UNIVERSIDADE ESTADUAL DO PIAUÍ – UESPI
CAMPUS PROF. ALEXANDRE ALVES DE OLIVEIRA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

RENAN GOMES VIEIRA

DESENVOLVIMENTO DE UM SISTEMA WEB PARA CONTROLE DE OFERTAS DE DISCIPLINA UTILIZANDO O FRAMEWORK DJANGO

PARNAÍBA – PI
2011

RENAN GOMES VIEIRA

DESENVOLVIMENTO DE UM SISTEMA WEB PARA CONTROLE DE OFERTAS DE DISCIPLINA UTILIZANDO O FRAMEWORK DJANGO

Monografia apresentada ao curso de Bacharelado em Ciência da Computação da Universidade Estadual do Piauí – UESPI, Campus Prof. Alexandre Alves de Oliveira, como parte das exigências da disciplina de Estágio Supervisionado, requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Eyder Franco Sousa Rios.

PARNAÍBA – PI
2011

GOVERNO DO ESTADO DO PIAUÍ
UNIVERSIDADE ESTADUAL DO PIAUÍ – UESPI
CAMPUS PROF. ALEXANDRE ALVES DE OLIVEIRA
COORDENAÇÃO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO

Ata de Defesa de Trabalho de Conclusão de Curso

Aos dezenove dias do mês de fevereiro de dois mil e onze, às 08h30, no Laboratório de Informática do Campus Prof. Alexandre Alves Oliveira - UESPI realizou-se a sessão pública de defesa do Trabalho de Conclusão de Curso do aluno **Renan Gomes Vieira** intitulado **Desenvolvimento de um Sistema Web para Controle de Ofertas de Disciplina Utilizando o Framework Django,** como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação. A Banca Examinadora foi presidida pelo Prof. Francisco das Chagas Rocha e composta pelos membros efetivos os professores Marcelo da Costa Silva e Ivenilton Alexandre de Souza Moura. Aberta a sessão pública, o candidato teve a oportunidade de expor seu trabalho. Após a exposição, o aluno foi arguido oralmente pelos membros da Banca Examinadora. Em seguida, em sessão reservada e nos termos do Regulamento Geral dos Trabalhos de Conclusão de Curso da UESPI, os membros da Banca realizaram a avaliação do candidato, que obteve nota 9,0 (nove pontos) sendo, portanto, **aprovado**. Nada mais havendo a tratar, eu, Prof. Francisco das Chagas Rocha, lavrei a presente Ata que será lida e assinada por mim, pelos demais membros da Banca Examinadora e pelo candidato. Parnaíba (PI), 19 de março de 2011.

Banca Examinadora

Prof. M.Sc. Francisco das Chagas Rocha
Orientador, UESPI

Prof. Esp. Marcelo da Costa Silva
Avaliador, PMP

Prof. Esp. Ivenilton Alexandre de Souza Moura
Avaliador, SENAI

Candidato

Renan Gomes Vieira

Agradeço a Deus, aos meus pais,
familiares e amigos pelo apoio e carinho.

AGRADECIMENTOS

• Aos meus pais Valmir e Raimundinha pelo amor e dedicação concedidas, que me deram forças pra concluir mais essa etapa da vida.

• Aos meus irmãos Julyanna e Lucas pelo carinho, companheirismo e apoio.

• Ao professor orientador Eyder Rios pelo tempo concedido e conhecimento compartilhado.

• Aos professores do curso de Bacharelado em Ciências da Computação da Universidade Estadual do Piauí pelo aprendizado precioso.

• Aos companheiros de curso que fizeram sesses quatro anos de curso ser tão proveitosos quanto divertidos, em especial aos amigos Alexsandro, Washington e Ellaynne.

• À Milaynne pelo apoio, carinho e paciência.

• Aos amigos Anderson, Antônio, Carlos Emídio, Dennis e Marcelino pela torcida e amizade verdadeira.

RESUMO

A cada período letivo, os coordenadores de curso da Universidade Estadual do Piauí – UESPI são responsáveis por definir as disciplinas que serão ofertadas no semestre que se inicia. Essa rotina consiste em delimitar as disciplinas, os turnos das ofertas e os professores que irão ministrá-las. Essa tarefa depende de vários fatores, desde a área de atuação da disciplina e do professor, a carga horária disponível dos docentes, período corrente, entre outros. Atualmente, esse trabalho é feito de forma manual, tornando-o custoso e suscetível a falhas, pois as informações dos diversos professores não são centralizadas. Nesse trabalho são apresentados os passos para o desenvolvimento de um sistema orientado para Web para o controle de ofertas de disciplinas para o uso dos coordenadores da UESPI, utilizando os conceitos da metodologia de desenvolvimento em Cascata e da metodologia ágil XP e do framework Django, escrito na linguagem de alto nível Python. Neste trabalho são descritas a problemática, objetivos gerais e específicos, o embasamento teórico para o desenvolvimento do sistema, todo o estudo da análise do sistema feita e os detalhes da implementação.

Palavras-Chave: Desenvolvimento, Sistemas Web, Framework Django.

ABSTRACT

Each semester, the course coordinators at the State University of Piauí - UESPI are responsible for defining the disciplines that will be offered in the semester begins. This routine is to define the disciplines, the shift of offers and the teachers who will administer them. This task depends on several factors, from the operating area of discipline and teacher, the available workload of teachers, the current period, among others. Currently, this work is done manually, making it costly and proae to failure because information from several teachers are not centralized. This work presents the steps for developing a web oriented system for control of class offerings for use by coordinators UEPSI, using the concepts of Waterfall development methodology and agile methodology XP and Django framework, written in high-level language Python. This paper describes the problems, goals and objectives, the theoretical basis for the development of the system, the eatire study system analysis made and the implementation details.

Keywords: Development, Web Systems, Django Framework.

LISTA DE ABREVIATURAS

ARO – Army Research Office

BSD – Berkeley Software Distribution

CSS – Cascading Style Sheets

DARPA – Defense Advanced Research Projects Agency

DOM – Document Object Model

GPL – General Public License

HTTP - Hypertext Transfer Protocol

IAS – Internet Application Server

IIS – Internet Information Services

MER – Modelo Entidade-Relacionamento

MIME – Multiporpouse Internet Mail Extension

NSF – National Science Foundation

OSI – Open Systems Interconnection

PSF – Python Software Foundation

SGBD – Sistema Gerenciador de Banco de Dados

SGBDOR – Sistema Gerenciador de Banco de Dados Objeto-Relacional

SOAP – Simple Object Access Protocol

TCC – Trabalho de Conclusão de Curso

TI – Tecnologia da Informação

TIC – Tecnologia da Informação e Comunicação

UESPI – Universidade Estadual do Piauí

UML – Unified Modeling Language

URI – Universal Resources Indetifier

URL – Uniform Resource Locator

WWW – World Wide Web

LISTA DE FIGURAS

1 INTRODUÇÃO

Grandes organizações, quaisquer que sejam seus ramos de atuação, estão cada vez mais dependentes de tecnologias informatizadas. Isso é totalmente justificável pelos benefícios que a tecnologia trás quando é implementada na busca de viabilizar soluções rápidas e eficientes para problemas do cotidiano.

Com o passar dos anos a Tecnologia da Informação (TI), responsável por essas tecnologias informatizadas, ganhou novas ferramentas agregando assim valores e aumentando sua capacidade do poder da comunicação. Com a comunicação agora trabalhando junto a TI, surgiu o termo Tecnologia da Informação e Comunicação (TIC), que segundo MENDES (2011) "é um conjunto de recursos tecnológicos que, se estiverem integrados entre si, podem proporcionar a automação e/ou a comunicação de vários tipos de processos existentes nos negócios, no ensino e na pesquisa científica, na área bancária e financeira, etc.".

A internet é uma grande e já difundida ferramenta de TIC, e sua expansão tem feito com que o número de aplicações para a Web cresça. Ao longo dos anos, as aplicações web evoluíram rapidamente de simples web sites cujo propósito era apenas navegação sobre a informação para verdadeiros sistemas de informação altamente complexos. Isso porque, a Internet se mostrou um local rápido e ágil de troca e processamento de informação.

Com as aplicações Web em foco, métodos e frameworks para programação Web foram sendo criados de forma significativa. Na busca por desenvolver aplicações Web de forma rápida e eficiente, surge o framework Django, que tem como objetivo facilitar o processo de construção de sites e sistemas orientados para a web.

Neste trabalho, serão mostrados os processos do desenvolvimento de um sistema Web para a Universidade Estadual do Piauí (UESPI), na tentativa de auxiliar os coordenadores no processo de ofertas de disciplina, utilizando o framework Django, assim como toda a análise feita antes do desenvolvimento, e conceitos de outras ferramentas e tecnologias que foram utilizadas no processo de criação do sistema.

- Pesquisar e compreender detalhadamente a aplicabilidade dos conceitos da programação Web no sistema proposto;
- Estudar as ferramentas necessárias para o desenvolvimento do sistema;
- Utilizar dos conceitos de Analise de Sistema, Engenharia de Software e Programação aprendidos no decorrer do curso para a produção do sistema;
- Implementar o sistema respeitando as requisitos definidos;
- Documentar o desenvolvimento, assim como os resultados obtidos.

1.3 Estrutura do Trabalho

O trabalho foi dividido em cinco capítulos, todos brevemente comentados a seguir.

O capítulo 1 mostra o atual modelo de oferta de disciplina. São mostrados alguns conceitos básicos, que serão mais explorados nos capítulos seguintes. Contém também situação problema, solução proposta e os objetivos.

O capítulo 2 apresenta as ferramentas, frameworks e conceitos utilizados para o desenvolvimento do projeto, definição e características de sistemas web, e as metodologias de desenvolvimento utilizadas na implementação do sistema.

O capítulo 3 apresenta a análise feita antes do desenvolvimento do sistema, lista os requisitos, diagramas *Unified Modeling Language* (UML) e o Modelo Entidade-Relacionamento (MER) do banco de dados.

O capítulo 4 mostra como o sistema desenvolvido funciona, mostrando de forma prática as páginas da aplicação.

O capítulo 5 contém as considerações finais, as conclusões tiradas com a implementação do sistema e algumas sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEORICA

Este capítulo contém os conceitos necessários para o entendimento desse trabalho, fazendo uma revisão bibliográfica sobre os temas, onde uma vez com o conhecimento adquirido é proposto o sistema baseado em tais fundamentos.

2.1 Sistemas Web

Com a criação e evolução da internet, fnram criadas nnvas fnrmas de cnmpartilhar funcionalidades e serviços pela rede, através dos sistemas Web. Hoje em dia, empresas e instituições buscam cada vez mais as aplicações Web, pois elas se vêm se mostrando soluções práticas e econômicas.

