UNIVERSIDADE FEDERAL DO PARÁ
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

LUIZ ALANO ALEXANDRE LOPES

DESENVOLVIMENTO DE UMA FERRAMENTA DE AUTORIA DE CONTEÚDO PARA APRENDIZADO VIA WEB

DM 11/2003

UFPA/CT/PPGEE
Centro Universitário do Guamá
Belém – Pará – Brasil
2003

UNIVERSIDADE FEDERAL DO PARÁ
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

LUIZ ALANO ALEXANDRE LOPES

DESENVOLVIMENTO DE UMA FERRAMENTA DE AUTORIA DE CONTEÚDO PARA APRENDIZADO VIA WEB

Dissertação submetida à Banca Examinadora do Programa de Pós-Graduação em Engenharia Elétrica da UFPA para obtenção do grau de Mestre em Engenharia Elétrica.

UFPA/CT/PPGEE
Centro Universitário do Guamá
Belém – Pará – Brasil
2003

L864d Lopes, Luiz Alano Alexandre

Desenvolvimento de uma ferramenta de autoria de conteúdo para aprendizado via *web* / Luiz Alano Alexandre Lopes, Roberto Célio Limão de Oliveira.-2003.

Dissertação (Mestrado) – Universidade Federal do Pará, Centro Tecnológico, Programa de Pós-Graduação em Engenharia Elétrica, Belém, 2003.

1. Engenharia de software. 2. Ensino auxiliado por computador. 3. Internet na educação. I. Título.

CDD – 21. ed. 005. 1

UNIVERSIDADE FEDERAL DO PARÁ
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

DESENVOLVIMENTO DE UMA FERRAMENTA DE AUTORIA DE CONTEÚDO PARA APRENDIZADO VIA WEB

AUTOR: LUIZ ALANO ALEXANDRE LOPES

DISSERTAÇÃO DE MESTRADO SUBMETIDA À AVALIAÇÃO DA BANCA EXAMINADORA APROVADA PELO COLEGIADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA DA UNIVERSIDADE FEDERAL DO PARÁ E JULGADA ADEQUADA PARA OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA ELÉTRICA NA ÁREA DE COMPUTAÇÃO APLICADA.

APROVADA EM 27/03/2003

BANCA EXAMINADORA:

Prof. Dr. Roberto Célio Limão de Oliveira
(ORIENTADOR – UFPA)

Prof. Dr. Elói Luiz Favero
(MEMBRO – UFPA)

Prof. Dr. Rodrigo Quites Reis
(MEMBRO – UFPA)

VISTO:

Prof. Dr. Evaldo Gonçalves Pelaes
(COORDENADOR DO PPGEE/CT/UFPA)

UFPA/CT/PPGEE

Em memória de meu avô, Miguel Alexandre, que hoje vive em paz junto de Deus.

Ao meu pai, Luiz Lopes, minha mãe Carmem Ruth e a minha querida irmã Rejane, por tudo.

A minha noiva, Mônica, minha psicóloga particular, que muito me ensinou a respeito da vida, por todo apoio.

AGRADECIMENTOS

Ao Prof. Dr. Roberto Limão do PPGEE/CT/UFPA, pela atenção e orientação dispensadas em todos os trabalhos e disciplinas do mestrado.

Aos meus chefes do SERPRO, Paulo Brasil e Ticiano Monteiro, não só pelo apoio na realização deste trabalho, mas também como de todo o mestrado. Aos meus amigos e companheiros de trabalho, pela excelente convivência que temos.

Aos meus amigos da VIATEC, Anamélia, Gerson, Gleisson, Leonardo e Viviane, não só pelo prazer de trabalharmos juntos, mas também pelo desejo de juntos construirmos algo útil para todos.

SUMÁRIO

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE SIGLAS

DOM	Document Object Model
EAD	Educação a distância
GCO	Gestão do conhecimento
HTML	Hypertext Markup Language
PDF	Portable document format
RUP	Rational Unified Process
UML	Unified Modeling Language
W3C	World Wide Web Consortium
WML	Wireless Markup Language
WWW	World Wide Web
XML	Extensible Markup Language
XSLT	Extensible Stylesheet Language

RESUMO

A *web* tem sido bastante utilizada ultimamente por empresas e instituições de ensino como forma de oferecer aprendizado. A criação de conteúdos para o aprendizado via *web* é uma atividade que tem demandado recursos humanos especializados em tecnologias e conceitos, de forma que os conteúdos criados não sejam restritos a ferramentas, plataformas e formatos de apresentação. Este trabalho apresenta o desenvolvimento de um protótipo de uma ferramenta de autoria de conteúdo para aprendizado via *web*, através da qual os conteúdos criados são independentes da ferramenta de autoria, da plataforma operacional e dos formatos de apresentação que sejam utilizados. Por ser um trabalho de desenvolvimento, um processo é definido e suas fases e atividades são utilizadas para organizar trabalho e documentar todas as decisões tomadas.

PALAVRAS-CHAVES: Engenharia de Software, Autoria de Conteúdo, Aprendizado via *Web*.

ABSTRACT

Web has been very used lately for companies and institutions of education as a way to offer learning. The creation of contents for the learning through web is an activity that has demanded human resources specialized in technologies and concepts, so that the contents created are not restricted the tools, platforms and formats of presentation. This work presents the development of a content authoring tool prototype for learning through web, in which the created contents are independents of the authoring tool, platform and presentation formats that are used. Being a development work, a process is defined and its phases and activities are used to organize the job and to register all the taken decisions.

KEYWORDS: Software Engineering, Content Authoring, Web Based Learning.

CAPÍTULO 1

INTRODUÇÃO

A utilização da Internet, popularizada através da *World Wide Web* (WWW), como ferramenta para transmitir conhecimento entre as pessoas revolucionou a maneira pela qual as empresas e instituições de ensino tratam este assunto, possibilitando a seus funcionários, alunos, professores, entre outros, mais uma forma de aprender, o que pode ser chamado de aprendizado via *web*. Atualmente, não se pode imaginar uma grande corporação que não possua recursos computacionais ligados em rede para disseminar informações importantes sobre o negócio entre equipes geograficamente distantes umas das outras, a chamada gestão de conhecimento (GCO), através de *sites* internos e externos. Também é quase impossível imaginar uma universidade que não possua projetos de educação a distância (EAD), com disciplinas e até mesmo cursos de graduação e pós-graduação totalmente *on-line*.

Para o acontecimento de tal revolução foi, e ainda é, necessário o envolvimento de muita tecnologia. Permitir que as pessoas utilizem a *web* como sala de aula, laboratório ou biblioteca exige recursos humanos bastante capacitados em padrões e ferramentas específicas para a construção de todo o conteúdo que ficará disponível para o aprendizado. A atividade de construção de conteúdos para o aprendizado via *web*, pode ser feita de várias maneiras. Algumas organizações possuem um setor específico para esta atividade. Outras simplesmente passam a responsabilidade para o setor de informática ou de comunicação social. Muitas vezes, por razões diversas, as organizações acabam não dando a devida importância aos conteúdos, que ficam disponíveis sem muita preocupação com formas, padrões de apresentação, atualização e manutenção.

A construção de um conteúdo para aprendizado via *web* requer, além dos conceitos pedagógicos envolvidos com esta atividade, a utilização de tecnologias que não restrinjam a manipulação do mesmo. Uma vez criado, a sua implantação não deve ser complicada e seus usuários devem possuir acesso de maneira simples. Além disto, sua manutenção deve ser independente do instrumento utilizado na sua criação, permitindo que ajustes necessários sejam feitos de forma rápida. Outro ponto importante é que na *web* um conteúdo pode possuir diversas formas de apresentação, seja pelos navegadores existentes, por dispositivos móveis como celulares ou simplesmente em formatos como os arquivos PDF (*Portable Document Format*) que, entre outras vantagens, permitem uma boa impressão em papel.

Baseando-se nestas considerações, a seguir apresentam-se os objetivos deste texto.

1.1 Objetivos

O principal objetivo deste trabalho é o desenvolvimento de um protótipo de uma ferramenta de autoria de conteúdo para aprendizado via *web* (que será chamado, no texto, de ferramenta de autoria de conteúdo para aprendizado via *web* ou somente ferramenta). Será definido e instanciado um processo de desenvolvimento, com base em processos iterativos e incrementais, com todas as fases e atividades que serão necessárias. O formato dos conteúdos a serem criados será definido no início deste processo e estes conteúdos deverão possuir as características citadas anteriormente

Como um processo de desenvolvimento requer a utilização e a documentação de uma série de conceitos e modelos, é também um objetivo que este trabalho possa servir como modelo de documentação para projetos de desenvolvimento de sistemas.

1.2 Organização do texto

No capítulo 2 apresenta-se uma definição de aprendizado e um histórico sobre a utilização de recursos computacionais para o mesmo. Apresentam-se alguns trabalhos realizados na área de educação a distância com intuito de demonstrar o estado da arte e os problemas atacados. O capítulo encerra-se com uma descrição informal da proposta deste trabalho.

No capitulo 3 inicia-se o processo de desenvolvimento da ferramenta. Define-se o processo de desenvolvimento e apresenta-se a fase de início do processo com a realização das atividades de especificação, casos de uso, estimativa e plano de iterações.

No capítulo 4 apresenta-se a fase de elaboração do processo com a realização das atividades de análise e projeto.

No capítulo 5 encerra-se o desenvolvimento da ferramenta, com a apresentação da fase de construção do projeto e suas respectivas atividades: modelo de dados, implementação e testes. O capítulo é encerrado com a amostra de uma evidência de execução da ferramenta ilustrando telas do aplicativo e de conteúdos gerados.

No capítulo 6 apresentam-se as conclusões tiradas deste trabalho e um direcionamento que pode ser utilizado em trabalhos futuros na área.

CAPÍTULO 2

APRENDIZADO, ESTADO DA ARTE E PROPOSTA

No intuito de contextualizar e definir o problema de que trata este trabalho, neste capítulo são apresentados um histórico sobre o aprendizado e a utilização de recursos computacionais no mesmo, o estado da arte de pesquisas sobre o aprendizado via *web* e a proposta de solução para o problema definido.

2.1 Aprendizado

O aprendizado é o ato ou efeito de aprender, que por sua vez é o fato de tomar conhecimento de algo, retendo na memória, graças a estudo, observação, experiência, etc. [Ferreira1985].

Há diversas outras definições de aprendizado na literatura, algumas mais gerais e outras mais específicas a uma determinada área. Por exemplo, uma boa definição de aprendizado para Redes Neurais Artificiais é o processo de ajustes sucessivos nos pesos das suas conexões e a verificação do padrão para o qual a rede converge e se estabiliza [ABO2000]. Para o propósito deste texto, a definição dada no primeiro parágrafo é suficiente.

O ato de aprender, ou seja, adquirir conhecimento, nasceu junto com a humanidade. Diariamente, mesmo que de forma involuntária, o ser humano aprende, pois o conhecimento lhe é transmitido de diversas formas. O homem para sobreviver, principalmente no mundo atual, precisa aprender sempre, manter-se atualizado, para que possa suprir as suas necessidades básicas da melhor forma.

Atualmente, muitos recursos podem ser utilizados no aprendizado, desde os mais tradicionais como a sala de aula, até os de última geração como as mídias em geral (televisão, rádio, hipertexto, etc.).

Nos dois próximos tópicos será mostrado um pouco o papel do computador e da Internet, mais precisamente a *web*, no aprendizado.

2.2 Aprendizado via computador

A utilização de tecnologia na transmissão de conhecimento vem de longa data. Na Segunda Grande Guerra, com soldados espalhados por diferentes partes do mundo, os Estados Unidos (exército e instituições de ensino) utilizaram bastante o filme de treinamento, para treiná-los em atividades como higiene pessoal, primeiros socorros, manutenção de armamento, entre outras atividades. Com os sucessos obtidos, não demorou muito para que, naquela época, a televisão fosse vista como próximo instrumento de aprendizado [Rosenberg2001].

Na década de 1980, o advento do computador pessoal trouxe enormes possibilidades para o aprendizado. Os desenvolvedores de treinamentos e cursos a distância possuíam na época um novo e valioso instrumento para realizarem seu trabalho. Entretanto, limitações tecnológicas como diferenças de plataformas, limitações de *hardware*, entre outras, dificultavam a implementação de bons programas de aprendizado via computador e os que eram implementados geralmente estavam fadados ao insucesso [Rosenberg2001].

A evolução das tecnologias computacionais alcançadas na década de 1990, principalmente o aumento da capacidade de processamento e armazenamento dos computadores pessoais permitiu que fossem criados programas de aprendizado com mais recursos gráficos e mais funcionalidades.

Aliada à evolução tecnológica da década de 1990 houve a popularização da Internet e tecnologias relacionadas, abrindo um novo leque de possibilidades para o aprendizado baseado em computador. No próximo tópico é apresentado um pouco do que foi realizado.

2.3 Aprendizado via *web*

Uma das tecnologias Internet mais interessantes é a *World Wide Web* (WWW ou *Web*). A *web* é um repositório *on-line* de informações em larga escala que os usuários podem visualizar usando um programa aplicativo interativo chamado navegador (*browser*). Tecnicamente é um sistema hipermídia distribuído no qual as informações são armazenadas em um conjunto de documentos (chamados de páginas) onde cada elemento pode possuir ponteiros (*links*) para os outros. A forma básica de representação das páginas é a HTML (*Hypertext Markup Language*) [Comer2001].

Atualmente a quantidade de informação disponível na *web* é imensa. Ela tornou-se uma das principais fontes de aquisição de conhecimento para todos. O aprendizado pode ser buscado e repassado de várias formas. Há milhares de locais que armazenam informações e sistemas de busca que facilitam o acesso a eles.

Muitas empresas e instituições de ensino atualmente utilizam as suas Intranets ou a Internet como meio para treinar seus empregados ou compartilhar informações estratégicas entre os mesmos e ensinar os seus alunos, respectivamente. A utilização deste recurso para o aprendizado tem sido chamado de *e-learning* [Rosemberg2001]. Para as empresas a Gestão de Conhecimento (GCO) utilizando o *e-learning* é um fator que proporciona o aumento da qualidade dos produtos e serviços prestados. Uma empresa que possui empregados com elevado grau de conhecimento e que fornece meios para estes empregados manterem-se atualizados, certamente obterá os melhores resultados em seus negócios. Para as instituições de ensino a Educação a Distância (EAD) é uma forma de oferecer para seus alunos mais conteúdo didático, oferecer mais vagas para a população e incentivar a pesquisa, entre outras vantagens. Algumas universidades já oferecem disciplinas utilizando *e-learning* para seus alunos. Outras, no exterior, oferecem cursos de graduação e pós-graduação parciais ou completos via *web*.

A seguir apresentam-se alguns trabalhos realizados no campo do aprendizado via *web*, objetivando demonstrar o estado da arte do desenvolvimento nesta área.

