Universidade Federal de Ouro Preto - UFOP

Instituto de Ciências Exatas e Biológicas - ICEB

Departamento de Computação - DECOM

SWAT - Sistema Web para Avaliação de Trabalhos

Aluno: Kayran dos Santos
Matricula: 08.1.4006

Orientador: David Menotti

Ouro Preto
9 de dezembro de 2010

Universidade Federal de Ouro Preto - UFOP

Instituto de Ciências Exatas e Biológicas - ICEB

Departamento de Computação - DECOM

SWAT - Sistema Web para Avaliação de Trabalhos

Relatório de atividades desenvolvidas apresentado ao curso de Bacharelado em Ciência da Computação, Universidade Federal de Ouro Preto, como requisito parcial para a conclusão da disciplina Monografia I (BCC390).

Aluno: Kayran dos Santos
Matricula: 08.1.4006

Orientador: David Menotti

Ouro Preto
9 de dezembro de 2010

Resumo

Apresentamos as atividades já concluidas e as atividades em processo de desenvolvimento para a elaboração do SWAT - Sistema Web para Avaliação de Trabalhos, a ser aplicado inicialmente na disciplina de Estrutura de Dados I (EDI), com grande possibilidade de expansão para outras disciplinas do ciclo básico da Computação voltadas para a programação. O sistema em construção auxiliará no desenvolvimento do novo plano pedagógico do DECOM no que diz respeito à otimização dos recursos didáticos.

Palavras-chave: Avaliação Automática, Trabalhos Práticos, Sistema Web.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

Uma das bases do aprendizado em Ciência da Computação é a confecção de Trabalhos Práticos (TPs). A disciplina de Estrutura de Dados I (EDI) utiliza esta base profundamente, exigindo dos alunos a entrega de no mínimo 4 TPs, sendo compostos de código fonte que resolva o problema proposto e documentação explicativa, aplicando os conceitos aprendidos na disciplina durante o período. A correção desses trabalhos é de extrema importância para o acompanhamento dos alunos em seu desenvolvimento na disciplina e exige muito tempo por parte do professor.

O grande número de alunos que ficam retidos nessa disciplina aumenta a quantidade de TPs a serem corrigidos e dificulta ainda mais esta correção, e além disso, existem prazos a serem cumpridos estabelecidos pela universidade. Automatizar o processo de correção do código fonte com um sistema *online* auxiliaria os professores a cumprir os prazos estabelecidos e facilitaria o acompanhamento do andamento dos alunos por eles mesmos e pelo professor.

Portanto, este trabalho propõe o SWAT (Sistema Web de Avaliação de Trabalhos), um sistema *online* que automatiza o processo de correção dos trabalhos.

2 Justificativa

Grande parte das disciplinas do curso de Ciência da Computação se desenvolve a partir de atividades extra classe, os chamados Trabalhos Práticos. As correções destes TPs são de extrema importância tanto para o aluno saber como anda seu desempenho na disciplina, como para o professor acompanhar o desenvolvimento da turma. Porém estas correções demandam considerável tempo do professor e ainda existe o risco de demorar mais do que é permitido pelas normas da universidade. O sistema Web proposto vem para auxiliar o professor e acelerar o processo de avaliação de TPs, e vem ao encontro do novo projeto pedagógico do curso de Ciência da Computação, no que diz respeito à otimização dos recursos didáticos.

3 Objetivos

3.1 Objetivo geral

O objetivo deste trabalho é desenvolver um sistema acessível pela internet para auxiliar a correção de trabalhos práticos da disciplina de Estrutura de Dados I com possível expansão para outras disciplinas.

O sistema será o responsável pela correção do código fonte do aluno. Este código será submetido em uma página Web e será verificado se o mesmo compila e executa corretamente. Caso execute a saída gerada será comparada com a saída esperada para as várias instâncias de teste que serão criadas para cada TP, para calcular a nota do trabalho.

3.2 Objetivos específicos

Os objetivos específicos são:

- Configuração de um servidor para armazenamento dos dados dos alunos, dos trabalhos submetidos e das páginas desenvolvidas;
- Elaboração do SWAT seguindo a metologia apresentada na seção seguinte;
- Execução de testes do sistema e monitoramento de seu funcionamento;
- Acompanhamento dos alunos durante a submissão de seus TPs.

