UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO

(Bacharelado)

SISTEMA DE GERENCIAMENTO DE CONTEÚDO PARA AMBIENTE WEB

TRABALHO DE CONCLUSÃO DE CURSO SUBMETIDO À UNIVERSIDADE REGIONAL DE BLUMENAU PARA A OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA COM NOME EQUIVALENTE NO CURSO DE CIÊNCIAS DA COMPUTAÇÃO — BACHARELADO

ALEXANDRE DOS SANTOS MORATELLI

BLUMENAU, JUNHO/2002

2002/1-06

SISTEMA DE GERENCIAMENTO DE CONTEÚDO PARA AMBIENTE WEB

ALEXANDRE DOS SANTOS MORATELLI

ESTE TRABALHO DE CONCLUSÃO DE CURSO, FOI JULGADO ADEQUADO PARA OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA DE TRABALHO DE CONCLUSÃO DE CURSO OBRIGATÓRIA PARA OBTENÇÃO DO TÍTULO DE:

BACHAREL EM CIÊNCIAS DA COMPUTAÇÃO

_____

Prof. Alexander Roberto Valdameri — Orientador na FURB

_____

Prof. José Roque Voltolini da Silva — Coordenador do TCC

BANCA EXAMINADORA

_____

Prof. Alexander Roberto Valdameri

_____

Prof. Paulo Roberto Dias

_____

Prof. Francisco Adell Péricas

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

API – *Application Program Interface*
ASP – *Active Server Pages*
CASE – *Computer Aided Software Engineering*
CGI – *Common Gateway Interface*
DFD – Diagrama de Fluxo de Dados
GLP – GNU *General Public License*
GNU – GNU's *Not* Unix
IMAP – *Internet Message Access Protocol*
LDAP – *Lightweight Directory Access Protocol*
MER – Modelo Entidade Relacionamento
NSCA – *National Center for Supercomputing Application*
ODBC – *Open Database Connectivity*
RFC – *Requests for Comments*
SQL – *Structured Query Language*
SNMP – *Simple Network Management Protocol*
STD – *State Transition Diagram*
WDDX – *Web Development Data Exchange*
XML – *eXtensible Markup Language*

RESUMO

Este trabalho apresenta um estudo sobre o gerenciamento de conteúdo para ambiente *web*, através da utilização de banco de dados para o armazenamento das informações, controle de acesso ao conteúdo e *templates* para a exibição das informações. Apresenta também a especificação e implementação de um sistema de gerenciamento de conteúdo para ambiente *web*, desenvolvido na linguagem PHP, utilizando o servidor *web* Apache httpd e o banco de dados MySQL.

ABSTRACT

This work presents a study about content management to the web environment, through utilization of database for information storage, access control to the content and templates to information exhibition. It also presents the specification and implementation of one content management system to the web environment, developed in PHP language, using the Apache httpd web server and MySQL database.

1 INTRODUÇÃO

Basicamente falando, segundo AppliedTheory (2001), gerenciamento do conteúdo é o controle – administração, gerenciamento do fluxo, acesso ao conteúdo e segurança – das informações de uma organização (sejam elas textos, imagens, gráficos, áudio ou vídeo).

A explosão na utilização da Internet tornou possível e desejável às organizações disponibilizarem seus bancos de dados na w*eb*. A imensa quantidade de produtos baseados na *web* facilita o desenvolvimento de aplicações para pesquisar e manipular bancos de dados. Estes produtos têm como alvo reduzir o esforço de desenvolvimento na criação de interfaces de pesquisa baseados em formulários. Entretanto, formulários não são ferramentas efetivas de pesquisa para muitos usuários da Internet.

Segundo Miller (1997) os usuários da *web* diferem no mínimo em três importantes detalhes dos usuários tradicionais de aplicações de bancos de dados. Primeiro, não estão familiarizados com o funcionamento das aplicações e podem não possuir experiência na utilização da interface da aplicação. Segundo, podem ser intolerantes com interfaces inflexíveis, porque tipicamente possuem alternativas (outros *sites*) disponíveis que oferecem serviços similares. Eles esperam métodos de navegação efetivos e eficientes que minimizem a quantidade de interação com o sistema, principalmente quando suas requisições ao sistema causam erros ou são incompletas. Finalmente, podem possuir diferentes demandas do mesmo banco de dados.

A maioria dos *sites* é formada por conjuntos fixos de páginas criadas e interconectadas manualmente para organizar a informação. Utilizando esta metodologia, os usuários não precisam saber a especificação exata da informação que necessitam, muito menos sua estrutura ou forma de distribuição. Preferencialmente, eles podem navegar pelos breves sumários da informação e sucessivamente ver mais e mais informações detalhadas. Esta maneira de navegar entre a informação possibilita a um novo usuário uma maneira efetiva de encontrar a informação.

De acordo com Deshpande (2001) um *site* cresce rapidamente para milhares de arquivos de computador interligados através de *links*. Então, um desenvolvedor de *sites* ou aplicações não precisa apenas cuidar da funcionalidade da aplicação (lógica do processo e gerenciamento de dados), mas também da estrutura e dos elementos que a constituem em um nível mais detalhado (como documentos e *links*). *Links* podem estar inativos ou não por uma

variedade de motivos, desde um simples nome de arquivo alterado, uma omissão, até a reestruturação de uma parte do *site*. Os visitantes precisam estar seguros de que a informação atualizada há seis meses continua válida. Desenvolvedores de *sites* podem não se comprometer com as rotinas de manutenção, deixando-as para o pessoal administrativo. Desta forma, existe a necessidade da ciência da computação e a engenharia de software criarem novos métodos de gerenciamento de conteúdos disponibilizados na Internet.

Kirda (2001) comenta que os requisitos básicos na engenharia w*eb* são: escalabilidade para permitir futuros serviços; mecanismos de atualização simples para os responsáveis pelo conteúdo; modelo de navegação consistente para os usuários; mecanismos de controle de versões de conteúdo, funcionalidades e *layout;* suporte multi línguas e capacidade para integrar componentes de softwares e dados em bases de dados legadas.

Este trabalho consiste na especificação e desenvolvimento de um sistema de gerenciamento de conteúdo para ambiente w*eb*, possibilitando o fácil e rápido gerenciamento de um *site* (inserção, atualização e exclusão), realizando o controle de acesso de usuários às informações publicadas, assim como a possibilidade de publicação de qualquer tipo de arquivo e a pesquisa de qualquer conteúdo contido no *site*.

1.1 OBJETIVOS DO TRABALHO

O objetivo almejado por este trabalho consiste em facilitar o gerenciamento de conteúdo de um *site*, através da implementação de um sistema de gerenciamento para ambiente w*eb*, possibilitando ao usuário manter e disponibilizar facilmente informações de maneira centralizada e estruturada, permitindo o controle de acesso às informações no nível desejado e a pesquisa das mesmas. Utilizar-se-á uma base de dados para o armazenamento das informações utilizadas pelo sistema.

Os objetivos específicos do trabalho são:

a) facilidade no gerenciamento (inserção, atualização, remoção) de conteúdo de um *site*;
b) facilidade de pesquisa do conteúdo;
c) automatização na estruturação dos conteúdos;
d) utilização de *templates* para separar o formato gráfico de exibição do conteúdo armazenado;

e) segurança no acesso à informação baseada no controle de usuários.

1.2 ESTRUTURA DO TRABALHO

O primeiro capítulo apresenta uma contextualização e os objetivos do trabalho.

O segundo capítulo é dedicado às fundamentações e conceituações sobre o gerenciamento de conteúdo, que serão utilizadas no desenvolvimento deste trabalho, como o controle da informação, administração do conteúdo, gerenciamento do *workflow*, acesso ao conteúdo, segurança, customização, bem como integração com sistemas legados.

O terceiro capítulo descreve as ferramentas utilizadas, o servidor *web* Apache httpd, a linguagem PHP e o servidor de banco de dados MySQL.

O quarto capítulo é dedicado a estruturação necessária para o desenvolvimento do sistema até seu estágio final, destacando a metodologia utilizada, a modelagem do sistema e considerações sobre sua implementação.

O quinto capítulo apresenta as considerações finais, as suas limitações e sugestões para trabalhos futuros.

2 GERENCIAMENTO DO CONTEÚDO

Segundo Radoff (2000), em 1991 a *web* ainda limitava-se aos laboratórios de pesquisa e universidades norte- americanas, umas poucas dúzias de *sites*, com talvez algumas centenas de acessos por dia. Menos de uma década depois, existem milhões de *sites*, muitos com milhares de acessos por hora.

Com este crescimento, o desafio de fornecer serviços de qualidade aceitável para milhares de usuários preocupou as organizações de tecnologia da informação. Hardware e largura de banda adequada têm sido preocupações para grandes organizações que dependem da *web* como um veículo de missão crítica para atender seus objetivos.

Como a importância estratégica dos *sites* corporativos tem crescido, o gerenciamento adequado do conteúdo eficientemente tem se tornado uma das maiores dores de cabeça para os administradores. Antigas soluções para o gerenciamento do conteúdo eram freqüentemente organizacionais – um *webmaster* em breve se tornaria *webdesigner*, desenvolvedor e produtor. Como as pessoas descobriram que gastavam muito tempo, além do grau de dificuldade para atualizar o conteúdo ou mesmo automatizar a publicação, muitas companhias desenvolveram suas próprias aplicações (utilizando CGI's, *scripts perl* ou aplicações servidoras) para resolver estes problemas.

Há não muito tempo atrás, de acordo com AppliedTheory (2001), os *sites* eram um custo, de resultados questionáveis para as organizações. Páginas estáticas, gráficos e interfaces grandes e desajeitadas eram a regra para as organizações apressadas em criar uma presença *online*. Como resultado, gerenciar e atualizar um *site* tornou-se algo caro que consumia muito tempo.

Hoje os *sites* são o coração de muitas operações de uma organização. Como parte de uma estratégia de *e-business*, um *site* pode ajudar uma companhia a atingir importantes objetivos, como:

a) proporcionar o marketing cliente a cliente;
b) conquistar e manter consumidores;
c) conduzir pesquisas de mercado;
d) educar consumidores;

e) proporcionar uma melhor comunicação com seus empregados;
f) oferecer serviços aos clientes;
g) comunicar-se com fornecedores;
h) gerar liderança de vendas.

Um fator crítico para o sucesso de qualquer *site* é a qualidade da informação nele disponível e a usabilidade de sua *interface* com o usuário. Seu conteúdo deve ser preciso, atualizado, intuitivo, e bem organizado para ser utilizado pelo público alvo.

Quando o conteúdo não atinge um destes critérios, sua credibilidade, e de seus patrocinadores podem ser questionada rápida e gravemente. O sucesso de um *site* depende fortemente do conteúdo e do seu gerenciamento.

Uma compreensão dos fatores críticos para o sucesso é importante, mas pode ser difícil começar na estrada do sucesso do *e-business*. Para isto, existem diversas definições de gerenciamento do conteúdo, assim como vendedores de ferramentas de gerenciamento de conteúdo. Também existem diferentes estratégias para gerenciar e aprimorar o conteúdo de uma organização, e o *e-business* está repleto de estratégias que não funcionaram.

2.1 CONTROLE DA INFORMAÇÃO

Basicamente falando, o gerenciamento do conteúdo é o controle – administração, gerenciamento do fluxo, acesso ao conteúdo e segurança – das informações de uma organização (sejam elas textos, imagens, gráficos, áudio ou vídeo).

Correta e estrategicamente implantada, o gerenciamento do conteúdo pode ajudar uma organização tornar-se mais competitiva, pela redução de despesas e aumento da sua rentabilidade. Pode ajudar a reduzir os custos de manutenção de um *site* através da definição de um *workflow*, automatizando tarefas e reduzindo a necessidade de softwares, suporte e treinamento. Outro benefício é a redução no tempo de produção desde a concepção até o conteúdo final da publicação.

Uma vez que a organização absorveu o pensamento do gerenciamento de conteúdo, o mesmo pode ser publicado em diferentes formatos maximizando a oportunidade de penetrar

em diversos segmentos. Também possibilita novos rendimentos através da utilização de serviços eletrônicos, como notícias e marketing por *email*.

2.2 COMPONENTES CHAVES NO GERENCIAMENTO DO CONTEÚDO

AppliedTheory (2001) relata que os quatro maiores componentes no gerenciamento de conteúdo são a administração do conteúdo, gerenciamento do *workflow*, acesso e segurança, bem como a customização e integração com sistemas legados.

A seguir existe uma pequena descrição de cada um destes componentes, delimitando assim sua importância no gerenciamento do conteúdo.

2.2.1 ADMINISTRAÇÃO DO CONTEÚDO

Consiste de todas as ferramentas necessárias para adicionar, modificar e remover o conteúdo do *site* de uma organização. Historicamente, administrar o conteúdo era uma tarefa desgastante e intensiva, envolvendo na maioria das vezes páginas estáticas, e, na melhor das hipóteses, ferramentas que automatizaram parte das tarefas. A diferença do passado e de hoje são as sofisticadas ferramentas de gerenciamento do conteúdo que utilizam bancos de dados.

2.2.1.1 ADMINISTRAÇÃO DO CONTEÚDO UTILIZANDO BANCO DE DADOS

Hoje em dia, o conteúdo é armazenado em um banco de dados junto com outras informações relacionadas, como seu autor, editor, aprovador, data inicial e final de publicação, histórico de revisões e usuários que podem visualizá-las. A informação é inserida e atualizada intuitivamente, através de interfaces baseadas em formulários.

O especialista em conteúdo de uma organização pode rápida e facilmente atualizar conteúdos sem precisar aprender programação ou técnicas de desenvolvimento da *web*. A administração do conteúdo utilizando banco de dados auxilia ainda mais os especialistas a aumentar a qualidade e diminuir o custo da publicação da informação. Outro grande benefício da utilização de bancos de dados na administração do conteúdo é que o conteúdo é armazenado separado do seu formato de publicação. Como resultado, o conteúdo pode ser publicado em diferentes páginas, ou até mesmo diferentes *sites*, utilizando *templates*, ou

modelos de formatos para cada diferente publicação. Os *templates* são um conjunto de regras, estilos e formato gráfico de apresentação de um *site*.

A administração de conteúdo utilizando banco de dados automatiza o processo de publicação enquanto garante consistência e uma aparência profissional para todo conteúdo. Mudanças no conteúdo são realizadas uma única vez e se refletem automaticamente em todas as publicações que aquele assunto aparece. Assim, um *site* pode manter-se recente através de alterações periódicas nos seus *templates* de exibição, ao invés de modificar o conteúdo.

A utilização de *templates* pode ser consistente com os padrões de comunicação da organização (ou identidade visual). Muitas organizações melhoram sua imagem e aumentam o reconhecimento de suas marcas aderindo a padrões específicos de formato e estilo nas suas comunicações externas. Isto inclui regras para a utilização do logotipo da organização, estilos e tamanho de fontes, *layouts* de páginas, esquemas de cores e regras editoriais.

A utilização de banco de dados na administração de conteúdo é essencial para tornar o conteúdo de uma organização e garantir uma aparência consistente profissional ao seu *site*. Também é importante ressaltar que a administração do conteúdo distribui o trabalho de criação para os maiores conhecedores do conteúdo dentro da organização, diminuindo assim a sobrecarga de trabalho do *webmaster*.

2.2.2 GERENCIAMENTO DO WORKFLOW

Normalmente, as organizações possuem políticas específicas sobre quem pode criar, editar e aprovar conteúdo para publicação. Infelizmente, muitas delas precisam lidar com estas funções através de discussões cara-a-cara, *emails*, conversas telefônicas, memorandos e outros processos ineficientes e sujeitos a erros. Devido ao tempo que estes processos tomam, revisar e aprovar conteúdo muitas vezes ocorre paralelamente à publicação deste conteúdo, ou até mesmo depois que o público já viu o conteúdo que não seria aprovado. Além disso, é virtualmente impossível rastrear o andamento destes processos em um certo momento.