2.4 Estado da arte do aprendizado via *web*

Os artigos citados abaixo, que foram apresentados em um congresso nacional da área de educação a distância, ilustram bem o estado da arte do aprendizado via *web*:

1) **A influência da plataforma de gerenciamento de EAD no desenvolvimento de programas de ensino via *web*: uma experiência do LATEC/UFRJ**: onde é avaliada a contribuição de uma ferramenta de gerenciamento de cursos on-line para a melhoria da produção e da implantação de cursos via *web* [PHR2002];
2) **Dinamizando as práticas pedagógicas através de um ambiente baseado na *web***: onde são apresentados resultados de experimentos realizados com a utilização de um ambiente de ensino, o AulaNet, em conjunto aplicação de uma Metodologia Construtivista para o processo de ensino-aprendizagem [FF2002];

3) **Maestro: uma ferramenta de planejamento e desenvolvimento de conteúdos em formato hipertexto para *e-learning***: onde também é apresentada uma ferramenta gerenciadora de cursos *on-line*, que visa facilitar para professores e especialistas em conteúdos, mesmo que sejam leigos em informática, o desenvolvimento de conteúdos hipertextuais para *e-learning* [BKL2002];
4) **Sistema de educação a distância baseada em serviços *web***: onde é apresentada uma proposta para as instituições de ensino oferecerem serviços de EAD (cursos, treinamentos, etc) para as empresas em geral que desejam criar um ambiente de aprendizagem permanente para seu pessoal, através da utilização de serviços *web* [AF2002].

As referências citadas acima, dentre muitas outras que podem ser facilmente encontradas em livros e na Internet, oferecem uma idéia das ferramentas e ambientes que estão sendo criados para o aprendizado via *web*. Além disto, há bastante pesquisa relacionada com a parte visual dos conteúdos educativos que são criados, como pode ser visto em [Romiszowski2002] e [Stefanelli2002].

2.5Proposta

Após o pequeno histórico sobre o aprendizado e a aplicação de recursos computacionais no mesmo e também após a ilustração do estado da arte do aprendizado via *web*, são levantadas aqui algumas questões sobre uma ferramenta de autoria de conteúdo para aprendizado via *web*.

Como foi apresentado na pesquisa sobre o estado da arte, existem muitos sistemas e/ou ambientes gerenciadores de cursos e de conteúdo para aprendizado via *web*. Entretanto, verificou-se que os conteúdos muitas vezes apresentam as seguintes características:

- **Dependentes da ferramenta que os gerou**: isto significa que os conteúdos gerados, em sua maioria, só podem ser modificados pela própria ferramenta. As modificações realizadas diretamente no conteúdo gerado dificilmente podem ser reaproveitadas pela ferramenta original;
- **Dependentes da plataforma para a qual foram concebidos**: isto significa que a geração e/ou apresentação dos conteúdos possuem características específicas, que as limitam a funcionarem em um determinado sistema operacional, banco de dados ou navegador;

- **Dependentes do formato de apresentação**: bastante relacionada com a característica anterior, esta significa que o conteúdo é gerado exclusivamente para um modelo único de apresentação, para uma única plataforma, sem levar em consideração, por exemplo, as oportunidades de apresentação em outros dispositivos como telefones celulares e computadores portáteis.

Desta forma, propõe-se o desenvolvimento de uma ferramenta computacional de autoria de conteúdo para aprendizado via *web*, cujos conteúdos gerados não contenham as características citadas anteriormente, ou seja, eles devem ser: gerados em uma tecnologia que permita modificações por outras ferramentas, independentes de plataforma, podendo ser executados em diferentes sistemas operacionais, bancos de dados ou navegadores e passíveis de adaptação para outras formas de apresentação.

Acredita-se que a utilização da linguagem XML (*Extensible Markup Language*) para a definição dos conteúdos garantirá o alcance dos objetivos citados no parágrafo anterior. A XML é uma linguagem de marcação projetada para a definição de dados na qual os elementos são criados de acordo com a necessidade [W3Schools2002].

Como a ferramenta será desenvolvida, nada mais natural que utilizar os conceitos da Engenharia de Software, principalmente um processo de desenvolvimento iterativo e incremental [Reed2000] e técnicas de modelagem como a UML (*Unified Modeling Language*) [Larman2000], para nortear todo o trabalho a ser realizado.

O capítulo seguinte apresenta o planejamento do desenvolvimento da ferramenta, definindo o processo que será utilizado, suas fases e atividades, além de iniciar os trabalhos com a execução da primeira fase.

CAPÍTULO 3

PLANEJAMENTO E FASE DE INÍCIO

Neste capítulo dá-se início ao desenvolvimento da ferramenta de autoria de conteúdo para aprendizado via *web*. Primeiramente, será apresentado o planejamento do desenvolvimento, com destaque para o processo de *software* que será utilizado, que indicará todas as fases e atividades que serão realizadas. Logo em seguida serão executadas a primeira fase do processo e suas atividades.

3.1 Planejamento

Para organizar todo o trabalho de desenvolvimento que será realizado é necessário que se lance mão de um processo de desenvolvimento (ou de *software*) que deverá indicar os rumos a serem tomados.

Atualmente as grandes organizações que desenvolvem *software* têm utilizado bastante os processos iterativos e incrementais. São chamados assim, pois o *software* a ser desenvolvido é dividido em subconjuntos de funcionalidade. Cada um destes subconjuntos é desenvolvido, seguindo todas as fases e atividades previstas no processo, o que caracteriza uma iteração. Em seguida outro subconjunto é desenvolvido da mesma forma, caracterizando um incremento, até que todas as funcionalidades tenham sido desenvolvidas. Um bom exemplo deste tipo de processo é o *Rational Unified Process* (RUP), que define, baseado nas melhores práticas da Engenharia de Software, um conjunto de fases, disciplinas e papéis para serem utilizados em projetos de desenvolvimento orientados a objetos [Rational2001]. Um outro processo iterativo e incremental, chamado *Synergy*, é definido e utilizado no trabalho de Paul Reed [Reed2000].

No desenvolvimento da ferramenta de autoria de conteúdo para aprendizado via *web*, serão utilizadas as idéias básicas dos processos citados acima, para definir as fases e atividades que nortearão este trabalho, ou seja, seu processo de desenvolvimento. Eles não foram utilizados por possuírem inúmeras atividades para serem realizadas o que tornaria oneroso o trabalho de desenvolvimento do protótipo. Desta forma, definiu-se um processo com as seguintes fases: Início, Elaboração e Construção. As atividades que serão realizadas em cada um destas fases são mostradas a seguir:

- **Início**
  - **Especificação:** onde é criada uma descrição textual de todas as funcionalidades previstas para a ferramenta;
  - **Diagrama e descrição dos casos de uso:** onde as funcionalidades descritas na especificação não mapeadas para um artefato da UML, o Diagrama de Casos de Uso, e descritas de acordo com um modelo;
  - **Estimativa:** utilizando-se de uma métrica que leva em consideração os casos de uso é estimado o esforço necessário para se desenvolver a ferramenta;
  - **Plano de iterações:** são os subconjuntos de funcionalidades, neste caso, casos de uso, que definirão a ordem de desenvolvimento dos mesmos;

- **Elaboração**
  - **Análise:** onde é criado o modelo conceitual para a solução, com base nas classes de objetos que permeiam o domínio do problema. Este modelo é apresentado utilizando um artefato UML, o Diagrama de Classes;
  - **Projeto:** onde é definida uma solução de implementação baseando-se no modelo conceitual. São criados diagramas de classes com detalhes mais específicos de implementação, como as classes que farão a frente com os usuários da ferramenta e as que serão utilizadas para fazer o acesso ao banco de dados, além da definição dos serviços (métodos) para as classes da análise;

- **Construção**
  - **Modelo de dados:** onde, com base na análise e no projeto, cria-se um modelo de dados que servirá como base para a implementação física do banco de dados que será utilizado na ferramenta;
  - **Implementação:** onde todas as classes previstas são construídas utilizando-se uma linguagem de programação;
  - **Testes:** onde as funcionalidades previstas serão validadas após as suas implementações, utilizando a especificação e os casos de uso.

Seguindo o raciocínio dos processos iterativos e incrementais, cada uma destas fases e atividades deve ser revista em cada iteração. Desta forma, cada documento ou artefato que

tenha sido criado deve ser atualizado com os incrementos de funcionalidades e/ou transformações que tenham acontecido durante a realização de cada iteração.

Dependendo do tamanho do projeto a realização das iterações pode variar. Por exemplo, se os requisitos forem estáveis, não há a necessidade de se revisar completamente a fase de Início. Para o desenvolvimento da ferramenta, na definição do plano de iterações será indicado o estilo que será adotado em seu desenvolvimento.

No tópico a seguir inicia-se o desenvolvimento da ferramenta utilizando o processo definido. A primeira atividade da fase de Início, a Especificação será apresentada.

3.2 Especificação

Atualmente existe uma infinidade de tecnologias para serem utilizadas na publicação de conteúdo para a *web*. Muitas decisões devem ser tomadas pelos desenvolvedores de conteúdo, o que requer bastante conhecimento dessas tecnologias por parte destes. Além disto, como foi apresentado no capítulo anterior, os conteúdos a serem criados devem possuir independência tanto de plataforma quanto de apresentação. A ferramenta de autoria de conteúdo para aprendizado via *web*, deve ocultar de seus usuários finais tais decisões e tecnologias envolvidas na elaboração de um conteúdo.

Para atender as necessidades citadas para a ferramenta e prever a utilização da mesma, as seguintes funcionalidades devem ser oferecidas:

1) Gerenciar vários conteúdos, ou seja, vários conjuntos de informação sobre determinados temas;
2) Manipulação dos dados de conteúdos, ou seja, cadastros, alterações e consultas;
3) Importação de conteúdos através de arquivos XML;
4) Geração, em arquivos XML, do conteúdo armazenado;
5) Geração, em arquivo XML, de um índice para todo o conteúdo gerado;
6) Utilizar páginas de apresentação para os conteúdos que permitam a total separação entre estas partes;
7) Utilização de uma estrutura básica de *site* de publicação de conteúdo;
8) Permitir a seleção de modelos (*templates*) para o *site* a ser utilizado.

O conteúdo a ser gerado deverá estar dividido em um formato básico, de forma a facilitar a sua geração pela ferramenta. O formato básico definido é o seguinte:

- Cada conteúdo deverá ser composto por diversos itens (títulos, subtítulos, capítulos, tópicos, etc);
- Cada item será uma página, um arquivo XML, que será gerado pela ferramenta e deverá ser formado por diversos componentes;
- Cada componente de um item poderá ser um texto, uma imagem ou um ponto de acesso (*link*) para outro item ou página da *web*.

No tópico a seguir, esta especificação textual será transformada em um diagrama de casos de uso, artefato que será utilizado para organizar todo o desenvolvimento da ferramenta.

3.3 Casos de uso

Um caso de uso é um documento narrativo que descreve a seqüência de eventos de um ator (um agente externo) que usa um sistema para completar um processo [Larman2000]. Um caso de uso é um artefato da UML que focaliza o quê o sistema deverá fazer, não como ele o fará [Reed2000].

Após a especificação textual das funcionalidades que um sistema deve possuir, seus casos de uso podem ser levantados. As referências citadas no parágrafo anterior apresentam formas de realizar este levantamento. Baseando-se na especificação da ferramenta de autoria de conteúdo para aprendizado via *web*, seus casos de uso foram identificados e seu diagrama é mostrado na figura 3.1. Cada funcionalidade descrita na especificação da ferramenta foi mapeada para algum dos casos de uso.

Para melhor documentar e facilitar o entendimento dos casos de uso será utilizado um modelo de descrição de casos de uso, baseado no trabalho de Paul Reed [Reed2000], através do qual cada caso de uso contido no diagrama será descrito. Completando o modelo, será acrescentado um item (funcionalidade atendida) que indicará para onde cada funcionalidade citada na especificação foi mapeada.

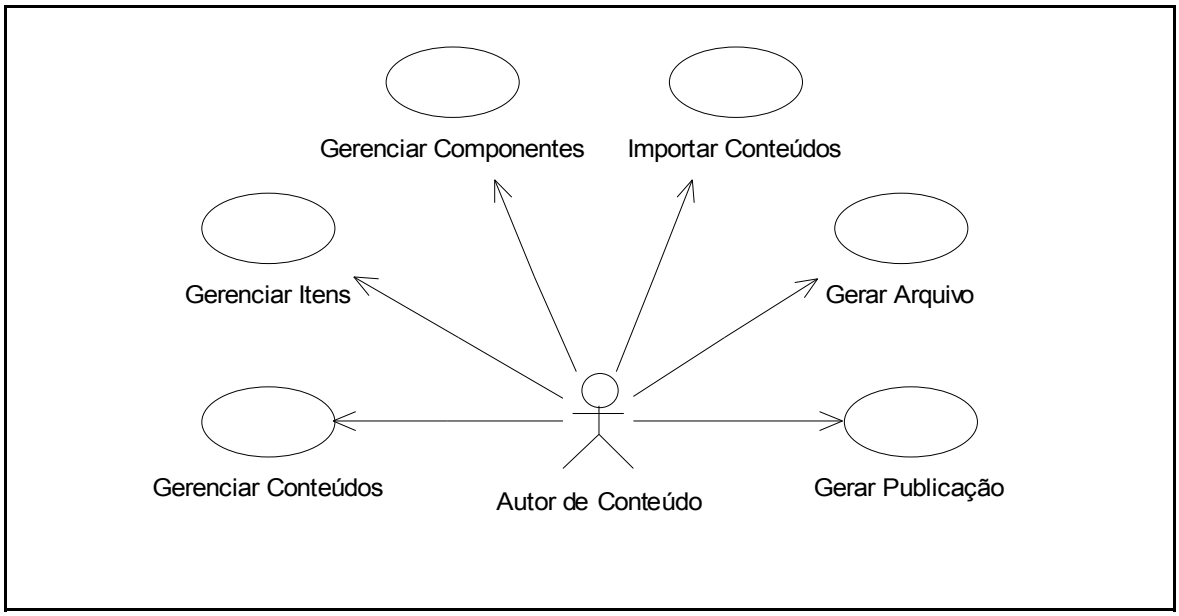

Fig. 3.1 – Diagrama de casos de uso da ferramenta

1. Gerenciar Conteúdos

- **Nome:** gerenciar conteúdos.
- **Objetivo:** gerenciar as funções básicas (cadastrar, listar, alterar, abrir e excluir) de manipulação dos dados dos conteúdos.
- **Ator:** autor de conteúdo.
- **Pré-condições:** nenhum conteúdo aberto.
- **Pós-condições:** função básica executada.
- **Fluxo primário:**
  - **Cadastrar:** informar os dados de identificação conteúdo. Gravar os dados na base.
- **Fluxos alternativos:**
  - **Listar:** apresentar os dados de identificação dos conteúdos cadastrados em uma lista. Permitir a seleção.
  - **Alterar:** selecionar um conteúdo na lista. Alterar os dados de identificação. Gravar novamente.
  - **Abrir:** selecionar um conteúdo na lista. Abrir o conteúdo (o que permitirá acesso ao caso de uso Gerar Item).
  - **Excluir:** selecionar um conteúdo na lista. Perguntar ao Ator se deseja realmente excluí-lo. Excluir o conteúdo em caso de resposta positiva.