4 Metodologia

A metodologia adotada para o desenvolvimento do sistema foi:

1. Em um primeiro momento foi feito a análise de requisitos do sistema para fazer o modelo Entidade - Relacionamento (ER) para o banco de dados seguindo os conceitos apresentados em [3] e uma esquematização do sistema, projetando suas interações com os usuários, como mostrado em [5, 13] visando uma melhor Interação Humano- Computador (IHC);

2. No momento os esquemas estão sendo seguidos de forma a implementar o sistema utilizando tecnologias Web [1, 12], em um banco de dados PostgreSQL [4];

3. Após haverá um período de testes do sistema em meio à disciplina de EDI com a submissão supervisionada de TPs;

4. Depois do período de testes na disciplina de EDI, o mesmo será aplicado em larga escala na disciplina e adaptado em outras disciplinas se possível.

4.1 Tecnologias Utilizadas

O sistema está sendo desenvolvido utilizando as tecnologias PHP (*Personal Home Page*) [11, 8] como linguagem base, *Javascript* [10] para uma melhor interação dos alunos e CSS (*Cascading Style Sheets*) [9] para definir os estilos das páginas do sistema. Todas as tecnologias estarão interligadas com um banco de dados PostgreSQL para armazenamento dos dados dos alunos. Este será hospedado em um servidor do DECOM configurado com Ubuntu/Linux e servidor Apache.

A escolha dessas tecnologias ocorreu pois elas são muito utilizadas no mercado atual visto que grande parte das páginas hoje em dia são feitas utilizando essas tecnologias, sem contar que possuem muitos materiais gratuitos de auxilio e estão em constante desenvolvimento. Também são tecnologias simples de utilizar e que produzem sistemas amigáveis aos usuários.

4.2 Organização do Sistema

O sistema será desenvolvido utilizando algumas ideias do BOCA (*BOCA Online Contest Administrator*) [2], sistema de submissão e correção automática utilizada pela ACM em sua Maratona de Programação [7, 6].

Basicamente o sistema será composto de um sistema Web de submissão de arquivos a ser acessado a partir de um "login". O aluno fará "login" no sistema a partir de seu Registro Acadêmico (RA), também conhecido como número de matrícula. Após o "login" o aluno terá acesso, principalmente, a uma página de perfil simples com o histórico de suas submissões e seus dados, uma página de submissão dos arquivos e uma pagina de alteração dos dados. Vale ressaltar que apenas o aluno, o professor e eventualmente um monitor da disciplina terão acesso às notas.

O arquivo submetido na página será armazenado no servidor até o fim do período para possíveis esclarecimentos. Após a submissão do arquivo comprimido, este mesmo será descompactado e compilado usando regras pré-estabelecidas a serem decididas e apresentadas aos alunos. A compilação vai gerar um arquivo executável Linux, que

vai rodar sobre uma base de testes preparada para cada problema. Esta base de testes será composta por um arquivo com possíveis entradas ao programa, que estarão padronizadas, sendo este padrão apresentado aos alunos assim que for apresentado um TP. O programa vai gerar uma saída de acordo com a entrada utilizada. Esta saída vai ser comparada com a saída esperada usando comandos do Linux. Caso a saída seja a esperada será atribuída ao TP do aluno nota total no que tange à execução do programa. Caso o programa apresente algum problema na compilação, ou exista algum problema de execução (Estouro de tempo, Saída mal formatada, Saída com valores errados), este será retornado ao aluno pela página Web e a forma de proceder será decidida em conjunto com o professor. Ao criar o TP o professor definirá um tempo limite para execução do problema, que será apresentado na especificação. Caso um trabalho execute em tempo maior do que o previsto, será caracterizado estouro de tempo.

O perfil do aluno será composto apenas de informações básicas para identificação do mesmo, sendo que estas informações também estarão disponíveis apenas para o professor e ocasionalmente o monitor. Além disso ele terá a página de histórico, onde os resultados dos TPs anteriores estarão disponíveis para que o mesmo possa controlar seu desempenho nos TPs.