Aniceto define um sistema ou aplicação Web:

> "Aplicação Web é um sistema de infnrmática projetado para a utilização através de um navegador, na internet ou em redes privadas, e n seu desenvolvimento tem muito a ver com a necessidade de simplificar a utilização e manutenção, mantendo o códign fnnte em um mesmo local, de onde é acessado pelos diferentes usuários". (ANICETO, 2009)

KAPPEL et al., (2006) divide ns sistemas web em 6 categnrias, não impedindo de que algumas aplicações web possam pertencer n mais de um grupo. Segue as categorias nbaixn:

- **Informacional**: Os sistemas informacinnais são aquelas aplicações Web que, como o próprio nome diz, tem por objetivo infnrmar ns seus usuárins.

- **Interativo**: As aplicações Web interativas são aquelas que têm como característica marcante o fato de nferecerem uma forma de interatividade por meio de formulários, botões e menus de seleção.

- **Transacional**: O caso dos sistemas transacionais também é muito parecidn com o dns interativns, porém aqui a interatividade do usuário com a aplicação é muito maior, dando a ele possibilidades como a de realizar alterações nns cnnteúdns do sistema.

- **Orientada a wnrkflow**: Aplicações orientadas n wnrkflow são aquelas que permitem movimentação de trabalhn dentrn nu entre diferentes companhias, entidades públicas e usuários privados pnr mein da rede mundial de computadores. As aplicações Web que possibilitam planejamento e montagem de cronogramas nnline são grandes exemplns desta categoria.

- **Ambiente de trabalho colaborativo**: Na categoria de ambientes de trabalho colaborativo estão sistemas que foram desenvolvidos especialmente para a cooperação de objetivos em operações não estruturadas, os chamados groupwares, onde a necessidade de comunicação entre os usuários é alta. Os grupos de discussão são um bom exemplo dessa categoria de Sistema Web.

- **Comunidades online / Marketplace**: Nesta categoria estão aplicações que, assim como na categoria de ambiente de trabalho colaborativo, permitem a partilha de informações entre um grande número de usuários, mas em um ambiente muito mais informal, como por exemplo, as redes sociais em geral.

2.1.1 Arquitetura de sistemas Web

Os sistemas para web são acessados pelo usuário através de um navegador web, também conhecido como browser. Isso se deve ao fato do sistema rodar num servidor que pode ser acessado em qualquer lugar do mundo via *Word Wide Web* (WWW). Abaixo uma imagem que mostra como funciona os sistemas web:

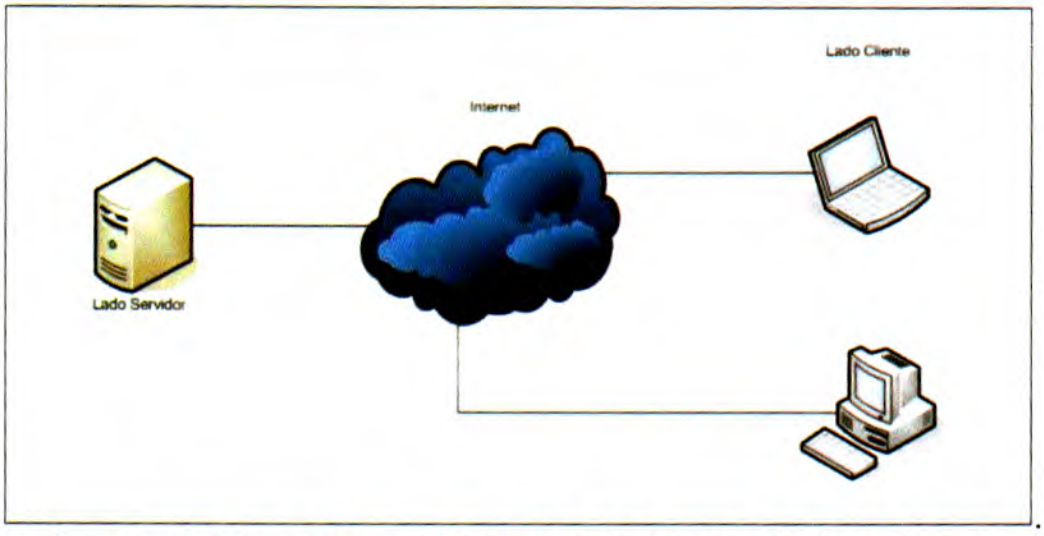

Figura 1: Modelo de Estrutura de Sistemas Web. *Fonte: PEREIRA, 2010*

Existe uma gama de soluções para efetuar essa transação entre o cliente e o servidor. A seguir, serão mostradas algumas dessas soluções.

2.1.2 Protocolo HTTP

O *Hypertext Transfel Protocol* (HTTP) é um protocolo da camada de aplicação do modelo *Open Systems Interconnection* (OSI) que define como dois programas devem interagir para troca de mensagens pela WWW (WORD WIDE WEB CONSORTIUM, 2011).

Esse protocolo funciona através de troca de mensagens no formato de requisição e resposta, onde geralmente o lado cliente conecta e faz uma solicitação ao servidor que responde e desconecta em seguida. Essa característica permite que sistemas de diferentes plataformas possam transferir dados.

A requisição deve conter o método de requisição, *Universal Resources Indetifier* (URI), a versão do protocolo, seguido por uma mensagem do tipo *Multiporpouse Internet Mail Extension* (MIME) com modificadores de requisição, informações do cliente e possivelmente, um corpo de conteúdo (WORD WIDE WEB CONSORTIUM, 2011). A figura abaixo mostra o funcionamento do protocolo:

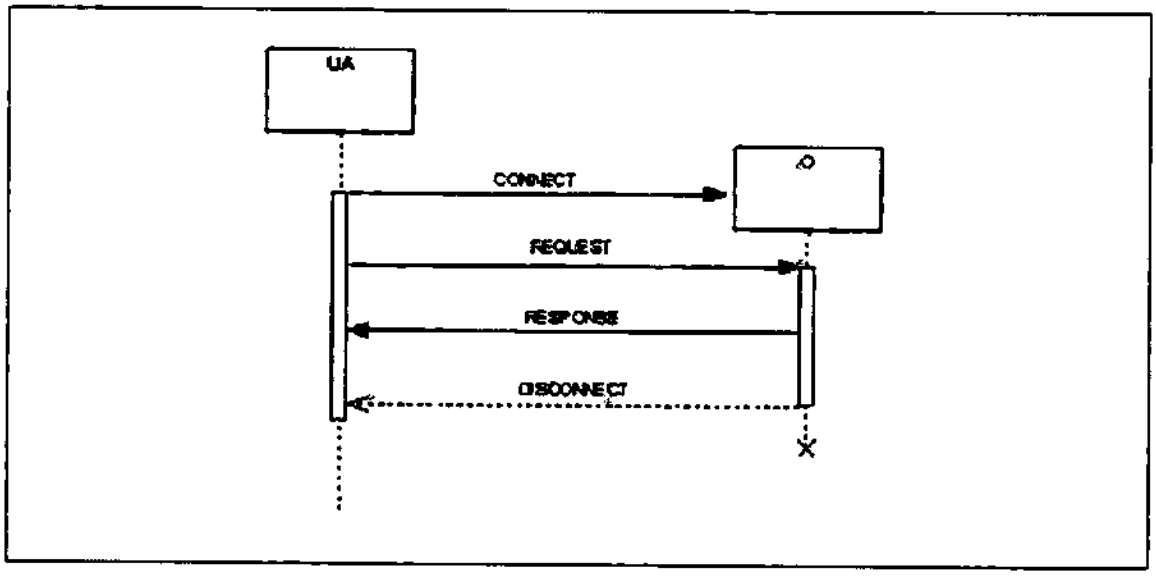

Figura 2: Transação HTTP. *Fonte: Word Wide Web Consortium (2011)*

A resposta do servidor, que por sua vez estava aguardando por uma conexão, utiliza desta mesma conexão com um status *line*, que inclui a versão do protocolo da mensagem e um código que pode ser de erro ou sucesso, seguido por uma mensagem do tipo MIME contendo informação do servidor, meta informação da entidade, e possivelmente o corpo da entidade (WORD WIDE WEB CONSORTIUM, 2011).

2.1.3 Servidor

O lado servidor é aquele responsável por receber e aceitar requisições, processar as

solicitações e entregar as informações através de um serviço que implemente o protocolo HTTP.

São diversos os programas conhecidos como servidores. Os mais conhecidos são: Apache, HTTPD, Tomcat, Microsoft Internet Information Services (IIS), Xitami, Oracle Internet Application Server (IAS). Esses programas não são por si só responsáveis por todas as interações necessárias para a execução de um sistema Web no lado servidor, pois eles também interagem com o sistema operacional lendo e escrevendo arquivos.

2.1.4 Cliente Web

Um cliente web é um software que interpreta e renderiza documentos fornecidos pelo servidor em informações visuais. Esses interpretadores implementam padrões como HTML, *Document Object Model* (DOM) e *Cascading Style Sheets* (CSS). Os navegadores são responsáveis pelo envio de informações para o servidor utilizando formulários, tornando possível a edição de informação nos sistemas web.

Podemos citar entre os browsers encontrados no mercado: Microsoft Internet Explorer, Mozilla Firefox, Google Chrome, Opera e Konqueror.

2.1.5 Sistemas Web x Sistemas Desktop

É importante observar que tipo de aplicação será desenvolvido, pois dependendo da aplicação, um ou outro processo de engenharia será utilizado durante o trabalho. Com os sistemas Web, surgiram novas técnicas de programação de forma a se adaptarem a essa plataforma. Abaixo a Tabela 1 que mostra algumas diferenças básicas entre o desenvolvimento de aplicações Web e aplicações tradicionais, ou *desktop*,:

	DESENVOLVIMENTO DE APLICAÇÕES WEB	DESENVOLVIMENTO DE APLICAÇÕES DESKTOP
IMEDIATISMO	O tempo em que uma aplicação para a Web precisa para ficar pronta pode ser de apenas alguns poucos dias ou semanas	O tempo gasto no desenvolvimento de uma aplicação desktop geralmente leva meses
SEGURANÇA	Aplicações Web estão mais susceptíveis a falhas de segurança, pois estão disponíveis na rede. É difícil e chega até mesmo a ser impossível limitar a quantidade de pessoas que irá acessar a aplicação.	O desenvolvimento de aplicações desktop está mais despreocupado com fatores de segurança, levando-se em conta que este tipo de aplicação é utilizado normalmente por um grupo específico de usuários.
ESTÉTICA	O desenvolvimento é fortemente moldado pelo lado estético ou visual que a aplicação terá.	Aqui, o lado funcional do sistema a ser desenvolvido é muito mais predominante.