- **Fluxos de exceção:**
  - **Interromper Cadastrar:** não permitir a operação e alertar o Ator ao se tentar gravar um conteúdo com informações obrigatórias não preenchidas.
  - **Interromper Alterar:** não permitir a operação e alertar o Ator ao se tentar gravar um conteúdo com informações obrigatórias não preenchidas.
  - **Interromper Excluir:** não permitir a operação caso o Ator opte por não excluir o conteúdo selecionado quando perguntado.
- **Requisito especial:** interface gráfica de fácil utilização.
- **Funcionalidade atendida:** 1.

2. Gerenciar Itens

- **Nome:** gerenciar itens.
- **Objetivo:** gerenciar as funções básicas (cadastrar, listar, alterar, abrir e excluir) de manipulação dos dados dos itens de um determinado conteúdo.
- **Ator:** autor de conteúdo.
- **Pré-condições:** o conteúdo deve estar aberto.
- **Pós-condições:** função básica executada.
- **Fluxo primário:**
  - **Cadastrar:** informar os dados do item de conteúdo. Gravar os dados na base.
- **Fluxos alternativos:**
  - **Listar:** apresentar os dados de identificação dos itens cadastrados em uma lista. Permitir a seleção.
  - **Alterar:** selecionar um item na lista. Alterar os dados de identificação. Gravar novamente.
  - **Abrir:** selecionar um item na lista. Abrir o Item (o que permitirá acesso ao caso de uso 3).
  - **Excluir:** selecionar um item na lista. Perguntar ao Ator se deseja realmente excluí-lo. Excluir o item em caso de resposta positiva.
- **Fluxos de exceção:**
  - **Interromper Cadastrar:** não permitir a operação e alertar o Ator ao se tentar gravar um item com informações obrigatórias não preenchidas.

  - **Interromper Alterar:** não permitir a operação e alertar o Ator ao se tentar gravar um item com informações obrigatórias não preenchidas.
  - **Interromper Excluir:** não permitir a operação caso o Ator opte por não excluir o item selecionado quando perguntado.
- **Requisito especial:** interface gráfica de fácil utilização.
- **Funcionalidade atendida:** 2.

3. Gerenciar Componentes

- **Nome:** gerenciar componentes.
- **Objetivo:** gerenciar as funções básicas (cadastrar, listar, alterar, e excluir) de manipulação dos dados dos componentes de um determinado item.
- **Ator:** autor de conteúdo.
- **Pré-condições:** o item deve estar aberto.
- **Pós-condições:** função básica executada.
- **Fluxo primário:**
  - **Cadastrar:** informar os dados do componente de item. Gravar os dados na base.
- **Fluxos alternativos:**
  - **Listar:** apresentar os dados de identificação dos componentes cadastrados em uma lista. Permitir a seleção.
  - **Alterar:** selecionar um componente na lista. Alterar os dados de identificação. Gravar novamente.
  - **Excluir:** selecionar um componente na lista. Perguntar ao Ator se deseja realmente excluí-lo. Excluir o componente em caso de resposta positiva.
- **Fluxos de exceção:**
  - **Interromper Cadastrar:** não permitir a operação e alertar o Ator ao se tentar gravar um componente com informações obrigatórias não preenchidas.
  - **Interromper Alterar:** não permitir a operação e alertar o Ator ao se tentar gravar um componente com informações obrigatórias não preenchidas.
  - **Interromper Excluir:** não permitir a operação caso o Ator opte por não excluir o componente selecionado quando perguntado.
- **Requisito especial:** interface gráfica de fácil utilização.
- **Funcionalidade atendida:** 2.

4. **Importar Conteúdo**

- **Nome:** importar conteúdo.
- **Objetivo:** realizar a importação de um determinado conteúdo a partir de arquivo.
- **Ator:** autor de conteúdo.
- **Pré-condições:** nenhum conteúdo aberto.
- **Pós-condições:** conteúdo importado.
- **Fluxo primário:**
  - **Importar:** selecionar o arquivo a ser importado. Importar as informações para a base.
- **Fluxo de exceção:**
  - **Interromper Importar:** não permitir operação e alertar o Ator se o arquivo for inválido.
- **Requisito especial:** interface gráfica de fácil utilização.
- **Funcionalidade atendida:** 3.

5. **Gerar Arquivo**

- **Nome:** gerar arquivo.
- **Objetivo:** gerar o arquivo XML com as informações de um determinado item.
- **Ator:** autor de conteúdo.
- **Pré-condições:** informações obrigatórias do item gravadas na base.
- **Pós-condições:** arquivo do item gerado.
- **Fluxo primário:**
  - **Gerar:** selecionar um item cadastrado. Gerar páginas.
- **Fluxo de exceção:**
  - **Interromper Gerar:** não permitir operação e alertar o Ator ao se tentar gerar o arquivo de um item que não contenha componente cadastrado.
- **Requisito especial:** interface gráfica de fácil utilização.
- **Funcionalidade atendida:** 4.

6. Gerar Publicação

- **Nome:** gerar publicação.
- **Objetivo:** geração do *site* de publicação do conteúdo.
- **Ator:** autor de conteúdo.
- **Pré-condições:** informações obrigatórias do conteúdo, itens e componentes gravadas na base.
- **Pós-condições:** *site* de publicação gerado.
- **Fluxo primário:**
  - **Gerar:** selecionar o conteúdo a ser publicado. Copiar o modelo selecionado (*template*). Gerar os arquivos dos itens. Gerar o arquivo de índice.
- **Fluxos de exceção:**
  - **Interromper Gerar 1:** não permitir a operação caso não haja itens cadastrados para o conteúdo selecionado.
  - **Interromper Gerar 2:** não permitir a operação caso não haja componente cadastrado para algum item do conteúdo selecionado e o Ator opta, quando informado, por não gerar a publicação.
- **Requisito especial:** interface gráfica de fácil utilização. Modelos (*templates*) criados e disponíveis para a ferramenta.
- **Funcionalidade atendida:** 5, 6, 7 e 8.

De posse do diagrama e das descrições dos casos de uso, pode-se estimar o esforço necessário para desenvolver a ferramenta de autoria de conteúdo para aprendizado via *web*, o que será apresentado no tópico a seguir.

3.4Estimativa

Para realizar a estimativa de prazo do desenvolvimento da ferramenta geradora de conteúdo, levou-se em consideração a métrica *Use Case Points* (Pontos de Casos de Uso). Esta métrica baseia-se em avaliações da complexidade dos atores, casos de uso, características técnicas e ambientais do projeto. O trabalho de Paul Reed [Reed2000] também oferece uma boa referência sobre esta métrica de software elaborada por Gustav Karner.

O resultado obtido com a aplicação da métrica é que a ferramenta pode ser desenvolvida em aproximadamente 426 horas. Levando em conta que a implementação será realizada por apenas um desenvolvedor, utilizando 6 horas por dia, tem-se um prazo de mais ou menos 3,5 meses para este trabalho.

No tópico seguinte, apresenta-se o plano de iterações da ferramenta.

3.5 Plano de iterações

O plano de iterações define a seqüência em que os casos de uso do sistema em questão serão desenvolvidos. Deve-se atentar para o fato de que é interessante atacar inicialmente aqueles casos de uso que oferecem mais risco para o desenvolvimento, pois podem causar eventuais atrasos no cronograma. Outro fato importante é que ao final de cada iteração pode haver uma avaliação de tudo que foi feito até o momento, onde é primordial que funcionalidades relevantes para a aplicação tenham sido implementadas, o que permitirá a realização de bons testes [Rational2001].

Como foi apresentado, para a ferramenta autoria de conteúdo para aprendizado via *web* foram identificados 6 casos de uso. Destes, os casos “Gerar Arquivo” e “Gerar Publicação” podem ser considerados os de maior risco, pois envolvem a criação de arquivos XML e a utilização de modelos pré-definidos de páginas para a apresentação, respectivamente. Entretanto, para se desenvolver a funcionalidade definida no caso de uso “Gerar Arquivo”, os casos de uso “Gerenciar Conteúdos”, “Gerenciar Itens” e “Gerenciar Componentes” devem ter sido desenvolvidos antes, como pode ser observado no item de pré-condições do caso de uso “Gerar Arquivo”. O caso de uso “Gerar Publicação” exige que os 4 já citados tenham sido desenvolvidos para que as funcionalidades do mesmo possam ser utilizadas. O caso de uso “Importar Conteúdo” não possui nenhuma implicação direta com a realização ou não dos outros. Com base nestas considerações, pode-se definir o plano de iterações, apresentado logo abaixo na Tabela 3.1, para o desenvolvimento da ferramenta.

Iteração	Casos de uso
1	Gerenciar Conteúdos, Gerenciar Itens, Gerenciar Componentes
2	Gerar Páginas, Gerar Publicação
3	Importar

Tabela 3.1 – Plano de iterações da ferramenta

Baseado na estimativa e de acordo com o plano de iterações, uma proposta de cronograma é apresentada logo a seguir na Tabela 3.2.

Iteração	Prazo
1	1 mês e meio
2	1 mês e meio
3	Metade de um mês

Tabela 3.2 – Proposta de cronograma

Acredita-se que os requisitos levantados na especificação da ferramenta de autoria de conteúdo para aprendizado via *web* sejam estáveis, portanto não haverá a necessidade de se rever a fase inicial em todas as iterações, a menos que haja alteração em alguma funcionalidade ou casos de uso. As demais fases e atividades deverão ser revistas e seus documentos e artefatos atualizadas no decorrer das iterações.

No capítulo seguinte serão apresentados os resultados da fase de elaboração através dos produtos (artefatos) finais gerados nas atividades de análise e projeto.

CAPÍTULO 4

FASE DE ELABORAÇÃO

Continuando o desenvolvimento da ferramenta de autoria de conteúdo para aprendizado via *web*, será apresentada neste capítulo a fases de elaboração e suas respectivas atividades, a saber: análise e projeto. Para cada atividade será mostrado o resultado final de sua execução, por exemplo, para a atividade de análise será mostrado o seu diagrama de classes final, resultante da realização de todas a iterações previstas no plano.

4.1 Análise

A análise enfatiza uma investigação do problema, de como uma solução pode ser definida. Na análise orientada a objetos esta ênfase é dada à descoberta e à descrição dos objetos (ou conceitos) do domínio do problema [Larman2000].

A definição citada anteriormente indica exatamente o que será realizado nesta atividade, cujo objetivo principal, segundo o processo definido, é a construção de um modelo conceitual para a ferramenta. Baseando-se na especificação e no modelo de casos de uso da ferramenta, pôde-se criar, utilizando um diagrama de classes da UML, o seguinte modelo, mostrado na figura 4.1.

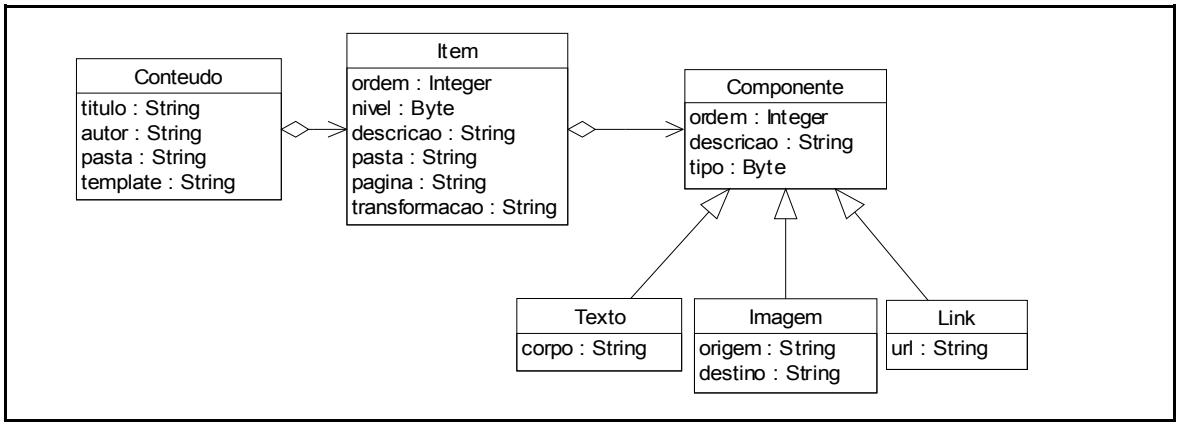

Fig. 4.1 – Modelo conceitual da ferramenta.

Este modelo mapeia diretamente as características do conteúdo citadas em sua especificação e, devido à orientação a objetos, já pode ser percebida uma possível solução para a sua construção. Na atividade de projeto isto será evidenciado.

A seguir são apresentadas as características de cada classe identificada:

Classe Conteudo

- Representa o conteúdo que será manipulado pela ferramenta;
- Associada a classe Item, através de uma agregação, o que indica que é composta por diversos objetos daquela;
- Possui os seguintes atributos: titulo, autor, pasta e template.

Classe Item

- Representa um item de conteúdo que será manipulado pela ferramenta;
- Associada, através de agregação com as classes Conteudo e Componente, sendo que compõe a primeira e é composta pelos objetos da segunda;
- Possui os seguintes atributos: ordem, nivel, descricao, pasta, pagina e transformacao.

Classe Componente

- Representa um componente de um item de conteúdo que será manipulado pela ferramenta;
- Associada, através de agregação, com a classe Item, de forma que seus objetos compõem esta. Associada com as classes Texto, Imagem e Link através de uma generalização, sendo a superclasse destas;
- Possui os seguintes atributos: ordem, descricao e tipo.

Classe Texto

- Representa um componente do tipo texto de um item de conteúdo que será manipulado pela ferramenta;
- Especialização da classe Componente, sendo sua herdeira;

- Possui os seguintes atributos: corpo, além dos de sua superclasse.

Classe Imagem

- Representa um componente do tipo imagem de um item de conteúdo que será manipulado pela ferramenta;
- Especialização da classe Componente, sendo sua herdeira;
- Possui os seguintes atributos: origem e destino, além dos de sua superclasse.

Classe Link

- Representa um componente do tipo *link* (ponto de acesso para algum item do conteúdo ou página da *web*) de um item de conteúdo que será manipulado pela ferramenta;
- Especialização da classe Componente, sendo sua herdeira;
- Possui os seguintes atributos: url, além dos de sua superclasse.

No tópico seguinte será apresentada a atividade de projeto, onde detalhes mais específicos, que levam a solução lógica do desenvolvimento mais para perto de sua implementação, são introduzidos.

4.2 Projeto

A fase de projeto enfatiza uma solução lógica para o problema. No projeto orientado a objetos existe uma ênfase na definição de elementos lógicos de *software*, os quais, em última instância, serão implementados em uma linguagem de programação orientada a objetos [Larman2000]. Tais elementos são aqui definidos e agrupados de acordo com as suas responsabilidades.