4.3 Período de Testes

Os testes do sistema, durante o periodo de desenvolvimento, serão feitos com problemas simples e ainda no servidor do Laboratório de Processamento Digital de Imagens (LaPDI). Após passar por esses testes preliminares, o mesmo será instalado no servidor do DECOM e disponibilizado para os alunos da disciplina de EDI pela internet. Ao disponibilizar o sistema, após entrar em acordo com o professor da disciplina, adaptaremos pelo menos um trabalho para testes do sistema. Depois de aprovado nestes testes, o sistema poderá ser incorporado à disciplina de EDI e poderemos estudar a sua implantação em outras disciplinas de programação básica.

5 Desenvolvimento

Nesta seção, será mostrado como está o andamento do desenvolvimento do sistema, quais passos foram concluídos e quais serão os próximos passos.

5.1 Passos concluídos

A especificação de requisitos do sistema, bem como sua modelagem de banco de dados e IHC foram concluídas. Com esses passos foram definidas todos os dados necessários no banco de dados da aplicação e quais funcionalidades cada usuário terá acesso.

5.1.1 Especificação de requisitos

Foi percebido que o sistema contém dois tipos de usuário básicos:

- Aluno: que representa os alunos de BCC202
- Admin: que representa o professor da disciplinas e/ou os monitores

Para ambos os tipos de usuário será armazenado seu Registro Acadêmico (RA, sendo o número de matrícula dos alunos e o código SIAPE dos professores), sua senha de acesso ao sistema, seu nome, sua idade e seu sexo. A diferença destes tipos de usuários é que o aluno também terá sua média armazenada, já o administrador não. Além disso, difere estes tipos de usuários suas funcionalidades. A ambos é permitido editar seus dados, já apenas os alunos poderão submeter códigos para resolver um determinado TP e terá acesso apenas ao seu histórico. Ao Administrador, e somente a ele, caberá a criação de um novo TP, com submissão de um arquivo de especificação do problema, um arquivo de teste e o arquivo resposta relacionado ao arquivo de teste. O administrador terá acesso também a dados estatísticos da turma.

Com relação aos TPs, será armazenado seus arquivos para especificação do problema, teste das submissões e respostas a serem comparadas além de seus identificadores que serão seu nome (que será no padrão TPx, onde x representa o número do TP), seu ano e semestre de aplicação.

Para cada submissão de trabalho feita para um aluno será armazenado o caminho dos códigos fonte enviados e a nota atribuída ao aluno.

5.1.2 Mapeamento do Banco de Dados

Partindo da especificação de requisitos da seção anterior foi feito um mapeamento para o diagrama ER que está representado na figura 1.

