

Trabalho de Conclusão de Curso apresentado como requisito para a obtenção do grau de Tecnólogo em Análise e Desenvolvimento de Sistemas pela Universidade Tecnológica Federal do Paraná - Campus Cornélio Procópio.

Profº. Eidy Leandro Tanaka Guandeline
Orientador

Profº. Eduardo Cotrin
Membro da banca examinadora

Profº. André Luis Przybysz
Membro da banca examinadora

DEDICATÓRIA

À minha mãe Elenice Aparecida Ferreira.

AGRADECIMENTOS

Agradeço à Deus, pois sem Ele minha vida não teria sentido e minhas conquistas não seriam alcançadas.

Agradeço a toda minha família e a minha namorada Jacqueline Gonçalves de Oliveira, pelo apoio, paciência e compreensão nos momentos que estive ausente durante o ano.

Agradeço ao meu orientador e professor Prof^o. Eidy Leandro Tanaka Guandeline, que me auxiliou nas etapas desse trabalho.

Agradeço meu grande amigo Bruno Capel que aceitou comigo a enfrentar o desafio deste trabalho.

Também agradeço aos meus amigos de turma Marcelo Sampaio e Harold Gruber que me ajudaram durante todo o curso e aos profissionais da BSI Tecnologia que convivem comigo no dia-a-dia me auxiliando em meu crescimento profissional.

EPÍGRAFE

*“Lâmpada para os meus pés é a tua palavra, e luz para o
meu caminho.”*

(Salmos 119:105)

RESUMO

Este trabalho apresenta um sistema para gerenciar a matéria de Trabalho de Conclusão de Curso. O software tem por finalidade gerenciar e centralizar as informações da matéria de TCC da Universidade Tecnológica Federal Campus Cornélio Procopio para todos os professores e alunos matriculados. A construção do software foi planejada em dois (02) módulos, tendo como referência neste trabalho, somente o módulo intitulado Aluno / Professor. Este módulo tem como responsabilidade gerenciar as atividades referentes aos alunos e professores, como acesso a agenda, correio, avisos e documentos. Para guiar o desenvolvimento, foi utilizado uma adaptação do processo de desenvolvimento *easYProcess*, juntamente com a linguagem de modelagem UML. As tecnologias e recursos empregados foram a linguagem de programação Visual Basic .NET na plataforma ASP .NET, com o Framework .NET 3.5 e banco de dados SQL Server 2005.

[Palavras chave: TCC, easYProcess e ASP. NET]

ABSTRACT

This paper presents a system to manage the matter of Finish Course Work (FCW). The software aims to manage and centralize information in respect of FCW of the "Universidade Tecnológica Federal Campus Cornélio Procópio" for all teachers and students. The construction of the software was planned in two (02) modules, with reference to this work, only the module Student / Teacher. This module has the responsibility to manage activities relating to students and teachers, such as access to calendar, mail, documents and notices. To guide the development, we used an adaptation of the development process easYProcess, along with the language of UML modeling. The technology and resources were employed with the programming language Visual Basic. NET platform in ASP. NET, with the Framework. NET 3.5 and SQL Server database, 2005.

[Key words: FCW, easYProcess and ASP. NET]

LISTA DE FIGURAS

2.1	O processo YP.....	10
2.2	Divisão de uma <i>Release</i>	13
3.1	Definição dos papéis.....	14
3.2	Diagrama de casos de uso geral - SGTCC.....	16
3.3	Atores.....	17
3.4	Diagrama de casos de uso para o Módulo Aluno / Professor.....	17
3.5	Arquitetura física - SGTCC.....	20
3.6	Arquitetura lógica (Camadas da aplicação) - SGTCC.....	20
3.7	Módulos - SGTCC.....	21
3.8	Modelo lógico de dados - Módulo Aluno / Professor.....	21
3.9	Modelo lógico de dados - SGTCC.....	22
3.10	Modelo estrutural da interface.....	23
3.11	Primeiro protótipo de tela.....	23
3.12	Segundo protótipo de tela.....	24
3.13	Cronograma de execução nas atividades do YP.....	25
3.14	Diagrama de Classes - Módulo Aluno / Professor.....	27
3.15	Diagrama de Classes - SGTCC.....	28
3.16	Diagrama de Seqüência.....	29
3.17	Estrutura de uma classe.....	30
3.18	Comentário de um método.....	31
3.19	Método de persistência dos dados utilizando DAAB.....	32
3.20	Controle Grid desenvolvido como um <i>User Control</i>	32
3.21	Autenticação configurada no <i>Web.config</i>	33
3.22	Tela Principal.....	34

3.23	Tela de Login	34
3.24	Tela de Gerenciamento de Arquivos	35
3.25	Tela de Agenda Pessoal.....	36
3.26	Tela de Correio	37

LISTA DE QUADROS

3.1	Usuários do Módulo Aluno / Professor	15
3.2	Descrição do Caso de Uso: Gerenciar Agenda	18
3.3	Descrição do Caso de Uso: Gerenciar Mensagens	18
3.4	Descrição do Caso de Uso: Gerenciar Documentos	18
3.5	Descrição do Caso de Uso: Gerenciar Avisos	19
3.6	Definição das <i>User Stories</i> para o Módulo Aluno / Professor	19
3.7	Tabela de Alocação de Tarefas (TAT)	26

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
CLR	<i>Common Language Runtime</i>
CSS	<i>Cascading Style Sheet</i>
DAAB	<i>Data Access Application Block</i>
IDE	Ferramenta Integrada de Desenvolvimento
MSP	Microsoft Student Partner
PAG	Arquitetura Prescritiva
TAT	Tabela de Alocação de Tarefas
TCC	Trabalho de Conclusão de Curso
UML	<i>Unified Modeling Language</i>
US	<i>User Stories</i>
UTFPR	Universidade Tecnológica Federal do Paraná
W3C	<i>World Wide Web Consortium</i>
WEB	<i>World Wide Web</i>

SUMÁRIO

DEDICATÓRIA.....	ii
AGRADECIMENTOS.....	iii
EPÍGRAFE.....	iv
RESUMO.....	v
ABSTRACT.....	vi
LISTA DE FIGURAS.....	vii
LISTA DE QUADROS.....	ix
LISTA DE ABREVIATURAS E SIGLAS.....	x
1 INTRODUÇÃO.....	1
1.1 APRESENTAÇÃO	1
1.2 OBJETIVOS	2
1.2.1 Objetivo geral	2
1.2.2 Objetivos específicos	3
1.3 JUSTIFICATIVAS	3
1.4 ORGANIZAÇÃO DO TRABALHO	4
2 REVISÃO BIBLIOGRÁFICA.....	5
2.1 TECNOLOGIAS	5
2.1.1 Microsoft .NET	5
2.1.2 <i>Visual Basic</i> .NET	5

2.1.3	ASP .NET.....	5
2.1.4	ASP .NET AJAX	6
2.1.5	<i>Javascript</i>	6
2.1.6	<i>Cascading Style Sheet (CSS)</i>	6
2.1.7	<i>Microsoft Visual Studio 2008</i>	6
2.1.8	<i>Microsoft Expression Web</i>	6
2.1.9	<i>Microsoft SQL Server 2005</i>	7
2.1.10	Tex, Latex e Utex	7
2.1.11	<i>Enterprise Architect</i>	7
2.2	PADRÕES DE PROJETO	7
2.2.1	<i>Microsoft Application Blocks</i>	7
2.2.2	<i>Data Access Application Block</i>	8
2.3	METODOLOGIAS	9
2.3.1	<i>Unified Modeling Language (UML)</i>	9
2.3.2	Processo de Desenvolvimento de Software <i>easYProcess</i> (YP)	9
3	APLICAÇÃO.....	14
3.1	DEFINIÇÃO DOS PAPÉIS	14
3.2	CONVERSA COM O CLIENTE	14
3.2.1	Documento de Visão	15
3.3	INICIALIZAÇÃO	15
3.3.1	Modelagem dos casos de uso	16
3.3.2	<i>User Stories</i>	19
3.3.3	Projeto Arquitetural.....	19
3.3.4	Modelo Lógico de Dados	21
3.3.5	Prototipação	22
3.4	PLANEJAMENTO	24

3.5	IMPLEMENTAÇÃO	26
3.5.1	Modelagem	26
3.5.2	Codificação	30
3.5.3	Testes	37
4	CONCLUSÃO	38
4.1	CONSIDERAÇÕES FINAIS	38
4.2	TRABALHOS FUTUROS	39
	REFERÊNCIAS.....	40
	Apêndice A – PROPOSTA DE TRABALHO DE CONCLUSÃO DE CURSO	41
	Apêndice B – GLOSSÁRIO	42
B.1	DEFINIÇÕES	42
	Apêndice C – PADRÕES DE CODIFICAÇÃO E DESENVOLVIMENTO	44

1 INTRODUÇÃO

1.1 APRESENTAÇÃO

A matéria de Trabalho de Conclusão de Curso (TCC) faz parte obrigatória da grade curricular dos cursos da Universidade Tecnológica Federal do Paraná (UTFPR). Tendo como finalidade, a elaboração de um projeto em que o aluno aplique de forma integrada, os conhecimentos obtidos entre as disciplinas lecionadas durante o curso ao qual o mesmo esteja matriculado. Seu principal objetivo é avaliar a capacidade do aluno em executar um projeto de pesquisa, estimulando assim a construção do conhecimento coletivo, a inovação tecnológica, sua interdisciplinaridade e ajudar na resolução de problemas existentes na sociedade. A execução do trabalho é acompanhada por um professor designado como orientador, sendo escolhido pelo aluno ou indicado pelo coordenador da matéria de TCC, para auxiliá-lo desde a idealização do projeto de pesquisa até a defesa e entrega da versão final do trabalho.

O processo de execução do TCC é feito por meio de 02 (duas) bancas. Nelas participam o orientador e dois professores convidados. Na primeira, o aluno tem como objetivo apresentar o tema que se propõe a resolver, gerando assim um plano de execução e os resultados esperados. E na última, é feita a defesa do trabalho final, por meio da entrega da redação final do TCC para avaliação da banca e apresentação de todo o trabalho desenvolvido, expondo os resultados obtidos e dificuldades encontradas. Nas regras atuais definidas pelo coordenador, a execução do trabalho de TCC deve ser feita no prazo máximo de 01 (um) ano.

Atualmente, o coordenador da matéria de TCC conta com um sistema bastante simples para o gerenciamento dos trabalhos ativos na instituição. Este software foi desenvolvido para ser utilizado em um ambiente *desktop* de uso exclusivo do coordenador. Por meio deste, é possível ter um controle simples dos alunos e professores da matéria, agendamento de bancas, sem nenhum filtro para professores e datas, e consultas de bancas, alunos e professores. O coordenador também conta com um website publicado no servidor da instituição onde são disponibilizadas algumas informações sobre a matéria, data de entrega dos trabalhos e bancas, e download de alguns documentos e modelos para auxiliar o aluno no desenvolvimento do TCC.

Mesmo não sendo ferramentas completas, o sistema *desktop* e o website possuem certas vantagens que ajudam a resolver algumas necessidades básicas do coordenador como: divulgação de informações importantes da matéria aos alunos de modo simples através de um meio de comuni-

cação aberto e bastante acessado por todos como a Internet, e poder contar com um acesso rápido a dados importantes sobre os matriculados na matéria por meio do sistema *desktop*. Apesar das vantagens que os dois sistemas proporcionam, ainda restam recursos para que os mesmos se tornem ferramentas completas para auxílio acadêmico.

1.2 OBJETIVOS

A definição dos objetivos ajudaram a entender as metas a serem atingidas para a solução do problema descrita neste trabalho.

1.2.1 Objetivo geral

O objetivo geral deste trabalho foi o desenvolvimento do sistema intitulado Sistema Gerenciador para Trabalhos de Conclusão de Curso (SGTCC), utilizando a plataforma ASP .NET com os conceitos da programação orientada a objeto e implementado para o ambiente *World Wide Web* (WEB).

O SGTCC é um sistema WEB a ser implantado na UTFPR - Campus Cornélio Procopio, visando unificar a idéia dos sistemas já existentes em um só, permitindo que funcionalidades e informações sobre a matéria sejam centralizadas em um único local para acesso dos alunos, professores e coordenador por meio da Internet. Propondo assim uma ferramenta de auxílio acadêmico que permitirá o coordenador da matéria de TCC manter o controle de quase todo o processo de andamento dos alunos matriculados e fazer com que tanto o aluno quanto o professor tenham mais integração no processo de elaboração dos trabalhos.

Para o desenvolvimento do SGTCC foi definida a divisão do sistema em dois (02) módulos denominados de:

- Módulo Aluno / Professor.
- Módulo Coordenador / Administrador.

O Módulo Aluno / Professor, chamado também de Módulo AP, tem como objetivo gerenciar todo o acesso e informações disponíveis para os alunos e professores. Também contam com o uso de uma agenda, correio, avisos e download de arquivos disponibilizados pelo coordenador da matéria. Os principais objetivos e funcionalidades deste módulo será detalhado no decorrer deste trabalho.

O Módulo Administrador / Coordenador, chamado também de Módulo AC, tem como objetivo gerenciar as informações sobre os usuários do sistema, os TCCs matriculados e as bancas de proposta e defesa final, assim como consultas e relatórios. O responsável pelo desenvolvimento deste módulo ficou a cargo do aluno Bruno Gustavo Carvalho Capel.

1.2.2 Objetivos específicos

O objetivo específico inicial desde trabalho foi o estudo das tecnologias e do processo de desenvolvimento que foram utilizados durante a elaboração e construção do sistema, visando assim sua adaptação. Uma síntese do processo foi elaborada para que se pudesse entender suas fases e mapear o estado do projeto em seu contexto.

Em segundo, foi o desenvolvimento das funcionalidades essenciais que estão agregadas a este módulo, sendo estas:

- Sistema de mensagens entre aluno e professor, permitindo a comunicação e centralização dos assuntos relacionados ao TCC;
- Agenda pessoal, permitindo gerenciar cronogramas, datas importantes e reuniões aluno e orientador;
- Área de visualização de avisos recentes sobre a matéria, e;
- Disponibilização de arquivos para downloads, como modelos de trabalhos, folhas de aprovação e trabalhos já apresentados.

Os acessos a determinadas áreas do sistema foram todas gerenciados por nível de usuário, ou seja, áreas disponíveis para o professor não serão acessadas por alunos.

