ************************)

(*-------------------------------------------------------------------*)
(*                          GLOSSÁRIO                                *)
(*-------------------------------------------------------------------*)
    GeradorCodigoXML.InsereTexto(Self.GeraXMLGlossario());
(*-------------------------------------------------------------------*)
    GeradorCodigoXML.InsereFimElemento('FRH_ROOT');
(************************ FECHA DOCUMENTO ****************************)
    SetLength(Atributos,0);

    //USA O COMPONENTE FRH
    //instancia
    FRH := TFRH.create(GeradorCodigoXML.GetTextoXML, XSL_path,
Relatorio_path,  TemplatesFRH_path,[ucNavegacaoHierarquica, ucFluxograma,
ucIndice, ucLinksAutomaticos, ucBusca],[gIndice, gLinksAutomaticos,
gFluxograma, gIncluiNoIndicePrincipal]);
    FRH.AplicaEstilos();
  Finally
    GeradorCodigoXML.Free();
    if FRH <> nil then
       FRH.Free();
  end;
  if Exibe then
     ExibeRelatorioHipermidia(Relatorio_path + 'index.html');
end;

(*-----------------------------------------------------------------*)
procedure TGeradorHipermidia.ExibeRelatorioHipermidia(IndexPath: string);
var fmPCMATHipermidia: TfmPCMATHipermidia;
begin
    fmPCMATHipermidia := TfmPCMATHipermidia.Create('PCMAT HIPERMÍDIA');
    try
      fmPCMATHipermidia.WebBrowser.Navigate(IndexPath);
      fmPCMATHipermidia.ShowModal();
    finally
      fmPCMATHipermidia.Free();
    end;
end;

end.
```

IV. Estilos XSLT criados para o PCMAT Hipermídia

atividade.xsl

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" encoding="ISO-8859-1"/>

<xsl:template match="atividade">
<HTML>
<HEAD>
<TITLE><xsl:value-of select="@FRH_TITULO"/></TITLE>
   <META http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1"/>
   <link rel="stylesheet" href="../css/padrao.css"/>
   <link ID="estilo_links" rel="stylesheet"
href="../css/estilo_links.css"/>
</HEAD>
<BODY class="FontePadrao">
<h4> Atividade: <xsl:value-of select="@FRH_TITULO"/> </h4>
<table align="center" width="94%" cellpadding="0" cellspacing="0"
border="0"><tr><td>
  Lista dos riscos e medidas preventivas contra acidentes: <br/><br/>
  <table style=" font-size: 10pt;" width="100%" class="Tabela"
cellpadding="2" cellspacing="1" border="0">
    <tr>
      <td colspan="2" align="center" style="background-color:
#EAECF2"><font size="3" color="#FF0000"><b>RISCOS</b></font></td>
      <td width="65%" align="center" style="background-color:
#EAECF2"><b><font size="3" color="#006666">MEDIDAS
PREVENTIVAS</font></b></td>
    </tr>
    <xsl:apply-templates select="child::risco"/>
 </table>
 </td></tr></table>
<br/><br/>
</BODY>
```

```
</HTML>
</xsl:template>

<xsl:template match="risco">
    <tr>
      <td width="3%" class="CelulaCabecalho"><xsl:value-of
select="numero"/></td>
      <td width="32%"><xsl:value-of select="nome"/></td>
      <td width="65%">
         <ul>
            <xsl:apply-templates select="child::medida_preventiva"/>
         </ul>
      </td>
    </tr>
</xsl:template>

<xsl:template match="medida_preventiva">
      <li><xsl:value-of select="nome"/></li>
</xsl:template>

</xsl:stylesheet>
```

etapa.xsl

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" encoding="ISO-8859-1"/>

<xsl:template match="etapa">
<HTML>
<HEAD>
<TITLE><xsl:value-of select="@FRH_TITULO"/></TITLE>
   <META http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1"/>
   <link rel="stylesheet" href="../css/padrao.css"/>
   <link ID="estilo_links" rel="stylesheet"
href="../css/estilo_links.css"/>
</HEAD>
<BODY class="FontePadrao">
```