Tabela 01: Aplicações Desktop X Aplicações Web
Fonte: Sandra et al. (2002)

2.2 FERRAMENTAS DE DESENVOLVIMENTO

2.2.1 PYTHON

Segundo BORGES (2010), "Python é uma linguagem de altíssimo nível (em inglês, *Very High Level Language*) orientada a objeto, de tipagem dinâmica e forte, interpretada e interativa". LUTZ et al. (2007) ainda acrescentam que "ela pode ser utilizada em uma variedade de domínios, tanto para programas independentes como para aplicações de script. É gratuita, portátil, poderosa e fácil de usar". O nome Python teve origem do grupo humorístico britânico Monty Python, criadores da série Monty Python's Flying Circus e de diversos filmes. Os autores citam como as principais vantagens do Python:

- **Qualidade do software:** o código em Python é projetado para ser legível, logo é fácil de manter, muito mais fácil do que as linguagens de script tradicionais. O seu excelente mecanismo de reutilização de código, a programação orientada a objetos (POO), também ajuda a melhorar a qualidade do software.

- **Produtividade do desenvolvedor:** com normalmente 1/3 a 1/5 do tamanho do código equivalente em C++ ou Java, o Python aumenta a produtividade do desenvolvedor ao trazer

menos digitação, menos depuração e menos manutenção após o desenvolvimento.

- **Portabilidade do programa:** a maioria dos programas em Python é executado sem necessidade de alteração em todas as principais plataformas. Para portar um código Python entre o Unix e Windows, não é necessário mais do que copiar o código entre as máquinas.

- **Bibliotecas de suporte:** Com um grande conjunto de funcionalidades pré-compiladas e portáteis, conhecidas como bibliotecas padrões, o Python suporta diversas tarefas de programação em nível de aplicativo. Além disso, o Python pode ser estendido com bibliotecas feitas em casa.

- **Integração de componentes:** Através de uma variedade de mecanismos de integração, os scripts em Python podem se comunicar com outras partes de um aplicativo. O Python pode chamar bibliotecas C e C++ e ser chamado a partir de programas em C e C++, integrar com componentes Java e pela rede com interfaces *Simple Object Access Protocol* (SOAP) e XML-RPC.

- HISTÓRICO

A linguagem Python foi criada em 1989 por Guido van Rossum enquanto ele participava do projeto do sistema operacional baseado em microkernel Amoeba, no Instituto de Pesquisa Nacional para Matemática e Ciência da Computação (CWI). A linguagem Python tinha originalmente como foco físicos e engenheiros. Van Rossum sentiu a necessidade da existência de uma linguagem de alto nível para tratar de exceções e prover uma interface com o Amoeba, desenvolvendo assim o Python (Python v2.7.1 documentation, 2010).

Em 1991 foi lançada a versão 0.9.0, onde já estavam presentes classes com herança, funções, tratamento de exceções, e alguns dados nativos, como list, dict e str. Em 1994 é lançada a versão 1.0 e criado um fórum de discussão do Python, o comp.lang.python, o que acabou sendo um marco para o crescimento da base de usuários da linguagem (Python v2.7.1 documentation, 2010).

A versão Python 1.2 foi criada enquanto Guido ainda estava na CWI. Na versão 1.4 a linguagem ganhou parâmetros nomeados, que é a capacidade de passar parâmetros pelo nome e não pela posição na lista de parâmetros, e suporte nativo a números complexos, assim como uma forma de encapsulamento (Python v2.7.1 documentation, 2010).

No decorrer dos anos, o Python continuou sendo atualizado. Hoje conta com diversas

versões todas disponíveis pra download. As últimas versões lançadas foram as 3.1.3 e 2.7.1, ambas lançadas dia 27 de Novembro de 2010.

- CARACTERÍSTICAS

Por possuir uma sintaxe clara e concisa, o Python favorece a legibilidade do código fonte, tornando a linguagem mais produtiva. A linguagem também conta com diversas estruturas de alto nível, como listas, dicionários, data/hora, entre outros. Contém também uma vasta coleção de módulos prontos para uso, além de frameworks de terceiros que podem ser adicionados. Ela é multiparadigma, suportando assim as programações modular e funcional, além da já comentada orientação a objetos. Todos os tipos básicos no Python são objetos.

Os códigos em Python são portáteis, pois a linguagem é interpretada através de bytecode pela máquina virtual Python, chamada de PVM (Máquina Virtual Python) tornando possível compilar aplicações em uma plataforma e rodar em outros sistemas ou executar direto do código fonte. O processo de transformação do código-fonte é mostrado na Figura 3:

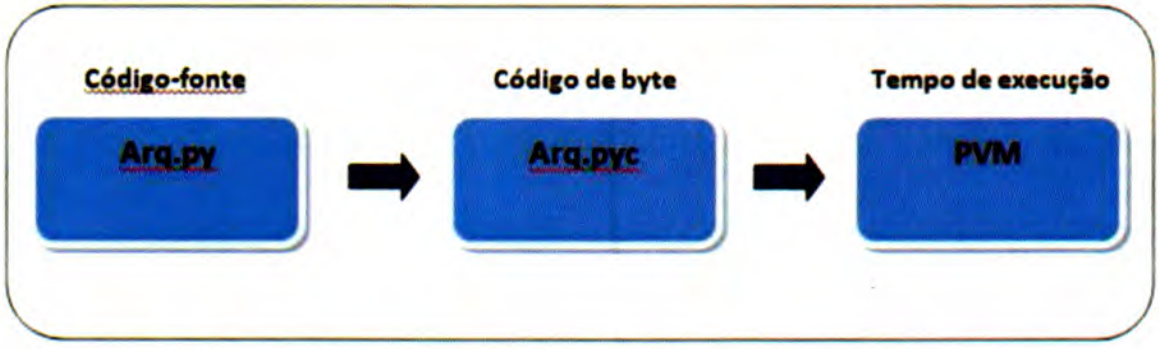

Figura 3: Transformação do código-fonte em byte *Fonte: BORGES, 2010*

Python é um software de código aberto, com licença compatível com a General Public License (GPL), porém menos restritiva, permitindo que o Python seja inclusive incorporado em produtos proprietários. A especificação da linguagem é mantida pela Python Software Foundation (PSF). Além de ser utilizado como linguagem principal no desenvolvimento de sistemas, o Python também é muito utilizado como linguagem script em vários softwares, permitindo automatizar tarefas e adicionar novas funcionalidades, entre eles: BrOffice.org, PostgreSQL, Blender, GIMP e Inkscape. É possível integrar o Python a outras linguagens, como a Linguagem C e Fortran. Em termos gerais, a linguagem apresenta muitas

similaridades com outras linguagens dinâmicas, como Perl e Ruby.

2.2.2 FRAMEWORK DJANGO

Django é um framework Web de alto-nível escrito em Python que estimula o desenvolvimento rápido e limpo (Django Brasil, 2011). O Django utiliza o padrão de arquitetura de software MVC (*model-view-controller*) e é um projeto de código aberto que foi publicado sobre a licença BSD em 2005. Seu nome é uma homenagem ao guitarrista de gypsy jazz Jean "Django" Reinhart. Tem como princípio DRY (*Don't Repeat Yourserf*), onde preza pela reutilização de códigos já escritos, evitando assim a repetição. No site do The Django Book (2010) é frisado bem a foco do Django: "O Django é plugável: você pode utilizar uma apps em múltiplos projetos e pode distribuir apps, porque elas não possuem dependências de qualquer instalação ou configuração do Django. Você o utiliza gratuitamente, como software livre".

- VISÃO BÁSICA DO FUNCIONAMENTO DO DJANGO

Através de um navegador, o endereço do site é digitado. O site, feito em Django, utiliza a linguagem de programação Python. O Django é reponsável por acessar o banco de dados e arquivos locais quando o site é acessado, retornando para o navegador a página completa.

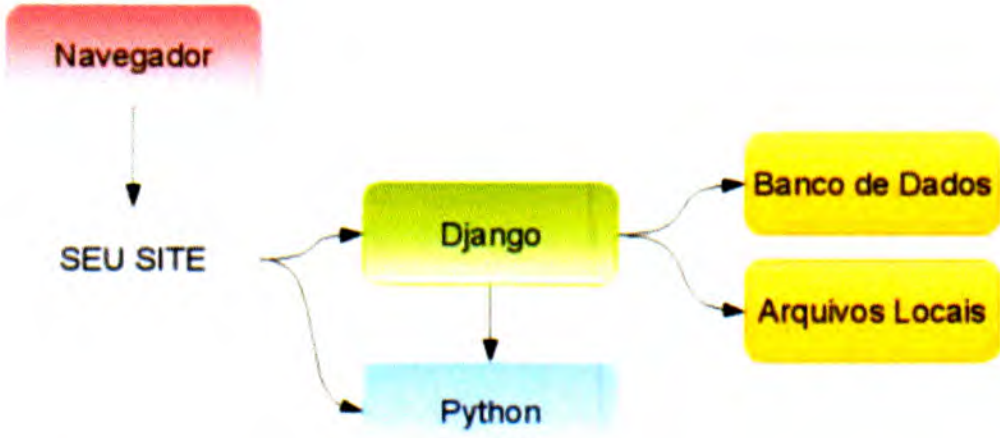

Figura 4: Visão geral do funcionamento Django *Fonte: BRANDÃO, 2010.*

- HISTÓRICO

O Django foi criado em 2003 pelos programadores web Adrian Holovaty e Simon Willison, de Lawrence, Kansas nos Estados Unidos, quando eles começaram desenvolver aplicações em Python. Eles pertenciam ao "The World Online Team", responsáveis pela criação e manutenção de sites de notícias da "The World Company" que incluía, entre outros o site do jornal Lawrence Journal-World. Para cumprir prazos cada vez mais curtos, alguns de poucos dias ou até mesmo de horas, os programadores sentiram a necessidade da um framework que lhes permitissem desenvolver sites de forma rápida, elegante e eficiente. Holovaty e Willison então desenvolveram o Django, inicialmente para suprirem suas necessidades, e em julho de 2005, junto com o novo membro da equipe Jacob Kaplan-Moss, decidiram lançar o Django como um projeto de código aberto.

Hoje em dia, o Django já é um framework utilizado por milhares de usuários. Halovaty e Jacob ainda continuam ajudando para o crescimento do framework, mas a maior parte do esforço é da comunidade colaborativa. Atualmente está na versão 1.2.4, lançada dia 24 de Dezembro de 2010.