O bom gerenciamento de conteúdo permite a automação dos processos de trabalho de uma organização. Tipicamente, processos como revisar editoriais, aprovar e publicar e entregar conteúdo, são beneficiados com a automação. A automatização do *workflow* facilita a interação entre criadores de conteúdo, editores, revisores e as aplicações da organização.

Processos de *workflow* podem ser iniciados pelos usuários para eventos específicos, ou outros critérios. É possível, por exemplo, uma aplicação automatizada gerar conteúdo e alimentar o processo de *workflow* adequado. O *workflow* também pode integrar clientes, vendedores e outros parceiros em seus processos.

Gerenciamento automatizado de *workflow* reduz o tempo total gasto desde o conceito até a publicação do conteúdo, acelerando o processo de revisão e aprovação. Ao mesmo tempo, aumenta a qualidade do conteúdo (e da imagem da organização) assegurando que o mesmo não foi publicado sem antes ter sido revisado e aprovado.

2.2.3 ACESSO AO CONTEÚDO E SEGURANÇA

O gerenciamento de conteúdo ajuda a assegurar a segurança da informação de uma organização pelo controle no acesso a publicação (para criar, editar e aprovar conteúdo) e acesso de leitura (para o público interno e externo). Segurança no acesso ao conteúdo aprimora, mas não substitui, os mecanismos de segurança de um *site* e/ou da rede da organização.

O controle de acesso na publicação especifica quem na organização tem acesso a diferentes categorias de informação e qual o seu nível de acesso. Com um esquema de controle de acesso na publicação, uma organização pode assegurar que as pessoas tenham acesso eficiente à informação desejada.

O controle de acesso à leitura controla as pessoas, dentro ou fora da organização, que terão permissão a acessar diferentes categorias de conteúdos na internet, intranet ou extranet da organização. Permissões de acesso devem ser especificadas individualmente, por departamentos e ou grupos, ou de acordo com as necessidades de segurança da organização.

O controle e segurança no acesso garantem, tanto para direitos de acesso a publicação ou leitura, que as pessoas tenham acesso ao conteúdo apropriado. Trabalham conjuntamente com o gerenciamento do *workflow* para garantir o acesso no ciclo de vida do conteúdo (criação, edição, revisão, aprovação, publicação, leitura, remoção, arquivamento).

2.2.4 CUSTOMIZAÇÃO E INTEGRAÇÃO COM SISTEMAS LEGADOS

Nenhuma organização é igual à outra. Elas diferem pela sua manufatura, práticas comerciais, políticas e procedimentos, objetivos e estrutura organizacional. Elas possuem seus ambientes computacionais, geralmente chamados sistemas legados, que também são diferentes. Um sólido gerenciamento de conteúdo precisa ser customizável para atender as necessidades específicas de uma organização e coexistir com os sistemas legados. Especificamente, um sistema de gerenciamento de conteúdo precisa se integrar totalmente com a infraestrutura de informação existente – os usuários devem poder utilizar o sistema de gerenciamento de conteúdo e seus sistemas legados como se fosse um único e grande sistema, adaptado para suas necessidades.

Tipicamente, os sistemas de gerenciamento de conteúdo precisam se integrar como sistemas de planejamento de recursos (E*nterprise Resource Planning – ERP*), sistemas de gerenciamento de *call center* ou outras aplicações internas ou externas (parceiros de negócio). O sistema de gerenciamento de conteúdo também pode ser integrado com um sistema de gerenciamento do relacionamento com o cliente (C*ustomer Relationship Management – CRM*) ou sistemas de estoque e financeiro, para criar aplicações de *e-commerce*.

O gerenciamento de conteúdo pode ser uma ponte entre potenciais usuários do conteúdo de uma organização e entre o conteúdo novo e o legado (através dos sistemas) da organização. Os usuários recebem informações personalizadas de acordo com suas necessidades e são autorizados a ver onde a informação é armazenada. Um sistema de gerenciamento de conteúdo deve esconder dos usuários a diferença entre conteúdo e aplicações, criando a desejável ilusão de um único sistema com uma única interface para o usuário.

O próximo capítulo descreve as ferramentas utilizadas no desenvolvimento de uma solução que contempla os componentes aqui descritos no gerenciamento de conteúdo.

3 TÉCNICAS E FERRAMENTAS UTILIZADAS

O processo de desenvolvimento de sistemas normalmente segue uma série de etapas e procedimentos relativos a cada uma delas. Existe uma série de metodologias que se aplicam geralmente a sistemas tradicionais ou sistemas de informação. A metodologia usada neste trabalho será a análise estruturada.

Este capítulo mostra os conceitos da análise estruturada e algumas ferramentas para o desenvolvimento do protótipo deste trabalho.

3.1 ANÁLISE ESTRUTURADA

A análise estruturada, como todos os métodos de análise de requisitos de software, é uma atividade de construção de modelos. Usando uma notação que é própria ao método de análise estruturada, criam-se modelos que retratam o fluxo e o conteúdo da informação, dividi-se o sistema em partições funcionais e comportamentais e descreve-se a essência daquilo que deve ser construído (Pressman 1995).

3.1.1 AS FASES DA ANÁLISE ESTRUTURADA

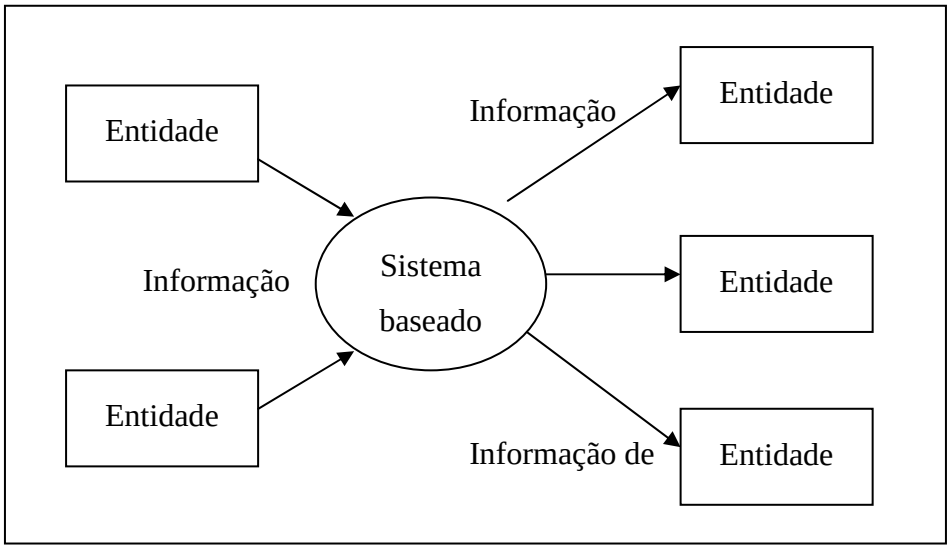
Serão examinados cada um dos passos que devem ser aplicados ao desenvolvimento de modelos completos e precisos por meio da análise estruturada.

3.1.1.1 DIAGRAMA DE FLUXO DE DADOS (DFD)

De acordo com Pressman (1995), à medida que se movimenta pelo software a informação é modificada por uma série de transformações. Um diagrama de fluxo de dados é uma técnica gráfica que descreve o fluxo de informação e as transformações que são aplicadas à medida que os dados se movimentam da entrada para a saída.

O DFD pode ser usado para representar um sistema ou software em qualquer nível de abstração. De fato, os DFDs podem ser divididos em partições de acordo com níveis que representem um crescente detalhamento funcional e do fluxo de informação. A forma básica de um diagrama de fluxo de dados é ilustrada na figura 1.

Figura 1 – Diagrama de Fluxo de Dados



Fonte: adaptado de PRESSMAN (1995), pg 279

3.1.1.2 MODELAGEM COMPORTAMENTAL

Segundo (Pressman 1995), a modelagem comportamental é um dos princípios fundamentais para todos os métodos de análise de requisitos. Contudo, somente as versões estendidas da análise estruturada oferecem uma notação para este tipo de modelagem. O Diagrama de Transição de Estado (*State Transition Diagram* - STD) representa o comportamento de um sistema, descrevendo seus estados e os eventos que fazem com que o sistema mude de estado. Além disso, o STD indica quais ações (por exemplo, ativação de processos) são executadas como conseqüência de um dado evento.

Para elaborar um STD, algumas definições são necessárias: estado, transição, ação e condição.

a) **estado:** de um sistema representa uma situação, um cenário ou um modo de comportamento em que encontramos um sistema ao observá-lo em determinado momento. Assim, se o sistema que estivermos observando for uma lâmpada elétrica, podemos encontrá-la em um dos seguintes estados: acesa e apagada;
b) **transição:** representa a passagem do sistema de um estado para outro. Assim, ao acendermos ou ao apagarmos uma lâmpada, estaremos provocando uma transição de estado;
c) **ação:** representa a atividade do sistema que efetua a transição de estado;

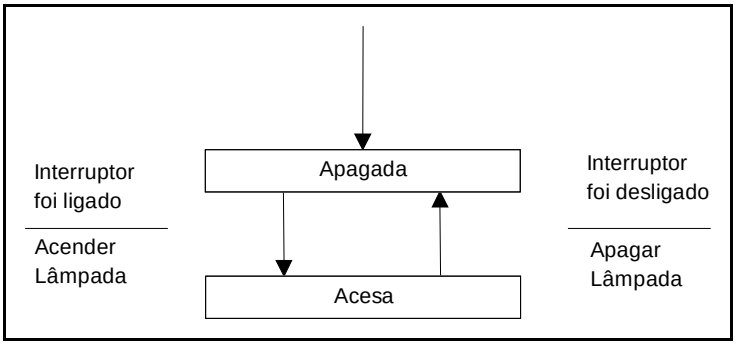
d) **condição:** representa a causa necessária para que haja a transição de estado. Decorre da ocorrência de um evento ou circunstância que propicia a transição de estado. Assim, ao apertarmos o interruptor, provocamos a condição para que seja executada a ação de acender ou de apagar a lâmpada, provocando desta forma uma transição de estado.

Para representar graficamente o modelo de transição de estados, diversos tipos de diagramas têm sido usados. Um deles é apresentado abaixo. Neste diagrama, usam-se as seguintes representações:

a) retângulos significam estados;
b) setas significam transições válidas (mudanças de estados);
c) uma linha horizontal ao lado da seta que representa a transição significa o par condição/ação, onde a condição aparece acima da linha mencionada e a ação desencadeada pela ocorrência da condição ou evento aparece abaixo da linha.

Exemplo: seja o caso simples de uma lâmpada. O STD é apresentado na figura 2:

Figura 2 – Diagrama de Transição de Estado



Fonte: adaptado de PRESSMAN (1995), pg 293

Neste diagrama, pode-se distinguir:

a) dois estados: apagada e acesa;
b) há duas transições possíveis: de apagada para acesa e de acesa para apagada;
c) há dois pares de condições/ações: a condição interruptor foi ligado, que dispara a ação acender lâmpada, provocando a transição do estado de apagada para acesa;
d) a condição interruptor foi desligado, que dispara a ação apagar lâmpada, provocando a transição de estado de acesa para apagada.

No diagrama apresentado visualiza-se uma seta na parte superior. Esta seta costuma ser usada por alguns autores para identificar o estado inicial do sistema.

3.1.1.3 MODELO ENTIDADE E RELACIONAMENTO

Segundo João (1993), o modelo entidade-relacionamento é um diagrama que detalha as associações existentes entre as entidades de dados e utiliza componentes próprios de semântica. Com ênfase para as entidades com que o sistema lida, bem como a relação entre as entidades.

Para Gane (1993), a modelagem de dados é necessária para melhor visualizar os relacionamentos entre os depósitos de dados com o objetivo de eliminar redundâncias.

3.1.1.4 DICIONÁRIO DE DADOS

Uma análise do domínio da informação seria incompleta se somente o fluxo dos dados fosse considerado. Cada depósito de dados freqüentemente é uma coleção de itens de dados individuais. Por conseguinte, algum método para representar o conteúdo de cada componente do DFD deve estar à disposição do analista.

O dicionário de dados foi proposto como uma gramática quase formal para descrever o conteúdo de objetos definidos durante a análise estruturada.

É uma listagem organizada de todos os elementos de dados que são pertinentes ao sistema, com definições precisas e rigorosas, de forma que tanto o usuário como o analista de sistemas tenham uma compreensão comum das entradas, das saídas, dos componentes dos depósitos de dados e dos cálculos intermediários.

O dicionário de dados quase sempre é implementado como parte de uma ferramenta CASE (*Computer Aided Software Engineering*). Embora o formato dos dicionários varie de ferramenta para ferramenta, a maioria contém as seguintes informações:

a) nome: o nome principal do item de dados ou item de controle, do depósito de dados ou de uma entidade externa;
b) alias: outros nomes usados para a primeira entrada;
c) descrição de conteúdo: uma notação para representar o conteúdo;

d) informação complementar: outras informações sobre tipos de dados, valores previamente estabelecidos, restrições ou limitações.

3.2 APACHE HTTPD

Conforme The Apache Software Foundation (2002a), o projeto Apache *Httpd* é um esforço de desenvolvimento colaborativo de software com o objetivo de criar um servidor *web* robusto, de qualidade comercial, e com o código fonte disponível gratuitamente. É um projeto gerenciado por um grupo de voluntários localizados ao redor do mundo, utilizando a internet para se comunicar, planejar e desenvolver o servidor e sua documentação. Estes voluntários são conhecidos como o Apache *Group*. Outras centenas de usuários têm contribuído com o projeto através de idéias, códigos e documentação.

Em fevereiro de 1995, o servidor de domínio público mais popular na *web* era o Apache, desenvolvido por Rob McCool no *National Center for Supercomputing Applications* (NCSA), localizado na Universidade de Illinois, Urbana-Champaign. Mas o desenvolvimento do servidor tinha praticamente parado quando Rob deixou o NCSA no meio de 1994, e muitos *webmasters* haviam desenvolvido as extensões e correções que eram necessárias em uma única distribuição. Um pequeno grupo destes *webmasters*, contatado via *email*, reuniu-se com o propósito de coordenar as mudanças necessárias no servidor (na forma de *patches*). Brian Behlendorf e Cliff Skolnick criaram uma lista de discussão, publicaram informações e criaram um local único para o desenvolvimento do núcleo, utilizado pelos desenvolvedores. No fim de fevereiro, oito colaboradores no desenvolvimento do *core* formaram o Apache *Group*.

Utilizando o servidor NCSA httpd 1.3 como base, eles adicionaram as correções e aprimoramentos que achavam necessário, testando o resultado em seus próprios servidores, e criando a primeira versão pública (0.6.2) do servidor Apache em abril de 1995. Por coincidência, a NCSA reiniciou o desenvolvimento do seu servidor durante o mesmo período, e os desenvolvedores do NCSA tornaram-se membros honorários do projeto Apache, compartilhando assim idéias e correções.

Segundo Netcraft (2002), menos de um ano depois do Apache *Group* estar formado, o servidor Apache tomou a liderança do NCSA httpd como servidor mais utilizado, posição que ocupa até hoje**.**

Em 1999, membros do Apach*e Group* formaram a Apache *Software Foundation*, para fornecer suporte organizacional, legal e financeiro para o servidor Apache. Ela tem colocado o software em um patamar sólido para desenvolvimento futuro.