```
<h2 style="font-size=13pt"> <xsl:value-of select="@FRH_TITULO"/> </h2>
<table align="center" width="94%" cellpadding="0" cellspacing="0"
border="0"><tr><td>
  Lista das atividades:<br/><br/>
  <table style=" font-size: 10pt;" width="100%" class="Tabela"
cellpadding="4" cellspacing="1" border="0">
    <xsl:apply-templates select="child::atividade"/>
  </table>
  <br/><br/>
</td></tr></table>
</BODY>
</HTML>
</xsl:template>

<xsl:template match="atividade">
    <tr>
      <td width="3%" class="CelulaCabecalho"><xsl:value-of
select="numero"/></td>
      <td width="95%"><xsl:value-of select="nome"/></td>
    </tr>
</xsl:template>

</xsl:stylesheet>
```

FRH_glossario_indice.xsl (padrão do FRH)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" encoding="ISO-8859-1"/>

   <xsl:template match="/">
      <xsl:apply-templates/>
   </xsl:template>

   <xsl:template match="FRH_GLOSSARIO">
      <HTML>
         <HEAD>
            <META http-equiv="Content-Type" content="text/html;
charset=ISO-8859-1"/>
```

```
            <LINK REL="stylesheet" HREF="css/glossario_indice.css"
TYPE="text/css"/>
         </HEAD>
         <BODY class="Indice_Body">
           <blockquote>

            <a name="a"/>
            <div class="Indice_Letra">A</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'A')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'a')]"/>

            <a name="b"/>
            <div class="Indice_Letra">B</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'B')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'b')]"/>

            <a name="c"/>
            <div class="Indice_Letra">C</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'C')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'c')]"/>

            <a name="d"/>
            <div class="Indice_Letra">D</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'D')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'd')]"/>

            <a name="e"/>
            <div class="Indice_Letra">E</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'E')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'e')]"/>

            <a name="f"/>
            <div class="Indice_Letra">F</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'F')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'f')]"/>
```

```
            <a name="g"/>
            <div class="Indice_Letra">G</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'G')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'g')]"/>

            <a name="h"/>
            <div class="Indice_Letra">H</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'H')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'h')]"/>

            <a name="i"/>
            <div class="Indice_Letra">I</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'I')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'i')]"/>

            <a name="j"/>
            <div class="Indice_Letra">J</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'J')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'j')]"/>

            <a name="k"/>
            <div class="Indice_Letra">K</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'K')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'k')]"/>

            <a name="l"/>
            <div class="Indice_Letra">L</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'L')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'l')]"/>

            <a name="m"/>
            <div class="Indice_Letra">M</div>
```

```
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'M')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'm')]"/>

            <a name="n"/>
            <div class="Indice_Letra">N</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'N')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'n')]"/>

            <a name="o"/>
            <div class="Indice_Letra">O</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'O')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'o')]"/>

            <a name="p"/>
            <div class="Indice_Letra">P</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'P')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'p')]"/>

            <a name="q"/>
            <div class="Indice_Letra">Q</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'Q')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'q')]"/>

            <a name="r"/>
            <div class="Indice_Letra">R</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'R')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'r')]"/>

            <a name="s"/>
            <div class="Indice_Letra">S</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'S')or(substring(normalize-
space(@FRH_TITULO),1,1) = 's')]"/>

            <a name="t"/>
```

```
            <div class="Indice_Letra">T</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'T')or(substring(normalize-
space(@FRH_TITULO),1,1) = 't')]"/>

            <a name="u"/>
            <div class="Indice_Letra">U</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'U')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'u')]"/>

            <a name="v"/>
            <div class="Indice_Letra">V</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'V')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'v')]"/>

            <a name="w"/>
            <div class="Indice_Letra">W</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'W')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'w')]"/>

            <a name="x"/>
            <div class="Indice_Letra">X</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'X')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'x')]"/>

            <a name="y"/>
            <div class="Indice_Letra">Y</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'Y')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'y')]"/>

            <a name="z"/>
            <div class="Indice_Letra">Z</div>
            <xsl:apply-templates select="Termo[(substring(normalize-
space(@FRH_TITULO),1,1) = 'Z')or(substring(normalize-
space(@FRH_TITULO),1,1) = 'z')]"/>
```

```
            </blockquote>
            <br/><br/><br/><br/><br/><br/><br/><br/><br/><br/>
         </BODY>
      </HTML>
   </xsl:template>

   <xsl:template match="Termo">
         <a class="Indice_Termos">
            <xsl:attribute name="href">
               Javascript:self.location =
self.top.gerente.DirGlossarioIndice_raiz +
self.top.gerente.DirGlossarioIndice_raiz +
self.top.gerente.DirNodosGlossario + "<xsl:value-of
select="concat(@FRH_CODNODO,'.html&quot;')" />
            </xsl:attribute>
            <xsl:value-of select="@FRH_TITULO" />
         </a>
         <br/>

   </xsl:template>

</xsl:stylesheet>
```