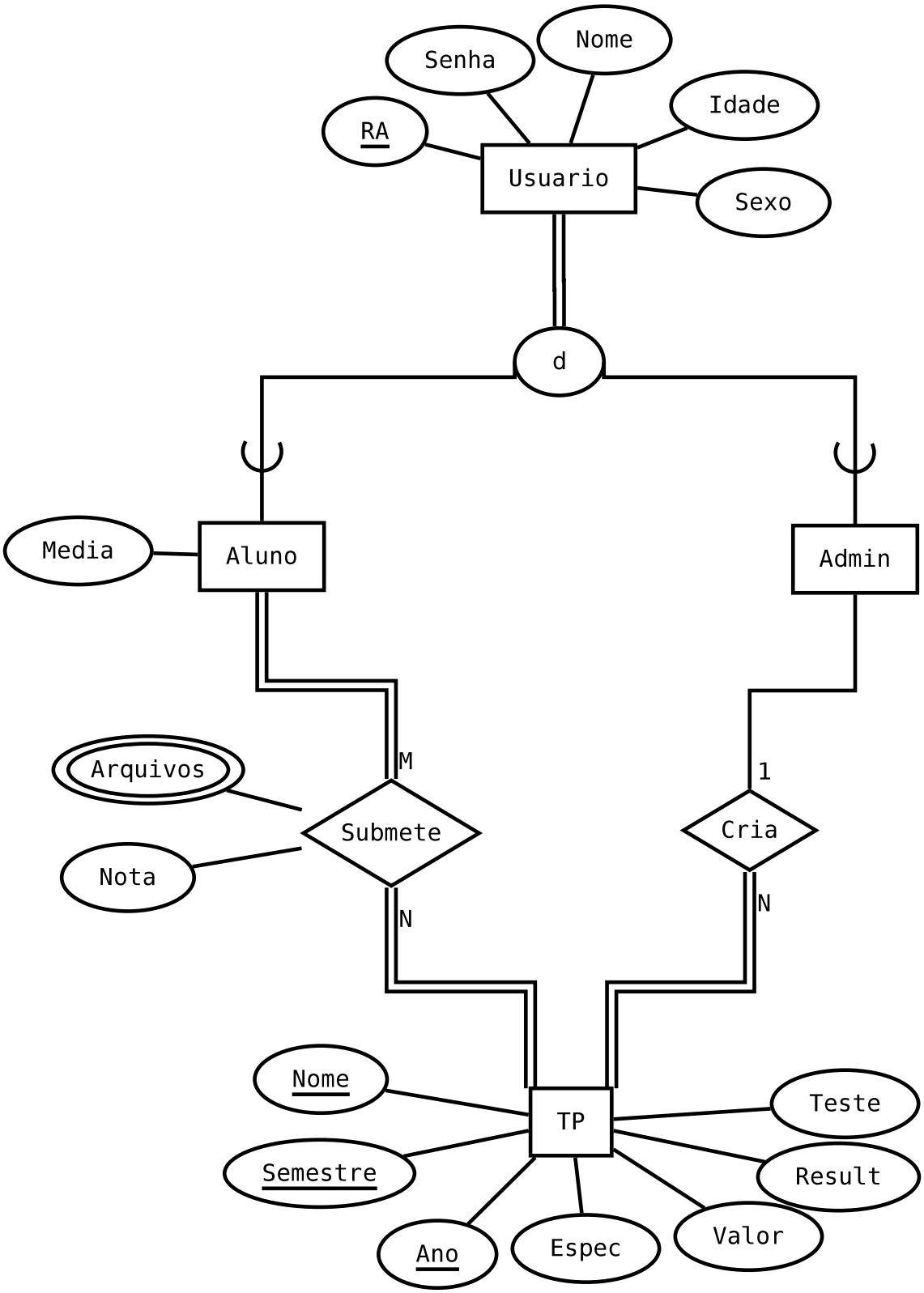

Figura 1: Diagrama ER

Como mostra o diagrama, foram mapeadas 4 entidades: *Usuário, Aluno, Admin e TP*. A entidade *Usuário* possui os atributos *RA* como chave, *senha, nome, idade e sexo*. Essa entidade se especializa em *Aluno* e *Admin*, sendo que a especialização é por disjunção. A entidade *Aluno* além de herdar os atributos de Usuário possui um atributo média. A entidade *Admin* apenas herda os atributos de Usuário. Esse tipo de especialização foi feito pois os usuários de alunos e administradores possuem funcionalidades próprias. A entidade *TP* possui como atributos Nome, Semestre e Ano como chave além de *Espec* (especificação), *Valor, Result* (Resultado esperado) e *Teste*.

Entre *TP* e Aluno há o relacionamento Submete com atributos Arquivo que é multi valorado, já que podem ser submetidos mais de um arquivo fonte para o mesmo problema, sendo que todos compilados em conjunto vão gerar o arquivo executável, e o atributo Nota que será atribuído assim que terminar a correção. Esse relacionamento possui cardinalidade M:N já que um aluno pode submeter vários trabalhos e um trabalho será submetido por vários alunos. Entre as entidades *Admin* e *TP* há o relacionamento Cria, que não possui atributos e possui cardinalidade 1:N já que cada trabalho será criado por apenas 1 administrador e cada administrador pode criar vários trabalhos.

Partindo deste diagrama ER foi feito o mapeamento para o Modelo Relacional, que representa as tabelas físicas do Banco de Dados. A tabela Usuario (Tabela 1) vai representar todos os usuários diferenciando a qual entidade especifica (Aluno ou Admin) que este pertence pelos atributos *booleanos* de tipo *EhAluno* e *EhAdmin*, sendo que caso *EhAdmin* seja *true*, *Media* ficará *null*. Esta tabela tem como chave primária o atributo *RA*.