O modelo conceitual concentra as responsabilidades do negócio de um sistema sendo, portanto, o principal componente de uma solução lógica. Entretanto, há outras responsabilidades que devem ser levantadas para um projeto de *software* ser completo. Classes que definem a frente (interface ou apresentação) através da qual os usuários utilizarão o sistema e as classes que definem o acesso as funcionalidades de banco de dados devem

também ser contempladas. Desta forma, o projeto pode ser dividido em camadas de serviços, cada uma com a sua responsabilidade específica.

O projeto da ferramenta de autoria de conteúdo para aprendizado via *web*, foi divido em três camadas de serviço: apresentação, negócio e acesso a dados. As classes da camada de apresentação representam os elementos através dos quais os usuários interagem com o sistema, informando ou recebendo dados. As classes da camada de negócio representam as entidades do domínio do problema e suas relações. As classes da camada de dados servem para que as informações processadas nas duas camadas anteriores possam ser armazenadas em um banco de dados.

A seguir são apresentados os diagramas de classes deste projeto, separados de acordo com as camadas de serviço a que pertencem. Para cada camada de serviço será apresentado o seu modelo e para cada classe haverá uma descrição, em linhas gerais, de suas responsabilidades.

Camada de apresentação

As classes desta camada estão relacionadas através de associações simples, como pode ser visto no diagrama apresentado na figura 4.2 abaixo. Estas associações existem, pois a classe FrmPrincipal pode instanciar os objetos da classe FrmConteudo que por sua vez pode instanciar os objetos da classe FrmItem que, enfim, pode instanciar os objetos da classe FrmComponente.

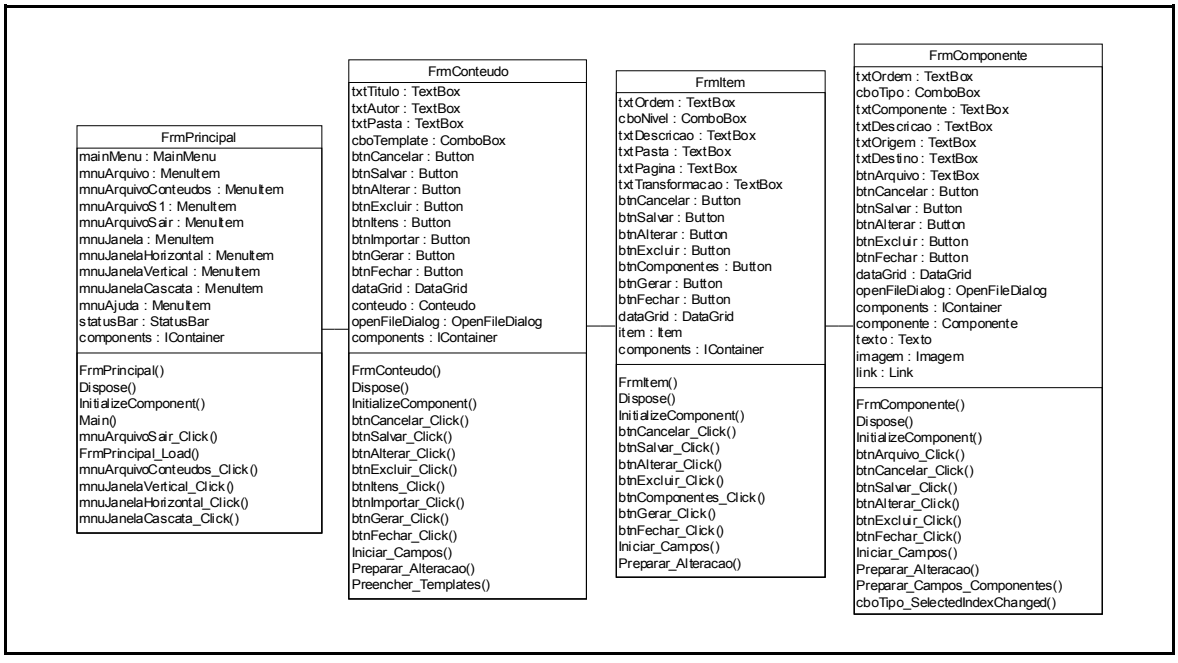

Fig. 4.2 – Diagrama de classes da camada de apresentação.

Classe FrmPrincipal

Representa o formulário principal da aplicação. A partir dele o formulário de conteúdo (FrmConteudo) é iniciado, sendo esta a principal atividade desta classe. As demais atividades são:

- Configurações de apresentação (posição e tamanho da tela);
- Posicionamento das janelas abertas;
- Instanciar o formulário de conteúdo (FrmConteudo);
- Acesso ao *Help* da ferramenta;
- Encerramento da aplicação.

Classe FrmConteudo

Por meio do formulário que esta classe representa o usuário da aplicação tem acesso as funcionalidades de manipulação geral do conteúdo, como:

- Manutenção (cadastro, alteração e exclusão) de seus dados iniciais (título, autor, pasta e *template*);
- Abrir o formulário de itens (FrmItem);
- Importação de conteúdo a partir de um arquivo XML;
- Gerar a publicação do conteúdo.

Para realizar suas atividades, esta classe utiliza os serviços da classe Conteúdo da camada de negócio.

Em especial, o método Preencher_Templates utiliza um arquivo (templates.xml, localizado no mesmo diretório da aplicação) para preencher o componente através do qual o usuário selecionará o *template* de publicação do conteúdo.

Classe FrmItem

O formulário desta classe, que foi instanciado pelo formulário de conteúdo é o responsável por oferecer ao usuário da aplicação os serviços relacionados aos itens que compõem um conteúdo. Dentre estes serviços, destacam-se:

- Manutenção (cadastro, alteração e exclusão) dos dados de um determinado item (ordem, nível, descrição, pasta, pagina e transformação);
- Abrir o formulário de componentes (FrmComponente);
- Geração do arquivo XML de um determinado item de conteúdo.

Para realizar suas atividades, esta classe utiliza os serviços da classe Item da camada de negócio.

Classe FrmComponente

Neste formulário o usuário realiza a manipulação dos componentes de um determinado item de conteúdo. Cada componente pode ser do tipo texto, imagem ou link. Os dados do componente variam de acordo com seu tipo.

Os serviços disponíveis são os de manutenção (cadastro, alteração e exclusão) dos dados. Os dados para cada componente são:

- Texto: ordem, tipo, descrição e corpo;
- Imagem: ordem, tipo, descrição, origem e destino;
- Link: ordem, tipo, descrição e URL.

Para realizar suas atividades, esta classe utiliza os serviço da classe Componente e de suas derivadas (Texto, Imagem e Link) da camada de negócio.

Camada de negócio

As classes desta camada possuem associações mais complexas se comparadas com as classes da camada anterior, como pode ser visto no diagrama apresentado na figura 4.3 abaixo. Neste diagrama, a associação entre as classes Conteudo, Item e Componente é de composição. A classe componente possui ainda três especializações que são as classes Texto, Imagem e Link. Existem ainda associações entre todas as classes que não são apresentadas no modelo para efeito de apresentação, mas são descritas na apresentação de cada classe.

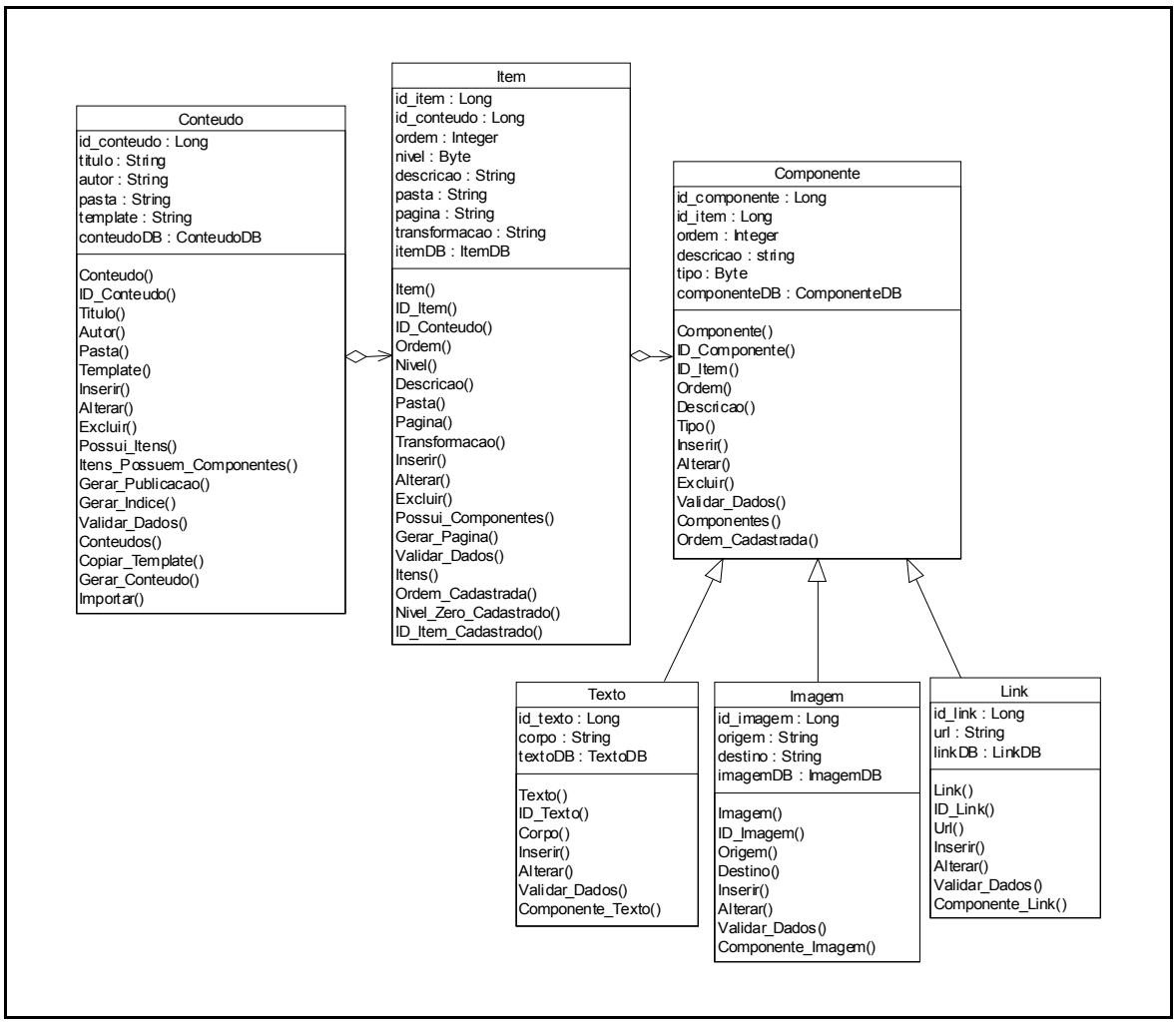

Fig. 4.3 – Diagrama de classes da camada de negócio.

Classe Conteudo

A classe Conteudo realiza os serviços oferecidos ao usuário pela classe FrmConteudo. Tais serviços são:

- Propriedades para cada um dos atributos (título, autor, pasta e *template*);
- Cadastro (método Inserir), alteração (método Alterar) e exclusão (método Excluir);
- Validação dos dados (método Validar_Dados);
- Verificar a existência de itens cadastrados para o conteúdo corrente (método Possui_Itens);
- Verificar a existência de componentes para cada item do conteúdo corrente (método Itens_Possuem_Componentes);
- Retornar um objeto da classe DataView com as informações de todos os conteúdos armazenados na base de dados (método Conteudos);
- Realizar a importação a partir de um arquivo XML (método Importar);
- Gerar a publicação do conteúdo (métodos Gerar_Publicacao, Copiar_Template, Gerar_Indice e Gerar_Conteudo, que gera as páginas de cada item).

Existe uma forte dependência entre a classe Conteudo e a classe Item, uma vez que esta é utilizada pelos seguintes métodos da primeira: Possui_Itens, Itens_Possuem_Componentes, Gerar_Indice, Gerar_Conteudo e Importar. Além desta dependência, o método Importar faz com que a classe Conteudo seja dependente também das classes Texto, Imagem e Link, pois utiliza os serviços destas para realizar a sua atividade.

A importação realizada pelo método Importar é feita a partir de um arquivo XML cuja origem é informada pelo usuário no formulário de conteúdo. Se o arquivo informado não for válido é gerada uma exceção.

Os métodos da classe Conteudo que necessitem de acesso ao banco de dados da aplicação utilizam os serviços de um objeto da classe ConteudoDB da camada de dados, que é membro da classe Conteudo, criando também uma forte dependência entre as duas.

Classe Item

A classe Item realiza os serviços oferecidos ao usuário pela classe FrmItem. Tais serviços são:

- Propriedades para cada um dos atributos (id_item, id_conteudo, ordem, nivel, descricao, pasta, pagina e transformacao);
- Cadastro (método Inserir), alteração (método Alterar) e exclusão (método Excluir). O método Inserir, em especial, é sobrecarregado possuindo uma implementação com validação dos dados e outra sem;
- Validação dos dados (método Validar_Dados);
- Verificar a existência de componentes cadastrados para o item corrente (método Possui_Componentes);
- Retornar um objeto da classe DataView com as informações de todos os itens armazenados na base de dados para um determinado conteúdo (método Itens);
- Verificar se a ordem dada a um item já esta cadastrada (método Ordem_Cadastrada);
- Verificar se existe um item de nível zero já cadastrado na base de dados (método Nível_Zero_Cadastrado);
- Retornar o ID de um item recém cadastrado na base (método ID_Item_Cadastrado);
- Gerar o arquivo XML correspondente a um item do conteúdo (método Gerar_Pagina).

Os métodos Gerar_Pagina e Possui_Componentes evidenciam a dependência da classe Item com as classes Conteúdo, Componente, Texto, Imagem e Link.

Os métodos da classe Item que necessitem de acesso ao banco de dados da aplicação utilizam os serviços de um objeto da classe ItemDB da camada de dados, que é membro da classe Item, criando também uma forte dependência entre as duas.

Classe Componente

A classe Componente realiza os serviços oferecidos ao usuário pela classe FrmComponente. Tais serviços são:

- Propriedades cada um dos atributos (id_componente, id_item, ordem, descricao, tipo);
- Cadastro (método Inserir), alteração (método Alterar) e exclusão (método Excluir);
- Validação dos dados (método Validar_Dados);
- Retornar um objeto da classe DataView com as informações de todos os componentes armazenados na base de dados para um determinado item (método Componentes);
- Verificar se a ordem dada a um componente já esta cadastrada (método Ordem_Cadastrada).

Os métodos da classe Componente que necessitem de acesso ao banco de dados da aplicação utilizam os serviços de um objeto da classe ComponenteDB da camada de dados, que é membro da classe Componente, o que cria uma forte dependência entre as duas.

Classe Texto

A classe Texto é derivada da classe Componente, ou seja, herda toda a implementação desta. Esta classe é específica para a manipulação de componentes de um determinado item do conteúdo que sejam textuais.