FRH_glossario_termo.xsl (padrão do FRH)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" encoding="ISO-8859-1"/>

<xsl:include href="insere_midias.xsl" />

<xsl:template match="Termo">
<HTML>
<HEAD>
<TITLE><xsl:value-of select="@FRH_TITULO"/></TITLE>
   <META http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1"/>
   <link rel="stylesheet" href="../css/padrao.css"/>
```

```
   <link ID="estilo_links" rel="stylesheet"
href="../css/estilo_links.css"/>
</HEAD>
<BODY class="FontePadrao">
<h4 style="color: #006666"> <xsl:value-of select="@FRH_TITULO"/>
<hr size="1"/>
</h4>
<table align="center" width="94%" cellpadding="0" cellspacing="0"
border="0"><tr><td>
  <P>
    <xsl:value-of select="Definicao"/>
  </P>
</td></tr></table>
<br/>
  <xsl:call-template name="insere_midia">
     <xsl:with-param name="width" select="350"/>
     <xsl:with-param name="height" select="300"/>
  </xsl:call-template>
<br/>
</BODY>
</HTML>
</xsl:template>

</xsl:stylesheet>
```

insere_midias.xsl

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:template name="insere_midia">
  <xsl:param name="width"/>
  <xsl:param name="height"/>
  <xsl:for-each select = "Midia">
     <xsl:choose>
        <xsl:when test="@tipo = 'imagem'">
           <xsl:apply-templates select="." mode="img"/>
        </xsl:when>
        <xsl:when test="@tipo = 'flash'">
```

```
            <xsl:apply-templates select="." mode="flash">
               <xsl:with-param name="width" select="$width"/>
               <xsl:with-param name="height" select="$height"/>
            </xsl:apply-templates>
         </xsl:when>
         <xsl:when test="@tipo = 'shockwave'">
            <xsl:apply-templates select="." mode="shock"/>
         </xsl:when>
      </xsl:choose>
  </xsl:for-each>
</xsl:template>

<xsl:template match="Midia" mode="img">
  <DIV align="center">
  <img>
     <xsl:attribute name="src">
        <xsl:value-of select="@path"/>
     </xsl:attribute>
  </img>
  </DIV>
</xsl:template>

<xsl:template match="Midia" mode="flash">
  <xsl:param name="width" select="300"/>
  <xsl:param name="height" select="270"/>

<div align="center">
  <object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.c
ab#version=4,0,2,0" width="{$width}" height="{$height}">
    <param name="movie" value="{@path}"/>
    <param name="LOOP" value="false"/>
    <embed src="{@path}"
pluginspage="http://www.macromedia.com/shockwave/download/index.cgi?P1_Prod
_Version=ShockwaveFlash" type="application/x-shockwave-flash"
width="{$width}" height="{$height}" loop="false">
    </embed>
  </object>
</div>
</xsl:template>
```

```
<xsl:template match="Midia" mode="shock">
<OBJECT align="center" classid="clsid:166B1BCA-3F9C-11CF-8075-444553540000"

codebase="http://download.macromedia.com/pub/shockwave/cabs/director/sw.cab
#version=8,5,1,0" ID="shockwave" name="shockwave" WIDTH="98%" HEIGHT="98%"
vspace="10" hspace="15" border="1">
  <param name="src" value="{@path}"/>
  <param name="swStretchStyle" value="fill"/>
  <param name="swRemote" value="swSaveEnabled='false' swVolume='true'
swRestart='true' swPausePlay='true' swFastForward='true'
swContextMenu='true' "/>
  <PARAM NAME="bgColor" VALUE="#FFFFFF"/>
  <EMBED name="shockwave" SRC="{@path}" align="center" bgColor="#FFFFFF"
vspace="10" hspace="15" border="1" swLiveConnect="TRUE" WIDTH="98%"
HEIGHT="98%" swRemote="swSaveEnabled='false' swVolume='true'
swRestart='true' swPausePlay='true' swFastForward='true'
swContextMenu='true' " swStretchStyle="fill" TYPE="application/x-director"
PLUGINSPAGE="http://www.macromedia.com/shockwave/download/">
  </EMBED>
</OBJECT>

</xsl:template>

</xsl:stylesheet>
```