USUARIO	RA	Senha	Nome	Idade	Sexo	EhAluno	Media	EhAdmin

Tabela 1: Tabela Usuário

A entidade *TP* foi mapeada para a tabela *TP* (Tabela 2), com todos os seus atributos, sendo chave primária os atributos Nome, Semestre e Ano. Ao mapear o relacionamento Cria entre *TP* e *Admin*, foi criado na tabela de *TP* uma chave estrangeira RA para relacionar o *TP* com seu usuário criador.

TP	Nome	Semestre	Ano	Espec	Result	Teste	Valor	*RA*
TP(RA) referencia USUARIO(RA)								

Tabela 2: Tabela *TP*

O relacionamento Submete foi mapeado como uma tabela que possui como chaves primárias RA, Nome, Semestre e Ano, sendo a primeira chave estrangeira de Usuário e as demais chave estrangeira de *TP*. Além disso tem o atributo Nota. O Atributo Arquivos por ser multi valorado necessitou ser mapeado como outra tabela, sendo que esta possui como atributos RA, Nome, Semestre, Ano e Arquivo, sendo todas chaves primárias, sendo as quatro primeiras chave estrangeira da tabela SUBMETE. A Tabela 3 mostra estes relacionamentos.

SUBMETE	*RA*	*Nome*	*Semestre*	*Ano*	Nota
SUBMETE(RA) referencia USUARIO(RA) SUBMETE(Nome,Semestre,Ano) referencia *TP*(Nome,Semestre,Ano)					
ARQUIVOS	*RA*	*Nome*	*Semestre*	*Ano*	Arquivo
ARQUIVOS(RA,Nome,Semestre,Ano) referencia SUBMETE(RA,Nome,Semestre,Ano)					

Tabela 3: Tabelas SUBMETE e ARQUIVO

A partir destas tabelas foi possível gerar o código SQL para criação das tabelas dentro do banco de dados. Este código *SQL* é mostrado no Programa 1.

```
create domain TRA as char(7);--Tipo RA
create domain TF as varchar(200);--Tipo File

--tabela que armazena os dados dos alunos
create table USUARIO{
  RA TRA NOT NULL,
  Senha varchar(12) NOT NULL,
  Nome varchar(100),
  Idade Int,
  Sexo char,
  EhAluno boolean DEFAULT true,
  Media float,
  EhAdmin boolean DEFAULT false,
  constraint USUSEX check(Sexo='M' OR Sexo='F'),
  constraint USUMED check(media>0),
  constraint USUPK primary key(RA)
};

--tabela que armazena dados basicos dos tps
create table TP{
  Nome char(3) NOT NULL,
  Semestre int NOT NULL,
  Ano int NOT NULL,
  Espec TF,
  result TF,
  teste TF,
  Valor float,
  RA TRA NOT NULL,
  constraint TPVAL check(Valor>0),
  constraint USUPK primary key(Nome,Semestre,Ano),
  constraint TPUSUFK foreign key(RA) references USUARIO(RA) on delete
      cascade on update cascade
};

--tabela que armazena as submiss es de cada aluno para cada tp
create table SUBMETE{
  RA TRA NOT NULL,
  Nome char(3) NOT NULL,
  Semestre int NOT NULL,
  Ano int NOT NULL,
  Nota float,
  constraint SUBNOT check(Nota>0),
  constraint SUBPK primary key(RA,Nome,Semestre,Ano),
  constraint SUBUSUFK foreign key(RA) references USUARIO(RA) on delete
```

```
        cascade on update cascade,
    constraint SUBTPFK foreign key(Nome,Semestre,Ano) references TP(Nome,
        Semestre,Ano) on delete cascade on update cascade
};

--tabela que armazena arquivos submetidos por alunos
create table ARQUIVOS{
    RA TRA NOT NULL,
    Nome char(3) NOT NULL,
    Semestre int NOT NULL,
    Ano int NOT NULL,
    Arquivo TF NOT NULL,
    constraint ARQPK primary key(RA,Nome,Semestre,Ano,Arquivo),
    constraint ARQSUBFK foreign key(RA,Nome,Semestre,Ano) references
        SUBMETE(RA,Nome,Semestre,Ano) on delete cascade on update cascade
};
```