3.2.1 DESCRIÇÃO E CARACTERÍSTICAS

The Apache Software Foundation (2002b), diz que o servidor Apache httpd *Server* é poderoso, flexível, e implementa os mais recentes protocolos, incluindo o HTTP/1.1 (RFC2616). É altamente configurável e extensível com módulos de terceiros. Pode ser personalizado através do desenvolvimento de módulos que utilizem a API do Apache. Fornece todo o código fonte e vem com uma licença irrestrita para utilização. É compatível com o Windows NT, Windows 95, Netware 5.x, OS/2 e muitas versões do Unix, assim como outros sistemas operacionais. Está em desenvolvimento constante. Implementa muitas características necessárias, incluindo:

a) utilização de bancos de dados para autenticação;
b) personalizar mensagens de erros e problemas;
c) configurar diversos arquivos como documentos quando um endereço é requisitado;
d) redirecionamento flexível e ilimitado de *aliases* e endereços;
e) permite a utilização de diferentes domínios, através da distinção entre requisições feitas para diferentes IP's ou endereços, mapeados na mesma máquina;
f) *logs* configuráveis e confiáveis.

3.3 PHP

Segundo The PHP Group (2002) a linguagem PHP foi concebida em meados de 1994 por Rasmus Lerdorf. Versões iniciais não lançadas foram usadas em sua *home page* para manter uma trilha de quem estava olhando o seu currículo virtual. A primeira versão usada por outros estava disponível em alguma época de 1995 e era conhecida como o *Personal Home Page Tools*. Consistia de um interpretador muito simplista que apenas fornecia algumas macros especiais e um número de utilidades que eram de uso comum em *home pages* daquela época. Um cadastro de visitantes, um contador, e algumas outras coisas. O interpretador foi reescrito em meados de 1995 e batizado PHP/FI Version 2. O FI veio de outro pacote que Rasmus tinha escrito que interpretava dados de formulários HTML. Ele combinou os scripts do *Personal Home Page Tools* com o *Form Interpreter* e adicionou suporte mSQL e o

PHP/FI nasceu. PHP/FI cresceu a um passo fantástico e pessoas começaram a contribuir para o código.

De acordo com The PHP Group (2002) é difícil dar qualquer estatística, mas é estimado que ao final de 1996 o PHP/FI estava em uso em pelo menos 15.000 *sites* ao redor do mundo. Em meados de 1997 este número tinha crescido para mais de 50.000. Em 1997 também houve uma mudança no desenvolvimento do PHP. Deixou de ser o projeto pessoal de Ramus para o qual muitas pessoas tinham contribuído, para se tornar em um esforço de um time mais organizado. O interpretador foi reescrito do zero por Zeev Suraski e Andi Gutmans e este novo interpretador formou a base do PHP versão 3. Muito do código de utilitários do PHP/FI foi portado para o PHP 3 e muito dele foi completamente reescrito.

3.3.1 DESCRIÇÃO E CARACTERÍSTICAS

O PHP é uma linguagem de padrão aberto, *server-side*, de *script* para criação de páginas dinâmicas para *e-commerce* e outras aplicações *web*, de acordo com Zend Technologies (2002). Uma página dinâmica é aquela que interage com o usuário, então cada usuário que visita uma determinada página receberá informações personalizadas. Páginas dinâmicas prevalecem em *sites* comerciais (*e-commerce*), onde o conteúdo exibido é gerado através de informações recuperadas de uma base de dados ou outra fonte externa.

O PHP oferece uma maneira simples, universal e fácil para a criação de páginas dinâmicas. Sua interface intuitiva permite aos programadores inserir comandos PHP diretamente em uma página HTML. Sua sintaxe é similar ao C e Perl, tornando fácil seu aprendizado por qualquer pessoa que possua um mínimo de experiência em programação. Seu *design* elegante torna significativamente fácil manter e aperfeiçoar uma aplicação, se comparado com aplicações desenvolvidas em outras linguagens.

Devido a sua distribuição para uma grande comunidade de usuários, a linguagem PHP é muito bem suportada. Como um produto de padrão aberto, a linguagem PHP recebe o suporte de um grande número de desenvolvedores de software de padrão aberto. A comunidade de desenvolvedores fornece uma excelente suporte técnico aos usuários, e os *bugs* encontrados são corrigidos rapidamente. O código é continuamente atualizado com melhorias e extensões da linguagem para expandir sua capacidade.

A versão 3, disponibilizada em junho de 1998, ganhou rápida popularidade e atualmente é utilizada por algumas das mais importantes organizações, como Mitsubishi, Redhat, Der Spiegel, MP3-Lycos, Ericsson e NASA.

Diferentemente de outras linguagens de script para desenvolvimento na *web*, PHP oferece excelente conectividade com a maioria dos bancos de dados mais comuns (incluindo Oracle, Sybase, MySQL, ODBC, e outros). Também oferece integração com várias bibliotecas externas, o que permite ao desenvolvedor realizar projeto, desde a geração de documentos no formato PDF, até a análise de XML. Talvez a maior vantagem de PHP, quando comparada com outras linguagens de *script*, como ASP ou ColdFusion, é o fato de ser de padrão aberto e independente de plataforma, satisfazendo as atuais necessidades nos diversos ambientes de rede heterogêneos.

Também é a escolha natural para quem desenvolve em Linux e utiliza o servidor Apache httpd, mas funciona igualmente bem em qualquer outro sistema operacional, como o UNIX ou Windows, com servidores *web* da Netscape ou Microsoft. O PHP suporta sessões http, conectividade com Java, expressões regulares, os protocolos LDAP, SNMP, IMAP, COM (sobre o Windows). Também suporta WDDX para troca de informações entre virtualmente todas as linguagens de programação para *web*.

3.4 MYSQL

É o sistema gerenciador de banco de dados relacional de padrão aberto mais popular, desenvolvido e fornecido pela companhia comercial MySQL AB.

MySQL AB Company (2002) comenta que um banco de dados é uma coleção estruturada de dados. Pode ser desde uma simples lista de compras para uma loja, até grande quantidade de informação de uma rede corporativa. Um banco de dados relacional armazena os dados em tabelas separadas, oferecendo velocidade e flexibilidade. As tabelas são interligadas através de relacionamentos, tornando possível criar combinações de dados de várias tabelas em uma requisição. O sufixo SQL, do MySQL, refere-se a linguagem SQL, o padrão mais comum utilizado para acessar bancos de dados.

O fato de MySQL ser um software de padrão aberto significa que qualquer um pode utilizá-lo sem pagar. Alguém interessado pode estudar o código fonte e alterá-lo de acordo

com suas necessidades. MySQL utiliza a licença GPL (GNU *General Public License*), para definir o que se pode ou não fazer com o software em diferentes situações. Se você sentir inseguro com a licença GPL, ou precisa utilizar o MySQL em uma aplicação comercial, pode-se comprar uma versão comercial licenciada.

O servidor de banco de dados MySQL é rápido, confiável e muito fácil de usar. Foi desenvolvido originalmente para manipular grandes bases de dados mais rapidamente do que as soluções existentes e tem sido utilizado em ambientes de desenvolvimento de alta necessidade por muitos anos. Através do constante desenvolvimento, MySQL oferece hoje uma ampla gama de funcionalidades. A conectividade, velocidade e segurança tornam o MySQL altamente desejável para bancos de dados acessadas através da internet.

O MySQL é um sistema cliente/servidor que consiste de um servidor SQL *multi-threaded* que suporta diferentes ambientes, programas clientes e bibliotecas, ferramentas administrativas e uma ampla gama de interfaces de programação (APIs). Também é fornecido como uma biblioteca *multi-threaded* que permite tornar uma aplicação pequena, rápida e fácil de gerenciar.

3.4.1 DESCRIÇÃO E CARACTERÍSTICAS

A lista a seguir descreve algumas das características mais importantes do MySQL:

a) escrito na linguagem C e C++. Testado com uma ampla gama de diferentes compiladores;
b) funciona em diferentes plataformas, dentre elas UNIX, Linux e Windows;
c) possui API's para as linguagens C, C++, Eiffel, Java, Perl, PHP, Python e Tcl;
d) totalmente *multi-threaded* utilizando *kernel threads*. Isto significa que ele pode utilizar o poder de processamento de múltiplas CPU's, se disponível;
e) rápido acesso a tabelas, armazenadas em disco através de árvores binárias com compressão de índices;
f) sistema de alocação de memória rápido baseado em *threads*;
g) funções SQL implementadas através de uma biblioteca de classes altamente otimizadas;
h) diversos tipos de colunas: inteiros com ou sem sinal com 1, 2, 3, 4 e 8 bytes de tamanho, *floats*, *doubles*, *chars*, *varchar*, *text*, *blob*, *date*, *time*, *datetime*, *timestamp*,

year, *set* e *enum*;

i) registros de tamanho fixo e variável;

j) todas as colunas possuem valores padrão. Você pode inserir um subconjunto de colunas em uma tabela e as colunas que não foram especificadas explicitamente terão seus valores atribuídos com seus valores padrão;

k) sistema de senhas e privilégios flexível e seguro, permitindo autenticação baseada em servidor. Senhas são seguras porque todo o tráfico de senha é criptografado quando é realizada a conexão com o servidor;

l) manipula grandes quantidades de dados;

m) é permitido um máximo de 32 índices por tabela. Cada índice pode consistir de 1 a 16 colunas ou partes de colunas. O tamanho máximo do índice é 500 bytes, mas pode ser alterado através da compilação do MySQL;

n) clientes podem conectar utilizando *sockets* do protocolo TCP/IP, Unix *Sockets* (Unix) ou *Named Pipes* (Windows NT);

o) suporte para ODBC (*Open-DataBase-Connectivity*) para Win32.

3.5 SYSTEM ARCHITECT

De acordo com Popkin Software (2002), o *System Architect* suporta diversas técnicas em Análise Estruturada, a saber, Gane & Sarson, *Information Engineering*, SSADM IV, Yourdan/De Marco e Ward & Mellor.

O *System Architect* permite um nivelamento automático de processos e subprocessos nas explosões no DFD. Por exemplo, além de sincronizar as entidades do modelo ER com depósitos de dados, há também a geração automática de diagramas de decomposição para representar uma estrutura hierárquica (de funções e de processos).

Outro recurso automatizado é a conversão de fluxos de controle no DFD do Ward & Mellor em condições e ações nos diagramas de transição de estados.

Para que os analistas / desenvolvedores possam lidar melhor com o *System Architect* e para que ele seja uma ferramenta de auxílio, a Popkin Software vem incrementando, versão a versão, novos recursos em checagem de integridade. Balanceamento vertical, entre processos-pai e processos-filho, e outras checagens de modelos seguindo regras de verificação, sempre

foram recursos disponíveis no *System Architect*. Tais checagens podem ou não ser automáticas.

Implementação do conceito de modelos e sub-modelos, ou *subject areas*. Com este conceito é possível montar projetos corporativos, trabalhando com *views* do modelo.

O *System Architect* trabalha com modelos separados para o MER-Lógico e o MER-Físico. Isto permite que o analista faça o modelo lógico com suas características de normalização e não redundância, mantendo sua coerência lógica.

Já no modelo físico o analista pode adaptar o modelo visando a sua implementação no banco de dados, desnormalizando e duplicando tabelas, por exemplo. É possível criar mais de um modelo físico a partir de um único modelo lógico, e isto é importante ao se trabalhar com mais de um sistema gerenciador de banco de dados.

Entre as facilidades na modelagem de dados, pode-se citar diagramação padrão em ER, herança de propriedades de dados a partir de *data domains* definidos pelos usuários, construir *constraints* de integridade referencial e *triggers* customizados, e propagação automática de chaves estrangeiras.

A seguir descreve-se o desenvolvimento do trabalho.

4 DESENVOLVIMENTO DO TRABALHO

Neste capítulo são apresentadas a especificação, modelagem, ferramentas utilizadas e a funcionalidade do sistema desenvolvido.

4.1 REQUISITOS PRINCIPAIS DO PROBLEMA A SER TRABALHADO

Um sistema de gerenciamento de conteúdo deve permitir a manutenção do conteúdo a ser disponibilizado, a manutenção de grupos, departamentos e usuários para possibilitar um efetivo controle de acesso ao conteúdo armazenado, além da separação do formato de exibição das informações a serem disponibilizadas.

4.2 ESPECIFICAÇÃO

O sistema de gerenciamento de conteúdo disponibiliza o conteúdo armazenado e operações nestes conteúdos ao usuário.

A metodologia utilizada no desenvolvimento da especificação foi a análise estruturada. Utilizou-se a técnica de Gane & Sarson na elaboração dos diagramas.

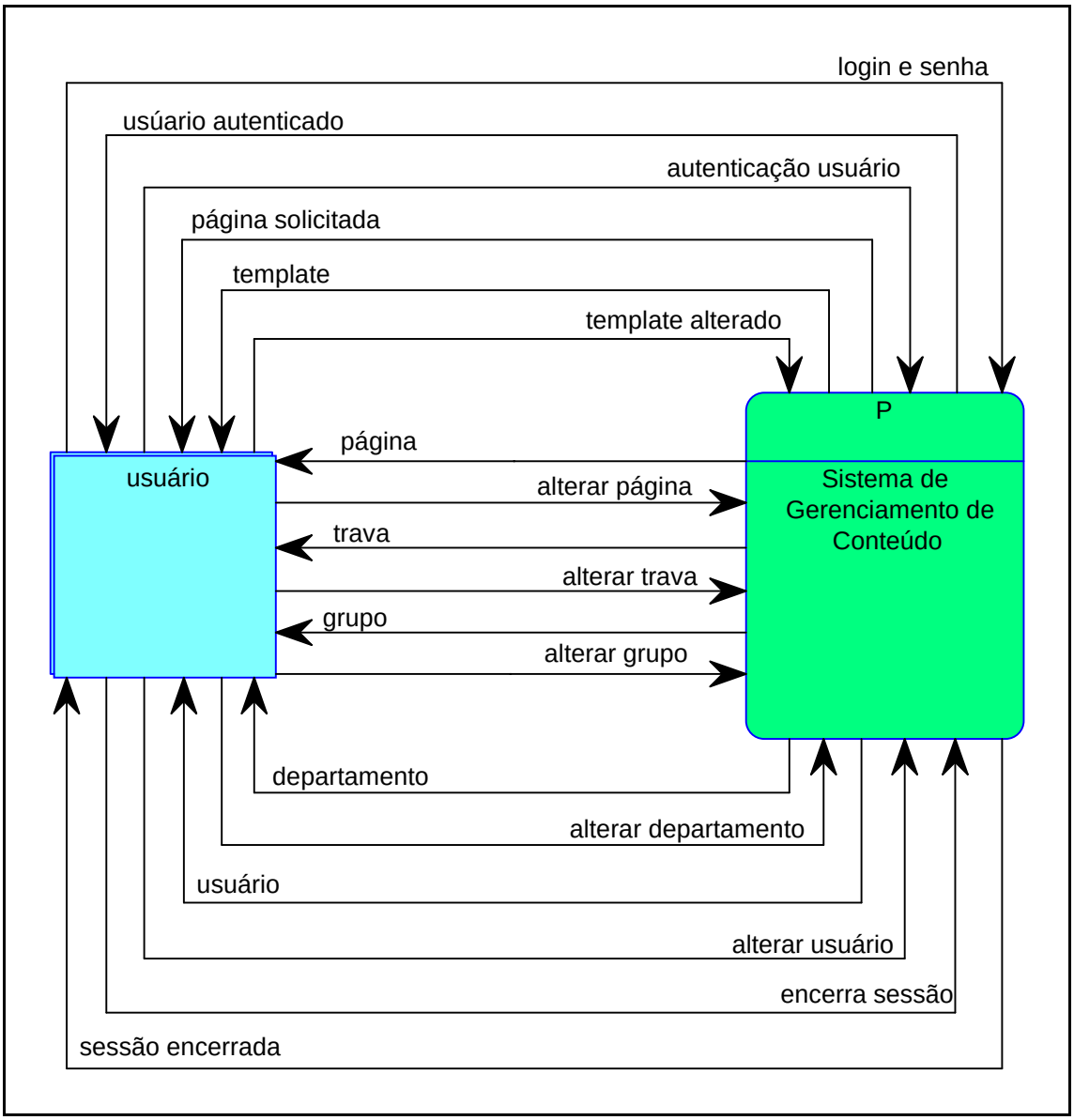
4.2.1 FERRAMENTA UTILIZADA

Para especificação do protótipo foi utilizada a ferramenta *System Architect*. Foram criados os diagrama de contexto, de fluxo de dados nível 0, modelo entidade relacionamento (MER) e o dicionário de dados das tabelas utilizadas pelo sistema de gerenciamento de conteúdo.