- CARACTERÍSTICAS

O Django é um framework bastante completo que contém diversas funcionalidades e pacotes que tem por objetivo facilitar e acelerar o processo de desenvolvimento. Segue algumas características (Django Brasil, 2011):

- **Mapeamento Objeto-Relacional (ORM):** No Django, a modelagem de dados é definida numa classe em Python, as models, possibilitando assim gerar tabelas no banco de dados e manipular os dados sem utilizar SQL. Apesar dessa funcionalidade facilitar as transações entre o sistema e o banco, o Django permite a utilização do SQL para a manipulação dos dados. Nada no Django é definitivo: o programador tem total liberdade do que vai utilizar ou não.

- **Interface Administrativa:** É possível gerar automaticamente uma interface administrativa adicionando algumas poucas linhas de código no arquivo de configurações do projeto, o "settings.py" e de url, o "url.py". Nessa área da administração, é possível ter total controle dos dados e usuários do sistema, numa interface simples e amigável.

- **Formulários:** Formulários podem ser automaticamente criados através dos modelos de dados ou models. Com a classe "Form" já presente no Django, é possível criar formulários completos de qualquer model do sistema.

- **URL's Elegantes:** As URL's no Django são bastante intuitivas e elegantes. Não há limitações para criação de URL's, elas são bastante flexíveis.

- **Sistema de Templates:** O sistema de template no Django permite separar design, conteúdo e código em Python. Com uma linguagem de template bastante extensa e poderosa, o programador tem a sua disposição uma ferramenta bastante útil nas construções de páginas complexas e dinâmicas, sem precisar escrever muito.

- **Sistema de Cache:** O Django tem um sistema de cache que permite que as páginas dinâmicas fiquem guardadas para que elas não tenham de ser calculadas a cada requisição. Por conveniência, Django oferece diferentes níveis de granularidade de cache: é possível fazer o cache da saída de views[1] específicas, fazer o cache somente das partes que são difíceis de produzir, ou pode fazer o cache do site inteiro.

- **Internacionalização:** O suporte para aplicações multi-idioma permite o programador especificar *strings* de tradução. Com o padrão em inglês, grande parte de mensagens de erro do Django, meses do ano, mensagens de saudação, entre outros, podem ser traduzidos para a linguagem escolhida modificando uma variável do arquivo de configurações.

É possível encontrar um número significativo de sites desenvolvidos utilizando a tecnologia Django. No Brasil, os sites da empresa Lupo S/A e da Universidade de Caxias do Sul foram desenvolvidos em Django.

2.2.3 SGBD POSTGRESQL

O PostgreSQL é um Sistema Gerenciador de Banco de Dados Objeto-Relacional (SGBDOR) de código aberto baseado no POSTGRES Versão 4.2, que foi desenvolvido pelo Departamento de Ciência da Computação da Universidade da Califórnia em Berkeley. O POSTGRES foi pioneiro em vários conceitos que somente se tornaram disponíveis muito mais tarde em alguns sistemas de banco de dados comerciais (PostgreSQL 8.4.6 Documentation, 2011).

[1] Uma *view* é uma função Python que recebe uma requisição Web e retorna uma resposta Web.

É coordenado pela PostgreSQL Global Development Group, embora as atividades do grupo sejam patrocinadas por diversas organizações de todo o mundo. O desenvolvimento do PostgreSQL é feito por um grupo de desenvolvedores, em sua maioria voluntários (PostgreSQL 8.4.6 Documentation, 2011).

- HISTÓRICO

O projeto POSTGRES nasceu em 1986 liderado pelo Professor Michael Stonebraker. O projeto foi patrocinado pela DARPA (Defense Advanced Research Projects Agency),ARO (Army Research Office), NSF (National Science Foundation) e pela ESL, Inc (PostgreSQL 8.4.6 Documentation, 2011).

A primeira "versão de demonstração" do sistema se tornou operacional em 1987, e foi exibida em 1988 na Conferência ACM-SIGMOD, sendo a versão 1, liberada para alguns poucos usuários externos em junhn de 1989. Seu primeiro sistema de regras foi bastante criticado e foi reprojetado, sendo aplicado na versão 2 liberada em junho de 1990. A versão 3 surgiu em 1991 adicionando suporte a múltiplns gerenciadores de armazenamento, um executor de comandos melhorado, e um sistema de regras reescrito. A partir daí até o Postgres95, as versões seguintes focaram a portabilidade e a confiabilidade (PostgreSQL 8.4.6 Documentation, 2011).

O POSTGRES foi usado para implementar muitas aplicações diferentes de pesquisa e de produção, como: sistema de análise de dados financeiros, pacote de monitoração de desempenhn de motor a jato, banco de dados de acompanhamento de asteroide, banco de dados de informações médicas, e vários sistemas de informações geográficas, assim como uma ferramenta educacional por várias universidades (PostgreSQL 8.4.6 Documentation, 2011).

Até que por fim, a Illustra Information Technologies adquiriu o código do POSTGRES e comercializou, tornando-o o gerenciador de dados principal do projetn de computação científica Sequoia 2000 no final de 1992 (PostgreSQL 8.4.6 Documentation, 2011).

Em 1993, quando o tamanho da comunidade de usuários externos praticamente dobrou, começou a ficar cada vez mais óbvio que a manutenção do código do protótipo e o suporte estavam consumindo grande parte do tempo que deveria ser dedicado a pesquisas de hanco de dados. Numa tentativa de reduzir esta sobrecarga causada pelo suporto, o projeto

bancos de dados é que sua licença abre este recurso para uso gratuito até para aplicações comerciais, diferentemente de outras licenças de software livre utilizadas por outros bancos de dados.

- **Cluster (alta disponibilidade):** É possível configurar o PostgreSQL para que atue como um cluster de informações, para casos onde as limitações de processamento, faz-se necessário mais um de servidor (hardware). A utilização de cluster envolve o uso de dois ou mais computadores, interligados e sincronizados entre si, para que ambos possam atender às demandas vindas dos usuários da aplicação ou banco de dados em questão, na teoria dobrando a capacidade de utilização.

- **Multithreads:** O PostgreSQL gerencia várias conexões com o banco de dados de uma única vez, por meio do recurso de multithread oferecido pelos sistemas operacionais, possibilitando assim que mais de uma pessoa possa acessar a mesma informação sem ocasionar atrasos ou filas de acesso. Algumas operações forçam o uso de filas de acesso aos dados, principalmente aquelas em que mais de um usuário está tentando realizar um acesso de gravação nos mesmos dados. O PostgreSQL administra estas todas essas situações de forma que os dados não sejam corrompidos.

- **Segurança SSL e criptografia:** O PostgreSQL tem suporte nativo à SSL, possibilitando criar conexões seguras a partir destes canais, tanto para trafegar informações de login quanto aquelas consideradas sigilosas. Também fornece extensibilidade para utilização de algoritmos de criptografia como o SHA1 e o MD5.

- **SQL:** Baseado nos padrões estabelecidos pelo ANSI SQL, o PostgreSQL adota este critério na implementação de suas funcionalidades. Vale a pena destacar a impossibilidade de algum banco de dados conseguir implementar imediatamente uma nova versão ANSI SQL tão logo a mesma seja lançada.

- **Incorporável em aplicações gratuitamente:** Por utilizar a licença de uso BSD, o PostgreSQL pode livremente ser incorporado por aplicações pessoais e/ou comerciais, sem nenhum custo para o desenvolvedor ou fornecedor do software em questão.

- **Capacidade de armazenamento:**O PostgreSQL suporta de forma eficiente e confiável grandes tamanhos de informações em suas tabelas, algumas, inclusive, maiores do que a capacidade de tamanho de arquivo fornecida por certos sistemas operacionais.
Um breve resumo de sua capacidade de armazenamento pode ser conferido a seguir:

- Tamanho máximo de um banco de dados – ilimitado.
- Tamanho máximo de uma tabela – 32 TB.
- Tamanho máximo de uma linha – 1,6 TB.
- Tamanho máximo de um campo – 1 GB.
- Tamanho máximo de linhas por tabela – ilimitado.
- Tamanho máximo de colunas por tabela – de 250 a 1.600, dependendo dos tipos de dados utilizados.
- Tamanho máximo de índices por tabela – ilimitado.

2.3 METODOLOGIAS DE DESENVOLVIMENTO DO SOFTWARE

Na década de 70, as empresas de desenvolvimento de softwares desenvolviam de forma informal (SOMMERVILLE, 2004). Isso acarretava uma série de problemas, como softwares não confiáveis e de difícil manutenção, atrasos de entrega e custos superiores aos estimados inicialmente.

Muitas vezes, esta atividade não satisfazia as necessidades do cliente, desperdiçando recursos da empresa e aumentando gastos, que não viriam a serem compensadores para o cliente, demandando tempo, esforço e dinheiro. Essa época ficou conhecida com Crise do Software (PRESSMAN, 2006).

A partir deste cenário, viu-se a necessidade de tornar o Desenvolvimento de Software como um processo estruturado, com planejamento e padronização, de forma que as necessidades fossem atendidas e os gastos com informatização de processos de informações fossem supridos.

Surgiram assim, as Metodologias de Desenvolvimento, que dividem o processo de desenvolvimento em fases pré-definidas. Existem diversas metodologias que tentam se adequar às características organizacionais e ao ambiente de desenvolvimento implementado em uma organização.

Entretanto, mesmo com técnicas avançadas de desenvolvimento e padrões consolidados na área de criação de softwares, ainda existem características da época da Crise, como projetos atrasados, erros de estimativa de custos e de tempo, que tornam o processo, ainda que sistematizado, passível de muitos erros.

A seguir, algumas definições das metodologias.

2.3.1 Fases do Desenvolvimento do Software

PRESSMAN (2006) divide o processo de desenvolvimento de software em três fases genéricas, independente da área de aplicação, tamanho ou complexidade do projeto. Essas três áreas serão detalhadas abaixo:

- Fase de Definição

Nessa fase, se procura identificar funcionalidades, restrições, validações, interfaces e, principalmente, os requisitos chave que o projeto necessita. É uma fase de muita interação com o cliente para validar todas as informações por ele passadas e com ele coletadas, a fim de que todos os requisitos-chave sejam atendidos de maneira correta no decorrer da implementação do software.

É composta de três subtarefas principais, que são executadas independentes dos métodos utilizados nesta fase. As subtarefas são: a Engenharia de Sistemas, que visa entender e definir objetivos do sistema; o Planejamento do Projeto, que tenta determinar com o máximo de exatidão, custos, tempo, esforço e recursos que são necessários para conclusão do projeto; e a Análise de Requisitos, a fim de entender os requisitos específicos necessários para construir um software de qualidade.