Programa 1: Código SQL para gerar as Tabelas

5.1.3 Modelagem da interação

Para modelar a interação dos usuários com o sistema, foi utilizada a linguagem de modelagem da interação como conversa (*MoLIC*). Nesta linguagem as ações e falas dos usuários são precedidas por *u:*, já as ações e falas do sistema são precedidas por *d:* representando o preposto do *designer*. A interação é baseada em cenas onde ocorrem a conversa do sistema e o usuário. Cada retângulo branco representa uma cena. Na parte superior se encontra o nome da cena e as demais informações representam o que cada um dos envolvidos na interação (usuário e preposto do designer) precisam informar para o sucesso da interação, sendo que apenas são obrigatórios os campos com "*". Os quadrados pretos indicam processamento do sistema, o círculo preto indica o inicio da interação, e as setas indica as interações entre o usuário e o sistema. O diagrama 2 mostra a interação entre os usuários e o SWAT.

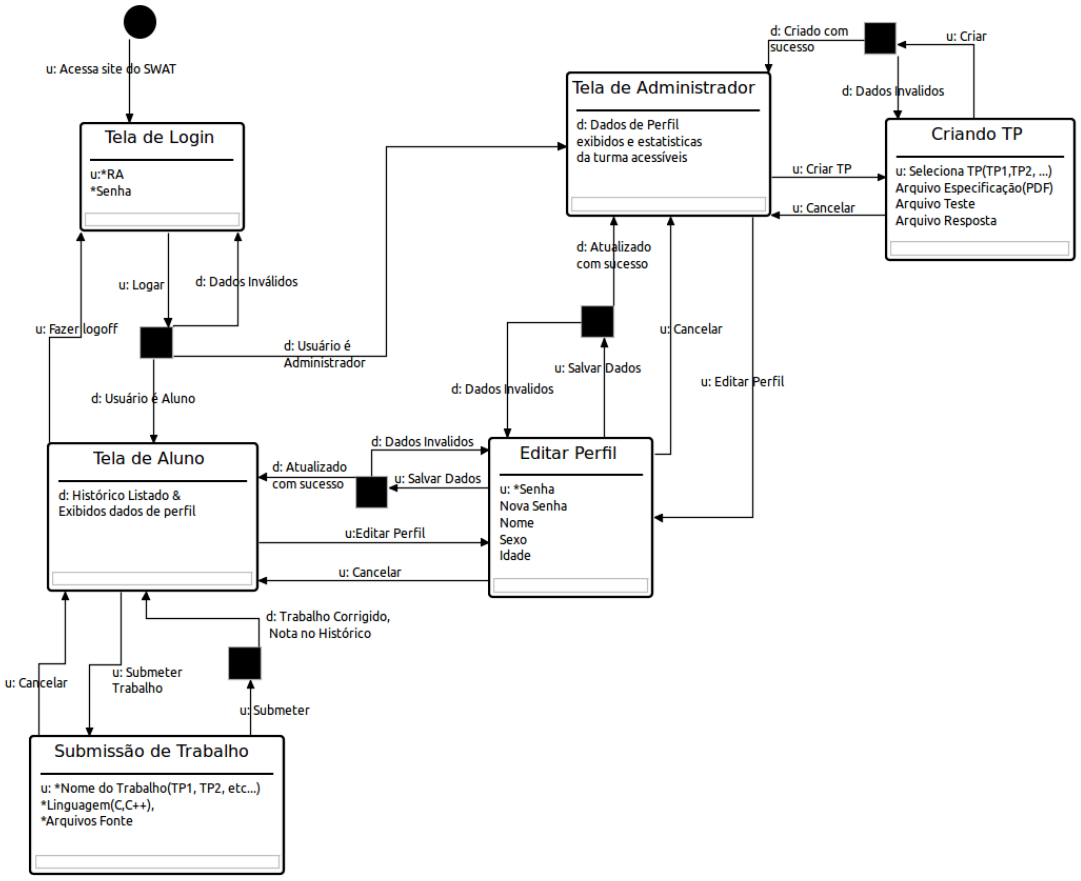

Figura 2: Diagrama MoLIC

A escolha do MoLIC se deu pelo fato de facilmente visualizar as várias telas do sistema, pelas cenas da interação. Seguindo o diagrama percebe-se que o sistema terá pelo menos 6 telas diferentes.