Outra funcionalidade desenvolvida que não fazia parte do escopo da proposta deste trabalho e não estava diretamente relacionada a nenhum dos módulos, foi o acesso de usuários públicos ao sistema. Essas funcionalidades tratam-se de conteúdo destinado a usuários que muitas vezes ainda não fazem a matéria, mas que tem permissão de acesso as áreas livre do sistema para consultar informações sobre a mesma e fazer o download de itens disponibilizados pelo coordenador. Essas funcionalidades em comum foram divididas em igual esforço entre os módulos.

1.3 JUSTIFICATIVAS

De modo geral, os seguintes itens justificaram as vantagens de se desenvolver um sistema acadêmico para gerenciamento da matéria de TCC:

- Unificar e centralizar em um único sistema de acesso livre, todas as informações sobre a matéria de TCC da UTFPR Campus Cornélio Procopio.
- Sistema reescrito do zero, utilizando-se das tecnologias e padrões de desenvolvimento mais recente disponíveis no mercado, garantindo um sistema robusto e de qualidade com o máximo de suporte caso o mesmo sofra manutenções ou sejam adicionadas funcionalidades que estão fora do escopo inicial do projeto.

Especificamente para o Módulo AP, a criação das novas funcionalidades irão garantir uma maior integração entre os envolvidos, para que além do coordenador, tanto o aluno quanto o professor possam ter acesso ao sistema e assim melhorar a comunicação entre ambos.

A decisão em construir um sistema para o ambiente WEB, se deu visto que os custos do desenvolvimento e implantação são bem menor comparado aos sistemas *desktop*, já que toda a aplicação deve ser instalada em uma única máquina agindo como servidor, tornando o usuário independente do sistema operacional. Sem contar a vantagem de manutenção em tempo real, ou seja, atualizando-se o servidor, os usuários acessam a última versão instantaneamente.

1.4 ORGANIZAÇÃO DO TRABALHO

Existem mais três capítulos além deste que resumem os objetivos do trabalho. Eles estão divididos da seguinte forma:

Capítulo 2: resume a fundamentação teórica do trabalho, e esta, consiste na revisão bibliográfica dos temas utilizados e na explicação do processo de desenvolvimento.

Capítulo 3: apresenta a aplicação das tecnologias e do processo de desenvolvimento adaptado na construção do Módulo Aluno / Professor. Além de ilustrar alguns dos produtos gerados e recursos utilizados.

Capítulo 4: as conclusões deste trabalho são apresentadas, e o relato dos objetivos alcançados, juntamente com o apontamento dos possíveis trabalhos futuros que podem ser realizados.

2 REVISÃO BIBLIOGRÁFICA

Neste trabalho, a pesquisa de tecnologias e metodologias utilizadas foram fundamentais para sua realização. Com isso, um resumo da revisão bibliográfica dos temas utilizados é detalhado nas seções seguintes.

2.1 TECNOLOGIAS

Nesta seção serão abordadas as tecnologias que foram de crucial importância para a realização deste módulo. Assim, é descrito uma visão geral sobre cada uma juntamente com um breve relato da fase em que a mesma foi utilizada durante o processo de desenvolvimento.

2.1.1 Microsoft .NET

Plataforma de desenvolvimento unificado utilizado na construção de vários tipos de aplicativos (BARWELL; CASE; FORGEY, 2004). Utilizada durante todas as etapas de implementação.

2.1.2 Visual Basic .NET

Uma das linguagens disponíveis na plataforma .NET. Criado com suas raízes em *BASIC*, ao qual herdou boa parte do seu legado, a linguagem Visual Basic surgiu com o objetivo de ser a maneira mais fácil para qualquer um programar qualquer coisa (MACDONALD, 2002). Utilizada durante todas as etapas de implementação.

2.1.3 ASP .NET

Plataforma que fornece suporte a um modelo de desenvolvimento WEB unificado que inclui os serviços necessários para que sejam criados aplicativos da WEB de nível empresarial (MICROSOFT, 2008c). Utilizada durante todas as etapas de implementação.

2.1.4 ASP .NET AJAX

Solução AJAX da Microsoft e refere-se a um conjunto de tecnologias de cliente e servidor, que se concentram na melhoria do desenvolvimento WEB com a ferramenta Visual Studio (GIBBS; WAHLIN, 2007). Utilizada na maior parte das etapas de implementação.

2.1.5 Javascript

Linguagem de programação destinada para o uso em páginas WEB do lado cliente. Permitindo ao desenvolvedor ter acesso aos elementos de uma página WEB fornecendo a capacidade de criar funcionalidades baseadas nas ações do usuário (FLANAGAN, 2004). Utilizada na maior parte das etapas de implementação.

2.1.6 Cascading Style Sheet (CSS)

Linguagem de estilos utilizada para a definição da aparência de páginas WEB. Com ela é possível definir fontes, cores, margens, posicionamento, tamanho, e outros, usando-se uma técnica diferente da convencional (W3C, 2008). Utilizada na maior parte das etapas de implementação.

2.1.7 Microsoft Visual Studio 2008

Ferramenta Integrada de Desenvolvimento (IDE) completa para construção de aplicações WEB ASP.NET, serviços WEB XML, aplicativos *desktop* e aplicativos móveis (MICROSOFT, 2008e). O Visual Studio 2008 traz avanços fundamentais para os desenvolvedores em três pilares principais: Produtividade, Colaboração e Experiência de usuário (MICROSOFT, 2008d). Utilizada na maior parte das etapas de implementação.

2.1.8 Microsoft Expression Web

Ferramenta de design WEB descendente do Microsoft FrontPage. Com o Expression Web é fácil diminuir a complexidade e facilitar a integração de dados usando robustas ferramentas de design e painéis de tarefa para criar designs para ASP.NET, PHP e XML (MICROSOFT, 2008a). Utilizada na maior parte das etapas de implementação.

2.1.9 *Microsoft SQL Server 2005*

Gerenciador de Banco de dados relacional e muito robusto. Entre os seus principais recursos, está a integração com o .NET Framework, que possibilita a construção de rotinas utilizando as linguagens do do .NET como VB.NET e C# (MICROSOFT, 2008b). Utilizada na maior parte das etapas de implementação.

2.1.10 *Tex, Latex e Utex*

O Tex é um sistema de editoração de textos, criado no final da década de 1970 por Donald Knuth, que oferece controle sobre a estrutura do documento a ser editado e, justamente por isso, traz uma complexidade na edição indesejável à maioria dos usuários comuns. LaTeX é um conjunto de macros escritas para o TeX que permite a criação de textos na mais alta qualidade tipográfica, usando um leiaute profissional pré-definido. Portanto, o LaTeX nada mais é que uma interface para o Tex, mas obviamente, mais simples de ser utilizada. O UTeX é um projeto realizado para a UTFPR unidade de Cornélio Procópio, e tem como objetivo padronizar as normas técnicas para elaboração de propostas e monografias de conclusão de curso de acordo com as normas da Associação Brasileira de Normas Técnicas (ABNT) (UTFPR, 2008). Utilizada durante a escrita da monografia.

2.1.11 *Enterprise Architect*

Ferramenta de análise, projeto e desenvolvimento de aplicações em Unified Modeling Language (UML) (SYSTEM, 2008). Utilizada nas fases de Inicialização e planejamento.

2.2 PADRÕES DE PROJETO

Os Padrões de Projeto, mais conhecido como *Design Patterns*, são conceitos e práticas que descrevem soluções para problemas no desenvolvimento de sistemas de software orientado a objetos. A utilização de padrões tem como objetivo facilitar o desenvolvimento de software e promover a reutilização de soluções de *design* (BISHOP, 2008).

2.2.1 *Microsoft Application Blocks*

Pensando nos problemas recorrentes de projetos de aplicações multicamadas, usando tecnologia .NET, a área de Arquitetura Prescritiva (PAG) da Microsoft passou a disponibilizar elementos chamados de *Application Blocks* ou Blocos de Aplicação, para agilizar o projeto e o desen-

volvimento de aplicações com maior qualidade e previsibilidade. Esses elementos foram criados com a preocupação de que os projetos na plataforma .NET tenham um nível de qualidade cada vez maior, reduzindo os riscos de insucesso com a utilização de abordagens algumas vezes inadequadas.

Os Blocos de Aplicação, ou simplesmente Blocos, são elementos estruturais desenvolvidos sobre o .NET e que complementam os recursos do *Framework* com implementações padrão baseadas em alguns dos *patterns* existentes e bastante conhecidos da comunidade (MANFREDI, 2003).

2.2.2 *Data Access Application Block*

O *Data Access Application Block* (DAAB) é uma camada de acesso a dados desenvolvida pela Microsoft utilizando-se de melhores práticas e padrões, garantindo ao desenvolvedor um alto desempenho em suas aplicações de uma maneira mais simples (MANFREDI, 2003). O DAAB foi projetado para solucionar estes problemas através do isolamento do código de chamada para as APIs de acesso a dados - ADO.NET - em um único componente, exigindo o mínimo de codificação por parte do desenvolvedor.

Em particular, este bloco de código auxilia o desenvolvedor com:

- Chamada a *stored procedures* e comandos SQL;
- Especificação de detalhes de parâmetros;
- Recebimento de objetos do tipo: *SqlDataReader*, *DataSet*, *XmlReader*, ou valores únicos;
- Utilização de tabelas e campos fortemente tipificados;
- Suporte a armazenamento em *cache* de parâmetros para casos onde exista esta necessidade;
- Inclusão de novas tabelas em um conjunto de dados existente - *datasets*;
- Atualização de um *dataset* com o uso de comandos especificados pelo usuário;
- Criação de objetos *SqlCommand*;
- Utilização de registro de dados fortemente tipificados como substituto a parâmetros.

2.3 METODOLOGIAS

2.3.1 *Unified Modeling Language* (UML)

A UML é uma linguagem visual para a elaboração da estrutura de projetos de software. A UML é uma ferramenta de análise. A análise é traduzir requisitos em componentes de software, descobrindo informações. Isso também é chamado de *design* ou modelagem.

A modelagem é um artefato importante quanto à implementação de um bom software, pois é um modelo que explica as características e comportamento do sistema a ser desenvolvido. Quando está se desenvolvendo um software, a modelagem servirá para identificar as características e funcionalidades e auxiliará no planejamento de sua construção (BOOCH; RUMBAUGH; JACOBSON, 2005).

2.3.2 Processo de Desenvolvimento de Software *easYProcess* (YP)

Para o desenvolvimento organizado de um sistema de informação, deve-se utilizar um processo de desenvolvimento bem definido, o qual corresponde a uma seqüência de atividades a serem seguidas em que cada etapa da atividade é executada assim que a anterior termina. Com a necessidade de projetos mais robustos e com um curto espaço de tempo no desenvolvimento, foram aplicados os vários paradigmas da engenharia de software por meio de diferentes modelos de processo de desenvolvimento, tais como o modelo cascata, o modelo espiral e prototipação. Esses modelos podem ser combinados entre si de modo que atendam as necessidades do desenvolvimento do software, assim como novas atividades podem ser criadas durante o processo (PRESSMAN, 2006).

A necessidade de utilizar melhores práticas para desenvolvimento de software no meio acadêmico, que possibilitem maior sucesso na implementação de projetos oferecidos em algumas disciplinas, foi a principal motivação para a utilização do processo de desenvolvimento designado *easYProcess* (GARCIA et al., 2004). Este processo de desenvolvimento de software auxilia no atendimento destas necessidades e apresenta uma estrutura de processo simplificado.

Adotando a filosofia de desenvolvimento ágil baseado em práticas do XP, RUP e *Agile Modeling*, o YP é ideal para projetos de pequeno e médio porte atendendo equipes menores que necessitem de organização durante todo o seu desenvolvimento.

De maneira geral, o YP possui um fluxo de trabalho que se inicia com o levantamento do escopo do problema a ser resolvido. Após isso, o planejamento do projeto busca adequar o problema dentro de um cronograma bem definido, visto que este seja quebrado em pequenos *releases*. Com

isso o sistema é construído e testado a cada *release* usando as tecnologias e ferramentas definidas na visão inicial do processo de negócio. Ao final, o sistema é entregue ao usuário passando por uma avaliação técnica garantindo assim a sua qualidade. Os detalhes das etapas propostas pelo YP são descritas nas seções seguintes. A Figura 2.1 ilustra as atividades aplicadas pelo YP usando um fluxo incremental.

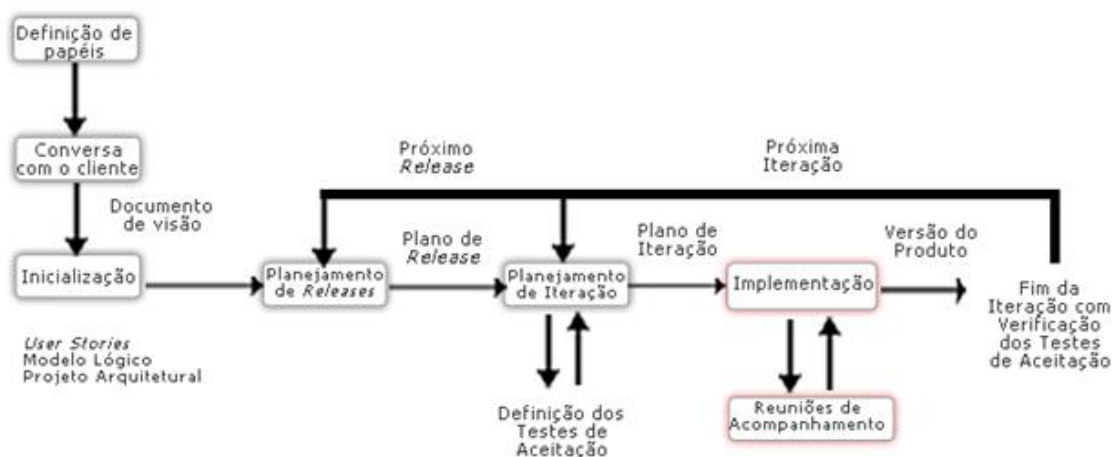


Figura 2.1: O processo YP.