As principais especializações realizadas na classe Texto em relação à classe Componente são:

- Inclusão de mais atributos (id_texto e corpo) e conseqüentemente propriedades para os mesmos;
- Sobrecarga dos métodos Inserir e Alterar, para realizarem a sua atividade no novo contexto. O método Inserir possui duas implementações uma com e outra sem a validação de dados;
- Sobrecarga do método Validar_Dados, para utilizar a implementação da classe base (Componente) e também realizar atividade específica;
- Retornar um objeto da classe DataRow com as informações de um componente texto (método Componente_Texto).

Os métodos da classe Texto que necessitem de acesso ao banco de dados da aplicação utilizam os serviços de um objeto da classe TextoDB da camada de dados, que é membro da classe Texto, o que cria uma forte dependência entre as duas.

Classe Imagem

A classe Imagem é derivada da classe Componente, ou seja, herda toda a implementação desta. Esta classe é específica para a manipulação de componentes de um determinado item do conteúdo que sejam arquivos de imagem.

As principais especializações realizadas na classe Imagem em relação à classe Componente são:

- Inclusão de mais atributos (id_imagem, origem e destino) e conseqüentemente propriedades para os mesmos;
- Sobrecarga dos métodos Inserir e Alterar, para realizarem a sua atividade no novo contexto. O método Inserir possui duas implementações uma com e outra sem a validação de dados;
- Sobrecarga do método Validar_Dados, para utilizar a implementação da classe base (Componente) e também realizar atividade específica;
- Retornar um objeto da classe DataRow com as informações de um componente imagem (método Componente_Imagem).

Os métodos da classe Imagem que necessitem de acesso ao banco de dados da aplicação utilizam os serviços de um objeto da classe ImagemDB da camada de dados, que é membro da classe Imagem, o que cria uma forte dependência entre as duas.

Classe Link

A classe Link é derivada da classe Componente, ou seja, herda toda a implementação desta. Esta classe é específica para a manipulação de componentes de um determinado item do conteúdo que sejam pontos de acesso a outras partes do conteúdo ou da *web*.

As principais especializações realizadas na classe Link em relação à classe Componente são:

- Inclusão de atributos adicionais (id_link e url) e conseqüentemente propriedades para os mesmos;
- Sobrecarga dos métodos Inserir e Alterar, para realizarem a sua atividade no novo contexto. O método Inserir possui duas implementações uma com e outra sem a validação de dados;

- Sobrecarga do método Validar_Dados, para utilizar a implementação da classe base (Componente) e também realizar atividade específica;
- Retornar um objeto da classe DataRow com as informações de um componente link (método Componente_Link).

Os métodos da classe Link que necessitem de acesso ao banco de dados da aplicação utilizam os serviços de um objeto da classe LinkDB da camada de dados, que é membro da classe Link, o que cria uma forte dependência entre as duas.

Camada de acesso a dados

As classes desta camada mapeiam o banco de dados utilizado pela aplicação assim como suas tabelas, cujas definições serão apresentadas no próximo capítulo, no sobre o modelo de dados. O diagrama é mostrado na figura 4.4 a seguir.

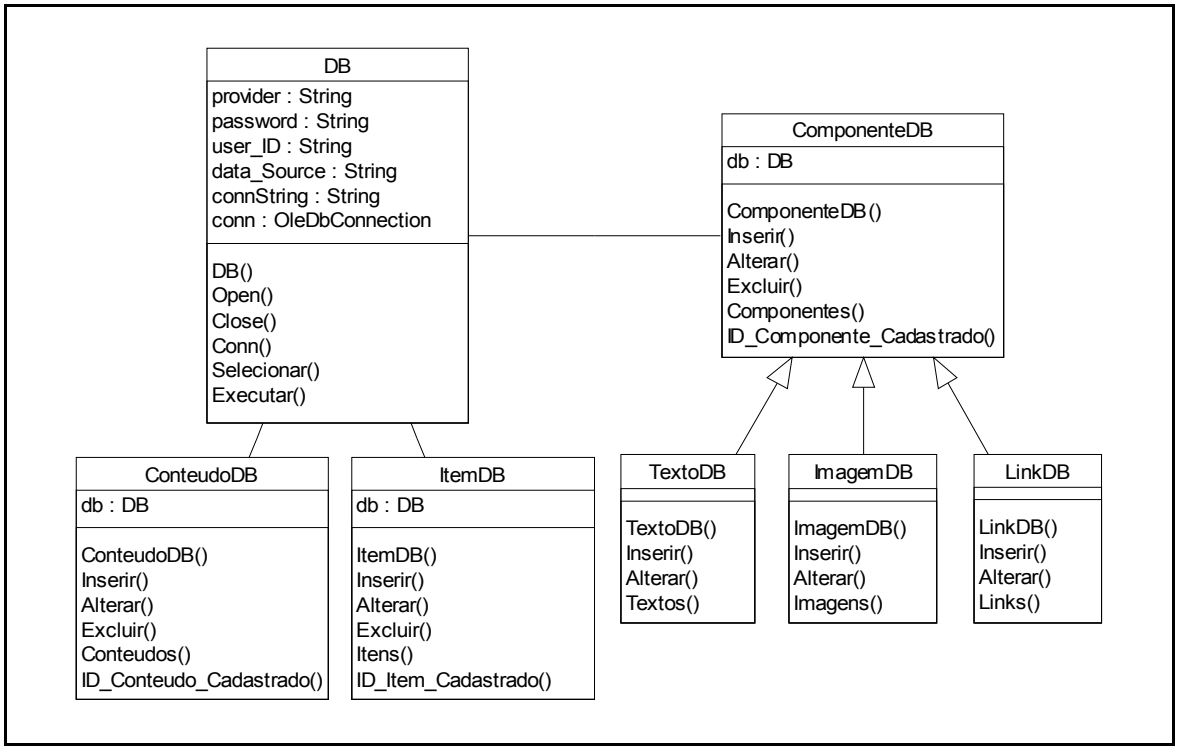

Fig. 4.4 – Diagrama de classes da camada de acesso a dados.

Classe DB

A classe DB implementa o acesso ao banco de dados utilizado na aplicação. Através desta classe é que todas as demais classes desta camada realizam operações no banco de dados. Tais operações são:

- Configurar o acesso ao banco de dados (construtor da classe);
- Abrir a conexão com o banco (método Open);
- Fechar a conexão com o banco (método Close);
- Retornar a conexão realizada com o banco (propriedade Conn);
- Retornar um objeto da classe DataSet após a execução de um comando de seleção no banco de dados (método Selecionar);
- Executar um comando de atualização no banco de dados (método Executar).

Classe ConteudoDB

A classe ConteudoDB implementa as instruções de manipulação de dados que realizam operações na tabela Conteudos do banco de dados da aplicação. Através desta classe as seguintes operações podem ser realizadas nesta tabela:

- Inserir registro (método Inserir);
- Alterar registro (método Alterar);
- Excluir registro (método Excluir);
- Retornar um objeto da classe DataSet com todos os registros da tabela (método Conteudos) ;
- Retornar ID do conteúdo recém cadastrado (método ID_Conteudo_Cadastrado).

Através de um membro da classe DB, todas as instruções definidas nesta classe podem ser executadas. Isto demonstra a associação entre estas duas classes.

Classe ItemDB

A classe ItemDB implementa as instruções de manipulação de dados que realizam operações na tabela Itens do banco de dados da aplicação. Através desta classe as seguintes operações podem ser realizadas nesta tabela:

- Inserir registro (método Inserir);

- Alterar registro (método Alterar);
- Excluir registro (método Excluir);
- Retornar um objeto da classe DataSet com todos os registros da tabela (método Itens) ;
- Retornar ID do item recém cadastrado (método ID_Item_Cadastrado).

Através de um membro da classe DB, todas as instruções definidas nesta classe podem ser executadas. Isto demonstra a associação entre estas duas classes.

Classe ComponenteDB

A classe ComponenteDB implementa as instruções de manipulação de dados que realizam operações na tabela Componentes do banco de dados da aplicação. Através desta classe as seguintes operações podem ser realizadas nesta tabela:

- Inserir registro (método Inserir);
- Alterar registro (método Alterar);
- Excluir registro (método Excluir);
- Retornar um objeto da classe DataSet com todos os registros da tabela (método Componentes) ;
- Retornar ID do componente recém cadastrado (método ID_Componente_Cadastrado).

Através de um membro da classe DB, todas as instruções definidas nesta classe podem ser executadas. Isto demonstra a associação entre estas duas classes.

Classe TextoDB

A classe TextoDB é uma especialização da classe ComponenteDB herdando, portanto, toda a implementação desta. A tabela do banco de dados que é manipulada por esta classe, além da tabela Componentes, é a tabela Textos.

Os métodos sobrecarregados e incluídos na implementação são:

- Inserir registro (método Inserir), que utiliza o método da classe base (ComponenteDB) para inserir as informações na tabela Componentes e realiza a sua implementação para inserir informações na tabela Textos;

- Alterar registro (método Alterar), que utiliza o método da classe base (ComponenteDB) para alterar as informações na tabela Componentes e realiza a sua implementação para alterar informações na tabela Textos;
- Retorna um objeto da classe DataSet com todos os registros da tabela Textos (método Textos).

O membro da classe DB que executa as instruções é herdado da classe ComponenteDB, não havendo associação direta entre estas classes.

Classe ImagemDB

A classe ImagemDB é uma especialização da classe ComponenteDB herdando, portanto, toda a implementação desta. A tabela do banco de dados que é manipulada por esta classe, além da tabela Componentes, é a tabela Imagens.

Os métodos sobrecarregados e incluídos na implementação são:

- Inserir registro (método Inserir), que utiliza o método da classe base (ComponenteDB) para inserir as informações na tabela Componentes e realiza a sua implementação para inserir informações na tabela Imagens;
- Alterar registro (método Alterar), que utiliza o método da classe base (ComponenteDB) para alterar as informações na tabela Componentes e realiza a sua implementação para alterar informações na tabela Imagens;
- Retornar um objeto da classe DataSet com todos os registros da tabela Imagens (método Imagens).

O membro da classe DB que executa as instruções é herdado da classe ComponenteDB, não havendo associação direta entre estas classes.

Classe LinkDB

A classe LinkDB é uma especialização da classe ComponenteDB herdando, portanto, toda a implementação desta. A tabela do banco de dados que é manipulada por esta classe, além da tabela Componentes, é a tabela Links.

Os métodos sobrecarregados e incluídos na implementação são:

- Inserir registro (método Inserir), que utiliza o método da classe base (ComponenteDB) para inserir as informações na tabela Componentes e realiza a sua implementação para inserir informações na tabela Links;
- Alterar registro (método Alterar), que utiliza o método da classe base (ComponenteDB) para alterar as informações na tabela Componentes e realiza a sua implementação para alterar informações na tabela Links;
- Retornar um objeto da classe DataSet com todos os registros da tabela Imagens (método Links).

O membro da classe DB que executa as instruções é herdado da classe ComponenteDB, não havendo associação direta entre estas classes.

No capítulo seguinte é apresentada a fase de construção da ferramenta, como o resultado das atividades de modelo de dados, implementação e testes. Além destas atividades, será apresentada uma evidência de execução da aplicação construída.

CAPÍTULO 5

FASE DE CONSTRUÇÃO

Neste capítulo é encerrado o desenvolvimento da ferramenta de autoria de conteúdo para aprendizado via *web*. Serão apresentadas as atividades da fase de construção: modelo de dados, implementação e testes. Na atividade de implementação, espera-se evidenciar o alcance dos objetivos deste trabalho, que foram definidos no capítulo 2, através da apresentação das características das tecnologias utilizadas para a definição e apresentação dos conteúdos. O capítulo encerra-se com a apresentação da execução da ferramenta, com a ilustração do cadastro de um conteúdo e a sua publicação em diferentes formas de apresentação.

5.1 Modelo de dados

Fundamental à estrutura de um banco de dados é o conceito de modelo de dados, uma coleção de ferramentas conceituais para descrição de dados, relacionamentos de dados, semântica de dados e restrições de consistência. De uma maneira geral, os modelos de dados são usados para especificar a estrutura do banco de dados e para fornecer uma descrição de alto nível da de sua implementação [KS1995].

O modelo de dados que será utilizado no desenvolvimento da ferramenta de autoria de conteúdo para aprendizado via *web* é o modelo relacional. Tal modelo representa os dados e seus relacionamentos através de um conjunto de tabelas [KS1995].

O modelo de dados da ferramenta reflete o modelo de classes da camada de negócio. Esta similaridade já era esperada uma vez que esta camada representa as entidades de domínio da aplicação, cujas informações necessitam de armazenamento persistente.

Na figura 5.1, logo a seguir, é mostrado o modelo de dados da ferramenta e, em seguida, a descrição de todas as suas tabelas.

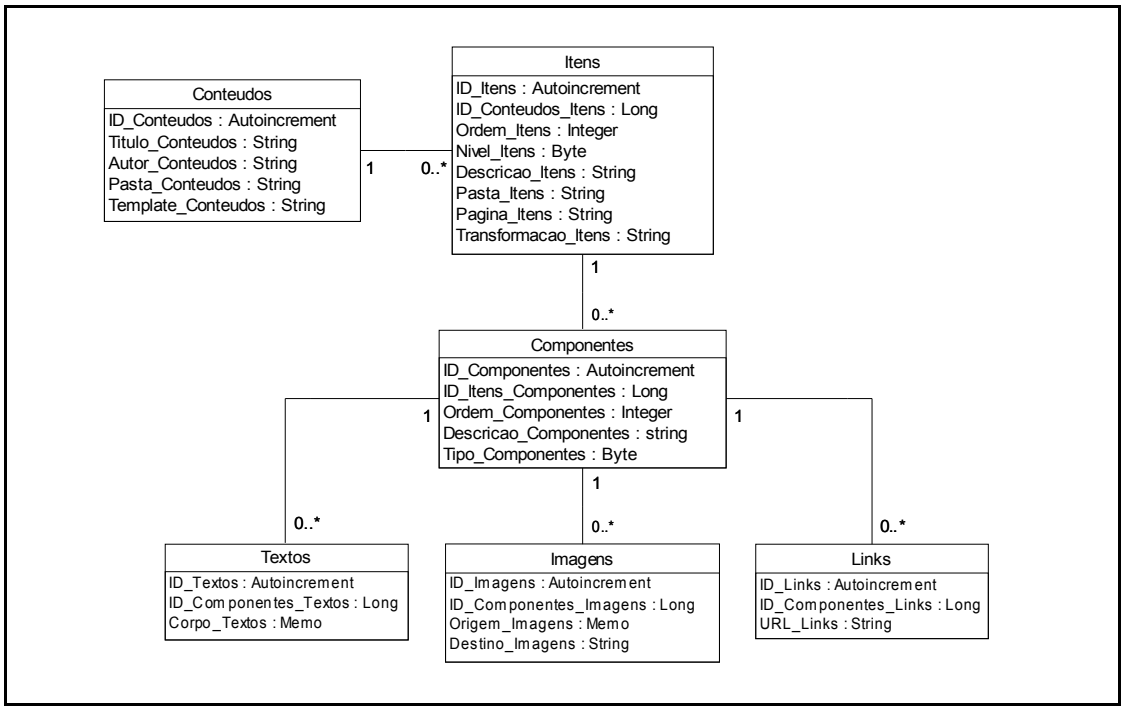

Fig 5.1 – Modelo de dados da ferramenta.