4.2.2 DIAGRAMA DE CONTEXTO

No diagrama de contexto (figura 3) apresenta-se a entidade externa envolvida no sistema de gerenciamento de conteúdo e seus relacionamentos com o sistema. Tem-se como entidade externa somente o usuário, pois o objetivo do sistema é fornecer ao usuário o conteúdo armazenado.

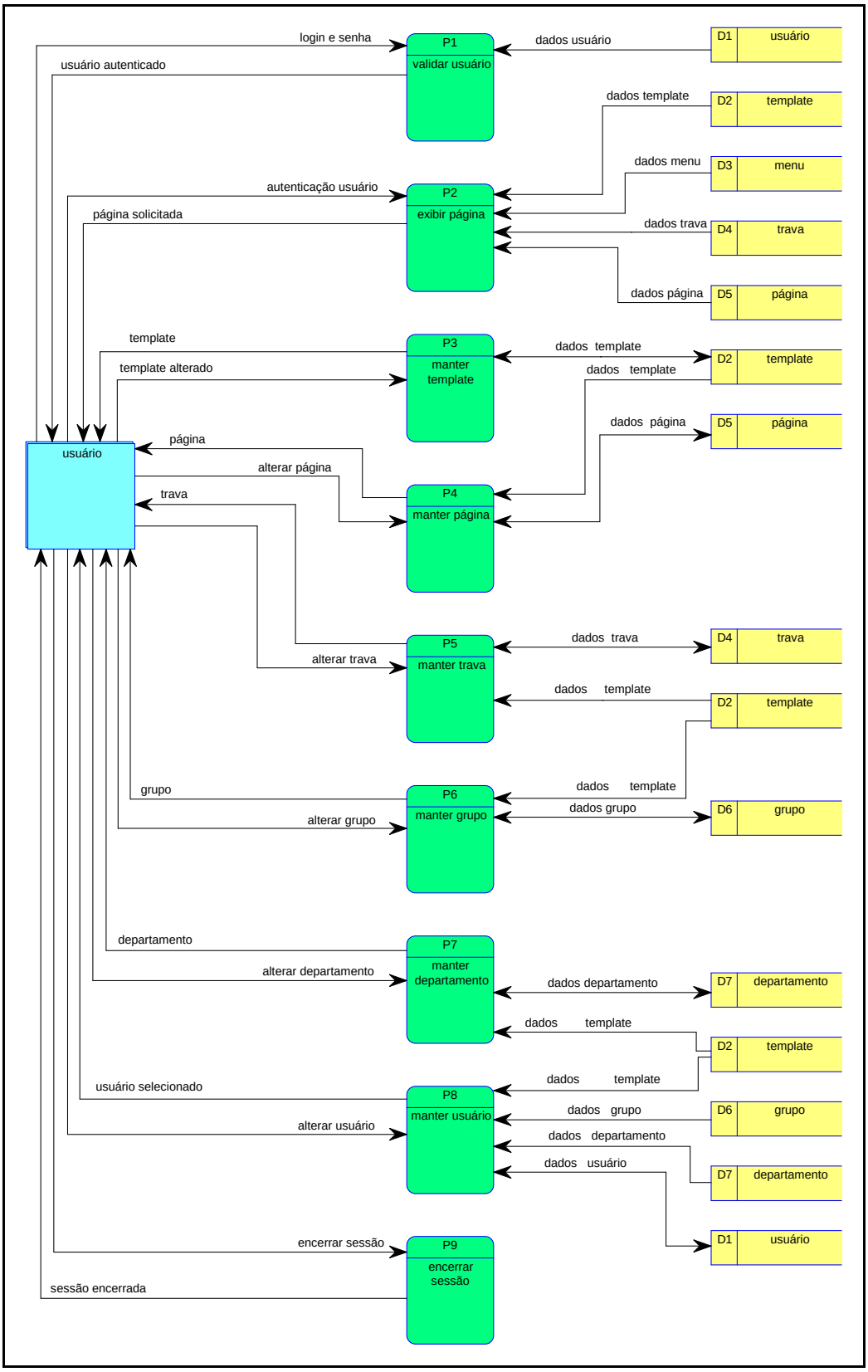
Figura 3 – Diagrama de Contexto



4.2.3 DIAGRAMA DE FLUXO DE DADOS NÍVEL 0

Na figura 4 apresenta-se o diagrama de fluxo de dados nível 0, onde observa-se as funcionalidades do sistema.

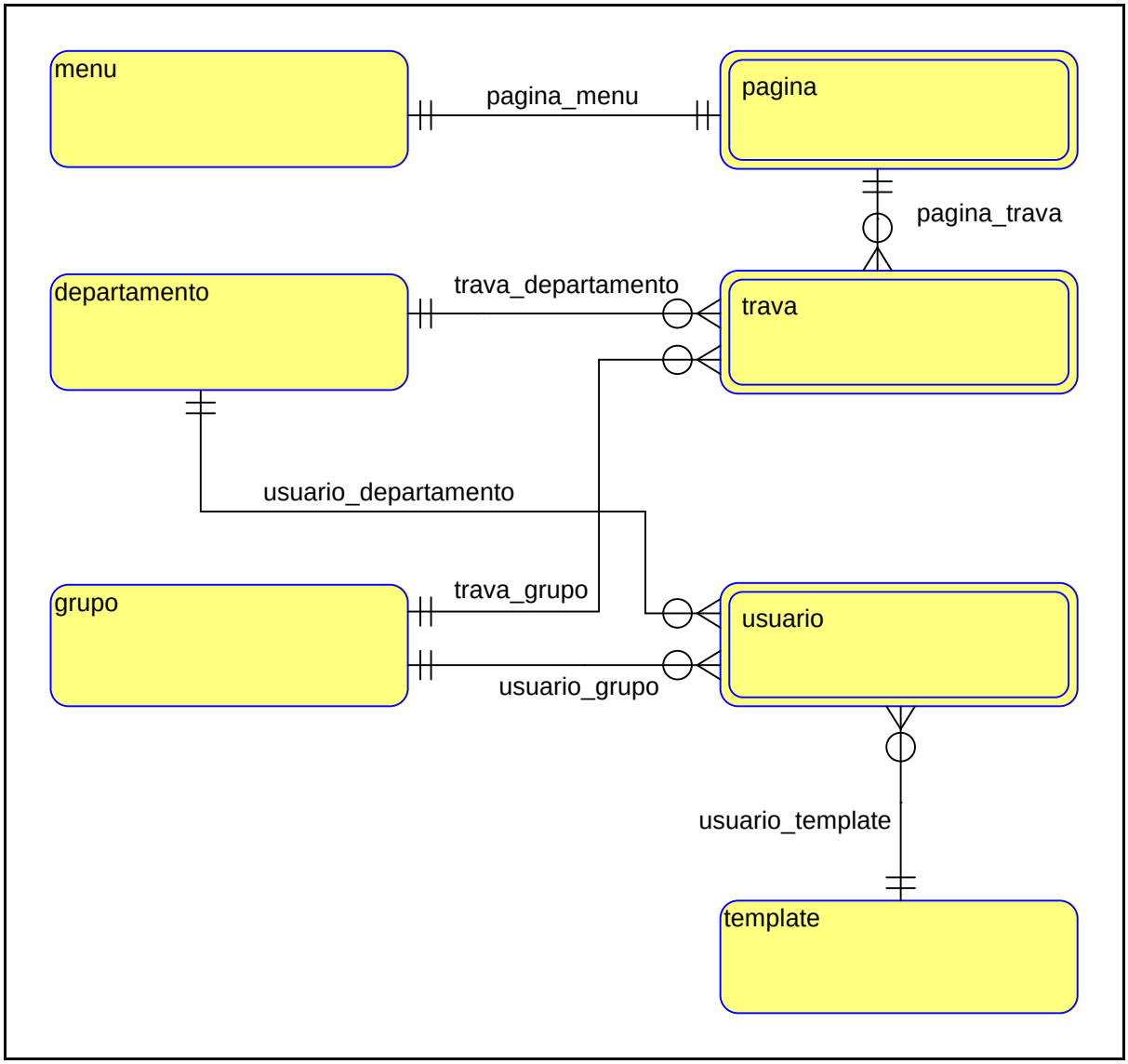
Figura 4 – Diagrama de Fluxo de Dados Nível 0



4.2.4 MODELO ENTIDADE RELACIONAMENTO

Na figura 5 apresenta-se o modelo entidade relacionamento (MER) das tabelas utilizadas no sistema.

Figura 5 – Modelo Entidade Relacionamento (MER)



O sistema é composto pelas seguintes entidades:

a) menu: hierarquia de páginas e sub-páginas;

b) pagina: informações da página;

c) trava: controle de acesso às páginas;

d) *template*: formatos de visualização do conteúdo;

e) departamento: informações dos departamentos de acesso;

f) grupo: informações dos grupos de acesso;

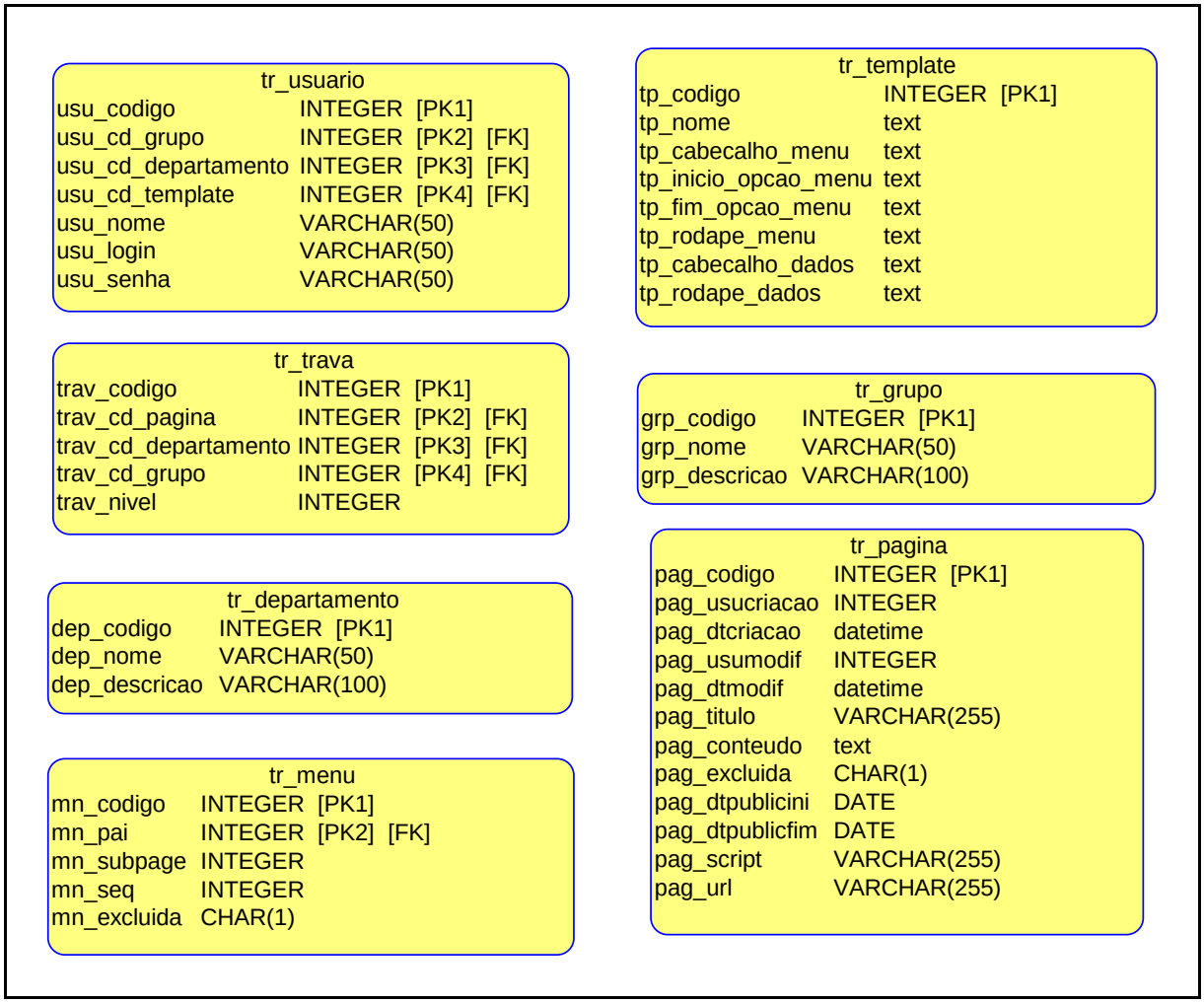
g) usuário: informações do usuário.

4.2.5 DICIONÁRIO DE DADOS

Na figura 6 apresenta-se o dicionário de dados das tabelas utilizadas no sistema.

A base de dados do sistema foi implantada no banco de dados MySQL.

Figura 6 – Dicionário de Dados



4.3 IMPLEMENTAÇÃO

Este item tem por objetivo abordar as ferramentas utilizadas na implementação e a operacionalidade da implementação.

4.3.1 TÉCNICAS E FERRAMENTAS UTILIZADAS

Para a implementação do sistema foram utilizadas as ferramentas: Apache httpd, a linguagem de programação PHP e o gerenciador de banco de dados MySQL. Todas as ferramentas foram instaladas na plataforma Windows 2000.

Utilizou-se o Apache httpd como servidor *web*, instalado como um serviço do Windows 2000, configurado com a linguagem PHP para a execução dos *scripts* PHP e geração dinâmica das páginas.

O MySQL serviu como gerenciador de banco de dados, armazenando os dados e conteúdos utilizados pelo sistema. Também foi instalado como um serviço do Windows 2000.

As páginas dinâmicas geradas pelo sistema são compatíveis apenas com o navegador Microsoft Internet Explorer, versão 4.1 ou superior.

4.3.1.1 CONFIGURANDO O APACHE HTTPD

A configuração do servidor *web* Apache httpd é realizada em um arquivo no formato texto chamado *httpd.conf*, localizado no diretório *conf* da sua instalação *default*. No Windows 2000, o diretório default é "*c:\arquivos de programas\apache group\apache*".

Neste arquivo, a configuração foi ajustada para que os arquivos *index.php* sejam enviados quando for solicitado um diretório e não um arquivo específico.

Configurou-se o Apache httpd como um serviço do Windows 2000, sendo iniciado automaticamente pelo mesmo.

4.3.1.2 INSTALANDO O PHP

Para a linguagem PHP trabalhar como um módulo configurou-se o Apache httpd, também no arquivo de configuração *httpd.conf*, para funcionar como um módulo do servidor *web*, permitindo que os arquivos com a extensão *.php* sejam tratados como scripts da linguagem PHP, sendo processados pelo módulo PHP, gerando assim o conteúdo dinâmico que, por sua vez, é repassado para o Apache httpd que finalmente envia ao navegador solicitante.