Fonte: (GARCIA et al., 2004)

Definição dos Papéis

Ao montar uma equipe de desenvolvimento de software sugere-se uma divisão de tarefas entre os membros da mesma, de forma que cada um assuma um determinado papel no desenvolvimento. Um papel constitui um conjunto de responsabilidades que determina qual será o comportamento de uma pessoa durante o processo. No YP recomenda-se a presença de cinco papéis: cliente, usuário, gerente, desenvolvedor e testador.

A definição dos papéis deve ser feita com base nas necessidades de cada projeto, levando-se em consideração as habilidades e características de cada membro envolvido. Uma equipe com uma estrutura bem definida aumenta as chances de se obter a maior produtividade possível.

É essencial que todos os papéis estejam presentes na equipe, caso contrário etapas importantes do processo de desenvolvimento podem ser esquecidas ou acabam não recebendo a atenção necessária, prejudicando assim o andamento do projeto e aumentando suas chances de insucesso. Neste caso, um papel não corresponde necessariamente a uma só pessoa da equipe. Uma mesma pessoa pode desempenhar vários papéis simultaneamente em casos de que a equipe seja bem menor, porém, deve-se ter o cuidado para que não aconteça um acúmulo de papéis, levando assim uma sobrecarga de tarefas para algum membro da equipe.

Papéis e responsabilidades na visão do processo YP:

Cliente: Papel desempenhado por quem solicitou o desenvolvimento do software. Um dos papéis mais importantes do projeto, pois, é pra quem o software vai ser desenvolvido e sua presença ativa no processo é de suma importância para que suas necessidades e preferências sejam coletadas e atendidas.

Usuário: É quem de fato irá utilizar o sistema a ser desenvolvido. Suas preferências muitas vezes não são atendidas, pelo fato de quem está desenvolvendo ignorar o perfil de uso do usuário e acaba por projetar com base no seu próprio perfil.

Gerente: É o responsável por coordenar as atividades de todos os outros membros da equipe. Tomada de decisões que impactam em riscos e sucesso para o projeto é uma das funções importantes que um gerente deve exercer.

Desenvolvedor: O papel do desenvolvedor vai desde a modelagem dos requisitos do sistema até a sua codificação.

Testador: E por fim, mas não menos importante, o testador tem como responsabilidade avaliar a qualidade do código gerado pelo desenvolvedor, elaborando testes que comprovem que o sistema faz o que foi planejado para ser feito.

Conversa com o Cliente

Logo após a definição dos papéis dos membros da equipe de desenvolvimento, o próximo passo a ser executado no processo é a primeira conversa com o cliente. Esta deve ser feita com o cliente e toda a equipe do projeto, tendo como objetivo extrair informações sobre o que é realmente importante.

Ao final da conversa, a equipe do projeto deve ter em mãos uma idéia bem definida do escopo do problema, assim como o perfil dos usuários do sistema e suas necessidades, os requisitos funcionais e não funcionais e os possíveis riscos que o projeto possa gerar. Com todas essas informações, deve ser criado um artefato contendo a visão sobre os processos de negócio do cliente chamado de documento de visão.

O documento de visão irá ajudar a:

- Avaliar se a equipe de desenvolvimento entendeu corretamente o domínio do problema;
- Ser uma forma de contrato entre o cliente e o desenvolvedor;
- Servir de ponto de referência caso o projeto sofra muitas mudanças e;

- Ser útil para avaliar se o desenvolvimento do projeto é viável.

Inicialização

A etapa de Inicialização, compreende as atividades de análise e identificação das funcionalidades do sistema, elaborando-se uma arquitetura que o descreva em alto nível e sua modelagem inicial. Os três itens a seguir descrevem como alcançar o objetivo desta etapa:

User Stories: As *User Stories* (US) são funcionalidades do sistema descritas a partir da visão do cliente, ou seja, o cliente conta uma "história", descrevendo o que o sistema deve fazer de acordo com a sua preferência. Nessa "história do usuário", ele pode sugerir funcionalidades parecidas com a de outros softwares que ele já usou. No processo de coleta das US não é necessário que todas sejam definidas inicialmente, pois, durante todo o andamento do projeto outras poderão surgir, assim como as já existentes podem sofrer alguma alteração ou até mesmo ser descartada.

Projeto Arquitetural: O propósito deste artefato é descrever as partes do sistema e suas dependências em alto nível de abstração. O YP sugere que para compor o projeto arquitetural seja construído um diagrama contendo a estrutura da arquitetura, de forma que as possíveis dependências entre o sistema que está sendo desenvolvido e demais sistemas sejam explicitadas.

Modelo Lógico de Dados: Caso um banco de dados esteja envolvido no projeto, deve ser elaborado cuidadosamente um modelo lógico de dados buscando sua estabilidade para que mudanças não ocorram frequentemente, evitando atividades trabalhosas e demoradas de migração de dados.

Planejamento

A etapa de planejamento serve para definir os *releases* do projeto. Os *releases* são períodos pré-definidos em que são alocadas tarefas específicas para o desenvolvimento da aplicação. O YP aconselha que esses *releases* sejam compostos por duas ou no máximo três iterações por período letivo, aos quais são definidas também o tempo entre elas. As US devem ser alocadas nos *releases* e a partir daí nas iterações, levando-se em consideração a priorização sugerida pelo cliente. Porém, a equipe de desenvolvimento deve alertá-lo para as US que tecnicamente devem ser primeiramente implementadas.



Figura 2.2: Divisão de uma *Release*.

O YP sugere fazer uso da Tabela de Alocação de Tarefas (TAT), para listar todas as tarefas da iteração, especificando o tempo de desenvolvimento e o responsável por cada uma delas, levando em consideração as habilidades de cada um e o seu tempo de dedicação ao projeto. Para auxiliar na gerência do andamento das tarefas, é recomendável a construção de um cronograma das iterações determinando a data de início e término de cada tarefa.

Implementação

Após todo o planejamento de como o sistema deve ficar, é iniciado a implementação das tarefas alocadas na fase de iteração. Nesta etapa são gerados os artefatos, como modelagem dos diagramas e código fonte, que darão continuidade ao projeto, já que uma iteração não começa sem que a anterior não esteja finalizada. Além da tarefa de codificação das US, esta etapa também aborda a execução de três atividades que serão explicadas a seguir, ajudando a garantir a qualidade do projeto.

Reunião de Acompanhamento: O gerente de projeto deve executar as reuniões de acompanhamento semanalmente com toda a equipe, visando avaliar os resultados obtidos até o momento.

Relatórios: Ao final de cada release é aconselhável que seja criado um relatório identificando cada artefato gerado, como código fonte, problemas encontrados e soluções, a fim de manter o projeto documentado.

Testes: A etapa de implementação também abrange os períodos de teste que devem ser feitos para verificar se as funcionalidades atendem ao que foi especificado. Os testes aplicam uma análise das pré-condições e pós-condições diante do contexto no qual está inserido o módulo a ser testado. Definir e elaborar testes antes de codificar é a prática correta, pois auxilia a pensar mais no código a ser desenvolvido.

3 APLICAÇÃO

No decorrer do processo de desenvolvimento de software, diversos artefatos foram produzidos, e o resultado final foi o sistema funcionando e atendendo as necessidades requisitadas. Em cada fase do processo, métodos forneceram a técnica de "como fazer" para construir o software de qualidade. Um conjunto de tarefas definiram o trabalho real para atingir os objetivos de uma ação do processo.

As atividades previstas para o projeto ocorreram durante 10 meses. Assim, um resumo das tarefas realizadas, conforme atividades do processo YP, é detalhado nas seções seguintes.

3.1 DEFINIÇÃO DOS PAPÉIS

No fluxo de trabalho do processo utilizado, deu-se início a definição dos papéis de cada envolvido de acordo com as necessidades do escopo do projeto, para que todas as etapas recebessem a atenção necessária. Por este trabalho se tratar de uma aplicação modulada em duas pessoas, foi necessário atribuir todos os papéis possíveis para cada.

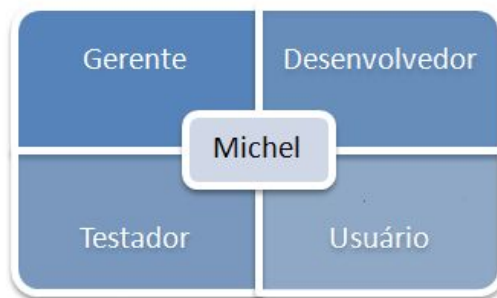


Figura 3.1: Definição dos papéis.

3.2 CONVERSA COM O CLIENTE

Durante a etapa de conversa com o cliente, um conjunto de tarefas propostas pelo YP foi realizado para a formulação do escopo inicial e análise do negócio a ser desenvolvido. Uma reunião com o cliente, no caso o coordenador da matéria de TCC, foi realizada para definir alguns pontos fundamentais para a construção do sistema, como as suas necessidades, características, restrições, entre outras. Com isso, uma descrição escrita do escopo foi elaborada e os requisitos e envolvidos

foram detectados.

3.2.1 Documento de Visão

Pelo fato do YP ser baseado em uma metodologia ágil, os produtos do documento de visão gerados por meio da conversa com o cliente foram usualmente simples. As informações obtidas foram registradas e serviram como base para as atividades de inicialização e planejamento.

À medida que a formulação teve início, foram respondidas questões referentes a principal necessidade do módulo a ser desenvolvido. Quais seriam suas principais funcionalidades e de qual forma deveria atender as necessidades dos envolvidos. A coleta de requisitos proporcionou a identificação do conteúdo e das funcionalidades esperadas. E também, meios para a definição dos cenários de interação para as diferentes classes de usuários.

Os usuários identificados para o Módulo AP, seus papéis e uma pequena descrição de sua atuação são detalhados no Quadro 3.1. Considerando-se que o usuário Coordenador também é um Professor, sua participação neste Módulo faz todo o sentido.

Nome	Descrição	Responsabilidade
Coordenador	Responsável pela matéria de TCC.	Responsável por gerenciar os TCC, bancas, usuários do sistema, disponibilização de avisos e arquivos.
Professor	Responsável por orientar e auxiliar o aluno durante a elaboração do TCC.	Responsável por gerenciar seus orientandos.
Aluno	É a pessoa que será matriculada na matéria de TCC.	Responsável por se comunicar com o seu orientador e acompanhar seu TCC.
Público	É o usuário que não possui nenhum vínculo com a matéria.	Só pode acessar informações gerais sobre a matéria e baixar documentos.

Tabela 3.1: Usuários do Módulo Aluno / Professor

3.3 INICIALIZAÇÃO

O Documento de Visão proporcionou dados para entender quais conteúdos e funções deveriam ser produzidas. Com isso, foram resolvidas questões referentes em:

- Como o módulo seria desenvolvido e configurado para a utilização;
- Qual seria o esquema navegacional de suas páginas e;
- Como deveria ser feito a divisão de suas funcionalidades.

Os detalhes do processo de Inicialização estão descritos nos próximos itens.

3.3.1 Modelagem dos casos de uso

Identificado os usuários e os seus requisitos de negócio, foi possível dar início a análise das primeiras abstrações por meio da modelagem dos casos de uso utilizando a ferramenta *Enterprise Architect*.

O Diagrama de Caso de Uso procura, por meio de uma linguagem simples, possibilitar a compreensão do comportamento externo do sistema por qualquer pessoa, tentando apresentar o sistema através de uma perspectiva do usuário (BOOCH; RUMBAUGH; JACOBSON, 2005). Os casos de uso ajudaram a entender as interações dos usuários com o Módulo AP e a identificar diretrizes importantes para os testes.

Partindo para uma visão geral do sistema, é possível visualizar na Figura 3.2 o caso de uso que abrange todos os módulos do SGTCC.

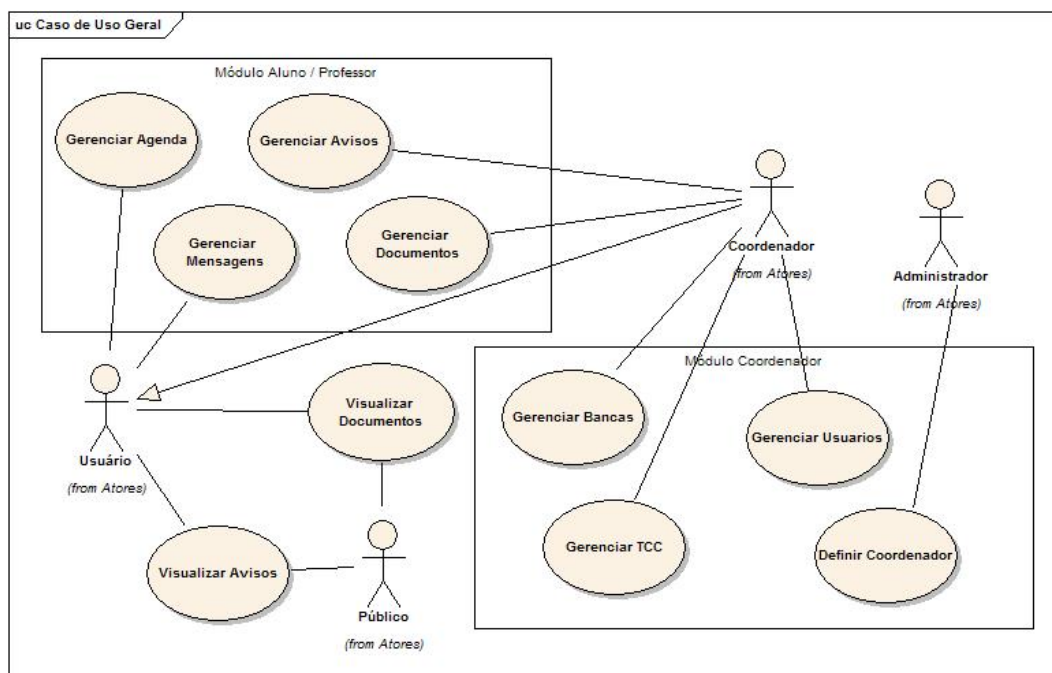


Figura 3.2: Diagrama de casos de uso geral - SGTCC.

A Figura 3.3 mostra os atores que interagem com o Módulo AP.