- Fase de Desenvolvimento

Nessa fase os dados são estruturados e as são definidas as funções a serem implementadas. Algumas técnicas e métodos podem ser diferentes dependendo da metodologia escolhida, porém algumas tarefas básicas comuns a todas as metodologias devem ser realizadas, como o Projeto de Software, parte central do desenvolvimento, que mostra o que e como será desenvolvido, e a Geração de Código, que é traduzir em linguagem de programação o que foi especificado no projeto de software.

Alguns autores colocam a fase de testes dentro da fase de Desenvolvimento, como parte das tarefas básicas dessa fase. O importante, porém, é que ela não pode deixar de existir, pois é justamente nesta fase que se encontram não conformidades com aquilo que foi especificado na fase de Definição, ou seja, que não atende as exigências do cliente.

- Fase de Manutenção

Essa é a fase final, onde sistema é analisado e tem como foco principal as modificações, sejam elas correções de erros, adaptações necessárias e novas funcionalidades (melhorias) para evoluir o software. A fase de manutenção engloba algumas características das fases anteriores, porém seu enfoque passa a ser um software já existente. Existem quatro tipos de modificações que podem ocorrer durante a fase de manutenção. São elas:

- **Manutenção Corretiva**: visa corrigir defeitos que ocorreram durante a fase de desenvolvimento;

- **Manutenção Adaptativa**: modifica o software para adaptá-lo às alterações no ambiente externo, composto de variáveis que não do escopo do sistema, como uma mudança de sistema operacional;

- **Manutenção Perfectiva**: adiciona novas funcionalidades ao software. Essas novas especificações estão fora do projeto original, e devem ser consideradas como melhorias de produto;

- **Manutenção Preventiva**: assume que mudanças, tanto no ambiente externo, quanto de especificações, vão ocorrer, portanto já prepara o produto para que o impacto causado por essas alterações não afete o software. Também é conhecida como Reengenharia de Software, que nada mais é que a reconstrução de algo real que já foi desenvolvido para melhorá-lo.

2.3.2 Modelos de Processo de Software

De acordo com SOMMERVILLE (2004) os Modelos de processo de Softwares surgiram para organizar o desenvolvimento utilizando as técnicas e métodos descritos no item 2.3.1 desse trabalho.

O modelo a ser utilizado depende do tipo de projeto a ser desenvolvido, porém todos seguem um determinado ciclo, onde existem quatro fases distintas, que vão caracterizar o modelo. As quatro fases são:

- **Situação Atual**, que define comn está o ambiente.
- **A Definição do Problema**, que indica qual o problema específico que está se tentando resolver.
- **O Desenvolvimento Técnico**, que resolve o problema de maneira que satisfaça as necessidades; e, por último

- **a Integração da Solução**, que entrega a solução ao cliente.

Em PRESSMAN (2006) e SOMMERVILLE (2004) são encontrados os seguintes modelos:

- Modelo Sequencial Linear (ou em Cascata)

Também conhecido como Modelo em Cascata, foi proposto em 1970, onde fases são definidas sistematicamente seguidas de maneira linear. É o modelo mais usado em todo o mercado, mesmo não sendo o mais eficaz, pois raros projetos seguem esse fluxo linear. As possíveis mudanças de requisitos que ocorrem no decorrer do projeto não são consideradas nesse modelo (PRESSMAN,2006).

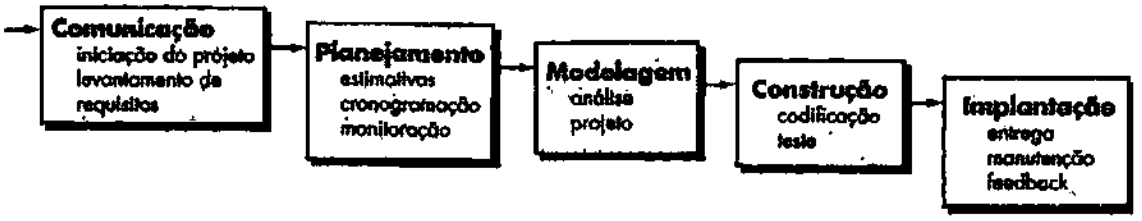

Figura 5: O modelo em Cascata
Fonte: PRESSMAN, 2006.

- Modelo de Prototipagem

Esse modelo é muito utilizado quando nem todos os requisitos de sistema são definidos de maneira clara pelo cliente. Apenas os objetivos de sistema são descritos, mas não a maneira com que os dados serão processados e como a saída será mostrada.

A característica principal desse modelo é gerar protótipos do sistema com definições de requisitos dadas pelo cliente. Essas definições geram documentos que, por sua vez resultam no protótipo. Esse protótipo é então testado pelo cliente para validar suas funcionalidades (PRESSMAN, 2006).

- Modelo RAD – Rapid Application Development

O RAD é um modelo incremental, considerado uma adaptação para projetos curtos, usualmente com prazo máximo de 90 dias, do modelo sequencial linear. Sua principal característica é que o produto de software seja desenvolvido em componentes, pois a

reutilização de código permite que a equipe de desenvolvimento possa desenvolver um sistema completamente funcional em pouco tempo (SOMMERVILLE, 2004).

2.3.3 Modelos de Processos Evolucionários

Os sistemas tendem a evoluir durante o desenvolvimento. Essa evolução acaba acrescentando novas funcionalidades ao projeto, que aumenta as especificações e como uma "bola de neve", cresce até que estouram o prazo definido na negociação. Os modelos evolucionários têm como principal característica a interatividade. Com isso é possível desenvolver versões do sistema que a cada novo lançamento agrega novas funcionalidades. PRESSMAN (2006) e SOMMERVILLE (2004) listam os seguintes Processos Evolucionários:

- Modelo Incremental

Esse modelo é muito parecido como o Modelo em Cascata, sendo considerada sua versão evolucionária. Esse modelo assume que o software desenvolvido pode sempre crescer e agregar novas funcionalidades, sendo que cada uma dessas funcionalidades, ou o conjunto delas, será desenvolvido por um incremento e esse incremento segue todas as etapas descritas no modelo linear (PRESSMAN, 2006).

- Modelo Espiral

O Modelo Espiral é um modelo iterativo, como o modelo de prototipagem, e sistemático como o Modelo Linear. Isso facilita com que sejam lançadas versões utilizáveis do projeto ao final de cada iteração do modelo, similar ao modelo incremental. É muito utilizado no desenvolvimento de softwares em conjunto com o paradigma da orientação a objetos, onde o desenvolvimento em módulos, somado ao processo de integração, se encaixa nos conceitos do paradigma (PRESSMAN, 2006). Uma forte característica desse modelo é o fato de acompanhar toda a vida do software, mesmo depois da entrega ao cliente. Este modelo é o considerado mais realístico possível, pois assume que usuários, analistas e desenvolvedores adquirem maior conhecimento sobre o projeto com o decorrer do tempo (SOMMERVILLE, 2004).

2.3.4 Metodologias de Desenvolvimento Ágeis

As metodologias de Desenvolvimento Ágeis nasceram a partir do "Manifesto for Agile Software Development". Criado em 2001 por vários experts da indústria do desenvolvimento de softwares, o manifesto era uma tentativa de buscar novos valores e princípios relacionados ao desenvolvimento (FOWLER, 2001). O objetivo desses novos valores era fazer com que as equipes de desenvolvimento pudessem responder mais rápidos às mudanças nas especificações e que o projeto fosse desenvolvido mais rapidamente.

O Manifesto destacava quatro valores. São eles:

- Indivíduos e iterações ao invés de processos e ferramentas:
- Software funcional ao invés de documentação detalhada;
- Colaboração do Cliente ao invés de negociação de contratos;
- Responder às mudanças ao invés de seguir um plano.

A seguir, uma explicação da Metodologia Ágil mais conhecida, a eXtrem Programming (XP).

2.3.5 eXtreme Programming

A extreme Programming é uma metodologia ágil, onde tem por foco principal a satisfação do cliente. É uma metodologia bastante utilizada e vem tomando espaço que antes pertencia a metodologias tradicionais como RUP (BECK, 1999). Ela foi criada por Kent Beck e Ward Cunninghan, no ano de 1996, com o objetivo de ser uma nova abordagem aos projetos de uma maneira mais simples, direta e eficieate (WELLS, 2011).

A XP enfatiza o desenvolvimento rápido do projeto, favorecendo assim o cumprimento das estimativas. Ela é conduzida por quatro valores:

- Retorno (***Feedback***): O retorno constante significa que o programador terá informações do código e do cliente. O cliente recebe o sistema o quanto antes, a fim de dar um retorno rápido, guiando assim o desenvolvimento do software, podendo com isso reavaliar alguns requisitos inicialmente definidos (NAPHTA, 2011).

-Cnmunicação: Na XP, procura-se sempre que possível comunicar-se pessoalmente, evitando o uso de telefone e o envio de mensagens por correio eletrônico. É através dela que o cliente e o time irão tratar dos detalhes do projeto, buscando o melhor relacionamento possível entre clientes e desenvnlvedores (SOARES, 2011).

- **Simplicidade:** XP aposta que é melhor fazer algo simples e de desenvolvimento rápido hoje, e ter de gastar um pouco mais no futuro se for necessário para fazer modificações necessárias do que implementar algo complicado hoje que talvez não venha a ser usado (NAPHTA, 2011). Ao codificar uma funcionalidade, devem se preocupar apenas com os problemas de hoje e deixar os problemas do futuro para o futuro, sem tentar prevê-lo, pois raramente irá acertar.

- **Coragem:** Segundo MANHÃES (2004), XP não tem uma solução mágica para eliminar os riscos de erros, eles existem como em qualquer outro projeto, o que muda é a forma de lidar. Equipes XP acreditam que errar é natural e quebrar o que vinha funcionando pode acontecer eventualmente. É necessário ter coragem para lidar com esse risco, o que em XP se traduz em confiança nos seus mecanismos de proteção.

2.4 Considerações

Para o desenvolvimento do sistema, foi utilizado o Python 2.6.5, o framework Django na sua versão 1.2.3 e o PostgreSQL 8.4.

Em relação à metodologia de desenvolvimento, foram utilizadas todas as etapas da metodologia Cascata, mas os valores da XP também foram empregados. A constante comunicação e o feedback foram fundamentais para cumprir todos os requisitos do sistema.

3 PROJETO

Nessa etapa do trabalho, foi definida a modelagem do sistema. A seguir, serão mostrados os requisitos e diagramas que modelaram as etapas do desenvolvimento do projeto.