Durante a instalação do PHP, é criado um arquivo *php.ini*. Neste arquivo encontram-se as configurações *default* da linguagem. A localização deste arquivo depende da plataforma onde a linguagem PHP é instalada. No Windows 2000, o diretório *default* de instalação é "*c:\php*". O módulo PHP é iniciado automaticamente pelo servidor *web* Apache httpd.

O sistema de gerenciamento de conteúdo não necessita de nenhuma configuração especial da linguagem PHP.

4.3.1.3 UTILIZANDO O MYSQL

Instalou-se o servidor de banco de dados MySQL no diretório *default*, "*c:\mysql*", não necessitando de nenhuma configuração especial para o funcionamento do sistema de gerenciamento de conteúdo desenvolvido.

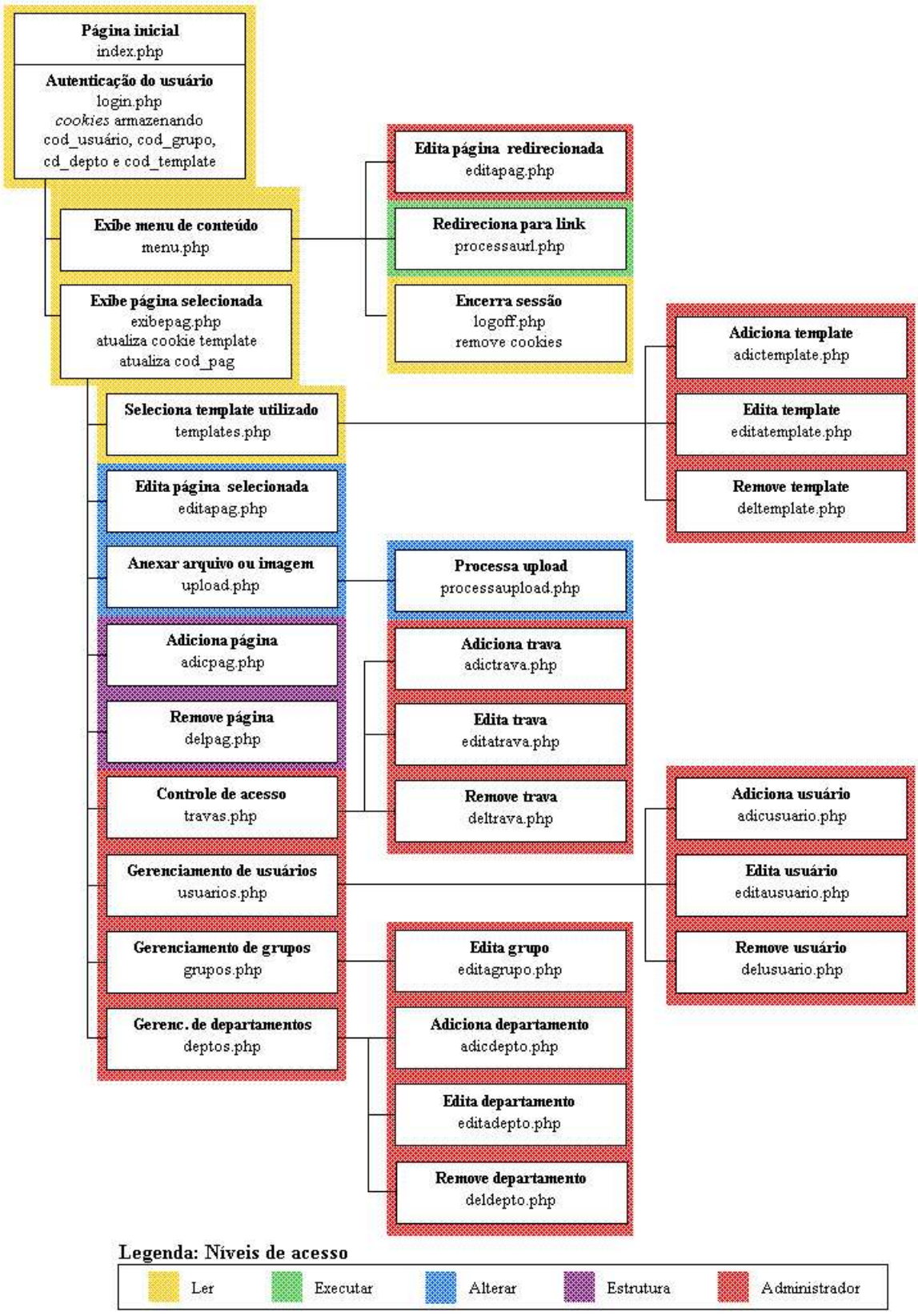
Também configurou-se o MySQL como um serviço do Windows 2000.

4.3.2 OPERACIONALIDADE DA IMPLEMENTAÇÃO

Este item trata da operacionalidade da implementação: como está estruturada e como estão organizados os arquivos do sistema. Um arquivo fonte está listado no anexo I.

Na figura 7 mostra-se o macro fluxo do sistema.

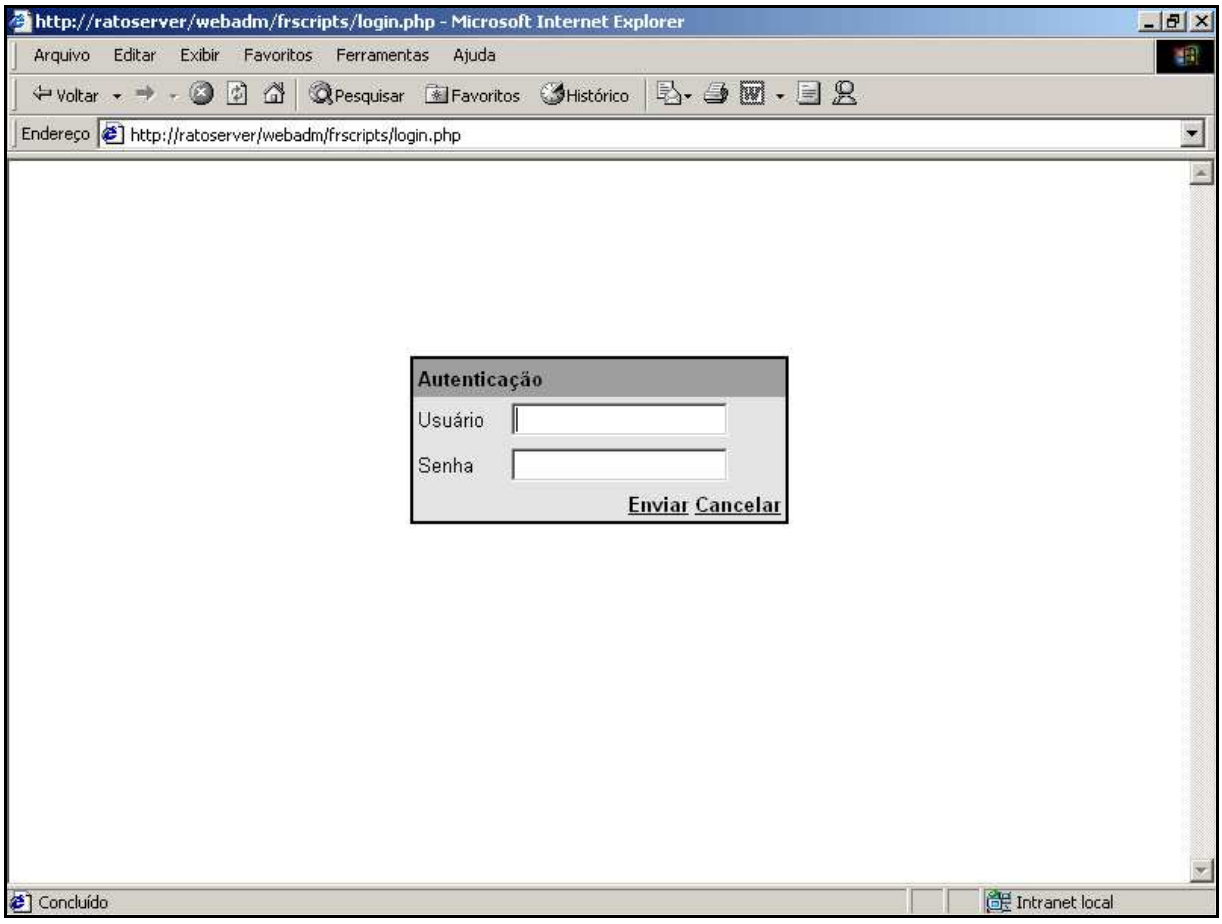
Figura 7 – Macro fluxo do sistema



Explicar-se-á a funcionalidade de cada item do macro fluxo (figura 7) abaixo:

Página inicial, é a página de acesso, que iniciará o sistema. Se o usuário estiver autenticado divide a tela em dois *frames*, executando *menu.php* no frame esquerdo e *exibepagina.php* no frame direito. Caso o usuário não esteja autenticado, ela o redireciona para tela de autenticação do usuário.

Figura 8 – Autenticação do usuário

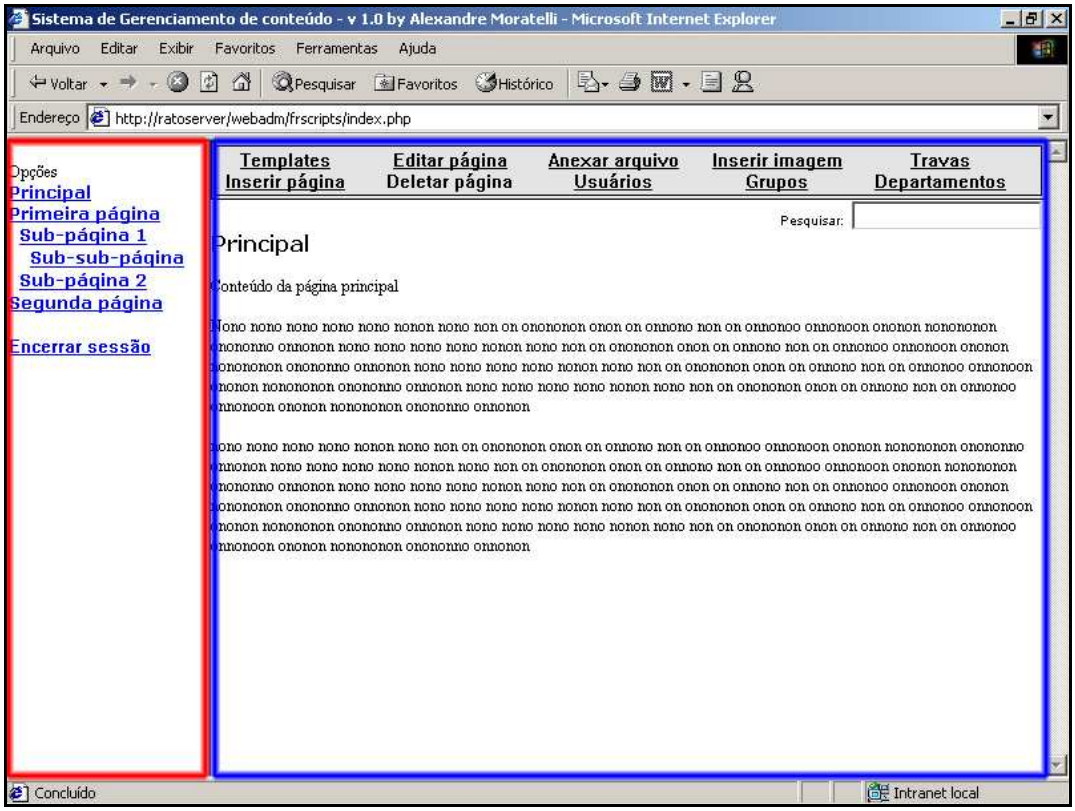


Autenticação do usuário, *script* de identificação do usuário. Sua tela é exibida na figura 8. Sendo um usuário válido, são criados *cookies* para armazenar o código do usuário (cod_usuário), código do template (cod_template) selecionado pelo usuário, código do grupo (cod_grupo) e código do departamento (cod_depto) ao qual o usuário pertence, e redireciona para página inicial. Se for um usuário inválido, retorna para autenticação do usuário.

Exibe menu de conteúdo, *script* que exibe *links* para o conteúdo, de acordo com a hierarquia de páginas e sub-páginas definida na inclusão das mesmas, que o usuário pode acessar, independente de qual seja seu nível de acesso: ler, executar, alterar, estrutura ou

administrador. Exibe um *link* para encerrar a sessão no sistema. Se o usuário possuir nível de acesso para alterar a página, e a página for um redirecionamento para outro *site* ou para um *script*, é exibido um *link* possibilitando editar a página. Caso receba como parâmetro "pag_url", redireciona o *frame* de conteúdo para o endereço indicado. A área demarcada em vermelho na figura 9, mostra um menu gerado pelo script.

Figura 9 – Exibe menu de conteúdo (vermelho) e exibe página selecionada (azul)



Exibe página selecionada, *script* que exibe o conteúdo da página. Antes de exibir qualquer conteúdo, verifica se recebeu como parâmetro código do *template* ou algum redirecionamento, atualizando o *cookie template* com o novo *template* escolhido, e redirecionando para *script* do usuário ou *site*, caso o usuário possua permissão executar ou superior. Após, exibe opções do sistema de acordo com o nível de acesso do usuário. A tabela 1 mostra os níveis de acesso e suas descrições.

Tabela 1 – Níveis de acesso

Código	Nível	Descrição
1	Ler	Visualizar conteúdo da página.
2	Executar	Redirecionar para scripts do usuário e sites.
3	Alterar	Alterar conteúdo da página.
4	Estrutura	Alterar estrutura da página e sub-páginas.
5	Administrador	Funções de administração do sistema

Os níveis de acesso mostrados na tabela 1 são acumulativos, ou seja, o nível mais alto acumula os níveis mais baixos. Por exemplo, um usuário com nível de acesso de administrador, acumula os níveis de acesso inferiores: estrutura, alterar, executar e ler.

Todos os usuários cadastrados recebem um nível de acesso, determinado pelo administrador do sistema. O sistema possui um usuário especial, chamado “admin”, que possui todos os níveis de acesso para todas as páginas cadastradas e também acesso às funções de administração do sistema.

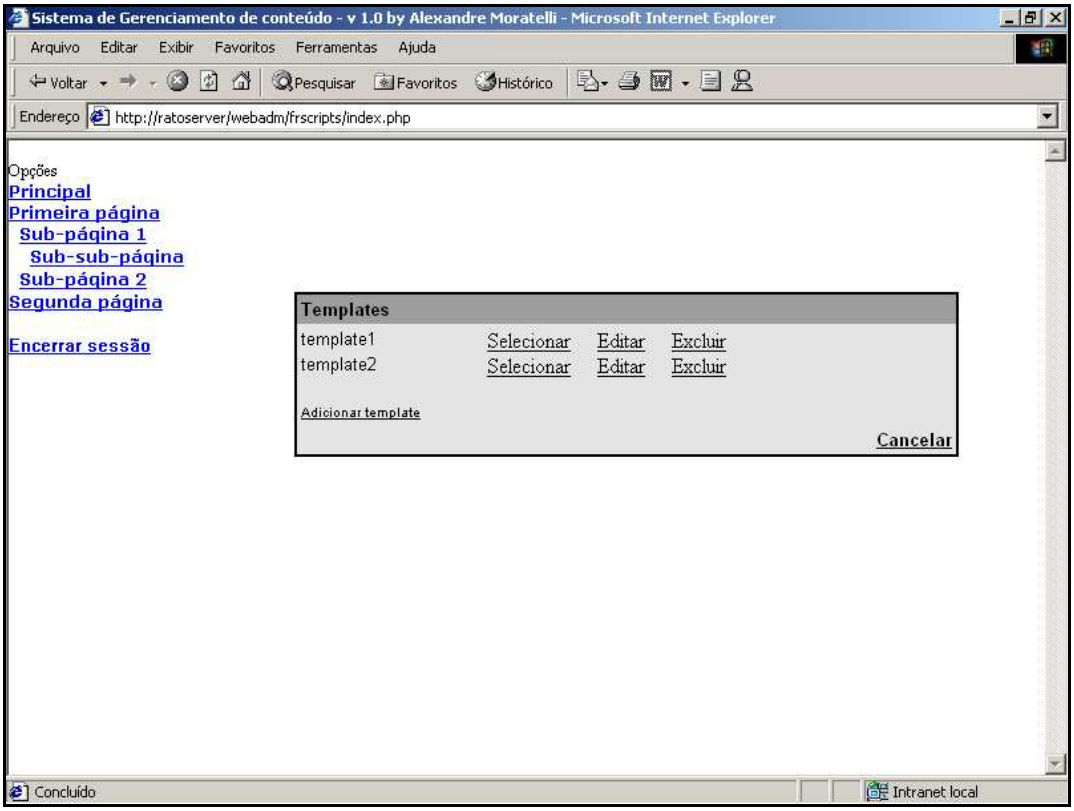
No macro fluxo (figura 7), estão exibidos as funções do sistema que cada nível de acesso pode executar.