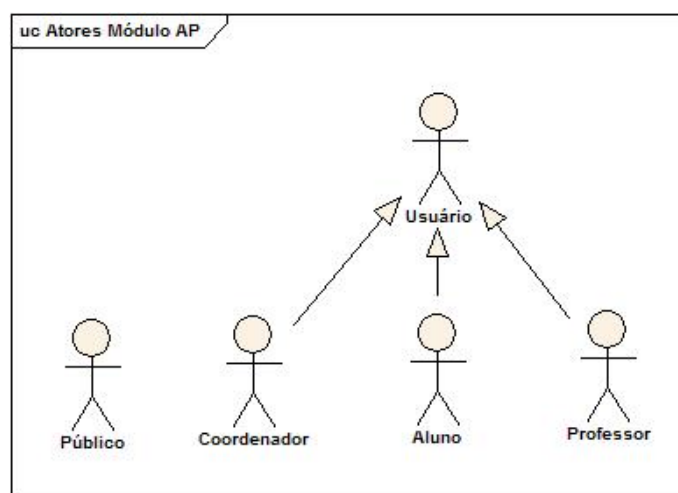


Figura 3.3: Atores.

A Figura 3.4 mostra o diagrama de caso de uso específico do Módulo AP.

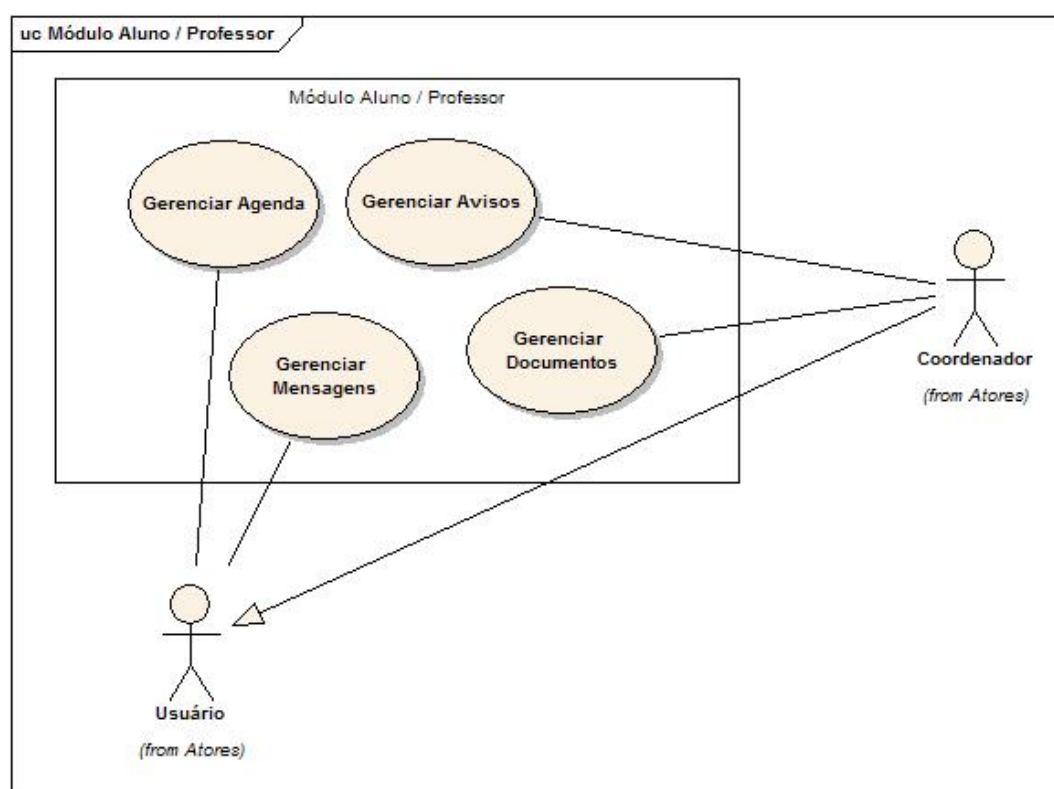


Figura 3.4: Diagrama de casos de uso para o Módulo Aluno / Professor.

A descrição textual dos casos de uso do é mostrado nos Quadros 3.2, 3.3, 3.4 e 3.5.

Caso de Uso: Gerenciar Agenda - Incluir Evento
01 - O usuário acessa a página de agenda pessoal.
02 - O usuário seleciona uma data no calendário.
03 - O usuário define o nome e descrição do evento.
04 - O usuário salva o evento.
Fluxo Alternativo A1 - Editar Evento
01 - Continua após o item 02 do processamento normal deste caso de uso.
02 - A tela é carregada com os dados do evento cadastrado.
03 - O usuário modifica os dados e salva o evento.

Tabela 3.2: Descrição do Caso de Uso: Gerenciar Agenda

Caso de Uso: Gerenciar Mensagens - Enviar Mensagem
01 - O usuário acessa a página de correio
02 - A página é carregada com os destinatários possíveis do usuário.
03 - O usuário seleciona o destinatário.
05 - O usuário envia a mensagem e recebe confirmação de envio.
Fluxo Alternativo A1 - Receber Mensagens
01 - Continua após o item 01do processamento normal deste caso de uso.
02 - O usuário seleciona o link "Mensagens Recebidas".
03 - O Grid de mensagens é atualizado com as mensagens recebidas.

Tabela 3.3: Descrição do Caso de Uso: Gerenciar Mensagens

Caso de Uso: Gerenciar Documentos - Novo documento
01 - O coordenador acessa a página de gerenciar documentos.
02 - A página é carrega com a árvore de diretório com todos os documentos do servidor e uma Grid de visualização.
03 - O coordenador envia um novo documento.
04 - O documento é aceito e enviado ao servidor.
Fluxo Alternativo A1 - Baixar documento
01 - Continua após o item 02 do processamento normal deste caso de uso.
02 - O usuário seleciona um documento.
03 - Um link é liberado e o usuário baixa o documento para sua máquina.

Tabela 3.4: Descrição do Caso de Uso: Gerenciar Documentos

Caso de Uso: Gerenciar Avisos - Incluir Aviso
01 - O coordenador acessa a página de gerenciar de avisos.
02 - O coordenador seleciona um novo aviso.
03 - O coordenador informa os dados do aviso.
04 - O coordenador salva o novo aviso no sistema.
Fluxo Alternativo A1 - Editar Aviso
01 - Continua após o item 02 do processamento normal deste caso de uso.
02 - O coordenador informa a correção do aviso.
03 - O coordenador salva as novas informações do aviso.

Tabela 3.5: Descrição do Caso de Uso: Gerenciar Avisos

3.3.2 User Stories

Como dito anteriormente, se tratando de um projeto desenvolvido em dois módulos, foi necessário que as funcionalidades do sistema, denominadas *User Stories*, fossem divididas de modo a ficar bem definido para ambos. Sendo, seus usuários, suas ações e esforço de trabalho, critérios predominantes para sua divisão. Com base na modelagem dos casos de uso, e por meio de uma reunião entre os integrantes do projeto, foi concordado que os seguintes US fossem implementados para o presente trabalho conforme Quadro 3.6.

	Descrição
US1	Gerenciar Agenda
US2	Gerenciar Avisos
US3	Gerenciar Mensagens
US4	Gerenciar Documentos
US5	Gerenciar Permissão de Acesso

Tabela 3.6: Definição das *User Stories* para o Módulo Aluno / Professor

3.3.3 Projeto Arquitetural

Com base nos limites e restrições levantados pelo documento de visão foi idealizado o modelo de arquitetura física e lógica da aplicação.

Para arquitetura física ficou definido que o usuário acessará o sistema por meio de um navegador de sua preferência, precisando de uma conexão com a *Internet*. O servidor responsável por abrigar o sistema e por receber as solicitações do usuário estará na UTFPR no qual ficará responsável por disponibilizar o acesso via *Internet*. A camada de base de dados irá armazenar todas as

informações persistentes do sistema.

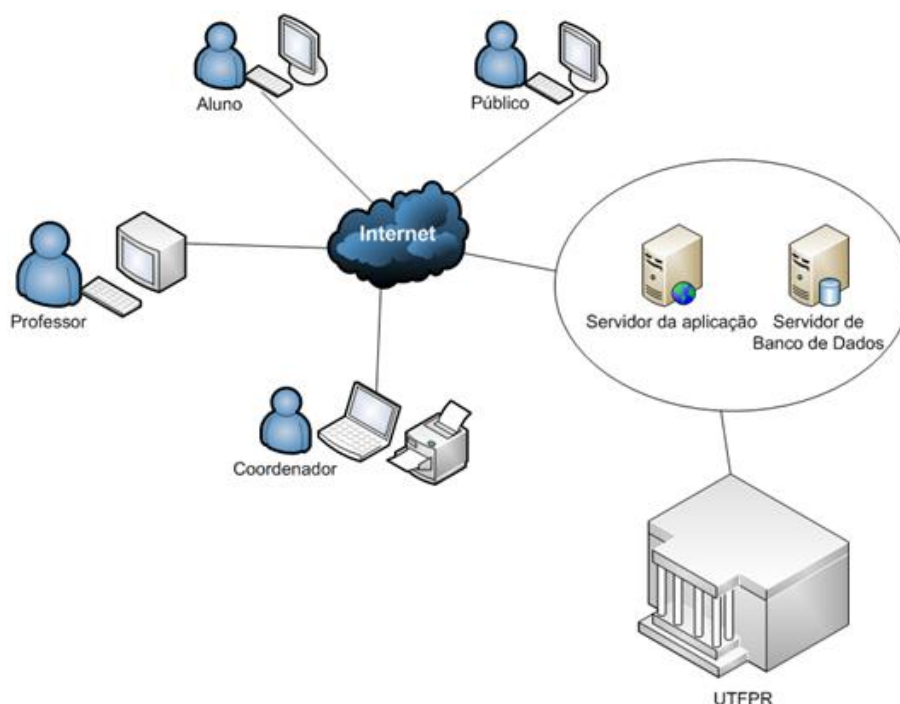


Figura 3.5: Arquitetura física - SGTCC.

Pensando na construção de uma aplicação totalmente orientada a objetos, a modelagem da arquitetura lógica ajudou a criar a divisão dos módulos e camadas que compõe a aplicação, procurando tornar o sistema organizado, escalável e de fácil manutenção. Os módulos foram resolvidos conforme a divisão do projeto especificado nas US. As camadas foram baseadas propondo um modelo três camadas definida em: apresentação (*WebInterface*), negócio (*Negocio*) e dados (*EnterpriseLibrary.Data*), representado pelo diagrama na Figura 3.6.

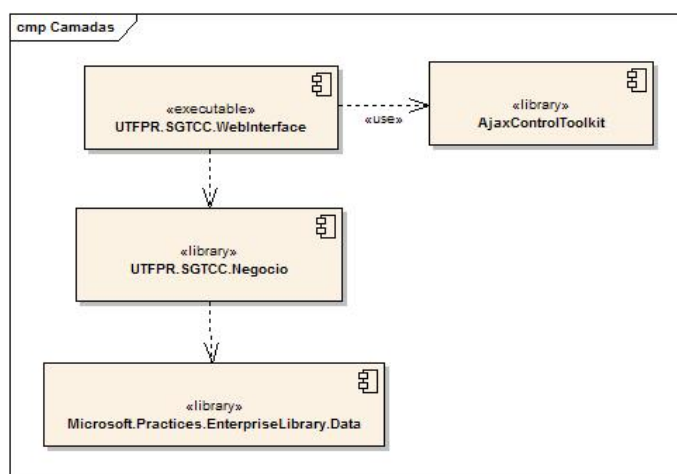


Figura 3.6: Arquitetura lógica (Camadas da aplicação) - SGTCC.

A Figura 3.7 representa, por meio de um diagrama, a divisão dos módulos na camada de

Negocio.

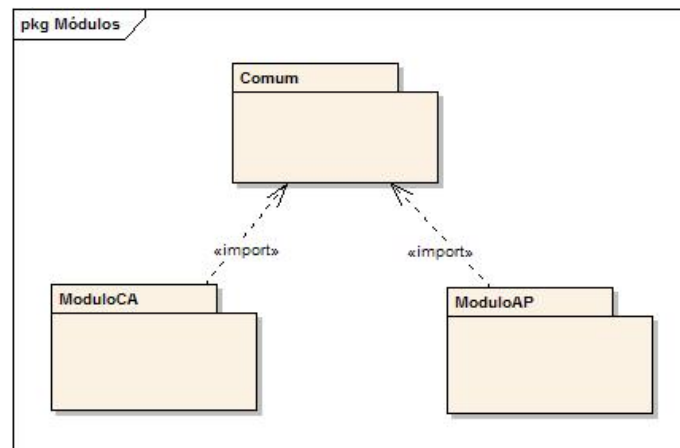


Figura 3.7: Módulos - SGTCC.

3.3.4 Modelo Lógico de Dados

O modelo de dados foi criado para entender o relacionamento das tabelas do banco de dados, ajudando no entendimento dos objetos de conteúdo. Este modelo também ajudou a identificar os objetos que seriam utilizados para a persistência, auxiliando assim na criação do diagrama de classes na etapa de implementação.

Um diagrama com a representação do modelo de dados utilizado no Módulo AP é mostrado na Figura 3.8.

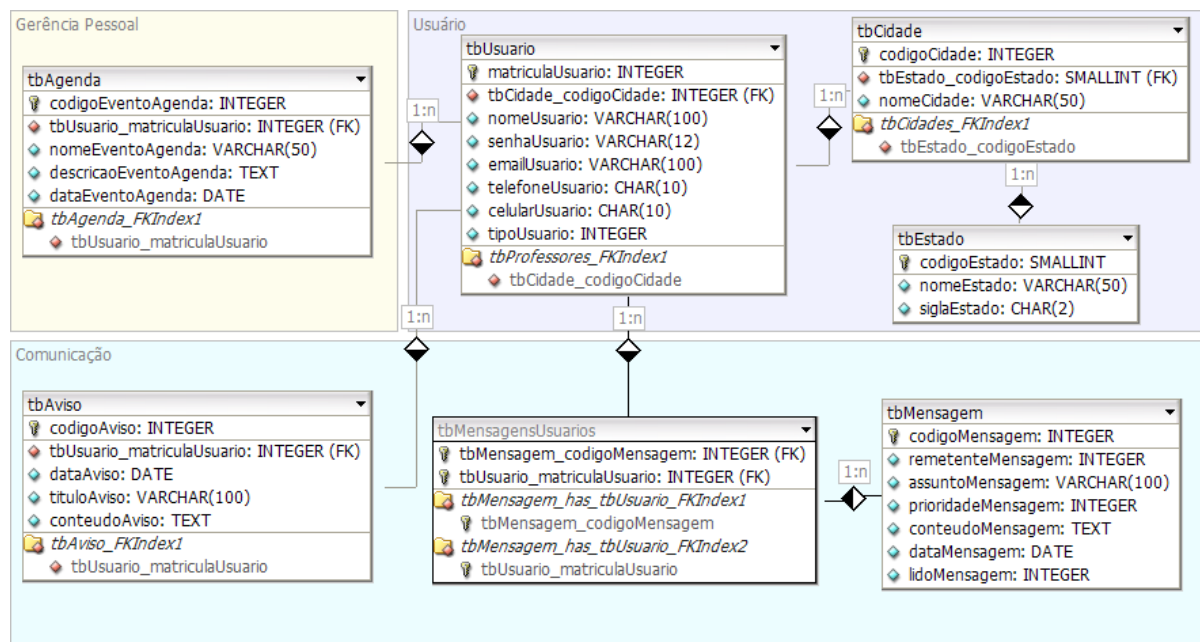


Figura 3.8: Modelo lógico de dados - Módulo Aluno / Professor.