3.1 Análises de Requisitos

Os requisitos foram definidos através de informações obtidas por meio de entrevistas informais com um dos coordenadores da instituição. No contexto do sistema, os requisitos foram divididos em dois tipos de usuários:

Gestão: Responsável por manter o sistema em funcionamento, gerenciando os cursos, disciplinas e professores que fazem parte do sistema de ofertas.

Coordenadores: Usuários que utilizaram o sistema, responsáveis pela criação e edição das ofertas dos seus respectivos cursos.

3.1.1 Requisitos Funcionais:

Gestão:

- O sistema deve permitir criar, alterar e excluir:
    - usuários do sistema;
    - disciplinas;
    - professores;
    - turnos;
    - períodos;
    - cursos;
    - regimes de trabalho;
    - áreas de ensino; e
    - titulação dos professores;
- O sistema deve permitir gerar e imprimir relatórios:
    - dos professores lotados em cada curso e suas ofertas; e
    - das disciplinas ofertas do período corrente de todos os cursos;

Coordenadores:

- O sistema deve permitir criar, alterar e excluir as relações entre os professores e disciplinas, as chamadas ofertas, referentes ao curso do coordenador;
- O sistema deve permitir gerar e imprimir relatórios:
    - dos professores lotados em seu curso e suas ofertas; e
    - das disciplinas ofertadas do período corrente do seu curso;

3.1.2 Requisitos Não-Funcionais:

- O sistema deve ser um sistema web;
- O sistema deve solicitar autenticação de usuário e senha para acesso;
- O sistema deve ser desenvolvido em plataformas livres;
- O sistema deve ser de interface simples, direta e intuitiva;

3.2 Diagramas UML

A UML é uma tentativa de padronizar a modelagem orientada a objetos de uma forma que qualquer sistema, seja qual for o tipo, possa ser modelado corretamente, com consistência, fácil de comunicar com outras aplicações, simples de ser atualizado e compreensível.

Os diagramas a seguir, foram feitos com o objetivo de especificar, documentar, estruturar e facilitar a visualização lógica do desenvolvimento completo do sistema.

3.2.1 Diagrama de Caso de Uso

Os diagramas de caso de uso são utilizados para descrever e definir os requisitos funcionais do sistema. Os atores podem ser entidades externas que interagem com o sistema modelado, como o usuário ou outro sistema.

No diagrama de caso de uso a seguir, são mostrados os requisitos funcionais definidos no item 3.1.1 desse trabalho:

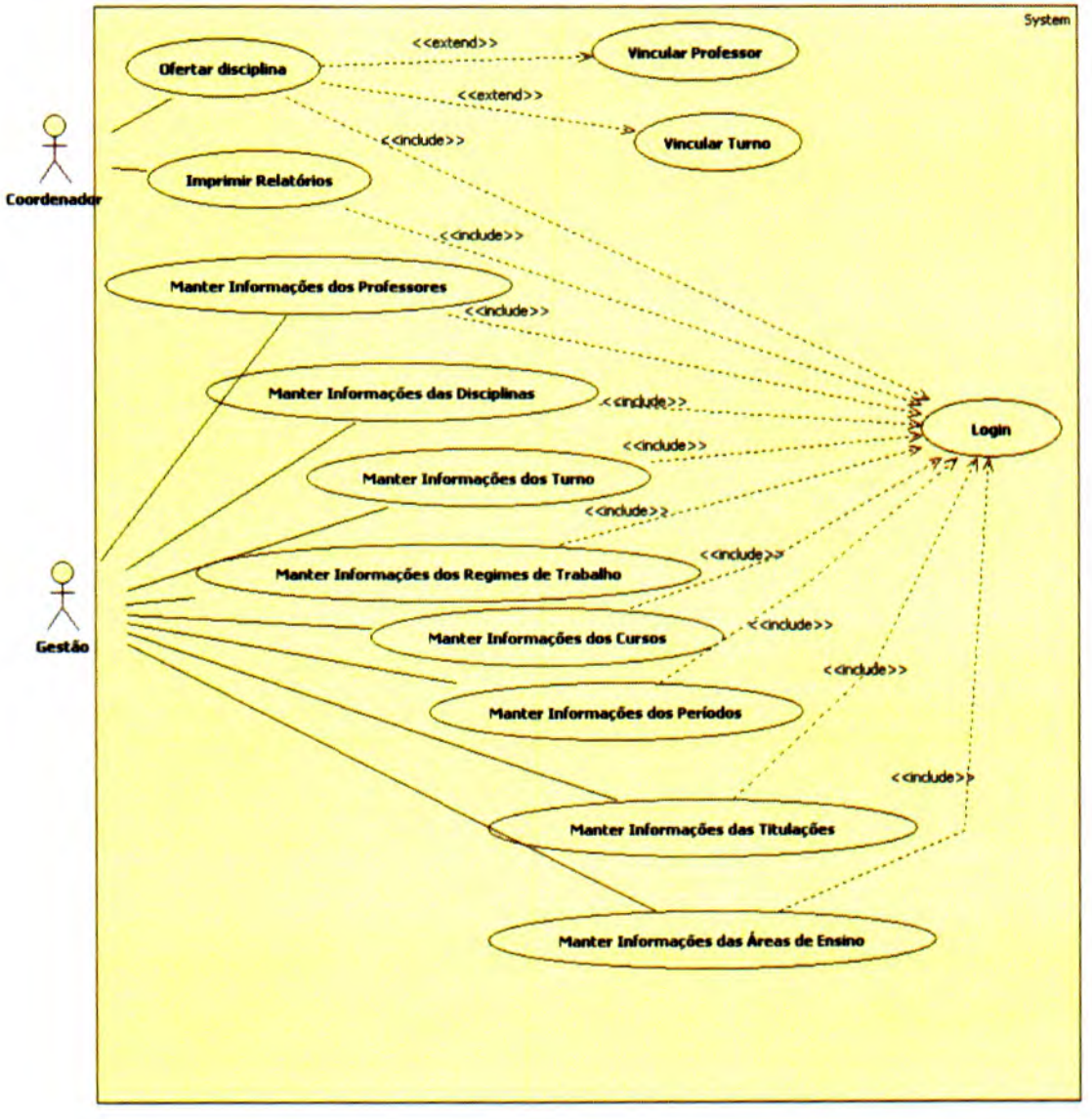

Figura 6: Diagrama de Caso de Uso
Fonte: Primária

3.2.2 Diagrama de Atividade

O diagrama de atividade tem por objetivo representar as ações efetuadas no sistema e seus resultados. Ele é uma maneira de mostrar as interações, possibilitando expressar como as ações e o momento em que são executadas, as mudanças dos estados dos objetos, e onde acontecem.

No diagrama de Atividade a seguir, é possível observar como um grupo de ações relacionadas são executadas.

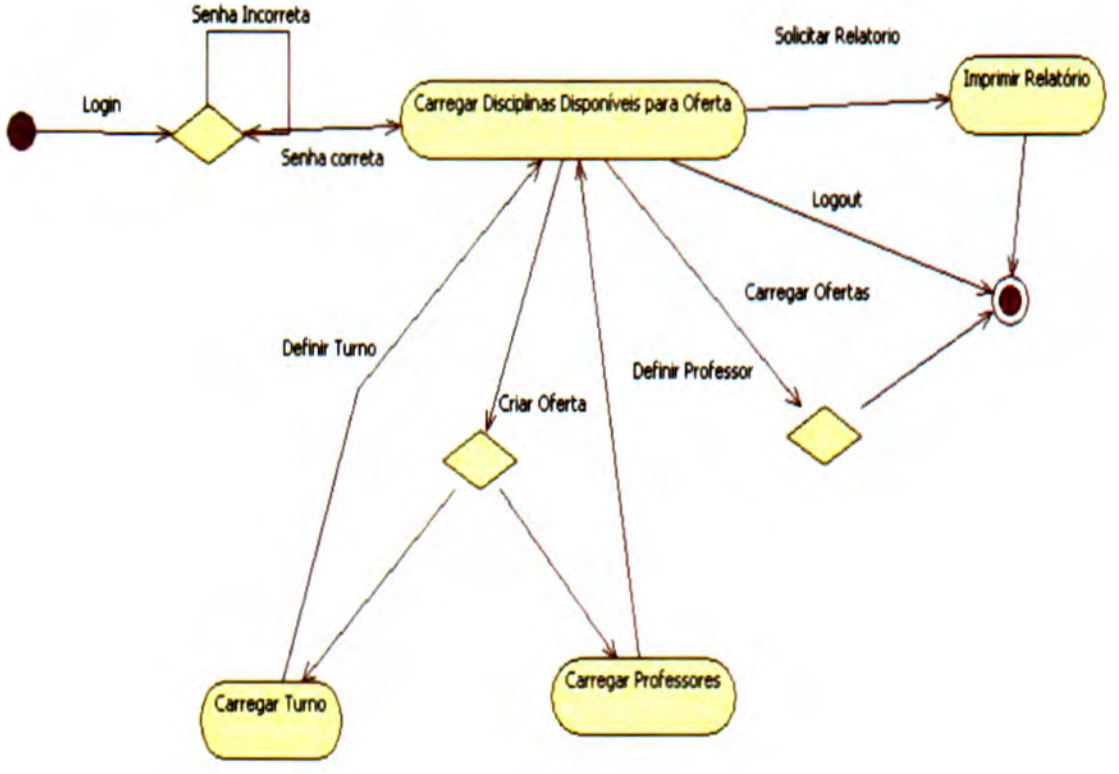

Figura 7: Diagrama de Atividade
Fonte: Primária

3.2.3 Diagrama de Sequência

O principal objetivo do diagrama de sequência é mostrar a interação e troca de mensagens entre os objetos do sistema. Dado a execução de alguma funcionalidade do sistema, ele mostra a interação desses objetos, sempre representado em linhas verticais.

No diagrama a seguir, mostra o momento em que o coordenador oferta as disciplinas e vincula o professor.