Caso a página selecionada não possua redirecionamento para um script do usuário ou *site*, seu conteúdo será então exibido.

A área demarcada em azul na figura 10, mostra o conteúdo da página exibido na tela para o usuário “admin”.

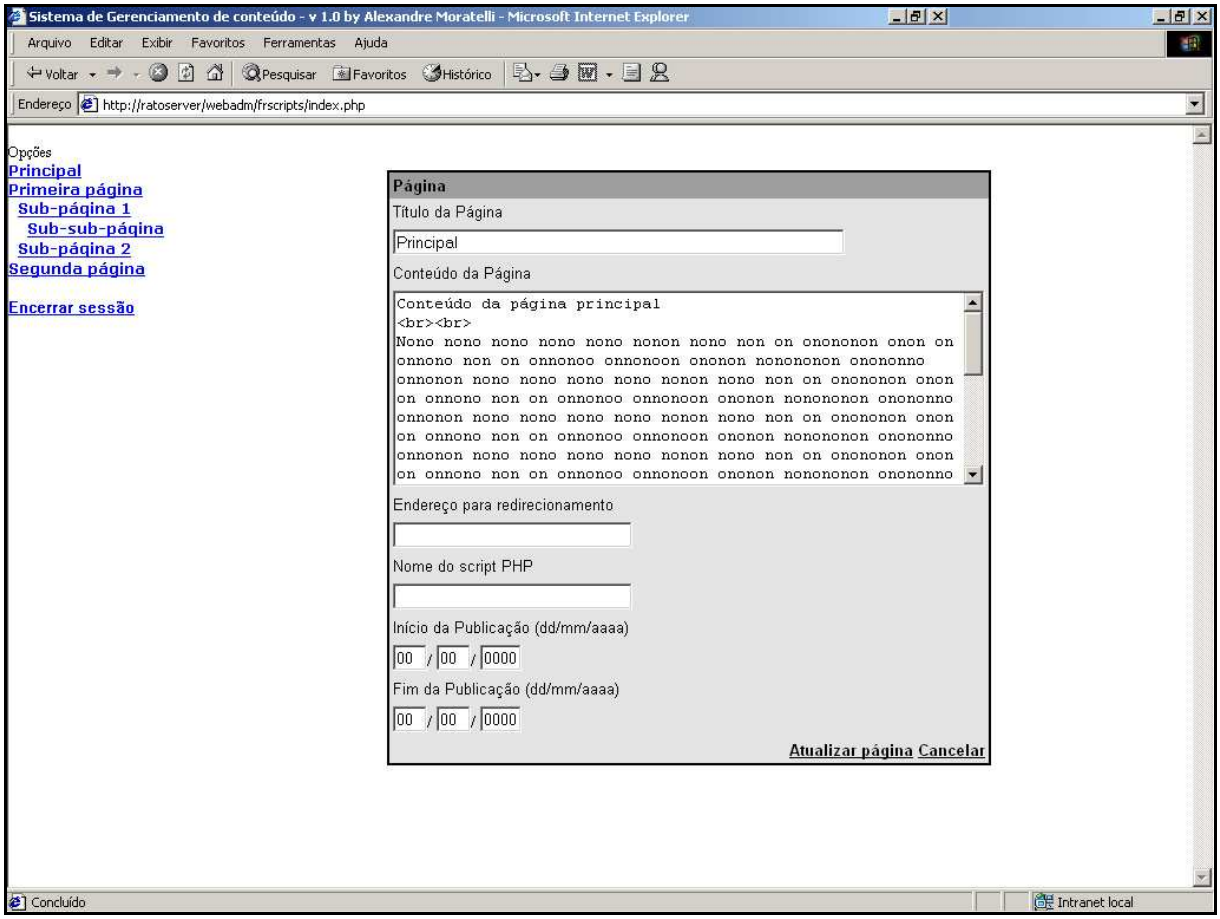
Seleciona *template* utilizado, *script* que exibe os *templates* disponíveis para utilização, exibindo *links* para a seleção, alteração, exclusão ou inclusão de novo *template*, de acordo com o nível de acesso do usuário. A figura 10 exibe a tela gerada pelo sistema para a seleção de *template*. Apenas os usuários com nível de acesso ler ou superior têm acesso a esta função.

Figura 10 – Seleciona *template* utilizado



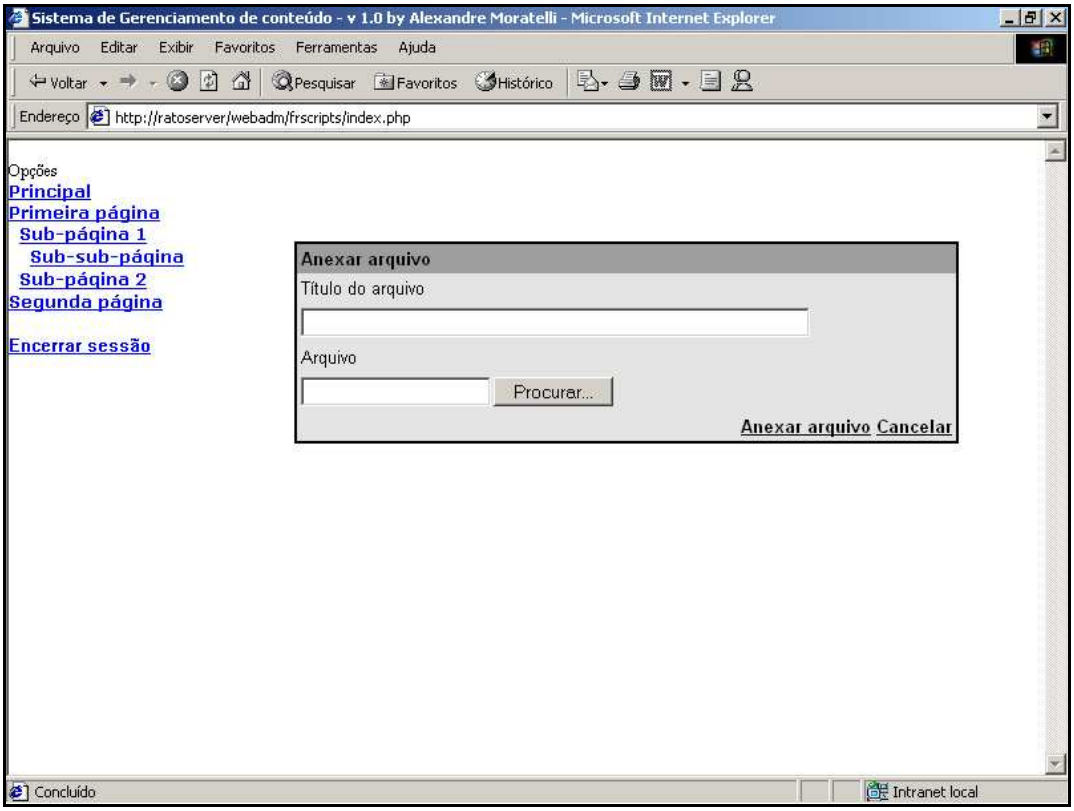
Edita página selecionada, *script* que permite a alteração das informações da página atualmente selecionada. Permite apenas a alteração do conteúdo, título, endereço para redirecionamento, *script* do usuário para execução, data inicial e final de publicação da página selecionada. A posição da página selecionada na hierarquia de páginas e sub-páginas não pode ser alterada. A figura 11 exibe a tela de edição da página selecionada. Apenas os usuários com nível de acesso alterar ou superior tem acesso à função de editar página.

Figura 11 – Edita página selecionada



Anexar arquivo ou inserir imagem, *script* que possibilita ao usuário escolher o arquivo que será inserido na página, como um documento anexado ou uma imagem. A figura 12 exibe a tela ao anexar arquivo ou inserir imagem na página selecionada. Apenas os usuários com nível de acesso alterar ou superior possuem acesso às funções de anexar arquivo ou inserir imagem.

Figura 12 – Anexar arquivo ou inserir imagem



Processa *upload*, *script* que realiza o *upload* do arquivo selecionado, inserindo-o à página atual como um documento anexado ou uma imagem. Este *script* não exibe nenhuma tela.

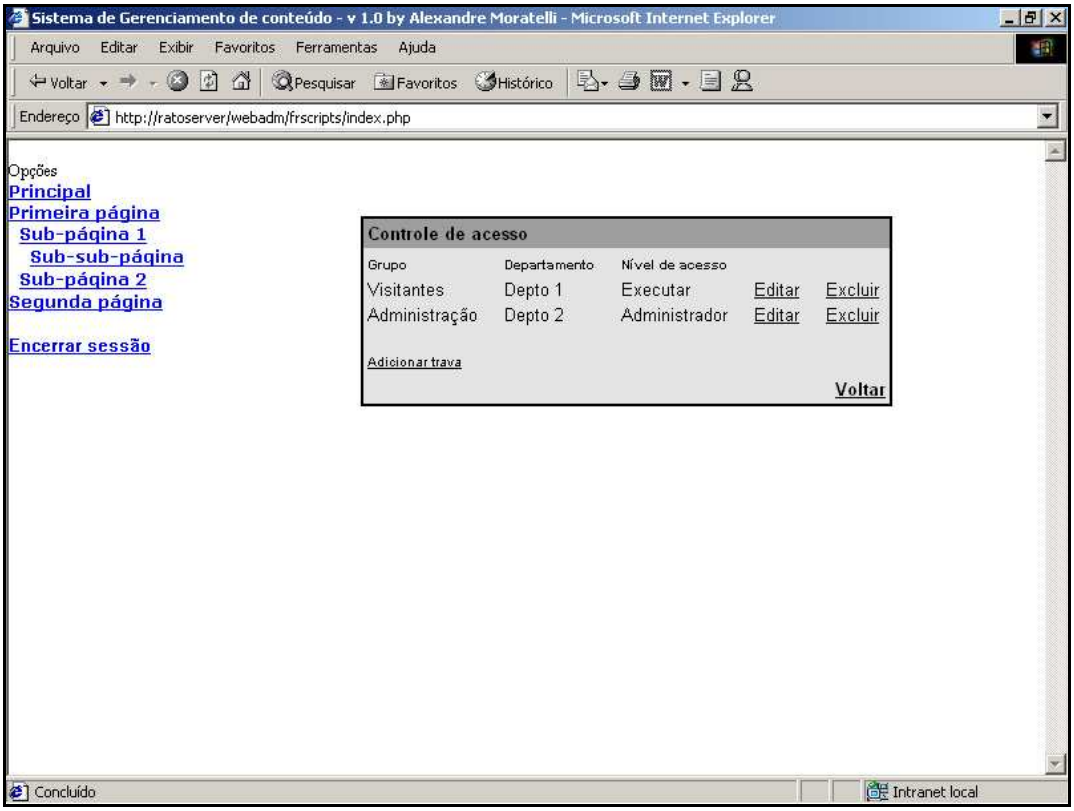
Adiciona página, *script* que permite ao usuário inserir uma sub-página da página atual. É a mesma tela exibida por edita página selecionada, apenas com os valores vazios para preenchimento das informações da nova página. A página será inserida na hierarquia de páginas e sub-páginas como uma sub-página da página selecionada. Se a página selecionada já possuir sub-páginas, ela será adicionada ao fim da lista de sub-páginas. Apenas os usuários com nível de acesso estrutura ou superior possuem acesso à função de adicionar página.

Remove página, *script* que remove a página atual. Não exibe nenhuma tela.

Controle de acesso, *script* que exibe as configurações de acesso da página atual, exibindo *links* que permitem adicionar, editar ou remover uma trava de acesso à página. Uma trava de acesso determina o nível de acesso dos membros de um determinado grupo e departamento. Os níveis de acesso estão descritos na tabela 1. A figura 13 mostra a tela de

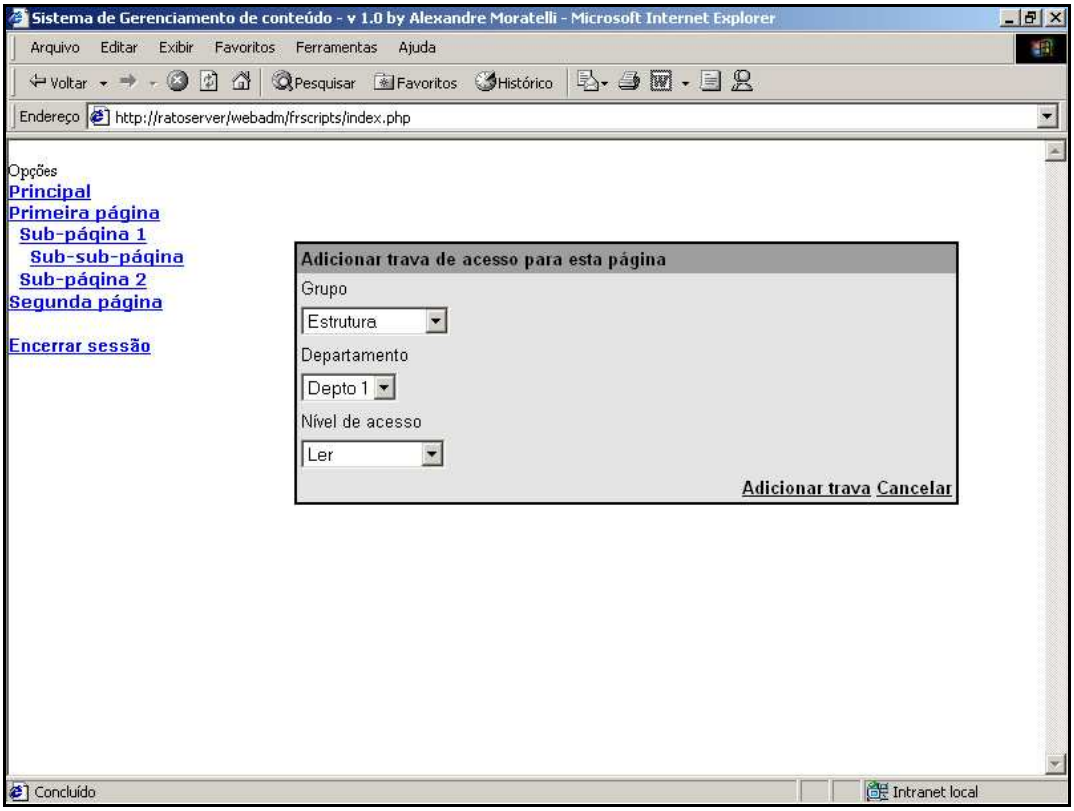
controle de acesso da página selecionada, que permite a configuração dos níveis de acesso. Apenas os usuários com nível de acesso administrador ou superior possuem acesso às funções de controle de acesso.