Na Figura 3.9 é possível visualizar o modelo de dados final, com as tabelas utilizadas pelo SGTCC.

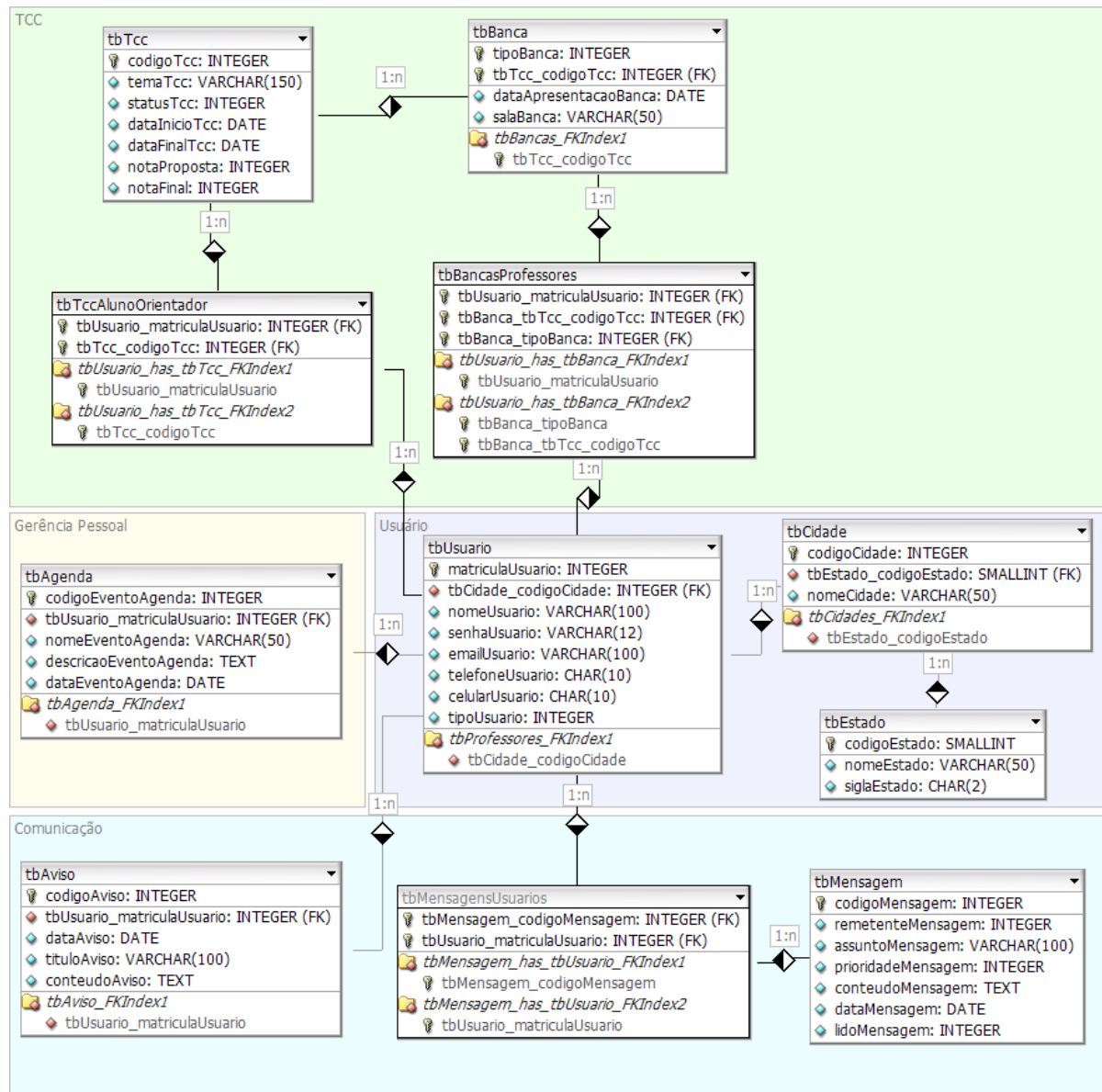


Figura 3.9: Modelo lógico de dados - SGTCC.

3.3.5 Prototipação

O cliente, freqüentemente, define um conjunto de objetivos gerais para o software, mas não identifica detalhadamente requisitos de entrada, processamento ou saída. Em outros casos, o desenvolvedor pode estar seguro da eficiência de um algoritmo, da adaptabilidade de um sistema ou da forma que a interação homem/máquina deve assumir. Nessa, e em muitas outras situações, um paradigma de prototipagem pode oferecer a melhor abordagem (PRESSMAN, 2006).

A prototipação não é uma atividade sugerida no YP, porém, já que adota-se uma metodologia

ágil, este é bastante flexível e adapta-se conforme a necessidade de cada projeto. O uso da prototipagem permitiu uma melhor avaliação do usuário quanto à interface gráfica, ajudando a definir a estrutura lógica do seu conteúdo e seu *design* visual.

O diagrama de representação da interface com o usuário, que definiu a estrutura da página mestre do SGTCC, é mostrado na Figura 3.10.

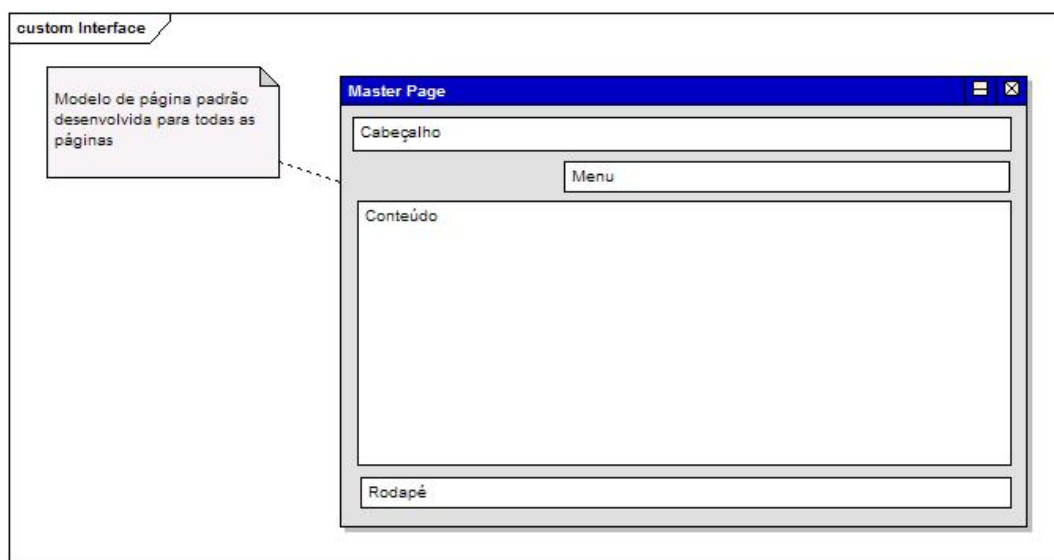


Figura 3.10: Modelo estrutural da interface.

Nas Figuras 3.11 e 3.12 é mostrado os primeiros passos para a definição do *layout* do sistema desenvolvido na ferramenta *Macromedia Fireworks*.



Figura 3.11: Primeiro protótipo de tela.

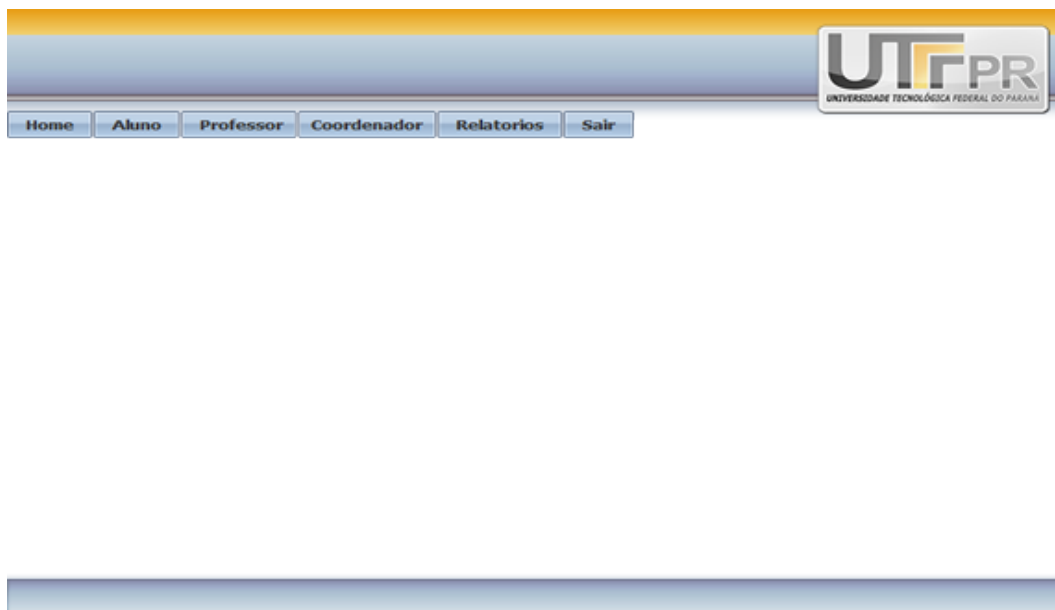


Figura 3.12: Segundo protótipo de tela.

3.4 PLANEJAMENTO

A comunicação com o cliente e a coleta de requisitos foram antecedentes essenciais para o planejamento do projeto. Depois do entendimento do escopo, foi feita uma definição da estratégia de projeto incremental. Dividiram-se então as atividades de cada etapa e seu tempo de desenvolvimento com base no prazo estipulado. Adaptando-se o processo, a divisão foi feita de modo a serem geradas três iterações com apenas um *release*.

Utilizando-se a ferramenta *Microsoft Project* 2003 foi possível criar um cronograma detalhando as etapas e suas iterações.

Na Figura 3.13 é possível visualizar as atividades do projeto e o período de duração que foi planejado para cada etapa. O cronograma em questão foi usado como base e parâmetro para a realização dos dois módulos, visto que o Módulo AP possuía algumas dependências de recursos que o Módulo AC deveria implementar, e vice-versa. Com isso, garantia-se que até certa etapa, os recursos dependentes de cada módulo estariam implementados para utilização.









Id		Nome da tarefa	Duração	Início	Término	Predecessoras
1		Definição de papéis	6 dias?	Sex 1/2/08	Sex 8/2/08	
2		Definição dos Stakeholders	6 dias?	Sex 1/2/08	Sex 8/2/08	
3		Conversa com Cliente	10 dias?	Seg 11/2/08	Sex 22/2/08	1
4		Documento de visão	10 dias?	Seg 11/2/08	Sex 22/2/08	
5		Perfil do Usuário	10 dias?	Seg 11/2/08	Sex 22/2/08	
6		Inicialização	25 dias?	Seg 25/2/08	Sex 28/3/08	3
7		Projeto Arquitetural	25 dias?	Seg 25/2/08	Sex 28/3/08	
8		Modelo Lógico de Dados	25 dias?	Seg 25/2/08	Sex 28/3/08	
9		User Stories	25 dias?	Seg 25/2/08	Sex 28/3/08	
10		Protótipo de Interface	25 dias?	Seg 25/2/08	Sex 28/3/08	
11		Estudo de Usabilidade	25 dias?	Seg 25/2/08	Sex 28/3/08	
12		Definir Iterações e Releases	25 dias?	Seg 25/2/08	Sex 28/3/08	
13		1ª Iteração	45 dias?	Seg 31/3/08	Sex 30/5/08	6
14		Planejamento	5 dias?	Seg 31/3/08	Sex 4/4/08	
15		Distribuir atividades na TAT	5 dias?	Seg 31/3/08	Sex 4/4/08	
16		Definição de testes de aceita	5 dias?	Seg 31/3/08	Sex 4/4/08	
17		Implementação	40 dias?	Seg 7/4/08	Sex 30/5/08	
18		Modelagem	40 dias?	Seg 7/4/08	Sex 30/5/08	
19		Codificação	40 dias?	Seg 7/4/08	Sex 30/5/08	
20		Testes contínuos	40 dias?	Seg 7/4/08	Sex 30/5/08	
21		Revisão de Código	40 dias?	Seg 7/4/08	Sex 30/5/08	
22		Controle de Qualidade	40 dias?	Seg 7/4/08	Sex 30/5/08	
23		Reuniões de Acompanhamer	40 dias?	Seg 7/4/08	Sex 30/5/08	
24		2ª Iteração	44 dias?	Seg 2/6/08	Qui 31/7/08	13
25		Planejamento	5 dias?	Seg 2/6/08	Sex 6/6/08	
26		Distribuir atividades na TAT	5 dias?	Seg 2/6/08	Sex 6/6/08	
27		Definição de testes de aceita	5 dias?	Seg 2/6/08	Sex 6/6/08	
28		Implementação	39 dias?	Seg 9/6/08	Qui 31/7/08	
29		Modelagem	39 dias?	Seg 9/6/08	Qui 31/7/08	
30		Codificação	39 dias?	Seg 9/6/08	Qui 31/7/08	
31		Testes contínuos	39 dias?	Seg 9/6/08	Qui 31/7/08	
32		Revisão de Código	39 dias?	Seg 9/6/08	Qui 31/7/08	
33		Controle de Qualidade	39 dias?	Seg 9/6/08	Qui 31/7/08	
34		Reuniões de Acompanhamer	39 dias?	Seg 9/6/08	Qui 31/7/08	
35		3ª Iteração	42 dias?	Seg 4/8/08	Ter 30/9/08	24
36		Planejamento	5 dias?	Seg 4/8/08	Sex 8/8/08	
37		Distribuir atividades na TAT	5 dias?	Seg 4/8/08	Sex 8/8/08	
38		Definição de testes de aceita	5 dias?	Seg 4/8/08	Sex 8/8/08	
39		Implementação	37 dias?	Seg 11/8/08	Ter 30/9/08	
40		Modelagem	37 dias?	Seg 11/8/08	Ter 30/9/08	
41		Codificação	37 dias?	Seg 11/8/08	Ter 30/9/08	
42		Testes contínuos	37 dias?	Seg 11/8/08	Ter 30/9/08	
43		Revisão de Código	37 dias?	Seg 11/8/08	Ter 30/9/08	
44		Controle de Qualidade	37 dias?	Seg 11/8/08	Ter 30/9/08	
45		Reuniões de Acompanhamer	37 dias?	Seg 11/8/08	Ter 30/9/08	
46		Finalização	23 dias?	Qua 1/10/08	Sex 31/10/08	35
47		Testes Finais	23 dias?	Qua 1/10/08	Sex 31/10/08	
48		Avaliação Final	23 dias?	Qua 1/10/08	Sex 31/10/08	
49		Implantação	23 dias?	Qua 1/10/08	Sex 31/10/08	
50		Documentação do Produto	23 dias?	Qua 1/10/08	Sex 31/10/08	

Figura 3.13: Cronograma de execução nas atividades do YP.