lista_etapas.xsl

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" encoding="ISO-8859-1"/>

<xsl:template match="ListaEtapas">
<HTML>
<HEAD>
<TITLE>Etapas do Processo Construtivo</TITLE>
   <META http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1"/>
   <link rel="stylesheet" href="../css/padrao.css"/>
```

```
   <link ID="estilo_links" rel="stylesheet"
href="../css/estilo_links.css"/>
</HEAD>
<BODY class="FontePadrao">
<h2> Etapas do Processo Construtivo </h2>
<table align="center" width="94%" cellpadding="0" cellspacing="0"
border="0"><tr><td>
  A tabela seguinte lista as Etapas do Processo Construtivo: <br/><br/>
  <table style=" font-size: 10pt;" width="100%" class="Tabela"
cellpadding="4" cellspacing="1" border="0">
      <xsl:apply-templates select="child::etapa"/>
  </table>
  <br/><br/>
</td></tr></table>
</BODY>
</HTML>
</xsl:template>

<xsl:template match="etapa">
    <tr>
      <td width="3%" class="CelulaCabecalho"><xsl:value-of
select="numero"/></td>
      <td width="94%"><xsl:value-of select="nome"/></td>
    </tr>
</xsl:template>

</xsl:stylesheet>
```

medida_preventiva.xsl

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" encoding="ISO-8859-1"/>

<xsl:include href="insere_midias.xsl" />

<xsl:template match="medida_preventiva">
<HTML>
<HEAD>
```

```
<TITLE><xsl:value-of select="@FRH_TITULO"/></TITLE>
   <META http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1"/>
   <link rel="stylesheet" href="../css/padrao.css"/>
   <link ID="estilo_links" rel="stylesheet"
href="../css/estilo_links.css"/>
</HEAD>
<BODY class="FontePadrao">
<h4 style="color: #006666"> Medida Preventiva:</h4>
<table align="center" width="94%" cellpadding="0" cellspacing="0"
border="0"><tr><td>
  <xsl:value-of select="@FRH_TITULO"/>
  <DIV align="center">
     <xsl:call-template name="insere_midia">
        <xsl:with-param name="width" select="350"/>
        <xsl:with-param name="height" select="250"/>
     </xsl:call-template>
  </DIV>
  <br/><br/>
</td></tr></table>
</BODY>
</HTML>
</xsl:template>

</xsl:stylesheet>
```

risco.xsl

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml" encoding="ISO-8859-1"/>

<xsl:include href="insere_midias.xsl" />

<xsl:template match="risco">
<HTML>
<HEAD>
<TITLE><xsl:value-of select="@FRH_TITULO"/></TITLE>
```

```
   <META http-equiv="Content-Type" content="text/html; charset=ISO-8859-
1"/>
   <link rel="stylesheet" href="../css/padrao.css"/>
   <link ID="estilo_links" rel="stylesheet"
href="../css/estilo_links.css"/>
</HEAD>
<BODY class="FontePadrao">
<h4> <xsl:value-of select="@FRH_TITULO"/> </h4>
<table align="center" width="94%" cellpadding="0" cellspacing="0"
border="0"><tr><td>
  <DIV align="center">
     <xsl:call-template name="insere_midia">
        <xsl:with-param name="width" select="350"/>
        <xsl:with-param name="height" select="250"/>
     </xsl:call-template>
  </DIV>
  <br/>
  <h5 style="color: #006666">MEDIDAS PREVENTIVAS</h5>
  <ol>
    <xsl:apply-templates select="child::medida_preventiva"/>
  </ol>
  <br/><br/>
</td></tr></table>
</BODY>
</HTML>
</xsl:template>

<xsl:template match="medida_preventiva">
      <li><xsl:value-of select="nome"/></li>
</xsl:template>

</xsl:stylesheet>
```