Figura 13 – Controle de acesso



Adiciona trava, *script* que permite ao usuário adicionar uma nova trava de acesso à página atual. Uma trava de acesso de uma página é configurada informando-se a combinação de grupo, departamento e o nível de acesso que os usuários pertencentes na combinação de grupo e departamento terão. A figura 14 exibe a tela para adicionar uma trava ao sistema.

Figura 14 – Adiciona trava

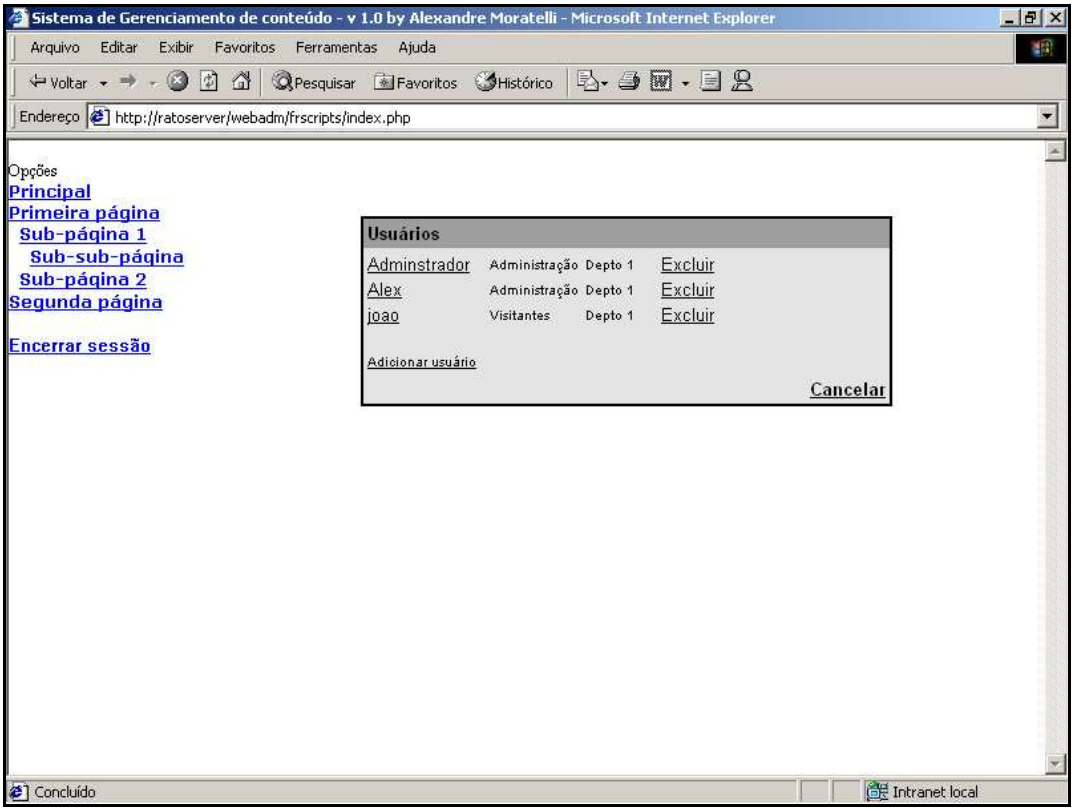


Edita trava, *script* que permite ao usuário editar a trava de acesso selecionada. A tela para editar uma trava é semelhante à de adicionar trava, possibilitando ao usuário alterar a combinação de grupo, departamento e nível de acesso.

Remove trava, *script* que permite ao usuário remover a trava de acesso selecionada. Este *script* não exibe nenhuma tela.

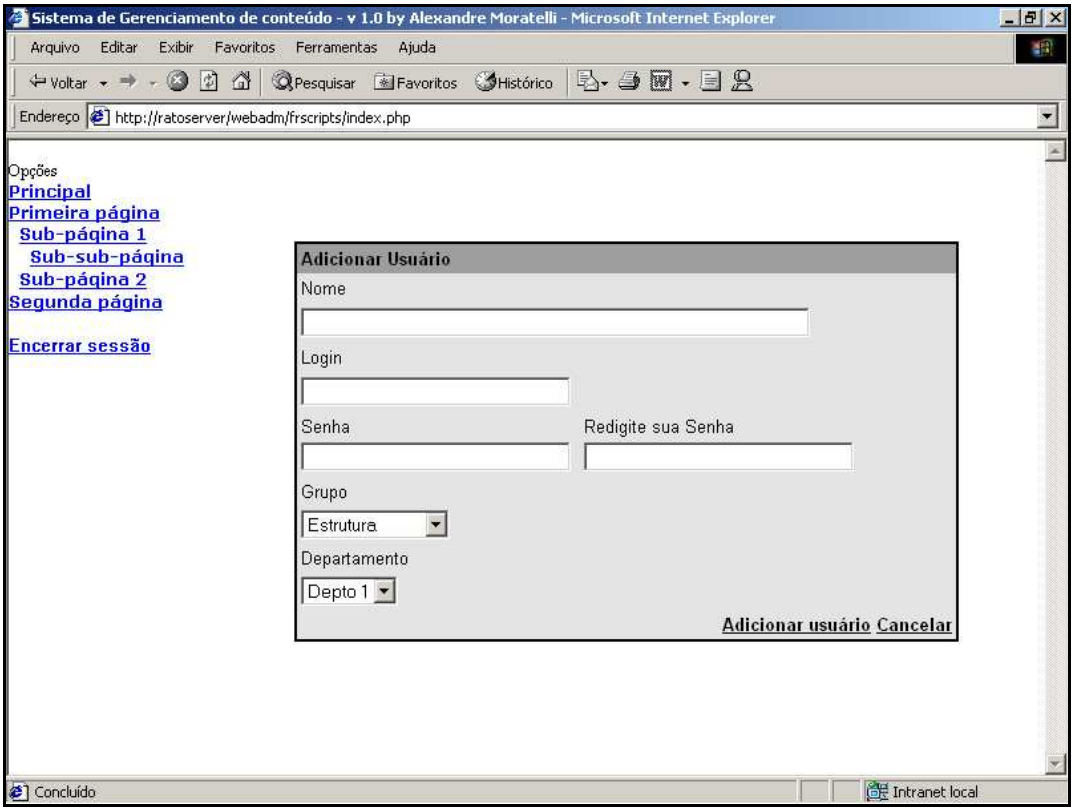
Gerenciamento de usuários, *script* que lista os usuários cadastrados no sistema, exibindo *links* para alterar, excluir ou adicionar um novo usuário. Somente usuários com nível de acesso de administrador possuem acesso a este *script*. A figura 15 exibe a tela de gerenciamento de usuários. Apenas os usuários com nível de acesso administrador ou superior possuem acesso às funções de gerenciamento de usuários.

Figura 15 – Gerenciamento de usuários



Adiciona usuário, *script* que permite adicionar um novo usuário, e suas informações, ao sistema. A figura 16 exibe a tela para adicionar um usuário.

Figura 16 – Adiciona usuário

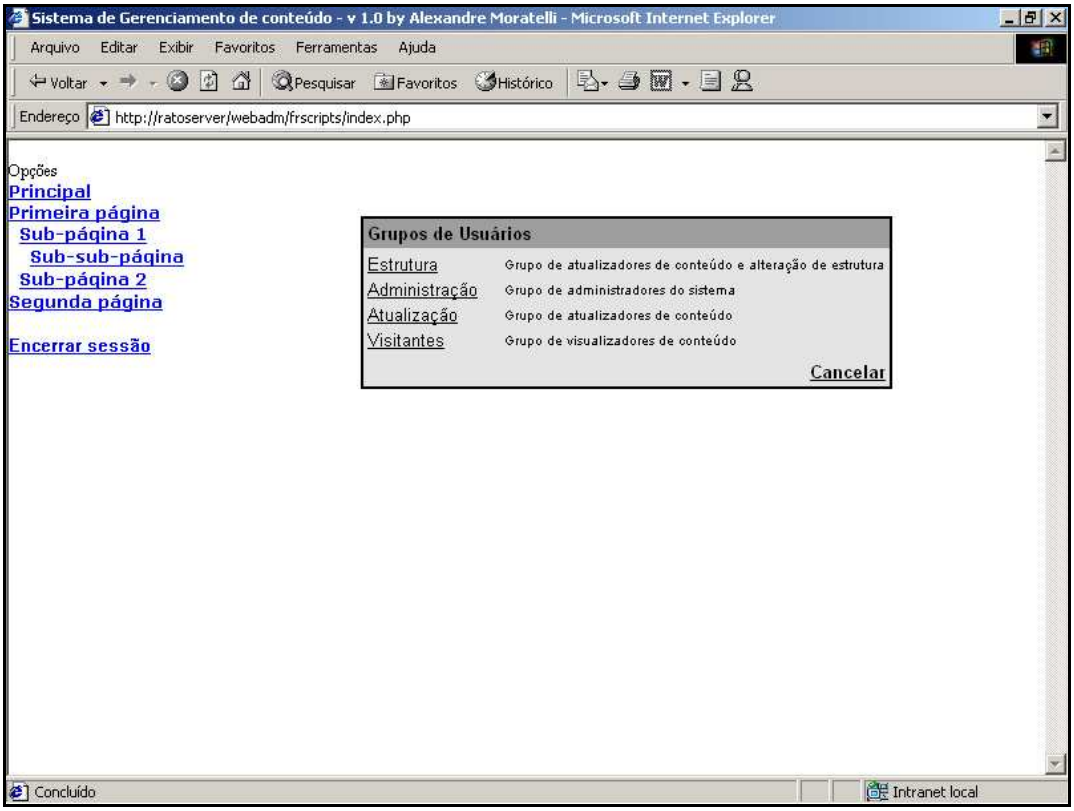


Edita usuário, *script* que permite a alteração dos dados do usuário selecionado. Exibe a mesma tela que adiciona usuário, porém permite alterar os dados do usuário selecionado.

Remove usuário, *script* que permite a remoção do usuário selecionado do sistema. Não exibe nenhuma tela.

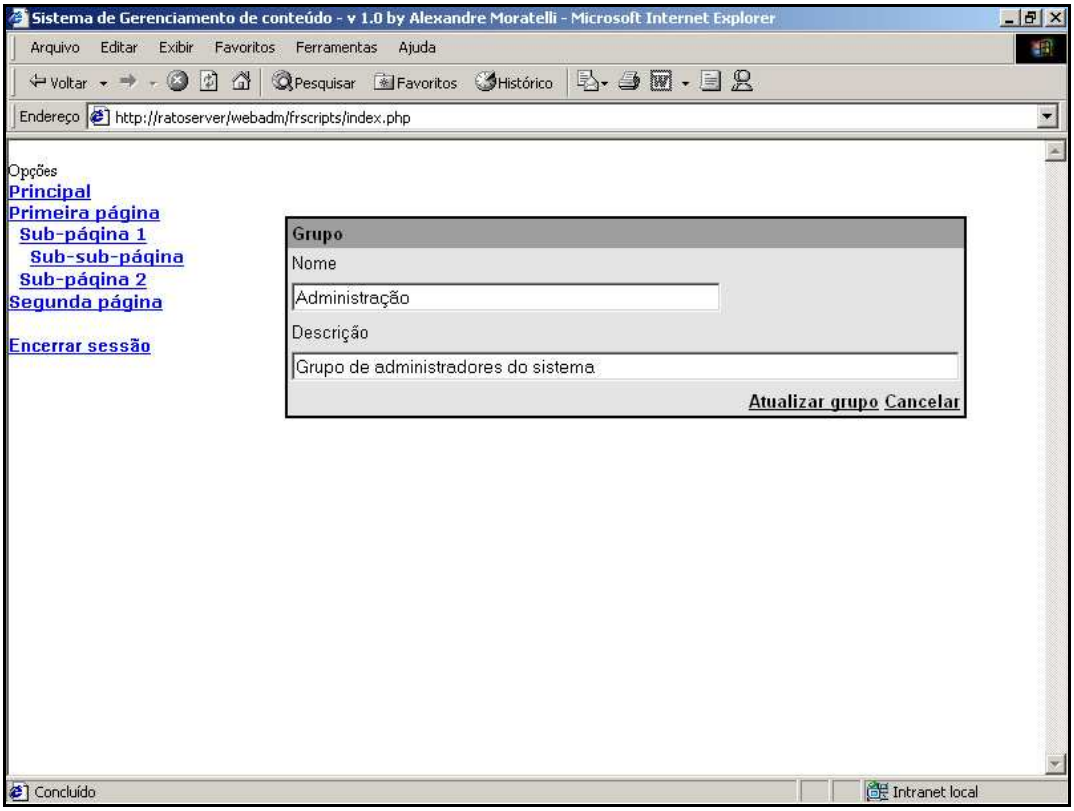
Gerenciamento de grupos, *script* que exibe os diferentes grupos de usuários, exibindo *link* para a alteração de um determinado grupo. A figura 17 exibe a tela de gerenciamento de grupos. Apenas os usuários com nível de acesso administrador ou superior possuem acesso às funções de gerenciamento de grupos.

Figura 17 – Gerenciamento de grupos



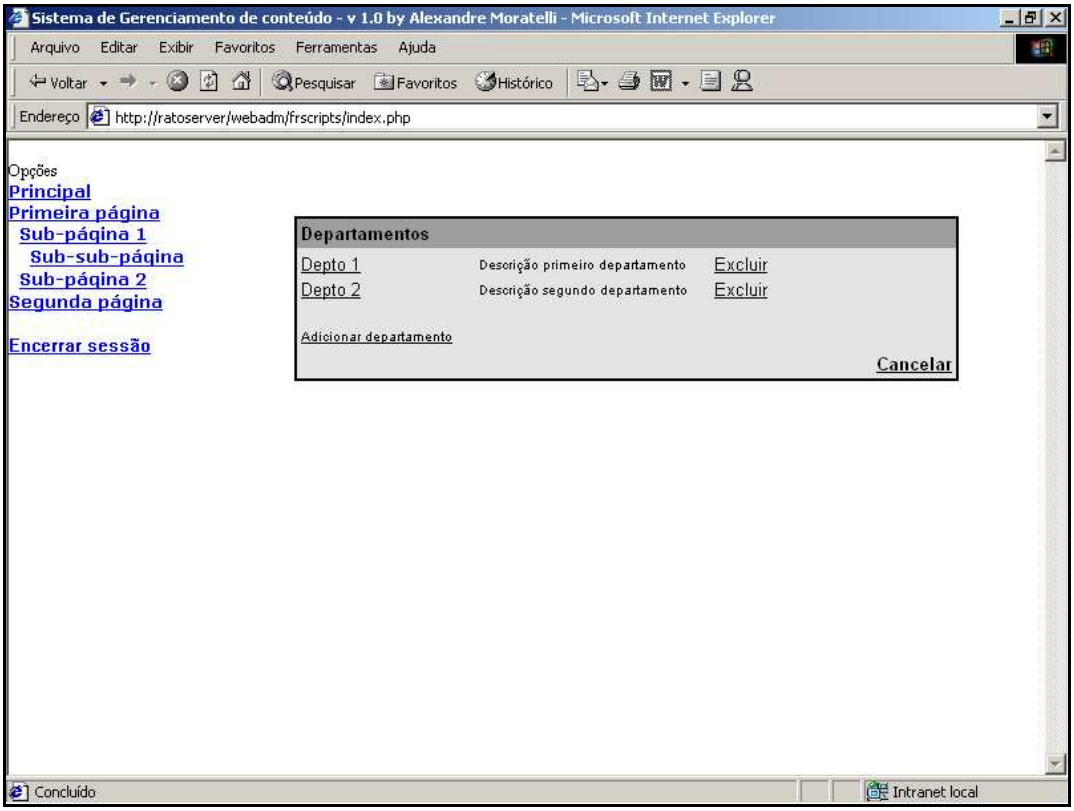
Edita grupo, *script* que permite a alteração das informações de um determinado grupo. A figura 18 exibe a tela de edição do grupo selecionado.