Na primeira iteração foram desenvolvidas as atividades estipuladas para as US Gerenciar

Agenda e Gerenciar Documentos. Para segunda iteração as US Gerenciar Mensagens e Gerenciar Avisos e finalmente na terceira iteração o controle de Permissão de Acesso dos usuários do sistema.

Após essa atividade, foi elaborado uma pequena tabela de tarefas chamada TAT. As atividades especificadas nas US foram quebradas em pequenas tarefas, e essas foram distribuídas entre as iterações. O Quadro 3.7 ilustra um exemplo desta tabela preenchida.

Release 01		
Iteração	Descrição	Status
01	Modelagem do Diagrama de Classes	Completo
01	Modelagem dos Diagramas de Sequência	Completo
01	Projetar Stored Procedures	Completo
01	Projetar modelo navegacional	Pendente
01	Implementar classes de Persistência	Pendente
01	Implementar as telas de cadastro	Pendente

Tabela 3.7: Tabela de Alocação de Tarefas (TAT)

3.5 IMPLEMENTAÇÃO

Com as atividades bem planejadas e requisitos definidos, deu-se início a implementação das tarefas alocadas na TAT.

3.5.1 Modelagem

Neste item estão descritas as técnicas de modelagem utilizando os diagramas da UML para representar a colaboração dinâmica entre os vários objetos do sistema durante o desenvolvimento das tarefas.

Antes da codificação das tarefas, foi modelado o mais importante e utilizado diagrama da UML, o Diagrama de Classes. Seu principal enfoque está em permitir a visualização das classes que compõem o sistema com seus respectivos atributos, bem como em demonstrar como as classes do diagrama se relacionam, complementam e transmitem informações sobre si (BOOCH; RUMBAUGH; JACOBSON, 2005).

Com base no modelo lógico de dados utilizado para este módulo, foi criado um diagrama de classes representando seus relacionamentos conforme Figura 3.14.

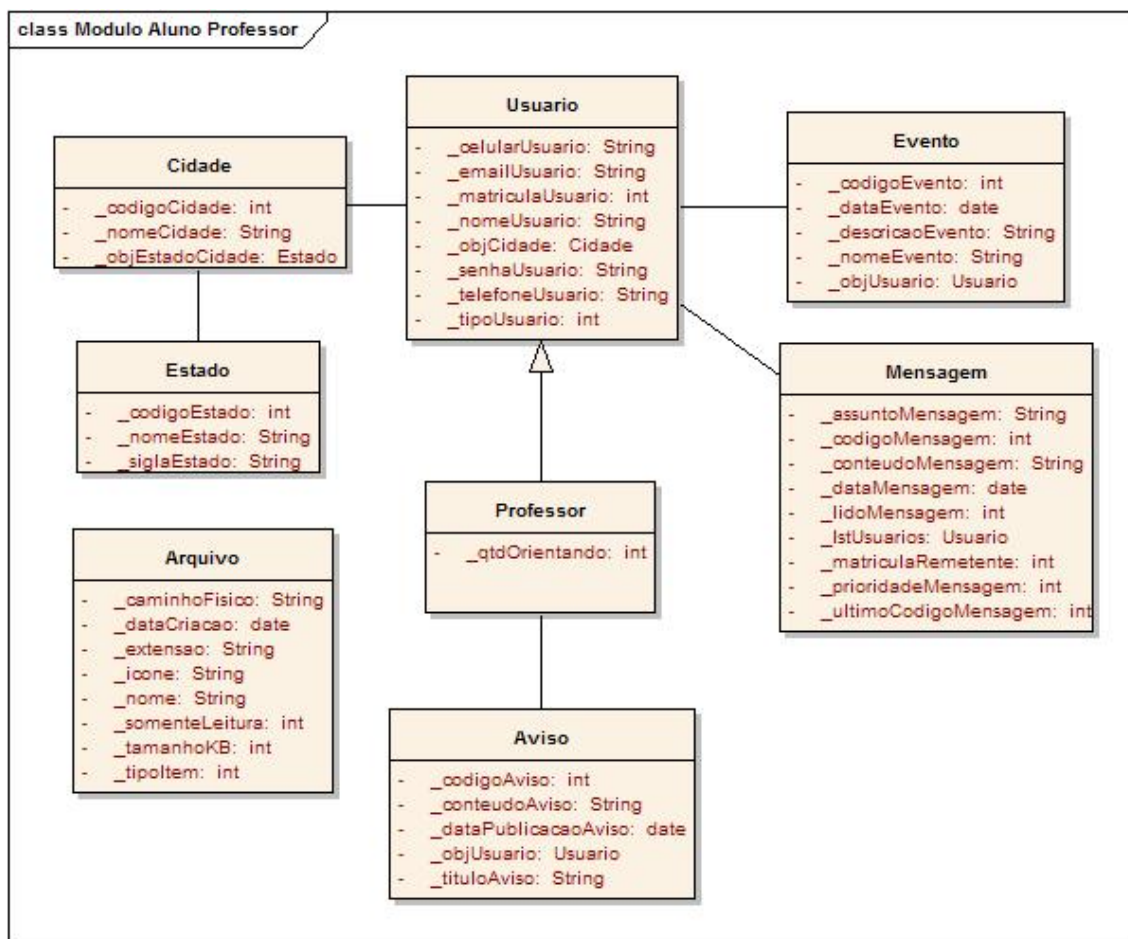


Figura 3.14: Diagrama de Classes - Módulo Aluno / Professor.

Na Figura 3.15, é possível visualizar o diagrama de classes final com a integração dos módulos.

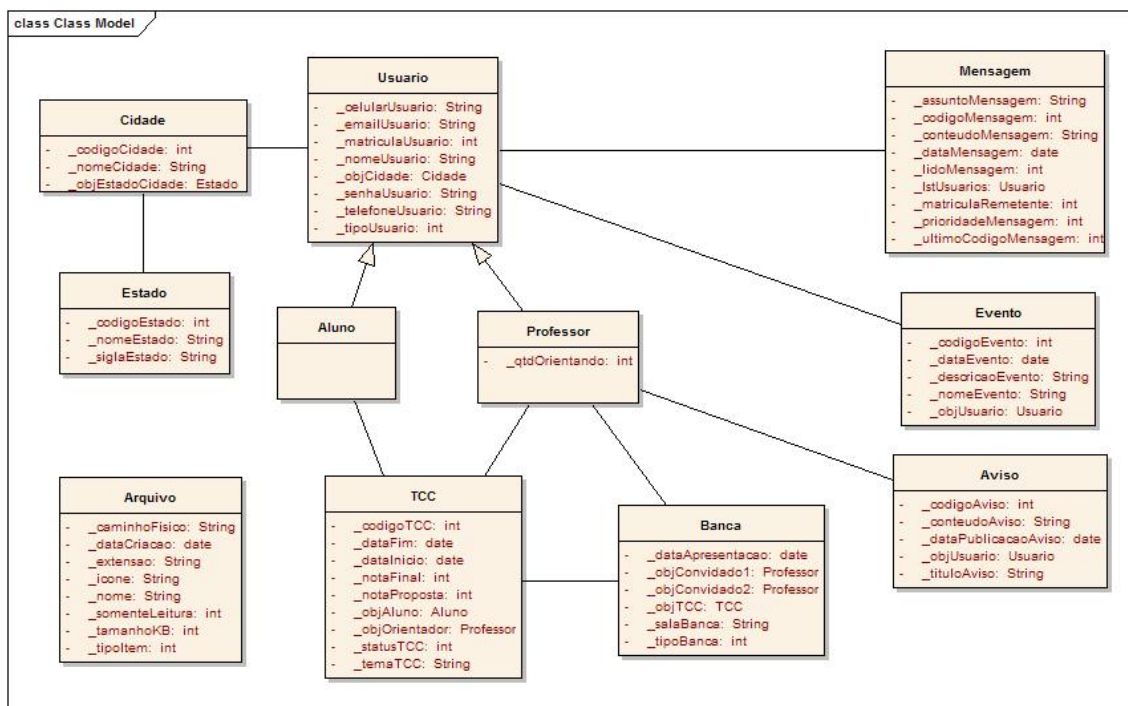


Figura 3.15: Diagrama de Classes - SGTCC.

Para cada iteração foram modelados os diagramas de sequência de cada caso de uso. Na UML, um diagrama de sequência tem o objetivo determinar a sequência de eventos que ocorrem em um determinado processo, ou seja, quais condições devem ser satisfeitas e quais métodos devem ser disparados entre os objetos envolvidos e em que ordem durante um processo específico (BOOCH; RUMBAUGH; JACOBSON, 2005). Esses diagramas ajudaram bastante a entender os passos que o sistema deveria executar para que uma funcionalidade do sistema fosse processada.

Um diagrama de sequência para o caso de uso "Gerenciar Documentos" é mostrado na Figura 3.16.

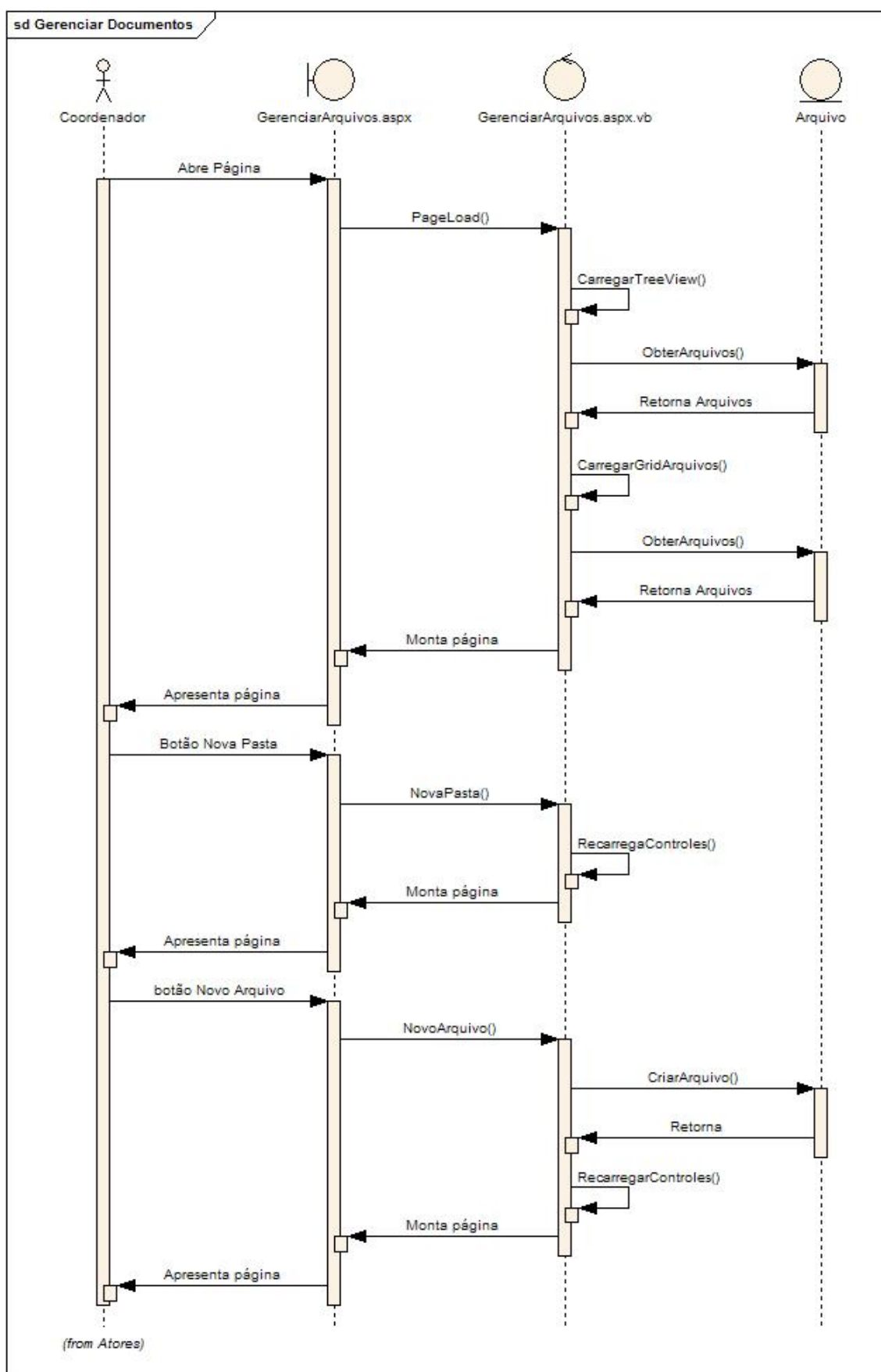


Figura 3.16: Diagrama de Seqüência.