Atributos	Descrição	Tipo
ID_Conteudos	Identificador do conteúdo	Long
Titulo_Conteudos	Título do conteúdo	String
Autor_Conteudos	Autor do conteúdo	String
Pasta_Conteudos	Pasta onde o conteúdo será armazenado	String
Template_Conteudos	Modelo de publicação selecionado para o conteúdo	String

Tabela 5.1 – Tabela Conteúdos.

Atributos	Descrição	Tipo
ID_Itens	Identificador do item	Long
ID_Conteudos_Itens	Identificador do conteúdo	Long
Ordem_Itens	Ordem do item dentro do conteúdo	Integer
Nivel_Itens	Nível do item	Byte
Descricao_Itens	Descrição do item	String
Pasta_Itens	Pasta onde a página do item será armazenada	String
Pagina_Itens	Nome do arquivo que será gerado para o item	String
Transformacao_Itens	Nome do arquivo que pode fazer a transformação do item	String

Tabela 5.2 – Tabela Itens.

Atributos	Descrição	Tipo
ID_Componentes	Identificador do componente	Long
ID_Itens_Componentes	Identificador do item	Long
Ordem_Componentes	Ordem do componente dentro do item	Integer
Descricao_Componentes	Descrição do componente	String
Tipo_Componentes	Tipo do componente (1 – Texto, 2 – Imagem e 3 – Link)	Byte

Tabela 5.3 – Tabela Componentes.

Atributos	Descrição	Tipo

ID_Textos	Identificador do texto	Long
ID_Componentes_Textos	Identificador do componente	Long
Corpo_Textos	Texto contido pelo componente	String

Tabela 5.4 – Tabela Textos.

Atributos	Descrição	Tipo
ID_Imagens	Identificador da imagem	Long
ID_Componentes_Imagens	Identificador do componente	Long
Origem_Imagens	Pasta e arquivo de origem da imagem	String
Destino_Imagens	Pasta e arquivo de destino da imagem	String

Tabela 5.5 – Tabela Imagens.

Atributos	Descrição	Tipo
ID_Links	Identificador do link	Long
ID_Componentes_Links	Identificador do componente	Long
URL_Links	Texto com link interno ou externo ao conteúdo	String

Tabela 5.6 – Tabela Links.

O modelo de dados da ferramenta foi implementado fisicamente no sistema gerenciador de banco de dados Microsoft® Access 2000, que ofereceu todos os recursos necessários para a construção deste banco de dados.

5.2Implementação

O propósito da implementação é a construção das classes em termos de componentes. Um componente pode ser um código fonte, um código objeto ou um executável, além de arquivos que contenham informações necessárias para a execução de um outro componente. Ao final da implementação pode ser construído o modelo de implementação [Rational2001].

O modelo de implementação da ferramenta de autoria de conteúdo para aprendizado via *web* será detalhado neste tópico. O diagrama pode ser visto na figura 5.2, logo abaixo. Durante o detalhamento dos componentes do modelo, serão introduzidas as características das linguagens e ambientes utilizados na implementação da ferramenta. Os componentes do modelo são Ferramenta.exe, Template e Templates.xml.

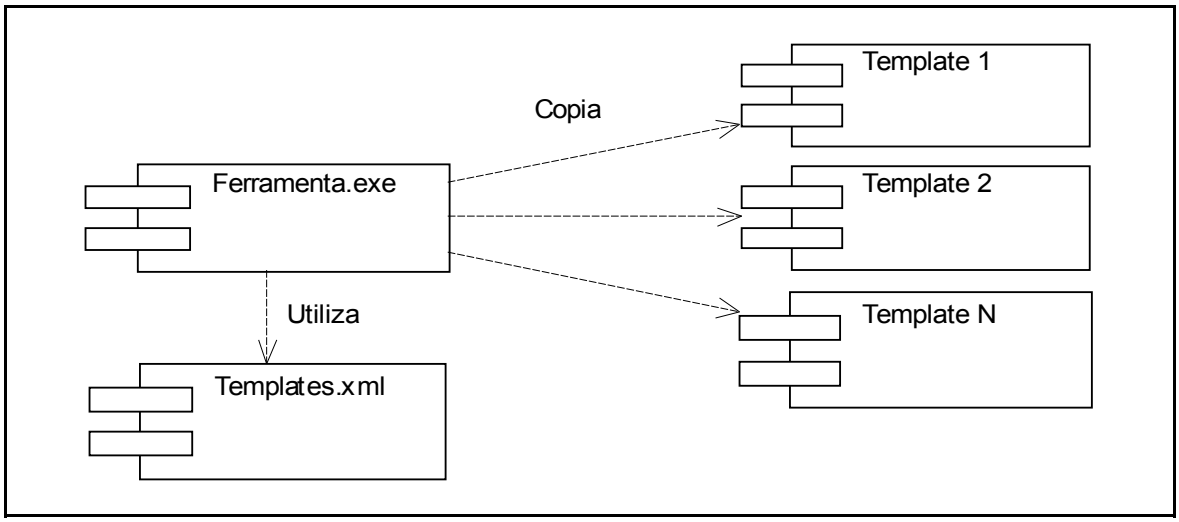

Fig. 5.2 – Modelo de implementação da ferramenta.

Ferramenta.exe

Este componente é o executável da ferramenta de autoria de conteúdo para aprendizado via *web*. Foi construído utilizando-se a tecnologia Microsoft® .Net, mais precisamente a linguagem de programação C#, o .Net Framework e o ambiente de desenvolvimento Visual Studio .Net.

A linguagem C#, por ser totalmente orientada a objetos, permitiu a implementação direta da solução definida na atividade de projeto. A utilização do .Net Framework permitiu a utilização de muitas classes que tornaram certas tarefas mais simples de serem realizadas como, por exemplo, a construção de uma interface gráfica para a aplicação utilizando o *namespace* System.Windows.Forms e a manipulação de arquivos XML utilizando o *namespace* System.Xml. Em especial este último *namespace* implementa internamente a interface de programação DOM (*Document Object Model*) para a manipulação total dos arquivos XML [DM2002]. A referência utilizada nesta atividade de implementação foi obtida no trabalho de Lima e Reis [LR2002] e nos seguintes cursos da Microsoft: [MS12002], [MS22002] e [MS32002].

Todas as classes definidas no projeto (camadas de apresentação, negócio e dados) foram implementadas e depois de compiladas geraram um único executável. Como apresentado no modelo de implementação, este componente depende dos componentes Template.xml e Template, que serão apresentados mais adiante.

Ao gerar os conteúdos, ou seja, os arquivos dos itens de conteúdo, este componente utiliza a linguagem XML (*Extensible Markup Language*). A XML tem como propósito

principal a descrição de informações. Esta capacidade é extremamente importante para armazenamento, recuperação e transmissão de informações. Ela permite que sejam colocados em um mesmo lugar os dados e os metadados, ou seja, as suas definições [Decio2001]. Através de XML, documentos podem ser estruturados com a ajuda de elementos de marcação sintática. Sua principal característica é ser extensível, ou seja, os elementos utilizados na estruturação de um documento não são fixos e novos podem ser acrescentados. Sendo bem formado, qualquer documento XML pode ser processado com um analisador XML padrão [DM2002]. Esta última característica é o que garante que um documento XML pode ser aberto em qualquer navegador *web* que atenda as recomendações para a XML do *World Wide Web Consortium* (W3C). No trabalho de David Carlson [Carlson2002] são apresentadas formas de se modelar conteúdos XML utilizando UML.

Os elementos de um arquivo XML gerado para um item de conteúdo pode ser visto na tabela 5.7 abaixo. Logo em seguida é mostrado um modelo estrutural do arquivo.

Elementos	Descrição
pagina	O início da página. Nó raiz.
nivel	Nível do item do conteúdo. Necessário na transformação em XSLT.
titulo	Título da página. Pode ser a descrição do item.
conteudo	Delimitador das informações sobre o conteúdo.
titulo_conteudo	Título do conteúdo.
autor	Nome do autor do conteúdo.
texto	Delimitador do componente texto.
imagem	Delimitador do componente imagem.
link	Delimitador do componente link.
ordem	Ordem do componente.
descricao	Descrição do componente.
corpo	Conteúdo do componente texto.
arquivo	Caminho e arquivo onde do arquivo do componente imagem.
url	Caminho ou endereço do componente link.

Tabela 5.7 – Elementos de um arquivo XML de item de conteúdo.

Modelo estrutural de um arquivo XML de item de conteúdo

```
<?xml version="1.0"?>
<pagina>
    <nivel></nivel>
    <titulo></titulo>
    <conteudo>
            <titulo_conteudo></titulo_conteudo>
            <autor></autor>
    </conteudo>
    <texto>
            <ordem></ordem>
            <descricao></descricao>
            <corpo></corpo>
```

```
    </texto>
    ...
    <imagem>
            <ordem></ordem>
            <descricao></descricao>
            <arquivo></arquivo>
    </imagem>
    ...
    <link>
            <ordem></ordem>
            <descricao></descricao>
            <url></url>
    </link>
</pagina>
```

Como previsto na especificação, a ferramenta realiza a importação de conteúdo através de um arquivo XML. Isto permite que outros conteúdos, gerados pela ferramenta ou não, possam ser reutilizados. Na tabela 5.8 são mostrados os elementos deste arquivo e em seguida os seus modelos estruturais.

Elementos	Descrição
conteudo	O início da página. Nó raiz.
titulo	Título do conteúdo.
autor	Autor do conteúdo.
item	Delimitador de item.
ordem	Ordem do item.
nivel	Nível do item.
descricao	Descrição do item.
componente	Delimitador de componente.
ordem	Ordem do componente.
tipo	Tipo do componente.
descricao	Descrição do componente.
corpo	Corpo do componente texto.
origem	Local do arquivo do componente imagem.
url	URL do componente link.

Tabela 5.8 – Elementos de um arquivo XML de importação.

Modelo estrutural do arquivo XML de importação

```
<?xml version="1.0"?>
<conteudo>
    <titulo></titulo>
    <autor></autor>
    <item>
            <ordem></ordem>
            <nivel></nivel>
            <descricao></descricao>
            <componente>
                    <ordem></ordem>
                    <tipo></tipo>
                    <descricao></descricao>
                    <corpo></corpo>
            </componente>
            <componente>
                    <ordem></ordem>
                    <tipo></tipo>
                    <descricao></descricao>
                    <origem></origem>
            </componente>
            ...
            <componente>
                    <ordem></ordem>
                    <tipo></tipo>
                    <descricao></descricao>
                    <url></url>
            </componente>
    </item>
    ...
    <item>
            <ordem></ordem>
            <nivel></nivel>
            <descricao></descricao>
            <componente>
                    <ordem></ordem>
                    <tipo></tipo>
                    <descricao></descricao>
                    <corpo></corpo>
            </componente>
            <componente>
                    <ordem></ordem>
                    <tipo></tipo>
                    <descricao></descricao>
                    <origem></origem>
            </componente>
            ...
            <componente>
                    <ordem></ordem>
                    <tipo></tipo>
                    <descricao></descricao>
                    <url></url>
            </componente>
    </item>
</conteudo>
```

Além do arquivo de item, este componente também gera um arquivo com informações para todos os itens de um determinado conteúdo, o indice.xml. A seguir, na tabela 5.9, são mostrados os elementos deste arquivo e em seguida o seu modelo estrutural.

Elementos	Descrição
pagina	O início da página. Nó raiz.
titulo	Título da página. Pode ser a descrição do item.
conteudo	Delimitador das informações sobre o conteúdo.
titulo_conteudo	Título do conteúdo.
autor	Nome do autor do conteúdo.
topico	Delimitador das informações do tópico do índice.
nivel	Nível do item do conteúdo. Necessário na transformação em XSLT.
descricao	Descrição do item
link	Caminho à página do item.

Tabela 5.9 – Elementos de um arquivo XML de índice.

Modelo estrutural do arquivo XML de índice (indice.xml)

```
<?xml version="1.0"?>
<pagina>
    <titulo></titulo>
    <conteudo>
            <titulo_conteudo></titulo_conteudo>
            <autor></autor>
    </conteudo>
    <topico>
            <nivel></nivel>
            <descricao></descricao>
            <link></link>
    </topico>
    ...
    <topico>
            <nivel></nivel>
            <descricao></descricao>
            <link></link>
    </topico>
</pagina>
```

Template.xml

Este componente é um arquivo XML que lista todos os *templates* disponíveis para serem utilizados pela ferramenta informando a localização de cada um destes. Sem este componente o Ferramenta.exe não pode gerar o *site* de publicação de um determinado conteúdo. A seguir, na tabela 5.10, são mostrados os elementos deste arquivo e em seguida o seu modelo estrutural.

Elementos	Descrição
templates	O início da página. Nó raiz.
template	Delimitador das informações do *template*.
nome	Nome do *template*.
pasta	Local onde está armazenado.
arquivo	Arquivo que faz parte do *template*.

Tabela 5.10 – Elementos do arquivo XML de templates.

Modelo estrutural do arquivo XML de índice (indice.xml)

```
<templates>
    <template>
            <nome></nome>
            <pasta></pasta>
            <arquivo></arquivo>
            ...
            <arquivo></arquivo>
    </template>
    ...
    <template>
            <nome></nome>
            <pasta></pasta>
            <arquivo></arquivo>
            ...
            <arquivo></arquivo>
    </template>
</templates>
```

Template

Os componentes template são responsáveis pela apresentação (*site* de publicação) dos conteúdos que são gerados pelo Ferramenta.exe.

Cada template é composto por vários arquivos, que podem ser imagens, arquivos de estilo e, principalmente, os arquivos que transformam arquivos XML dos itens de conteúdo e do índice para a tecnologia de apresentação desejada.

Para realizar estas transformações, foi utilizada a XSLT (*Extensible Stylesheet Language*). A finalidade principal de XSLT é permitir transformações controladas por folha de estilo de um formato XML (independente de apresentação), para formatos específicos como HTML, WML ou PDF [DM2002].

A seguir é mostrado o formato de um arquivo que transforma um arquivo XML de um item de conteúdo gerado pela ferramenta para o formato HTML. No tópico 5.4, sobre a execução da ferramenta, serão evidenciadas as características da utilização da XSLT, com a apresentação de um mesmo conteúdo em diversos formatos de apresentação.