Figura 18 – Edita grupo



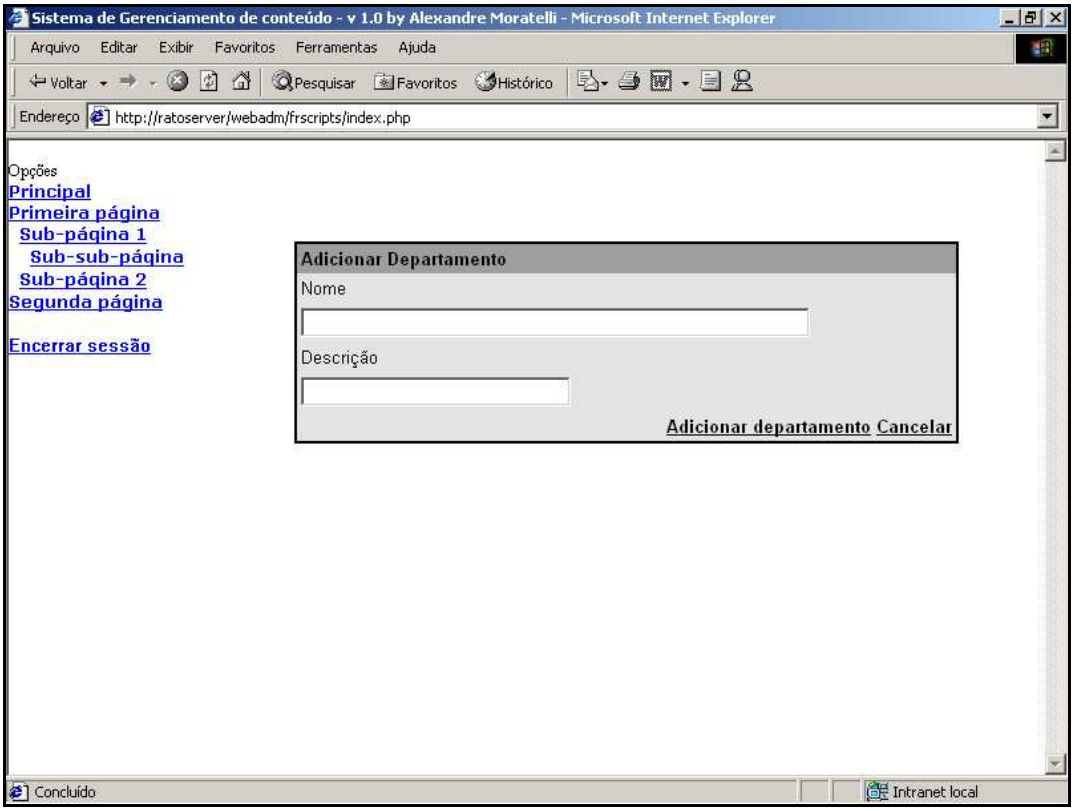
Gerenciamento de departamentos, *script* que exibe os departamentos cadastrados nos sistema, e exibe links permitindo adicionar um novo departamento, alterar ou remover o departamento selecionado. A figura 19 exibe a tela de gerenciamento de departamentos. Apenas os usuários com nível de acesso administrador ou superior possuem acesso às funções de gerenciamento de departamentos.

Figura 19 – Gerenciamento de departamentos



Adiciona departamento, *script* que permite adicionar um novo departamento e suas informações ao sistema. A figura 20 exibe a tela para adicionar um departamento.

Figura 20 – Adiciona departamento



Edita departamento, *script* que permite a alteração das informações do departamento selecionado. Utiliza a mesma tela de adiciona departamento, permitindo a edição das informações do departamento selecionado.

Remove departamento, *script* que permite a remoção do departamento selecionado. Não exibe nenhuma tela.

5 CONSIDERAÇÕES FINAIS

Este capítulo apresenta as conclusões, limitações e sugestões referentes ao trabalho desenvolvido.

5.1 CONCLUSÕES

O gerenciamento de conteúdo é um assunto em destaque, merecendo ser objeto de maiores estudos. Mostra-se uma área que envolve diversas tecnologias, como controle de acesso, utilização de banco de dados no armazenamento das informações, utilização de *templates* gráficos para exibição das informações de forma independente do conteúdo e servidores *web*.

Os objetivos do trabalho foram alcançados. O sistema proposto foi desenvolvido e seu funcionamento está de acordo com a especificação, atendendo todos os seus requisitos. O sistema desenvolvido facilita o gerenciamento de conteúdo, a pesquisa deste conteúdo, a estrutura o conteúdo de maneira automática, garante o acesso das informações através do controle de usuários e a utilização de *templates* para separar o formato gráfico de exibição do conteúdo armazenado.

Sobre as tecnologias envolvidas, acredita-se que ainda exista muito a ser estudado. O gerenciamento de conteúdo é uma disciplina que envolve muita discussão sobre a melhor maneira de se realizar este gerenciamento. Se existe alguma certeza em relação ao gerenciamento de conteúdo, é a necessidade das organizações de gerenciarem as informações geradas de maneira simples e eficaz, permitindo a disseminação da informação, obedecendo a critérios específicos no acesso e formato de exibição do conteúdo.

A utilização de *templates* possibilita a separação do formato gráfico de exibição das informações do conteúdo propriamente dito, facilitando a alteração e criação de novos formatos sem a necessidade de alteração do sistema ou das informações.

As ferramentas utilizadas mostraram-se eficientes na realização de suas tarefas. A perfeita integração entre o servidor *web* Apache httpd e o módulo da linguagem PHP, e entre a linguagem PHP e o banco de dados MySQL, criou um ambiente perfeito para a criação de páginas dinâmicas através da recuperação de informações do banco de dados.

O Apache httpd mostrou-se estável e confiável na execução dos *scripts .php*, tornando sua utilização eficiente para este tipo de aplicação.

Quanto ao servidor de banco de dados MySQL, o mesmo mostrou-se confiável no armazenamento das informações e sua performance foi satisfatória para os objetivos do sistema.

Por sua vez, a linguagem PHP mostrou-se versátil e capaz de solucionar praticamente qualquer problema computacional, sempre respeitando os limites do protocolo HTTP.

5.2 LIMITAÇÕES

Como limitações têm-se os seguintes itens:

a) inclusão de sub-página na última posição. Uma sub-página sempre será inserida como última, não permitindo que seja colocada entre outras sub-páginas existentes;
b) necessidade de configuração de níveis de acesso. Ao incluir uma página deve-se configurar também a(s) trava(s) de acesso para esta página, pois sem ela(s) nenhum usuário, a não ser o admin, terá acesso ao seu conteúdo;
c) não realização do gerenciamento do *workflow*. O trabalho desenvolvido não contempla o gerenciamento do *workflow*;
d) o acesso a sistemas legados é permitido através da implementação de *scripts* do usuário, que poderão realizar quaisquer operações (inclusões, alterações e consultas) sobre as bases de dados de sistemas legados, de acordo com as necessidades levantadas.

5.3 SUGESTÕES

Como sugestões para trabalhos futuros, tem-se as seguintes possibilidades:

a) estudar a novas formas de utilização de *templates* gráficos para a exibição do conteúdo;
b) estudar aspectos de segurança no acesso às informações, como criptografia das informações do usuário;
c) estudar formas mais efetivas de pesquisa do conteúdo;
d) realizar testes de utilização com o sistema contendo uma grande quantidade de

conteúdo e acesso simultâneo de vários usuários para a avaliação de performance;

e) possibilitar o armazenamento e exibição de conteúdos em diferentes formatos de arquivo, por exemplo: vídeos, músicas, animações, etc;
f) estudar o gerenciamento do *workflow*, possibilitando a implementação de uma estrutura que permita o seu fácil gerenciamento.

REFERÊNCIAS BIBLIOGRÁFICAS

APPLIEDTHEORY CORPORATION. **Content Management White Paper**, New York, 2001. Disponível em: <http://www.appliedtheory.com/promotions/contentmanagement.pdf>. Acesso em: 21 abr. 2002.

DESHPANDE, Yogesh; Hansen, Steve. Web Engineering: Creating a Discipline among Disciplines. **IEE Multimedia**, Nova York, v. 8, n. 2, p. 82-87, apr./jun. 2001.

GANE, Chris. **Análise estruturada de sistemas**. Rio de Janeiro: Livros Técnicos e Científicos, 1993.

JOÃO, Belmiro do Nascimento. **Metodologias de desenvolvimento de sistemas**. São Paulo: Érica, 1993.

KIRDA, Engin et al. Experiences in Engineering Flexible Web Services. **IEE Multimedia**, Nova York, v. 8, n. 1, p. 58-65, jan./mar. 2001.

MILLER, Renée J.; Tsatalos, Odysseas G.; Williams, John H. DataWeb: Customizable Database Publishing for the Web. **IEE Multimedia**, Nova York, v. 4, n. 4, p. 15-21, oct./dec. 1997.

MYSQL AB COMPANY. **MySQL | Documentation | MySQL | By chapter | Page**, [S.l.], 2002**.** Disponível em: <http://www.mysql.com/documentation/mysql/bychapter/manual_Introduction.html#What-is>. Acesso em: 22 abr. 2002.

NETCRAFT. **May 2002 Netcraft Web Survey**, [S.l.], 2002**.** Disponível em: <http://www.netcraft.com/Survey/Reports/0205/byserver/index.html>. Acesso em: 12 jun. 2002.

POPKIN SOFTWARE. **Popkin Software Structured Analysis and Design:** A process-oriented approach designed to model business, [S.l.], 2002**.** Disponível em: <http://www.popkin.com/products/system_architect/structured_ad.htm>. Acesso em: 18 mai. 2002.

PRESSMAN, Roger S. **Engenharia de software**. São Paulo: Makron Books, 1995.

RADOFF, Jon. **Best Practices for Web Content Management**, Framingham, 2000. Disponível em: <http://www.eprise.com/eprise/main/uploads/pdfs/WhitePapers/Paper_BestPractices.pdf>. Acesso em: 21 abr. 2002.

THE APACHE SOFTWARE FOUNDATION. **About the Apache HTTPD Server Project** – The Apache HTTPD Server Project, [S.l], 2002. Disponível em: <http://httpd.apache.org/ABOUT_APACHE.html>. Acesso em: 22 abr. 2002.

THE APACHE SOFTWARE FOUNDATION. **Apache Server Frequently Asked Questions**, [S.l], 2002. Disponível em: <http://httpd.apache.org/docs/misc/FAQ.html#what>. Acesso em: 22 abr. 2002.

THE PHP GROUP. **PHP: Manual: History of PHP and related projects**, [S.l.], 2002. Disponível em: < http://www.php.net/manual/fi/history.php>. Acesso em: 22 abr. 2002.

ZEND TECHNOLOGIES. **ZEND / About PHP**, [S.l.], 2002**.** Disponível em: <http://www.zend.com/zend/aboutphp.php>. Acesso em: 22 abr. 2002.

ANEXO I

Em anexo está o código fonte do *script* login.php, que faz uma conexão com o banco de dados MySQL, e seta *cookies* com informações do usuário.

```
<?php
// BIBLIOTECA DE CONFIGURAÇÃO FÍSICA DO SISTEMA
$biblioteca = "../include/sistema.inc";
if (file_exists ($biblioteca)) {
        include($biblioteca);
} else {
        echo 'Erro. ', $biblioteca, ' não encontrada.<br><br>';
}

// CONEXAO COM BANCO DE DADOS
$conexao = @mysql_connect($chost,$cuser,$cpass) or die("Erro na leitura da página");
mysql_select_db($cdb,$conexao);

// VERIFICA CHAMADA
if ($REQUEST_METHOD=="POST") {
        // VERIFICA SE $login e $senha EXISTEM E SETA COOKIE COM CODIGO DO USUARIO
        AUTENTICADO
        // SENAO PEDE LOGIN NOVAMENTE
        $query = "SELECT usu_codigo, usu_cd_template, usu_cd_grupo, usu_cd_departamento
FROM tr_usuario WHERE usu_login='$login' AND usu_senha='$senha'";
        $result  =  @mysql_query($query,$conexao)  or  die  ("Erro:  SELECT  tr_usuario
$result");
        if(@mysql_num_rows($result) != 0) {
                if       (list($usu_codigo,       $usu_cd_template,       $usu_cd_grupo,
$usu_cd_departamento) = mysql_fetch_row($result)) {

                        // SETA COOKIE "COD_USUARIO" COM O CODIGO DO USUARIO
                        setcookie("usuario_autenticado", $usu_codigo, 0, "/", "", 0);

                        // SETA COOKIE "COD_GRUPO" COM O CODIGO DO GRUPO DO USUARIO
                        setcookie("cod_grupo", $usu_cd_grupo, 0, "/", "", 0);

                        // SETA COOKIE "COD_DEPARTAMENTO" COM O CODIGO DO DEPARTAMENTO
                        DO USUARIO
                        setcookie("cod_depto", $usu_cd_departamento, 0, "/", "", 0);

                        // SETA COOKIE "COD_TEMPLATE" COM O TEMPLATE DO USUARIO
                        setcookie("cod_template", $usu_cd_template, 0, "/", "", 0);
                        $location = "Location: index.php";
                } else {
                        $location = "Location: login.php";
                }
        } else {
                        $location = "Location: login.php";
        }
        Header($location);
} else {
        $cod_template = 2;

        // BIBLIOTECA DE RECUPERACAO DE TEMPLATE
        $biblioteca = "../include/template_dados.inc";
        if (file_exists ($biblioteca)) {
                include($biblioteca);
        } else {
                echo 'Erro. ', $biblioteca, ' não encontrada.<br><br>';
        }
?>
<br><br>
<br>
<center>
<font face="arial" size=3 color="#FFFFFF"><b>Sistema de Gerenciamento de<br>
Conteúdo para Ambiente Web</b></font>
<br><br><br>
```

```
	<table	border=1	bordercolor="#000000"	cellpadding=0	cellspacing=0	width="250"
bgcolor="#E4E4E4">
		<form method="post" name="login" action="<?=$PHP_SELF?>">
		<input type=hidden name="cod_pag" value="<?=$cod_pag?>">
		<tr>
			<td>
				<table width=100% cellpadding=3 cellspacing=0 border=0>
					<tr bgcolor="#9E9E9E"valign="top">
						<td colspan=2>
							<font		face="arial"		size=2
color="#000000"><b>Autenticação</b></font>
						</td>
					</tr>
					<tr>
						<td>
							<font		face="arial"		size=2
color="#000000">Usuário</font>
						</td>
						<td>
							<font		face="arial"		size=2
color="#000000"><input type="text" name="login" value="" size="20"></font>
						</td>
					</tr>
					<tr>
						<td>
							<font		face="arial"		size=2
color="#000000">Senha</font>
						</td>
						<td>
							<font		face="arial"		size=2
color="#000000"><input type="password" name="senha" value="" size="20"></font>
						</td>
					</tr>
					<tr valign="top">
						<td colspan=2 align=right>
							<font		face="arial"		size=2
color="#000000">
							<a			style="color:#000000"
href="javascript:document.login.submit()"><b>Enviar</b></a>
							<a			style="color:#000000"
href="javascript:history.go(-1);"><b>Cancelar</b></a>
							</font>
						</td>
					</tr>
				</table>
			</td>
		</tr>
	</form>
	</table>

	<?
	}

	echo $rodape_dados;

	// ENCERRA CONEXÃO COM BANCO DE DADOS
	mysql_close($conexao);
	?>
```