3.5.2 Codificação

A codificação do SGTCC utilizou tecnologias e ferramentas que possibilitaram a reusabilidade dos componentes na medida em que foram construídos. Os códigos fontes na linguagem *Visual Basic .NET* foram auto documentados seguindo os padrões definidos pelos integrantes deste trabalho por meio do documento Padrões de Codificação disponível no Apêndice C.

```

1 Imports System.Data.Common
2 Imports System.Data.SqlClient
3 Imports UTFPR.SGTCC.Negocio.Comum
4 Imports System.Collections.Generic
5 Imports Microsoft.Practices.EnterpriseLibrary.Data
6
7 Namespace ModuloAP
8
9     ''' <summary>
10     ''' ***** <BR/>
11     ''' Nome.....: Usuario. <BR/>
12     ''' Objetivo.....: Classe responsável por manipular e acessar aos dados <BR/>
13     ''' dos usuários. <BR/>
14     ''' ***** <BR/>
15     ''' </summary>
16     ''' <remarks>Classe que implementa a interface INegocioPersistencia</remarks>
17     Public Class Usuario
18         Implements INegocioPersistencia
19
20     Atributos
21
22     23
24     Propriedades
25
26     131
27     132 Construtor
28
29     179
30     Métodos
31
32     613
33     614 End Class
34
35     615
36     616 End Namespace

```

Figura 3.17: Estrutura de uma classe

A Figura 3.17 mostra a estrutura de organização de um classe definida para este projeto.

```

182      ''' <summary>
183      ''' Método para cadastrar um novo usuário no sistema.
184      ''' </summary>
185      ''' <param name="matricula">Número da matrícula</param>
186      ''' <param name="nome">Nome do usuário</param>
187      ''' <param name="email">Endereço de Email</param>
188      ''' <param name="telefone">Número do Telefone</param>
189      ''' <param name="celular">Número do Celular</param>
190      ''' <param name="tipo">Código identificador do tipo do usuário</param>
191      ''' <param name="codigoCidade">Código identificador da cidade</param>
192      ''' <remarks></remarks>
193      Public Sub CadastrarUsuario(ByVal matricula As Integer, _
194                                ByVal nome As String, _
195                                ByVal email As String, _
196                                ByVal telefone As String, _
197                                ByVal celular As String, _
198                                ByVal tipo As TipoUsuario, _
199                                ByVal codigoCidade As Integer)
200
201          ' Carrega as propriedades do objeto
202          Me.Matricula = matricula
203          Me.Nome = nome
204          Me.Senha = "123"
205          Me.Email = email
206          Me.Telefone = telefone
207          Me.Celular = celular
208          Me.Tipo = tipo
209          Me.Cidade = New Cidade(codigoCidade)

```

Figura 3.18: Comentário de um método

A Figura 3.18 mostra um método de uma classe com a documentação de toda sua estrutura. A documentação em forma de XML permite, ao final do projeto, que seja gerado toda documentação técnica da aplicação em formato HTML.

Os componentes da biblioteca *AjaxControlToolkit* do ASP .NET AJAX permitiram a construção de páginas mais dinâmicas, melhorando a experiência do usuário em sua utilização. Um exemplo de componente utilizado é o *UpdatePanel*. Com ele foi possível fazer com que parte específica de uma página fosse processada e atualizada de forma transparente ao usuário, evitando aquelas "piscadas" inconvenientes do browser quando é feito uma requisição ao servidor da aplicação.

O Bloco de Acesso a Dados, disponibilizado pelo *Microsoft Application Blocks*, agilizou o processo de codificação de acesso a dados com soluções padronizadas e centralizadas, obtendo assim o máximo desempenho de comunicação com o SGBD utilizado. A Figura 3.19 apresenta um código simples da utilização desse componente.

```

''' <summary>
''' Método que persiste os dados do usuário na base de dados
''' </summary>
''' <remarks>
''' Método implementado da interface INegocioPersistencia
''' </remarks>|
Private Sub Salvar() Implements INegocioPersistencia.Salvar

    Try

        ' Define o objeto DataBase
        Dim objDtBase As Database = DatabaseFactory.CreateDatabase()

        ' Define um objeto DbCommand para executar a stored procedure
        Dim objCommand As DbCommand = objDtBase.GetStoredProcCommand("proc_UsuarioInserir")

        ' Define os parametros usados na stored procedure
        objDtBase.AddInParameter(objCommand, "@matriculaUsuario", DbType.Int32, Me.Matricula)
        objDtBase.AddInParameter(objCommand, "@nomeUsuario", DbType.String, Me.Nome)
        objDtBase.AddInParameter(objCommand, "@senhaUsuario", DbType.String, Security.Cifrar(Me.Senha))
        objDtBase.AddInParameter(objCommand, "@emailUsuario", DbType.String, Me.Email)
        objDtBase.AddInParameter(objCommand, "@telefoneUsuario", DbType.String, Me.Telefone)
        objDtBase.AddInParameter(objCommand, "@celularUsuario", DbType.String, Me.Celular)
        objDtBase.AddInParameter(objCommand, "@tipoUsuario", DbType.Int32, Me.Tipo)
        objDtBase.AddInParameter(objCommand, "@codigoCidade", DbType.Int32, Me.Cidade.Codigo)

        ' Executa a query
        objDtBase.ExecuteNonQuery(objCommand)

    Catch ex As Exception
        Log.GravarLog("Usuario", "Salvar", ex.Message, TipoErro.Critico)
        Throw New Exception
    End Try

End Sub

```

Figura 3.19: Método de persistencia dos dados utilizando DAAB

O *User Control* é um recurso disponível no .NET *Framework* que permitiu a criação de controles personalizados e reutilizáveis na construção das páginas e suas funcionalidades. A Figura 3.20 mostra um controle personalizado criado com este recurso para listagem de registros.

Filtro de Consulta:

Tema:

Orientador:

Aluno: [IbErroDuasMatriculas]

Status: ☐ Proposta ☐ Matriculado ☐ Banca ☐ Concluido ☐ Desistente

Buscar

Tema	Aluno	Orientador	Status	Data Inicio	Data Entrega
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound

Página de

[IbMensagem]

Figura 3.20: Controle Grid desenvolvido como um *User Control*

O *Forms Authentication* é uma forma de autenticação de usuários disponível em aplicativos ASP .NET. Utilizando-se desse recurso foi possível definir o processo de autenticação e autor-

ização de usuários de forma simples e rápida configurando suas definições em um arquivo XML chamado *Web.config*. A Figura 3.21 ilustra o arquivo configurado com as áreas de acesso para cada usuário do sistema.

```
<!-- Controle de acesso de todos usuários Cadastrados -->
<location path="site/comum">
  <system.web>
    <authorization>
      <deny users="?" />
      <allow users="Aluno" />
      <allow users="Professor" />
      <allow users="Coordenador" />
      <allow users="Administrador" />
    </authorization>
  </system.web>
</location>

<!-- Controle de acesso do usuário Professor -->
<location path="site/professor">
  <system.web>
    <authorization>
      <deny users="?" />
      <deny users="Aluno" />
      <allow users="Professor" />
      <allow users="Coordenador" />
      <allow users="Administrador" />
    </authorization>
  </system.web>
</location>

<!-- Controle de acesso do usuário Coordenador -->
<location path="site/coordenador">
  <system.web>
    <authorization>
      <deny users="?" />
      <deny users="Aluno" />
      <deny users="Professor" />
      <allow users="Coordenador" />
      <allow users="Administrador" />
    </authorization>
  </system.web>
</location>
```

Figura 3.21: Autenticação configurada no *Web.config*

Algumas das telas e funcionalidades importantes do SGTCC são mostradas nas Figuras a seguir.

A Figura 3.22 mostra a página principal do SGTCC. A página Home exibe as principais informações sobre a matéria de TCC de uma forma simples, clara e objetiva.



Figura 3.22: Tela Principal

A Figura 3.23 mostra a página na qual o usuário fará o acesso ao conteúdo reestruturado a alunos e professores matriculados na matéria. Neste caso, para acessar o sistema, o usuário deve ter seu número de matrícula cadastrado no SGTCC por meio do coordenador da matéria.

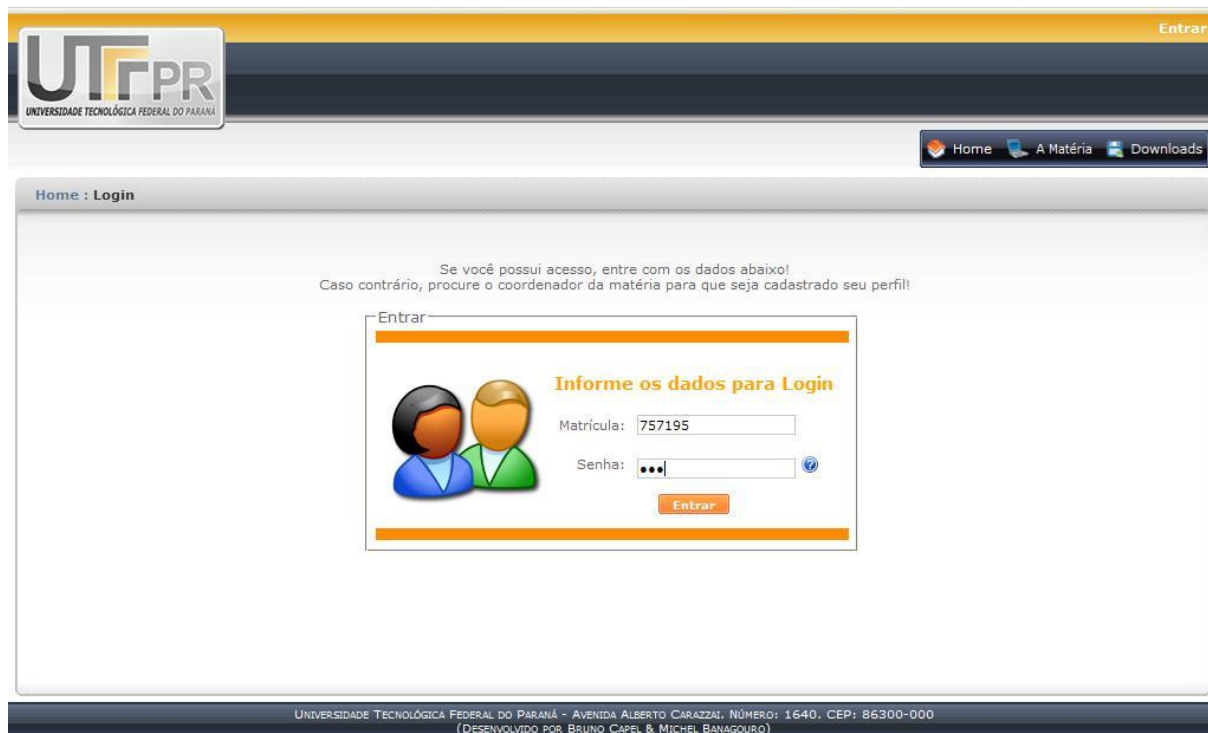


Figura 3.23: Tela de Login

A Figura 3.24 mostra a área onde o coordenador da matéria gerencia todos os arquivos que ficarão disponíveis na área de downloads do SGTCC. Nessa tela o coordenador pode definir a estrutura lógica dos seus arquivos conforme o seu critério, com a possibilidade de criar pastas e subpastas.

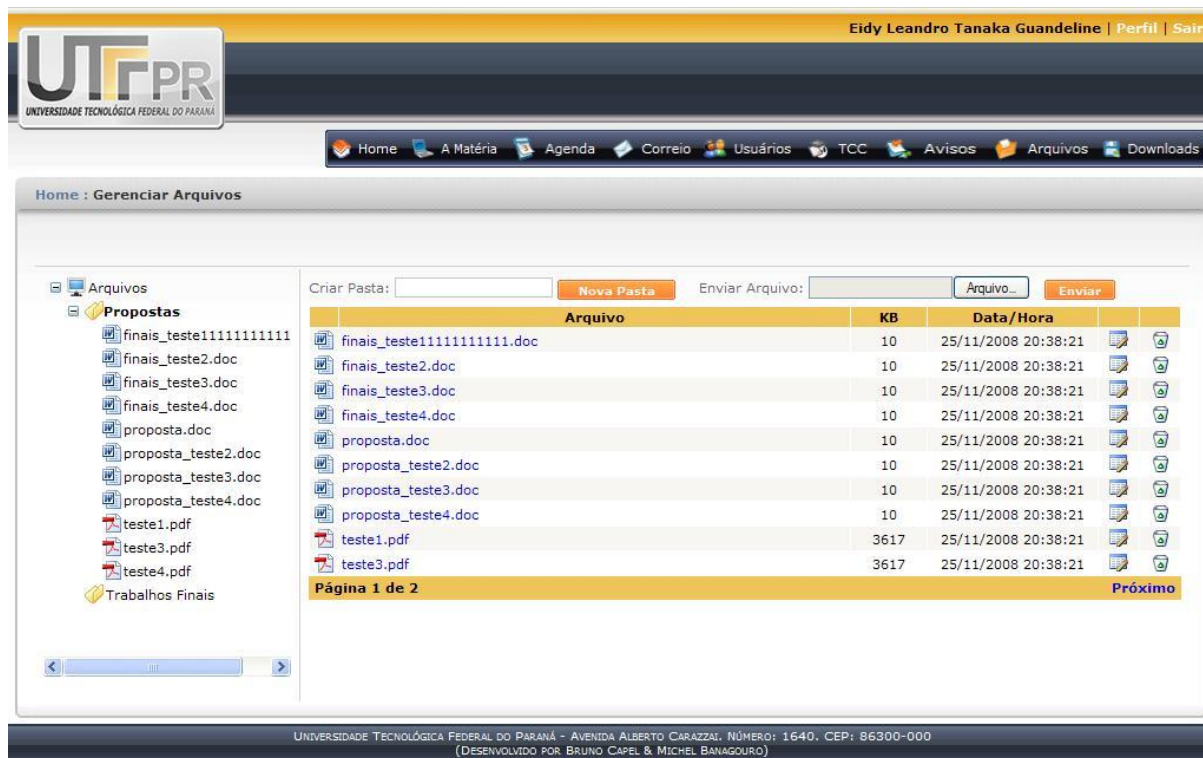


Figura 3.24: Tela de Gerenciamento de Arquivos

A Figura 3.25 mostra a área de agenda do usuário autenticado na aplicação. Uma agenda bem simples onde o usuário pode marcar seus eventos e tarefas referentes ou não ao TCC.