Estrutura de um arquivo XSLT que transforma um arquivo XML para HTML

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output  method=”html”/>
<xsl:template match="/pagina">

<html>
    <head>
            <title><xsl:value-of select="titulo"/></title>
    </head>
    <body>
            <h1><xsl:value-of select="conteudo/titulo_conteudo"/></h1>
            <xsl:apply-templates/>
    </body>
</html>

</xsl:template>
<xsl:template match="nivel">
</xsl:template>
<xsl:template match="conteudo">
</xsl:template>
<xsl:template match="titulo">
    <h2><xsl:value-of select="."/></h2>
</xsl:template>
<xsl:template match="texto">
    <p><xsl:value-of select="corpo"/></p>
</xsl:template>
<xsl:template match="imagem">
    <p align="center">
            <img border="0" src="images/{arquivo}"></img><br/>
            <xsl:value-of select="descricao"/>
    </p>
</xsl:template>
<xsl:template match="link">
    <p align="center">
            <a href="{url}"><xsl:value-of select="descricao"/></a>
    </p>
</xsl:template>
</xsl:stylesheet>
```

No tópico a seguir são feitas considerações sobre como foram realizados os testes após a implementação de cada caso de uso da ferramenta de autoria de conteúdo para aprendizado via *web*.

5.3 Testes

No RUP, o objetivo principal da atividade de testes é avaliar a qualidade através das seguintes práticas [Rational2001]:

- Procurar e documentar falhas na qualidade do software;
- Geralmente aconselhar sobre a qualidade do software;
- Validar as funcionalidades de acordo com o projeto;
- Validar se os requisitos foram implementados apropriadamente.

Um bom foco para a criação de um plano de testes é a avaliar se as regras de negócio e os fluxos previstos nos casos de uso se verificam. Ao final de cada iteração, faz-se esta observação para cada caso de uso previsto no plano de iterações e ao final do desenvolvimento faz-se uma avaliação geral. Este método é o que foi utilizado no desenvolvimento da ferramenta de autoria de conteúdo para aprendizado via *web*. Os resultados obtidos foram satisfatórios e a ferramenta executa todas as funcionalidades da maneira projetada. Caso se desejasse futuramente utilizar a ferramenta em escala comercial, ajustes e uma atividade de testes mais formal seriam necessários.

No tópico a seguir é apresentada uma evidência de execução da ferramenta construída.

5.4 Execução da ferramenta

Para evidenciar a construção da ferramenta e o alcance de alguns objetivos previstos, no capítulo 2, para este trabalho, apresentam-se aqui algumas telas de execução e de publicação de conteúdos realizados pela ferramenta.

Na figura 5.3 é apresentada a tela principal da ferramenta de autoria de conteúdo para aprendizado via *web*. Esta tela é representada pela classe FrmPrincipal da camada de apresentação do projeto da ferramenta.

Nas figuras 5.4, 5.5 e 5.6 são apresentadas as telas de cadastro de conteúdos, itens e componentes, respectivamente. Estas telas são representadas no projeto pelas classes FrmConteudo, FrmItem, FrmComponente, respectivamente, da camada de apresentação. São todas as telas da aplicação, através das quais o Autor de Conteúdos, ator previsto no modelo de casos de uso da ferramenta, tem acesso a todas as funcionalidades da ferramenta.

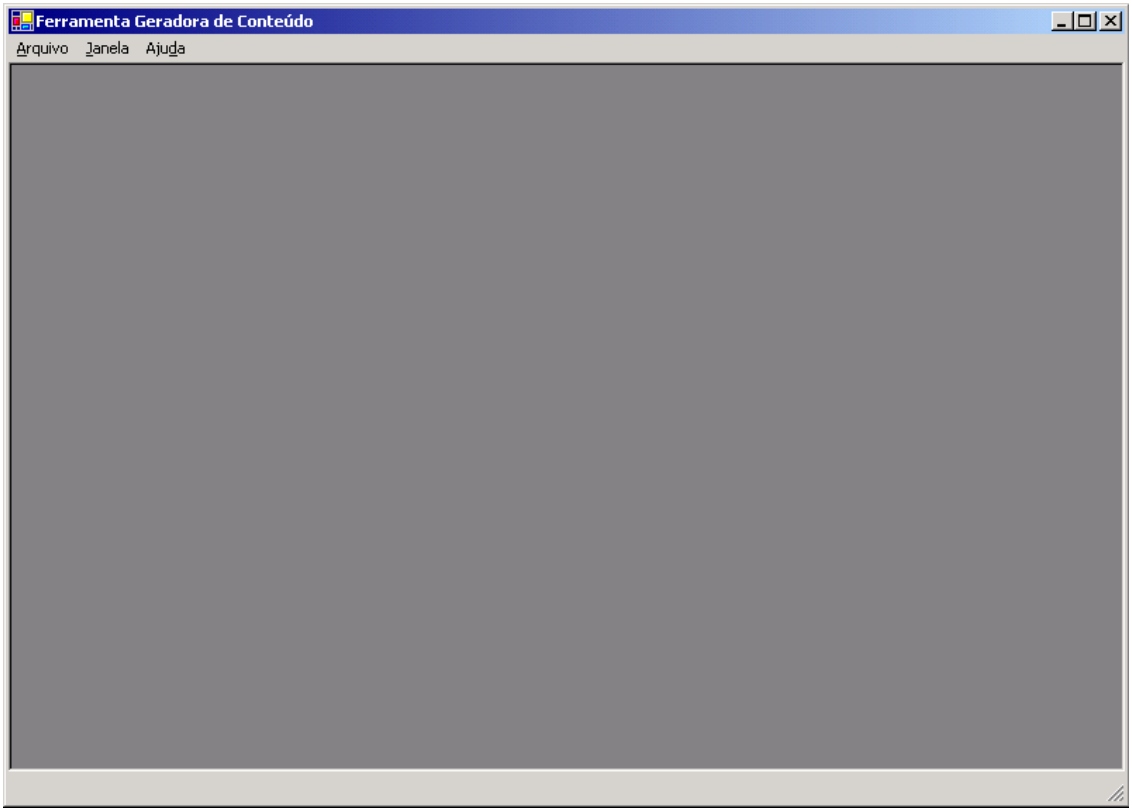

Fig. 5.3 – Formulário principal da ferramenta.

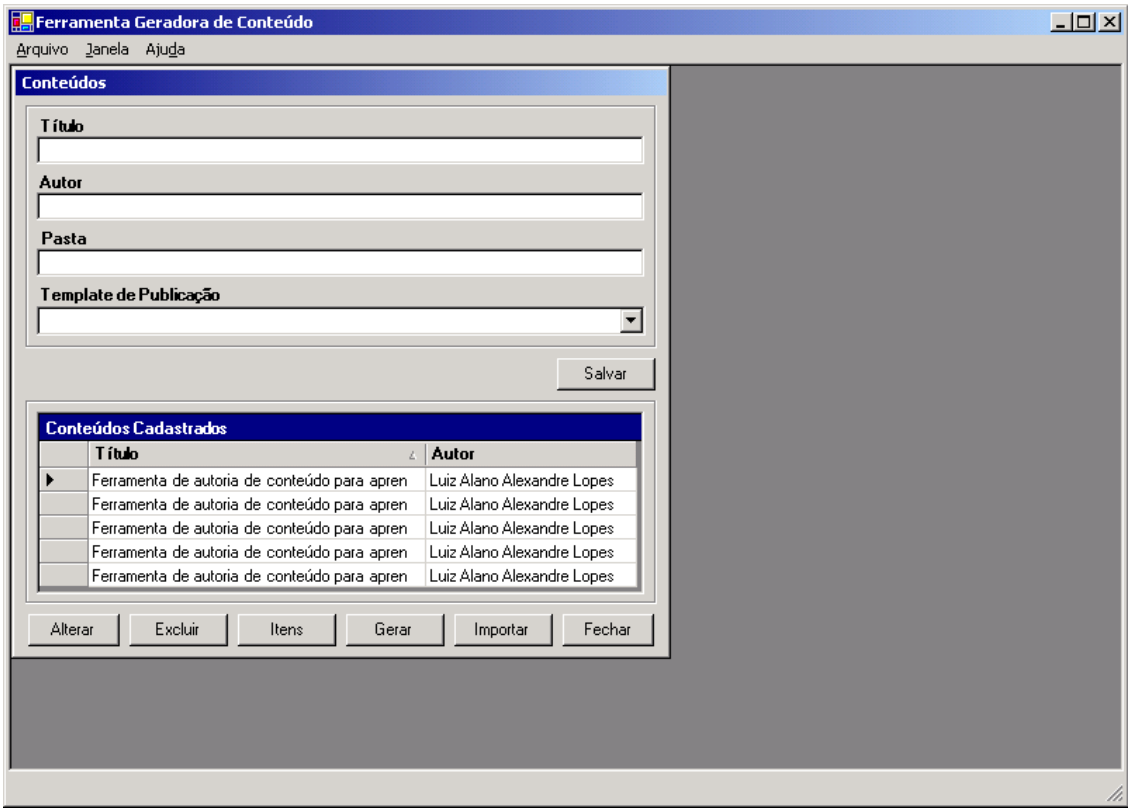

Fig. 5.4 – Formulário de cadastro de conteúdo.

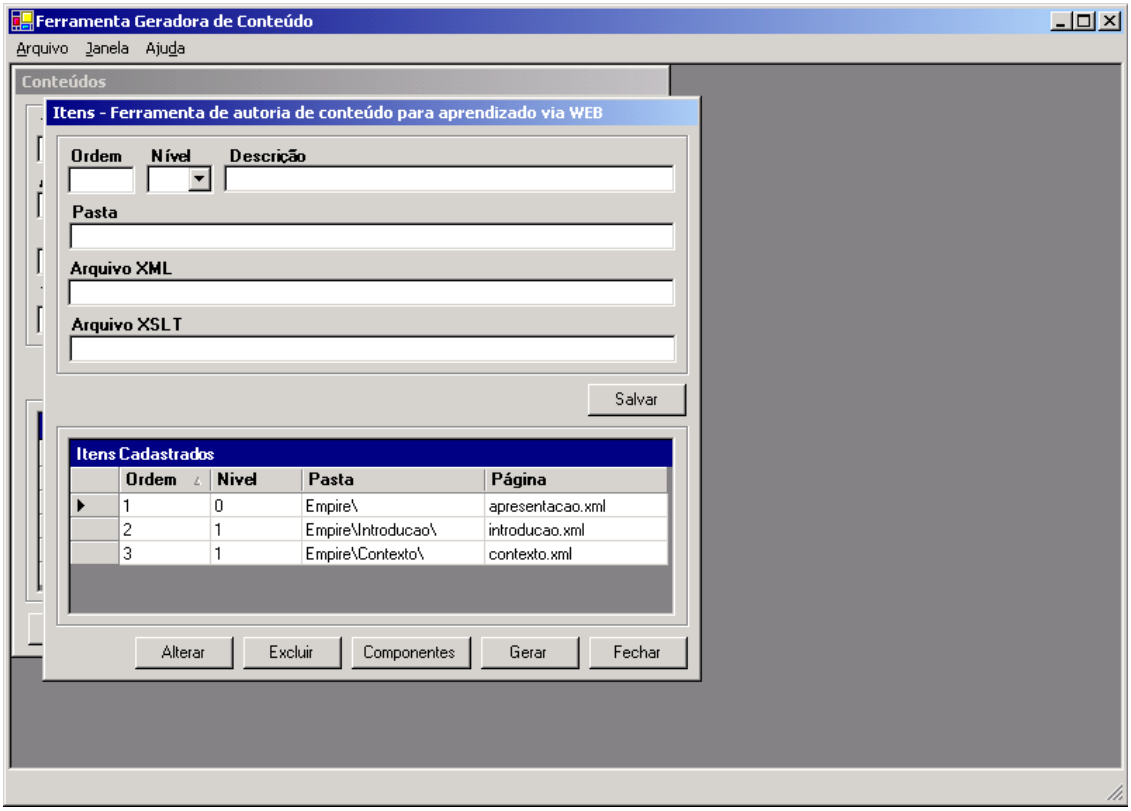

Fig. 5.5 – Formulário de cadastro de itens.

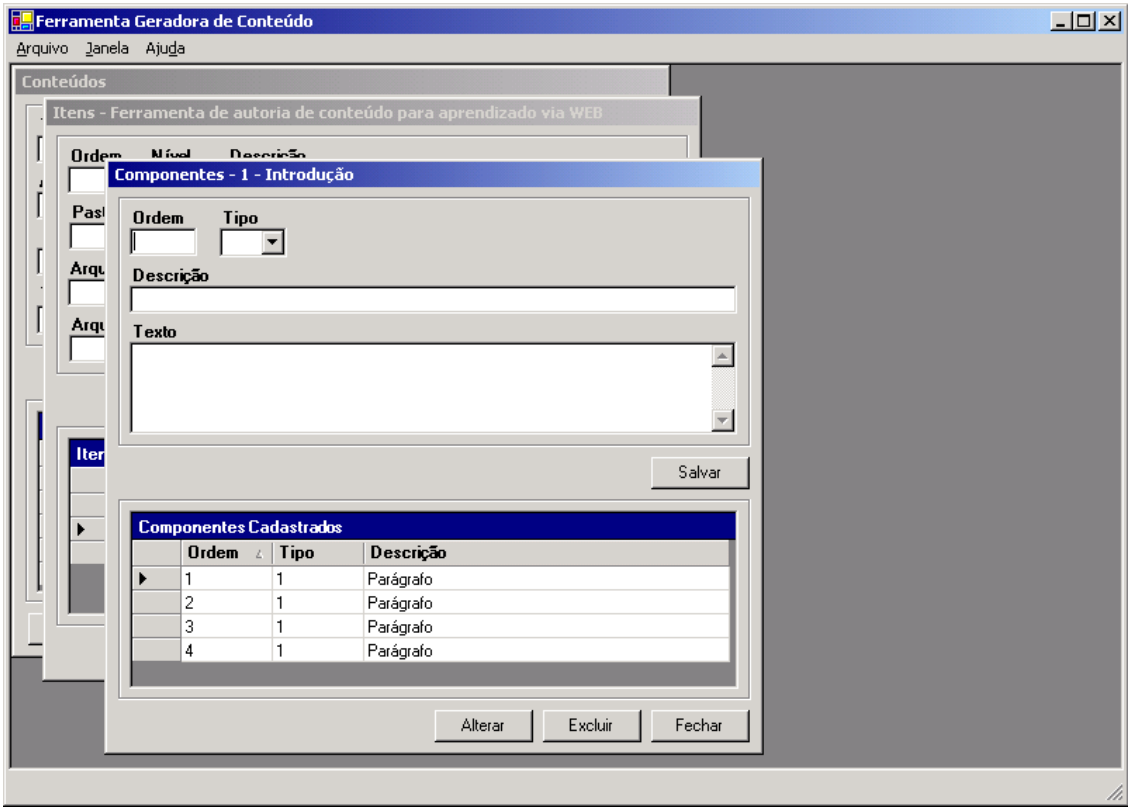

Fig. 5.6 – Formulário de cadastro de componentes.

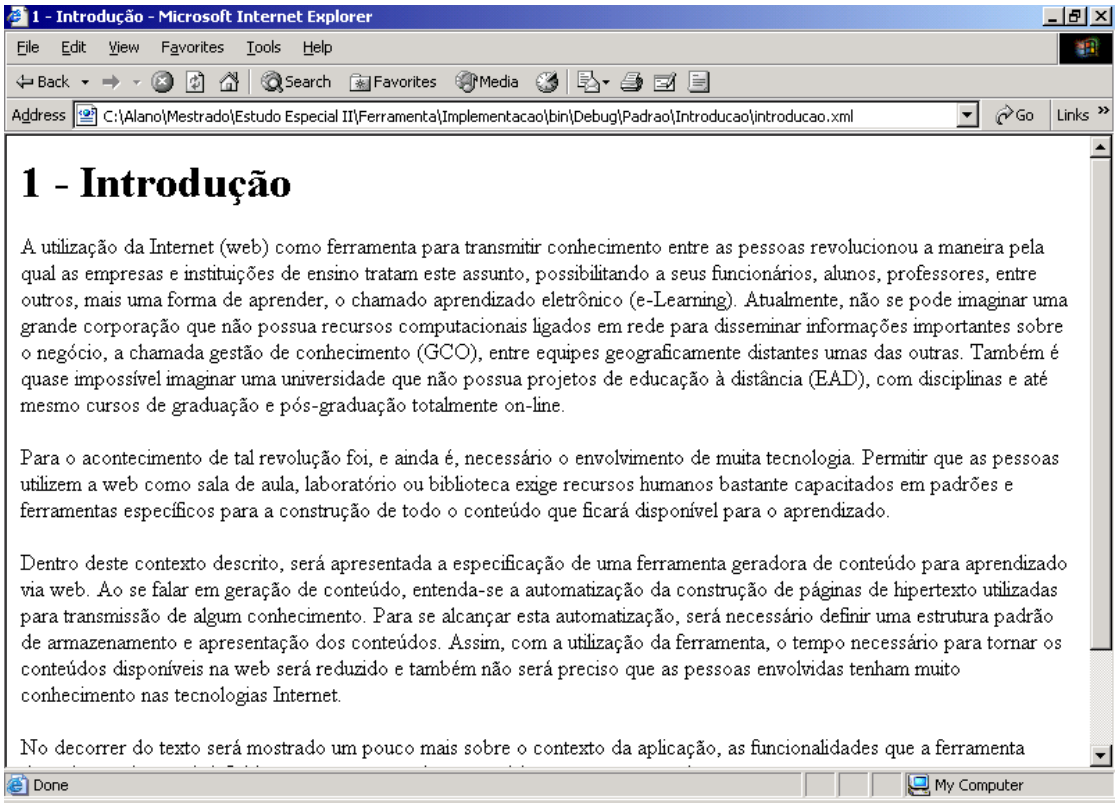

Fig 5.7 – Apresentação de uma publicação em um formato padrão.

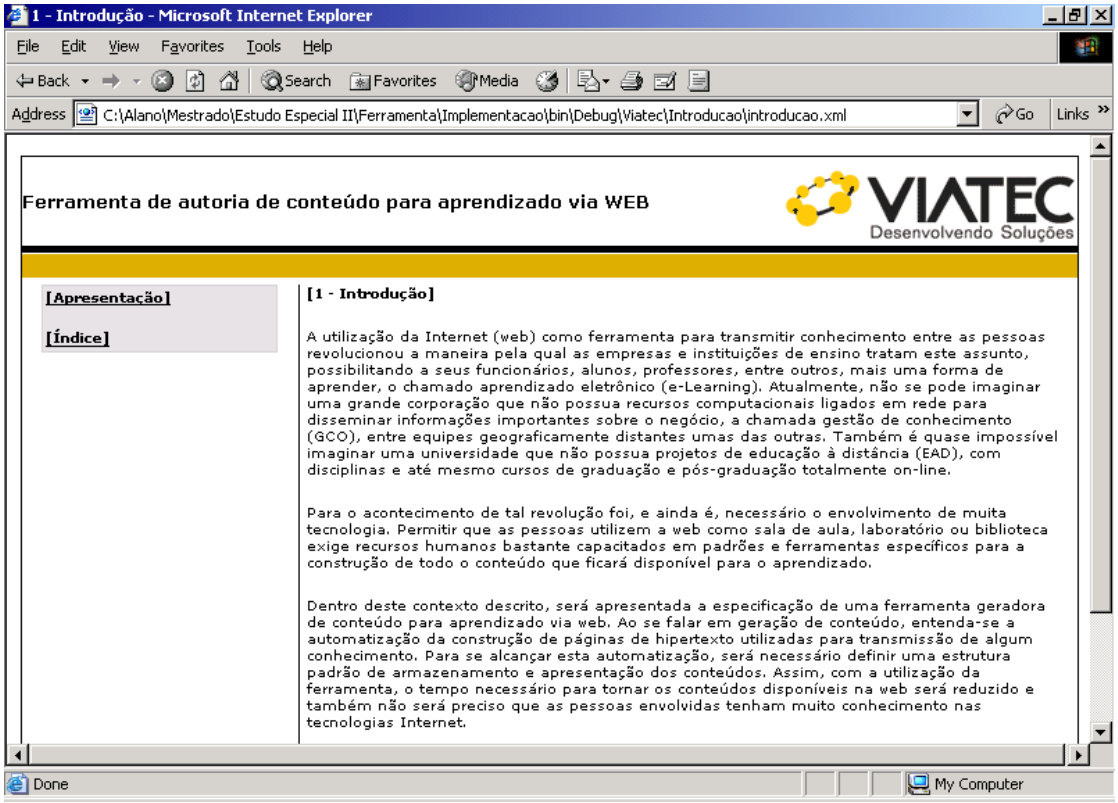

Fig 5.8 – Apresentação de uma publicação em um formato específico.

Fig 5.9 – Apresentação de uma publicação em um formato WML.

A figura 5.7 é a apresentação, em um navegador *web*, da publicação de um conteúdo em um formato padrão da ferramenta. A figura 5.8 ilustra este mesmo conteúdo gerado em outra publicação, desta vez mais elaborada. Esta mudança é feita apenas com a escolha de outro *template* para o conteúdo selecionado. Estas duas figuras ilustram a transformação do conteúdo, através da XSLT, para o formato HTML. A figura 5.9 ilustra a transformação deste mesmo conteúdo para o formato WML.

No capítulo seguinte apresenta-se a conclusão deste trabalho e são feitas algumas considerações sobre trabalhos futuros sobre o tema.

CAPÍTULO 6

CONCLUSÃO

A ferramenta de autoria de conteúdo para aprendizado via *web* traz importantes conceitos que podem ser empregados em todas as iniciativas relacionadas com o aprendizado via *web*, como *sites* de educação a distância, treinamento organizacional, entre outros.

Os objetivos traçados no capítulo 2 para os conteúdos a serem gerados foram alcançados. Observa-se claramente que os conteúdos são:

- **Independentes da ferramenta:** pois todos os arquivos são gerados utilizando a XML, portanto podem ser editados por qualquer editor que suporte esta linguagem e até mesmo editores de texto tradicionais;
- **Independentes da plataforma:** para que sejam acessados, basta que os usuários utilizem um navegador *web* que suporte as recomendações mais atuais sobre a XML do W3C. Atualmente os navegadores mais atuais atendem a estas recomendações;
- **Independentes do formato de apresentação:** como foi evidenciado no capítulo 5, pôde-se criar formatos de apresentação diversos para um mesmo conteúdo gerado, tanto em HTML quanto em outras linguagens, como WML. Com um pouco mais de pesquisa e a utilização de outros recursos, poder-se-ia criar também apresentações para PDF.

O resultado final do processo de desenvolvimento da ferramenta, baseado em um processo de desenvolvimento iterativo e incremental, é uma boa documentação de todas as decisões tomadas. Ao se observar o código gerado se verifica que ele é realmente a tradução para uma linguagem de programação do projeto realizado. Pode-se afirmar que este texto é um bom modelo de documentação para projetos de desenvolvimento de sistemas.

A estimativa de tempo realizada no Capítulo 2 utilizando a métrica *Use Case Points* (Pontos de Casos de Uso) foi atingida. Foram necessárias até menos horas que o previsto, entretanto esta mensuração não foi realizada. Pode-se afirmar que todas as atividades previstas no plano de iterações foram realizadas dentro prazo estipulado para cada uma. Mais informações sobre essa métrica podem ser obtidas no trabalho de Paul Reed [Reed2000].