5.2 Próximos Passos

Baseando no diagrama MoLIC será feito o desenho das telas do sistema utilizando as cenas do diagrama e seus atributos. Em seguida será feita a codificação em HTML dos desenhos das telas. Este código será interligado com o banco de dados utilizando PHP, linguagem que também permitirá fazer a compilação dos códigos fonte utilizando o *GCC* e seus parâmetros, além da comparação de arquivos utilizando o comando *diff* do Linux. Após esta sincronização já poderão ser executados pequenos testes do sistema. Enquanto isso será feito o aprimoramento gráfico do sistema com a utilização de CSS e Javascript. Depois o sistema estará pronto para a execução de testes mais pesados, sendo o mesmo disponibilizado aos alunos de forma não obrigatória. Após a resolução dos possíveis problemas do sistema, o mesmo será aplicado em larga escala na disciplina BCC202 e sua aplicação em outras disciplinas será estudada com os outros professores.

6 Cronograma de atividades

As atividades desenvolvidas e a serem desenvolvidas são mostradas na lista abaixo:

1. Estudo das tecnologias a serem utilizadas;
2. Modelagem do sistema;
3. Desenvolvimento do sistema;
4. Teste preliminares;
5. Testar o sistema em um trabalho;
6. Resolução de problemas observados;
7. Aplicar o sistema em EDI;
8. Elaboração do relatório de atividades;
9. Apresentação do relatório de atividades;
10. Redigir a Monografia;
11. Apresentação da monografia para banca;
12. Escrita de um artigo.

Na Tabela 4, apresenta-se o cronograma para desenvolvimento das atividades da lista apresentada acima.

Atividades	**Nov**	**Dez**	**Jan**	**Fev**	**Mar**	**Abr**	**Mai**	**Jun**
1	Concluído							
2	Concluído							
3	X	X	X	X				
4	X	X	X					
5				X	X			
6	X	X	X	X	X	X	X	X
7					X	X	X	X
8	X							
9		X						
10					X	X	X	X
11								X
12			X	X	X	X	X	X

Tabela 4: Cronograma de Atividades.

Referências

[1] Cerli Antônio da Rocha. *Desenvolvendo Websites Dinâmicos PHP - ASP - JSP*. Editora Campus, 2007. `http://www.naredemundial.com.br/livro/` visitado em 9 de dezembro de 2010.

[2] Cássio P. de Campos and Carlos E. Ferreira. BOCA: um sistema de apoio a competições de programação. *SBC Workshop de Educação em Computação*, Agosto 2004.

[3] Ramez Elmasri and Shamkant B. Navathe. *Sistema de Banco de Dados*. Addison Wesley, 4th edition, 2005.

[4] André Milani. *PostgreSQL Guia do Programador*. Editora Novatec, 2008.

[5] Raquel Oliveira Prates and Simone Diniz Junqueira Barbosa. Introdução à teoria e prática da interação humano computador fundamentada na engenharia semiótica. In *T Kowaltowiski and K. Breitman (orgs.) Jornada de Atualização em Informática, JAI 2007*, pages 263–326, 2007.

[6] XV maratona de programação. `http://maratona.ime.usp.br/`, 2010. Visitado em 9 de dezembro de 2010.

[7] The ACM-ICPC international collegiate programing contest web syte sponsored by IBM. `http://icpc.baylor.edu/`, 2010. Visitado em 9 de dezembro de 2010.

[8] PHP: Hypertext preprocessor. `http://www.php.net/`, 2010. Visitado em 9 de dezembro de 2010.

[9] Cascading Style Sheets - Wikipédia. `http://pt.wikipedia.org/wiki/Cascading_Style_Sheets`, 2010. Visitado em 9 de dezembro de 2010.

[10] JavaScript - Wikipédia. `http://pt.wikipedia.org/wiki/JavaScript`, 2010. Visitado em 9 de dezembro de 2010.

[11] PHP - wikipédia. `http://pt.wikipedia.org/wiki/PHP`, 2010. Visitado em 9 de dezembro de 2010.

[12] Maurício Samy Silva. *Construindo Sites com CSS e (X)HTML: sites controlados por folhas de estilo em cascata*. Editora Novatec, 2008.

[13] Élton José da Silva. Modelagem de Projeto de IHC. In *Sistemas Interativos*, pages 92–112. Élton José da Silva, 2006.