UT FPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

Michel Cesar Leme Banagouro | Perfil | Sair

Home A Matéria Agenda Correio Downloads

Home : Agenda

Próximos Eventos

quarta-feira, 7 de janeiro de 2009
Reunião c/ Orientador

Calendário de Eventos

novembro						janeiro	
domingo	segunda-feira	terça-feira	quarta-feira	quinta-feira	sexta-feira	sábado	
30	1	2	3	4	5	6	
7	8	9	10	11	12	13	
14	15	16	17	18	19	20	
21	22	23	24	25	26	27	
28	29	30	31	1	2	3	
4	5	6	7	8	9	10	

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ - AVENIDA ALBERTO CARAZZINI, NÚMERO: 1640. CEP: 86300-000
(DESENVOLVIDO POR BRUNO CAPEL & MICHEL BANAGOURO)

Figura 3.25: Tela de Agenda Pessoal

A Figura 3.26 mostra a parte de envio de mensagens, onde é possível visualizar as mensagens enviadas e recebidas do orientador ou aluno, assim como a possibilidade de enviar mensagens ao orientador do TCC por parte do aluno, e envio de mensagens aos alunos por parte do professor orientador.

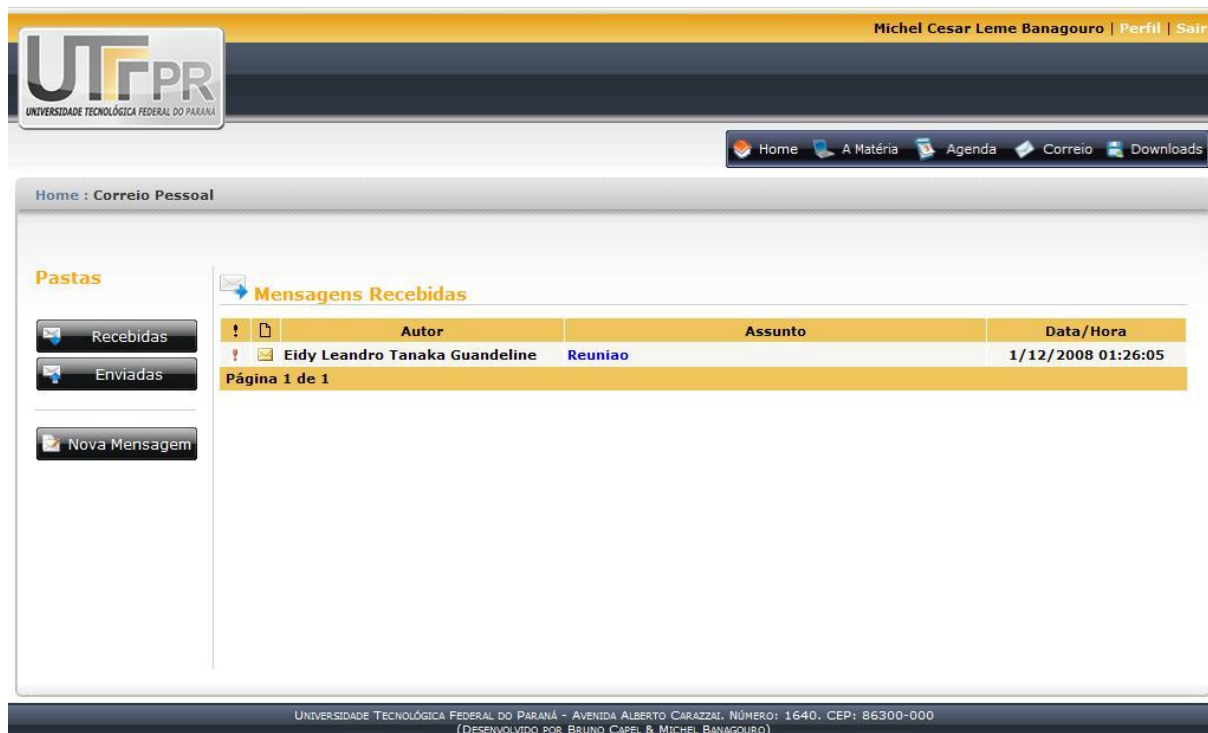


Figura 3.26: Tela de Correio

3.5.3 Testes

Nos testes, foram exercitadas várias dimensões de qualidade do Módulo AP, com a intenção de encontrar erros ou descobrir tópicos que poderiam levar a falhas de qualidade. Os testes focaram conteúdo, função, estrutura, usabilidade, navegabilidade, desempenho e segurança. A estratégia adotada para exercitar cada uma destas dimensões de qualidade foi inicialmente um exame das unidades de conteúdo, funcionalidade ou navegação. Para isso, os testes foram guiados pela perspectiva do usuário e a informação dos casos de uso.

Ao final de cada iteração do processo de desenvolvimento, foram aplicados testes em cada página para que suas funcionalidades estivessem conforme os requisitos. Durante os testes foi possível identificar falhas de design, erros de lógica e pontos de tratamento de *bugs*.

Uma vez validadas as unidades e módulos individuais, o foco foi para os testes que exercitaram o SGTCC como um todo, onde ao final foi feito a integração do módulo desenvolvido pelo aluno Bruno Gustavo Carvalho Capel com o presente módulo. Os testes integrados garantiram que inconsistências fossem eliminadas e tratadas, e a revisão do padrão de codificação do SGTCC fosse aplicada para que o sistema estivesse com a máxima qualidade.

4 CONCLUSÃO

4.1 CONSIDERAÇÕES FINAIS

Ao longo do desenvolvimento do Módulo Aluno / Professor, e conseqüentemente do SGTCC, os conhecimentos e experiências adquiridos foram de essencial ajuda para crescimento profissional e pessoal. Toda a dedicação e empenho aplicados para o aprendizado das tecnologias Microsoft proporcionaram o surgimento de várias oportunidades acadêmicas e profissionais.

A nomeação como Microsoft Student Partner (MSP) por parte da própria Microsoft foi uma das conquistas adquiridas durante o desenvolvimento do trabalho. O programa MSP tem como objetivo proporcionar aos estudantes participantes um grande destaque dentro do meio acadêmico, nas empresas de TI e na própria Microsoft, além de outros benefícios. Como MSP foi possível a criação de uma célula acadêmica Microsoft no SENAI na cidade de Londrina para estudo de tecnologias no desenvolvimento da plataforma .NET. Além da célula, também foi possível a ministração de palestras em faculdades da região e aulas voluntárias de .NET para alunos do SENAI.

A mudança para atuar na plataforma Microsoft na empresa BSI Tecnologia também foi uma conquista como recompensa de toda dedicação aos estudos e pesquisas das tecnologias Microsoft.

Utilizando-se de todas as formas de estudo como blogs, livros, fóruns e amigos de trabalho, foi possível a criação de um sistema atendendo aos requisitos do cliente, tendo alguns ajustes desde a proposta inicial, tais como acesso a controle de estatística pelo professor, neste caso o cliente preferiu que as estatísticas das informações só fossem disponíveis para o coordenador e geradas em gráficos pelos relatórios, função essa ficando a cargo do Módulo Administrado / Coordenador.

Durante a pesquisa e desenvolvimento foi possível utilizar ferramentas e componentes que antes eram desconhecidas, ajudando a incrementar a qualidade visual e funcional do sistema. Dentre elas pode-se citar o *software "Blumentals Easy Button and Menu Maker Pro"*, no qual foi possível criar menus com muita facilidade e com ótimo *design*.

Ao final deste trabalho, é entregue a implementação do Módulo Aluno / Professor, sendo este, parte importante de um sistema único chamado SGTCC - Sistema para Gerência de Trabalhos de Conclusão de Curso o qual tem como objetivo permitir o controle geral dos TCC's em andamento e finalizados na UTFPR pelo coordenador, assim como a possibilidade da participação tanto do aluno como do orientador para acompanhamento da matéria. Além de proporcionar fácil acesso as informações referentes ao TCC e trabalhos já desenvolvidos pela comunidade acadêmica.

4.2 TRABALHOS FUTUROS

O grande desafio agora é o de acrescentar mais funcionalidades ao sistema, para que a ferramenta torne-se mais poderosa em relação a implementada neste trabalho de conclusão de curso. São inúmeras as possibilidades de melhoras no SGTCC e seus módulos. O processo de desenvolvimento e as tecnologias utilizadas permitem que as mudanças continuem e fornece suporte para isso.

Ao término do trabalho, foi se pensado em novas funcionalidades que aumentassem os recursos do Módulo Aluno / Professor, tais como: envio de email de aviso quando é realizado publicações de novos avisos, alertas da agenda e cadastro de novos usuários. Outro ponto de melhoria está relacionada a usabilidade na área de agenda. Essas são algumas idéias quem visam tornar o Módulo AP cada vez mais completo e de qualidade.

REFERÊNCIAS

- BARWELL, F.; CASE, R.; FORGEY, B. *Professional Visual Basic .NET: de programador para programador*. 1. ed. [S.l.]: São Paulo: Pearson Education do Brasil, 2004.
- BISHOP, J. *C# 3.0 Design Patterns*. 1. ed. [S.l.]: California: O'Reilly, 2008.
- BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. *UML: guia do usuário*. 2. ed. [S.l.]: Rio de Janeiro: Campus, 2005.
- FLANAGAN, D. *JavaScript: o guia definitivo*. 4. ed. [S.l.]: Porto Alegre: Bookman, 2004.
- GARCIA, F. P. et al. *easYProcess: Um Processo de desenvolvimento para Uso no Ambiente Acadêmico*. 1. ed. [S.l.]: Programa de Educação Tutorial, Universidade Federal de Campina Grande, 2004.
- GIBBS, M.; WAHLIN, D. *Professional ASP .NET AJAX*. 1. ed. [S.l.]: Rio de Janeiro: Editora Alta Books, 2007.
- MACDONALD, M. *O Livro de VB.NET: O essencial de .NET para desenvolvedores VB*. 1. ed. [S.l.]: Rio de Janeiro: Editora Ciência Moderna, 2002.
- MANFREDI, P. *Elementos Estruturais para o Projeto de Aplicações Multicamadas (Application Blocks)*. 1. ed. [S.l.]: Microsoft Corp., 2003.
- MICROSOFT. *A ferramenta profissional de design da WEB*. 2008. Disponível em: <<http://www.microsoft.com/brasil/msdn/expression/products/Overview.aspx?key=web#page-top>>. Acesso em: 25/05/2008.
- MICROSOFT. *SQL Server*. 2008. Disponível em: <<http://www.microsoft.com/brasil/msdn/sql/default.msp>>. Acesso em: 10/05/2008.
- MICROSOFT. *Visão Geral do ASP .NET*. 2008. Disponível em: <<http://msdn.microsoft.com/pt-br/library/4w3ex9c2.aspx>>. Acesso em: 15/05/2008.
- MICROSOFT. *Visão geral do Visual Studio 2008*. 2008. Disponível em: <<http://msdn.microsoft.com/pt-br/vstudio/products/bb931331.aspx>>. Acesso em: 20/05/2008.
- MICROSOFT. *Visual Studio*. 2008. Disponível em: <<http://msdn.microsoft.com/pt-br/library/52f3sw5c.aspx>>. Acesso em: 18/05/2008.
- PRESSMAN, R. S. *Engenharia de software*. 6. ed. [S.l.]: São Paulo: McGraw-Hill Brasil, 2006.
- SYSTEM, S. *Enterprise Architect*. 2008. Disponível em: <<http://www.sparxsystems.com.au/>>. Acesso em: 15/05/2008.
- UTFPR. *UTEX*. 2008. Disponível em: <<http://inf.cp.utfpr.edu.br/tex/>>. Acesso em: 10/05/2008.
- W3C. *Cascading Style Sheets*. 2008. Disponível em: <<http://www.w3.org/Style/CSS/>>. Acesso em: 18/04/2008.

APÊNDICE A – PROPOSTA DE TRABALHO DE CONCLUSÃO DE CURSO

Esse apêndice traz a proposta apresentada para este trabalho de conclusão de curso.

APÊNDICE B – GLOSSÁRIO

B.1 DEFINIÇÕES

Framework	Coleção abstrata de classes, interfaces e padrões, dedicados a resolver uma família de problemas semelhantes, por meio de uma arquitetura flexível e extensível.
WEB	É a interface gráfica que nos permite ver o conteúdo da <i>Internet</i> por meio de algum <i>software</i> .
Ambiente de Desenvolvimento Integrado (IDE)	Um software composto por um editor, um compilador e um depurador.
API	Em linguagens antigas possui o significado de bibliotecas, ou seja, é um conjunto de interfaces que permite a construção de novos aplicativos com algum já existente.
artefato	Uma informação que é usada ou produzida por um processo de desenvolvimento de software. Um artefato pode ser um modelo, uma descrição ou um software. Sinônimo: produto.
ator	Alguém ou algo fora do sistema que interage com ele.
banco de dados	Um conjunto de dados armazenados juntos e gerenciados por um sistema de gerenciamento de banco de dados.
caso de uso	Especificação de uma sequência de ações (incluindo variantes) que um sistema (ou outra entidade) pode executar, interagindo com atores do sistema.
cenário	Sequência específica de ações que ilustra comportamentos. Um cenário pode ser usado para ilustrar uma interação ou a execução da instância de um caso de uso.
desenvolvedor	Uma pessoa responsável pelo desenvolvimento da funcionalidade necessária de acordo com os procedimentos e os padrões adotados no projeto.
diagrama	Uma representação gráfica parcial ou total de um modelo.

diagrama de caso de uso

Um diagrama que mostra o relacionamento entre atores ande casos de uso em um sistema.

diagrama de classes

Um diagrama que mostra uma coleção de elementos de modelo declarativos (estáticos), como classes, tipos, seus relacionamentos e conteúdo.

diagrama de seqüência

Um diagrama que mostra interações de objetos organizadas em uma seqüência temporal. Mostra principalmente os objetos que participam da interação e a seqüência de mensagens trocadas.

APÊNDICE C – PADRÕES DE CODIFICAÇÃO E DESENVOLVIMENTO

Esse apêndice traz o documento de padrões de codificação e desenvolvimento criado para este trabalho.