Como não foram realizadas experiências com usuários finais, pois o objetivo era desenvolver o protótipo que gerasse conteúdos *web* com as características citadas, não se pode

falar sobre vantagens e desvantagens em relação a outras ferramentas do ponto de vista dos usuários finais.

Analisando este trabalho como um todo o resultado foi muito bom. Os objetivos traçados foram alcançados e a apresentação das tecnologias envolvidas e da forma com que foram aplicadas é de fácil entendimento para possível reutilização. Evidentemente existem pontos de melhoria, como, por exemplo, a questão da utilização e validação por usuários finais.

No tópico seguinte, são apresentadas outras sugestões de melhorias e trabalhos futuros que podem dar continuidade ao tema deste texto.

6.1 Trabalhos futuros

As sugestões de trabalhos futuros relacionados com a ferramenta de autoria de conteúdo para aprendizado via *web* estão no campo das melhorias, acréscimo de funcionalidade e migração para outras plataformas de desenvolvimento. A seguir são apresentadas algumas sugestões:

- **Versão para a *web*:** a versão atual da ferramenta é de um aplicativo *desktop*, ou seja, opera instalada em um microcomputador. A criação de uma versão *web* do aplicativo tornaria transparente o processo de publicação dos conteúdos na Internet, uma vez que hoje, depois de gerados, eles devem ser copiados para um servidor manualmente;
- **Editor de conteúdos:** atualmente são utilizados formulários simples para a manipulação dos dados dos conteúdos. A criação de um editor tornaria a atividade de criação mais simples para os usuários. Tal editor deve funcionar na versão *web* do aplicativo também;
- **Integração com outros ambientes:** como foi apresentado na pesquisa do capítulo 2, existem muitos ambientes gerenciadores de cursos *on-line*. A utilização dentro destes ambientes dos conceitos utilizados na ferramenta pode trazer para eles as mesmas vantagens alcançadas;

- **Mudança de tecnologia:** a ferramenta, devido a utilização da linguagem C#, bastante semelhante à linguagem Java, pode ser facilmente migrada para esta plataforma, ou apenas as suas classes de negócio, permitindo assim a sua reutilização em outros projetos. Por possuir uma boa documentação e utilizar a orientação a objetos, esta migração pode ser feita também para outras linguagens que não sejam similares a C#. Neste trabalho utilizou-se esta linguagem devido a familiaridade do desenvolvedor com esta tecnologia, o que tornou alta a produtividade.

REFERÊNCIAS BIBLIOGRÁFICAS

[Ferreira1985] Ferreira, A., Novo dicionário da língua portuguesa, Nova Fronteira, 1985.

[ABO2000] Azevedo, F., Brasil, L. e Oliveira, R., Redes neurais com aplicações em controles e em sistemas especialistas, Bookstore, 2000.

[Rosenberg2002] Rosemberg, M. J, e-Learning – Implementando com sucesso aprendizado on-line na sua empresa, Makron Books, 2002.

[Comer2001] Comer, E. C., Redes de computadores e Internet, 2º edição, Bookman, 2001.

[PHR2002] Pará, T., Haguenauer, C., e Rivero, M., A influência da plataforma de gerenciamento de EAD no desenvolvimento de programas ensino via web: uma experiência do LATEC/UFRJ, [on-line] Disponível na Internet via WWW, IX Congresso Internacional de Educação a Distância,
URL: http://www.abed.org.br/congresso2002/trabalhos/texto30.htm, Acessado em 19 de fevereiro de 2003.

[FF2002] Fernandes, M. e Fernandes, J., Dinamizando as práticas pedagógicas através de um ambiente baseado na web, [on-line] Disponível na Internet via WWW, IX Congresso Internacional de Educação a Distância,
URL: http://www.abed.org.br/congresso2002/trabalhos/texto18.htm, Acessado em 02 de fevereiro de 2003.

[BKL2002] Barreto, L., Kawagoe, A. e Lacerda, F., Maestro: uma ferramenta de planejamento e desenvolvimento de conteúdo em formato hipertexto para e-learning, [on-line] Disponível na Internet via WWW, IX Congresso Internacional de Educação a Distância,
URL: http://www.abed.org.br/congresso2002/trabalhos/texto17.htm, Acessado em 02 de fevereiro de 2003.

[AF2002] Alves, F., e Furtado, E., Sistema de educação a distância baseada em serviços web, [on-line] Disponível na Internet via WWW, IX Congresso Internacional de Educação a Distância,
URL: http://www.abed.org.br/congresso2002/trabalhos/texto46.htm, Acessado em 02 de fevereiro de 2003.

[Romiszowski2002] Romiszowski, H., Avaliação no design e desenvolvimento de multimídia educativa: estratégia de apoio ou parte do processo?, [on-line] Disponível na Internet via WWW, VII Congresso Internacional de Educação a Distância,
URL: http://www.abed.org.br/congresso2000/texto08.doc, Acessado em 02 de fevereiro de 2003.

[Stefanelli2002] Stefanelli, J., A importância do profissional de comunicação gráfica na produção de material de EAD, [on-line] Disponível na Internet via WWW, VII Congresso Internacional de Educação a Distância, URL: http://www.abed.org.br/congresso2000/texto04.doc, Acessado em 02 de fevereiro de 2003.

[W3Schools2002] W3Schools online web tutorials, [on-line] Disponível na Internet via WWW, URL: http://www.w3schools.com, Acessado em 19 de fevereiro de 2003.

[Reed2000] Reed, P., Desenvolvendo aplicativos com Visual Basic e UML, Makron Books, 2000.

[Larman2000] Larman, C., Utilizando UML e padrões – Uma introdução à análise e ao projeto orientados a objetos, Bookman, 2000.

[Rational2001] Rational University, Rational Unified Process Fundamentals, Rational Software Corporation, 2001.

[KS1995] Korth, H. e Silberschatz, A., Sistema de banco de dados, 2º edição, Makron Books, 1995.

[LR2002] Lima, E. e Reis, E., C# e .Net – Guia do desenvolvedor, Campus, 2002.

[MS12002] Microsoft Official Curriculum, Programming in C#, Microsoft Corporation, 2001.

[MS22002] Microsoft Official Curriculum, Introduction to ASP .Net, Microsoft Corporation, 2001.

[MS32002] Microsoft Official Curriculum, Programming in Microsoft .Net Framework, Microsoft Corporation, 2001.

[Decio2001] Décio, O., XML – Guia de consulta rápida, Novatec, 2001.

[DM2002] Daum, B., e Merten, U., Arquitetura de sistemas com XML, Campus, 2002.

[Carlson2002] Carlson, D., Modelagem de aplicações XML com UML – Aplicações práticas de e-business, Makron Books, 2002.