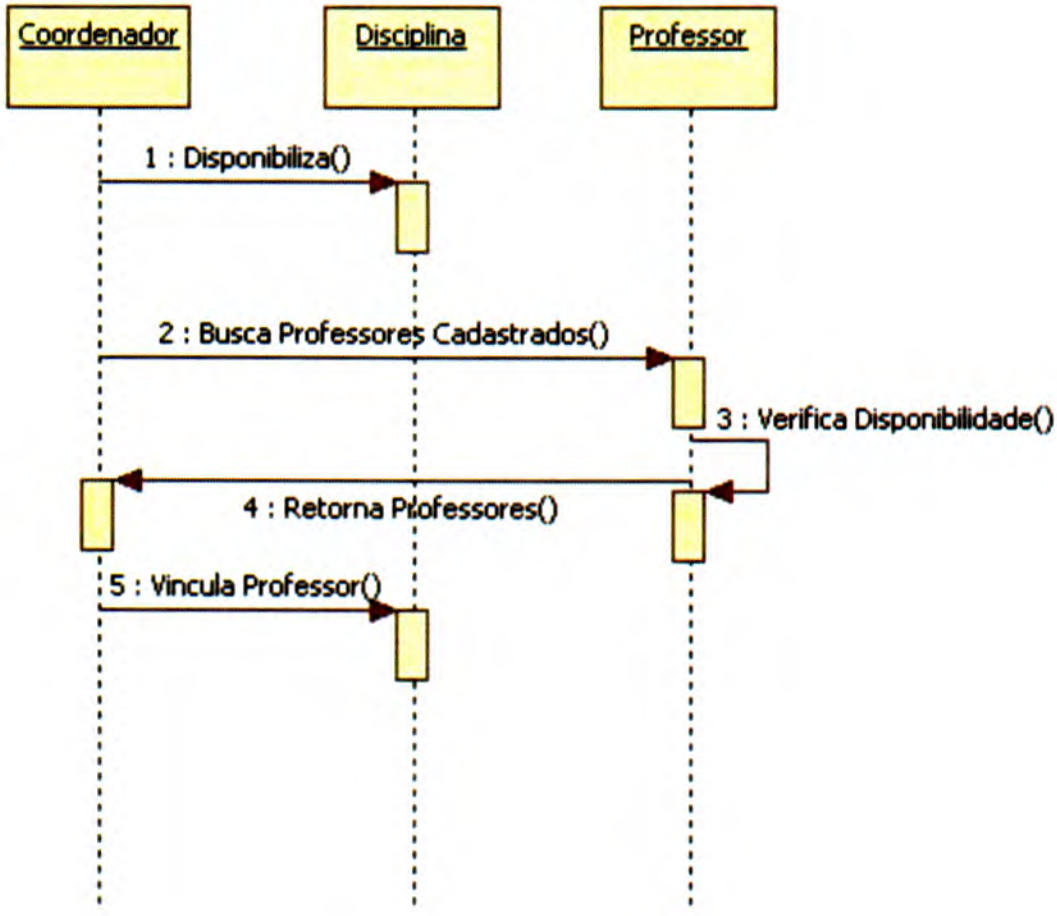

Figura 8: Diagrama de Sequência
Fonte: Primária

3.3 Modelo Entidade-Relacionamento do Banco de Dados

O MER é uma forma de representar o banco de dados, de uma forma descritiva e de maneira conceitual, que será utilizado em um sistema.

O MER a seguir, representando o Banco de dados do Sistema proposto no trabalho, foi utilizado principalmente na hora da criação dos models[2], pois como foi visto no item 2.2.2, o Django utiliza do MVC. As tabelas e seus respectivos atributos foram criando como models no sistema e criados pelo Django a partir das especificações definidas no arquivo models.py.

Segue abaixo o Modele Entidade-Relacionamento do banco de dados do Sistema de controle de Ofertas.

[2] Um model é a fonte definitiva de dados no Django. Ele contém os campos e comportamentos essenciais dos dados do banco de dados. Geralmente, cada model representa uma tabela no banco de dados.

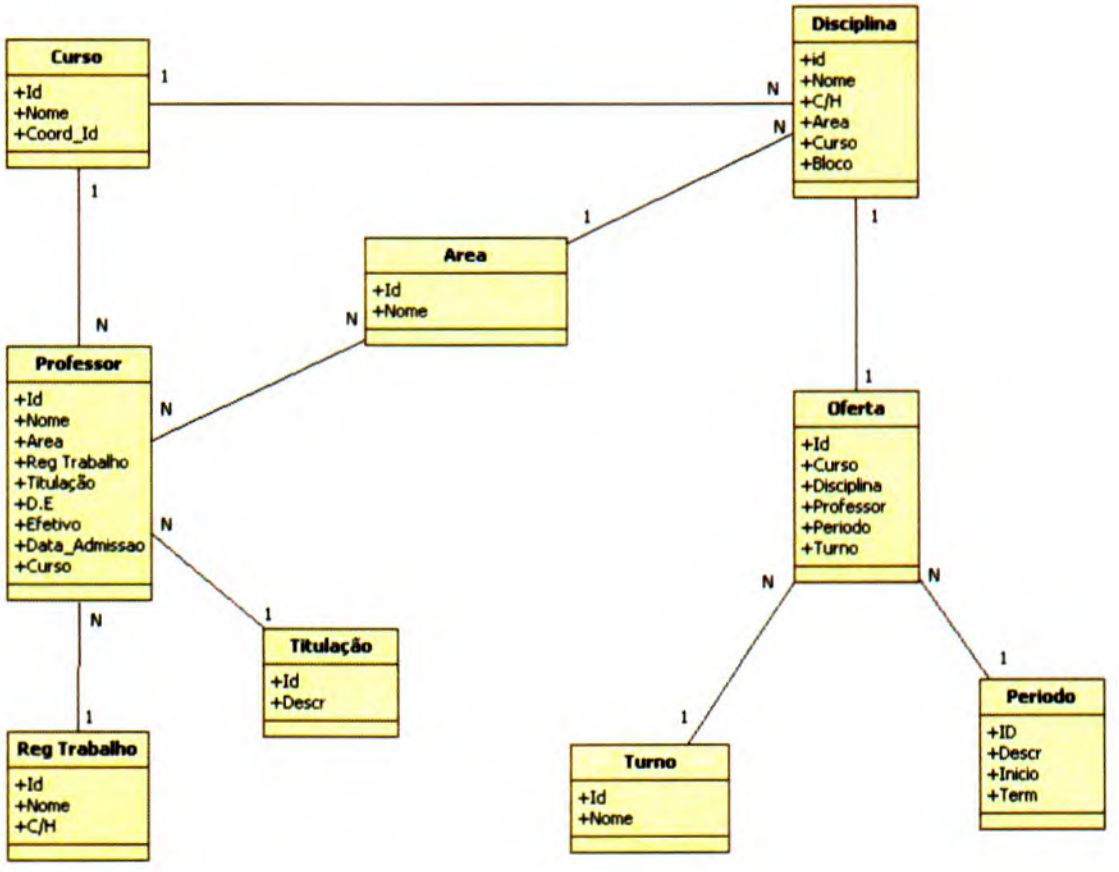

Figura 9: MER do Banco de Dados
Fonte: Primária

Dicionário de dados:

Curso: id=int; Nome=varchar(30); Coord_Id= int;

Professor: Id=int; Nome=varchar(30); D.E=boolean; Efetivo=boolean; Data_Admissao=date

Area: Id=int; Nome=varchar(15)

Reg_Trabalho: Id=int; Nome=varchar(10); C/H=int;

Titulação: Id=int; Descr=varchar(10)

Turno: Id=int; Nome=varchar(8)

Oferta: Id=int;

Periodo: ID=int; Descr=varchar(8); Inicio=date; Term=date

Disciplina: id=int; Nome=varchar(30); C/H=int; Bloco=int;

4 O SISTEMA DE CONTROLE DE OFERTAS DE DISCIPLINAS

Nesse capítulo, é mostrado o funcionamento e os detalhes de cada uma das páginas do sistema.

Ao entrar no sistema, o usuário é direcionado a página de autenticação. Todas as coordenações terão um "usuário" e "senha" que irão direcioná-los a página que contém as disciplinas do curso.

Figura 10: Tela de Login
Fonte: Primária

Uma vez autenticado, a página com todas as disciplinas cadastradas do curso do usuário (coordenador) é carregada. É nessa página que o coordenador irá escolher as disciplinas que serão ofertadas e, se desejar, o turno dessas ofertas. Há uma caixa de seleção (*checkbox*) ao lado de cada disciplina, e uma em cima das disciplinas ("Marcar Todas"), ao lado do Bloco. Também há outras informações da disciplina como carga horária e a área de ensino da disciplina. Abaixo, duas figuras das páginas: uma de disciplinas do curso e uma mostrando as disciplinas já selecionadas:

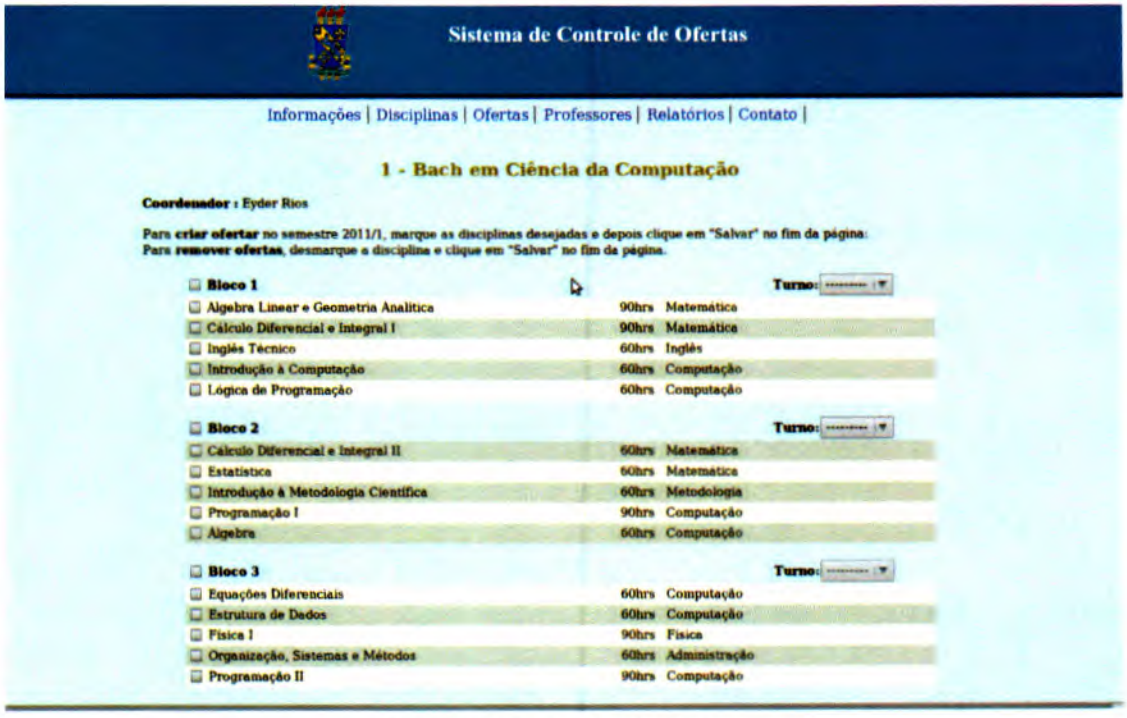

Figura 11: Tela de Disciplinas
Fonte: Primária

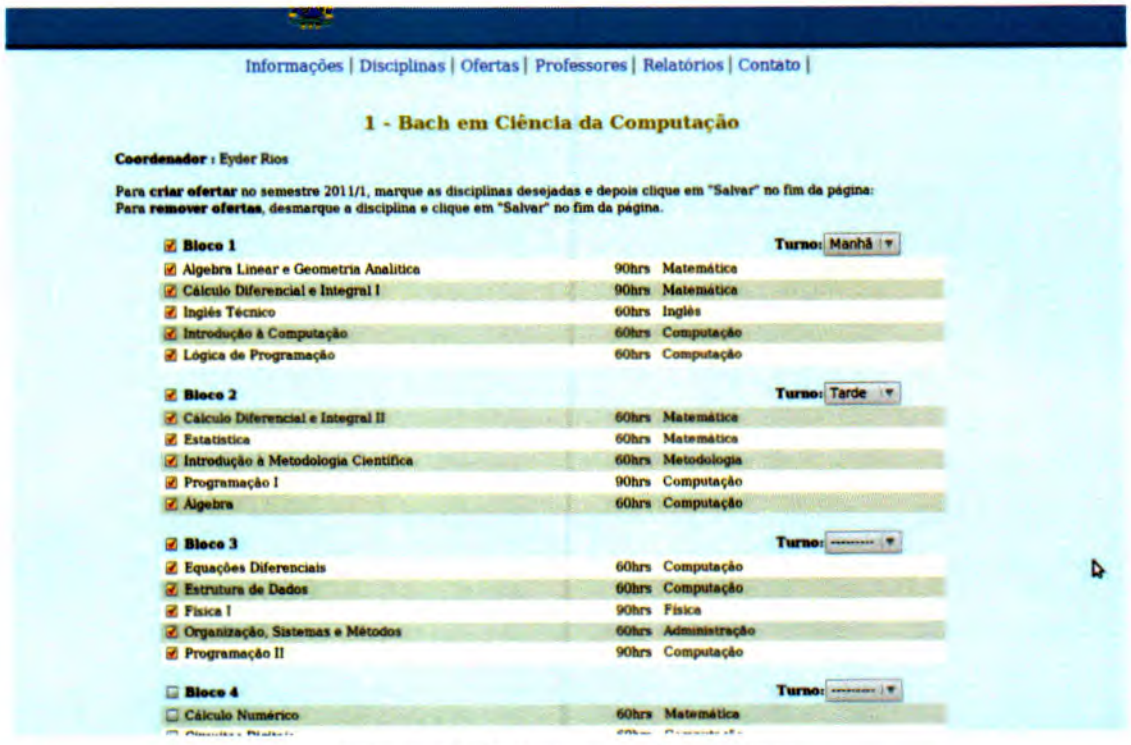

Figura 12: Selecionando Disciplinas
Fonte: Primária

Após selecionar as disciplinas e o turno (que é opcional), o usuário deverá ir no final na página e clicar no botão salvar. As alterações são serão consideradas se salvas.

Figura 13: Salvando Ofertas
Fonte: Primária

Para excluir ofertas já criadas, o processo é o inverso: o usuário desmarca as ofertas que deseja excluir e então clica em salvar. Uma vez que o usuário clica em "salvar", o sistema carrega a página de ofertas do curso. Nela é possível adicionar ou excluir vínculos dos professores e alterar (ou adicionar, caso não tenha sido escolhido nenhum no ato de criação da oferta) turnos. Tanto na coluna "Turno" quanto na coluna "Professor", há um link informativo que leva a uma janela pop-up de edição desses campos da oferta.

Figura 14: Tela de Ofertas
Fonte: Primária

Ao clicar no link da coluna "Professor", uma janela pop-up é aberta e exibe uma lista de professores. O usuário pode escolher em visualizar os "Todos os professores", "Professores da Área da Disciplina", "Professores Lotados no Curso". Uma pequena caixa com informações dos regimes de trabalho e suas respectivas cargas horárias está disponível na parte superior direita da pop-up. Na coluna "Horas de Aula" contém a quantidade de horas/aula que o professor já tem. Ao lado do professor há um *tooltip*[3] informativo representador por "(?)", que exibe informações do professor correspondente, como o curso em que foi lotado, suas áreas de atuação e titulação. Uma vez que o usuário decide qual será o professor da disciplina, ele deve clicar em "Vincular" correspondente ao professor escolhido.

Para alterar um professor que já está vinculado, a tela e o processo é o mesmo.

[3] Tooltip (terminologia ou dica de contexto, em inglês) é uma moldura pop-up que abre quando um usuário passa o mouse sobre um elemento HTML contendo normalmente uma informação adicional sobre o mesmo.

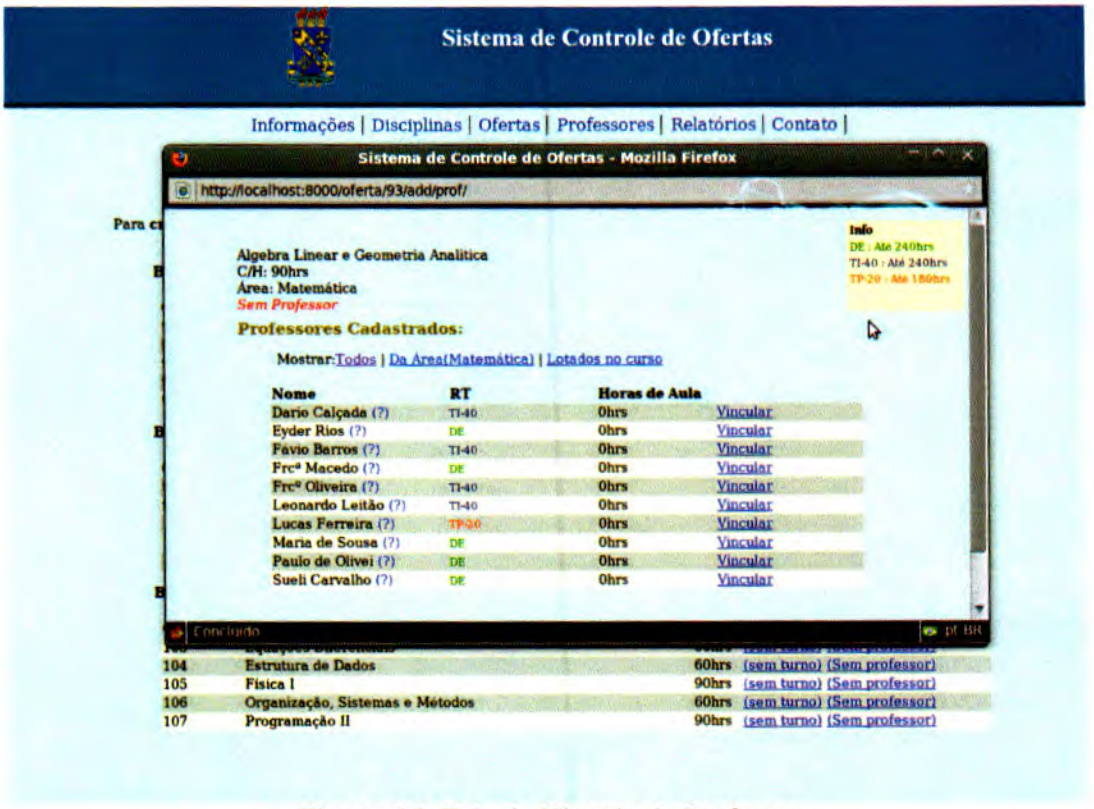

Figura 15: Tela de Vinculo de Professor
Fonte: Primária

Ao clicar no link da coluna "Turno", outra janela pop-up abre contendo os turnos cadastrados, como visto a seguir:

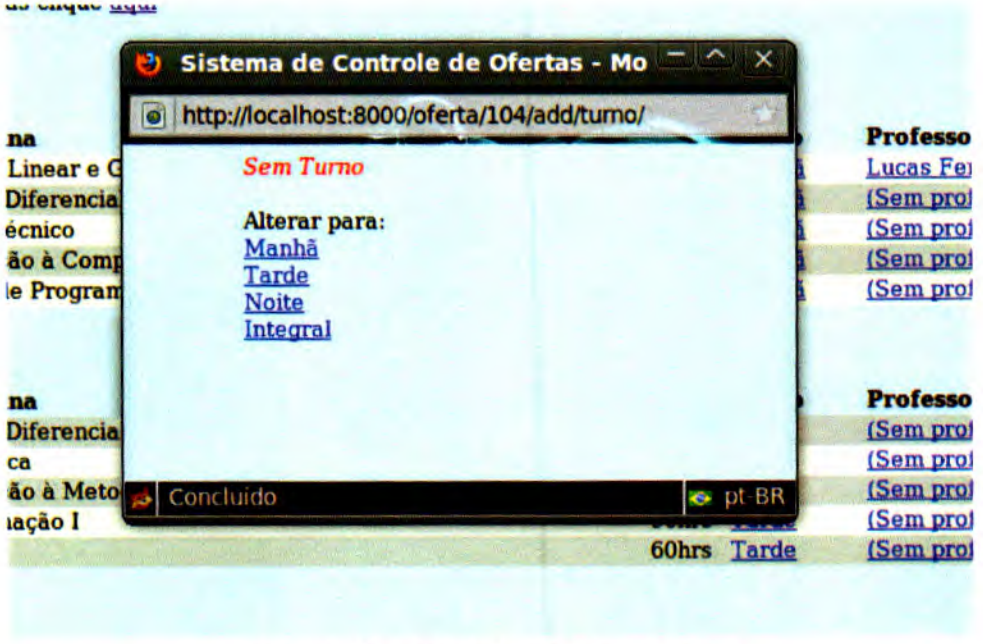

Figura 16: Tela de Definição de Turno
Fonte: Primária

No menu superior do sistema contém outras opções. Na página de "Professores", contém as opções de buscar "professor pelo nome", "mostrar todos os professores de um curso" ou "todos os professores da área". A tela da página "Professores" é exibida a seguir:

Figura 17: Página de Professores
Fonte: Primária

Ao fazer uma busca por nome ou escolher visualizar professores por curso ou área, é carregada uma página de resultados, reproduzida na figura 18 abaixo:

Figura 18: Resultado da Busca de Professores
Fonte: Primária

Ao clicar em um professor, o sistema exibe uma página com as informações do professor, como áreas de atuação, regime de trabalho, titulação, data de admissão e o curso em que foi lotado. Também é possível visualizar as disciplinas que ele ministra no semestre, como mostra a figura abaixo:

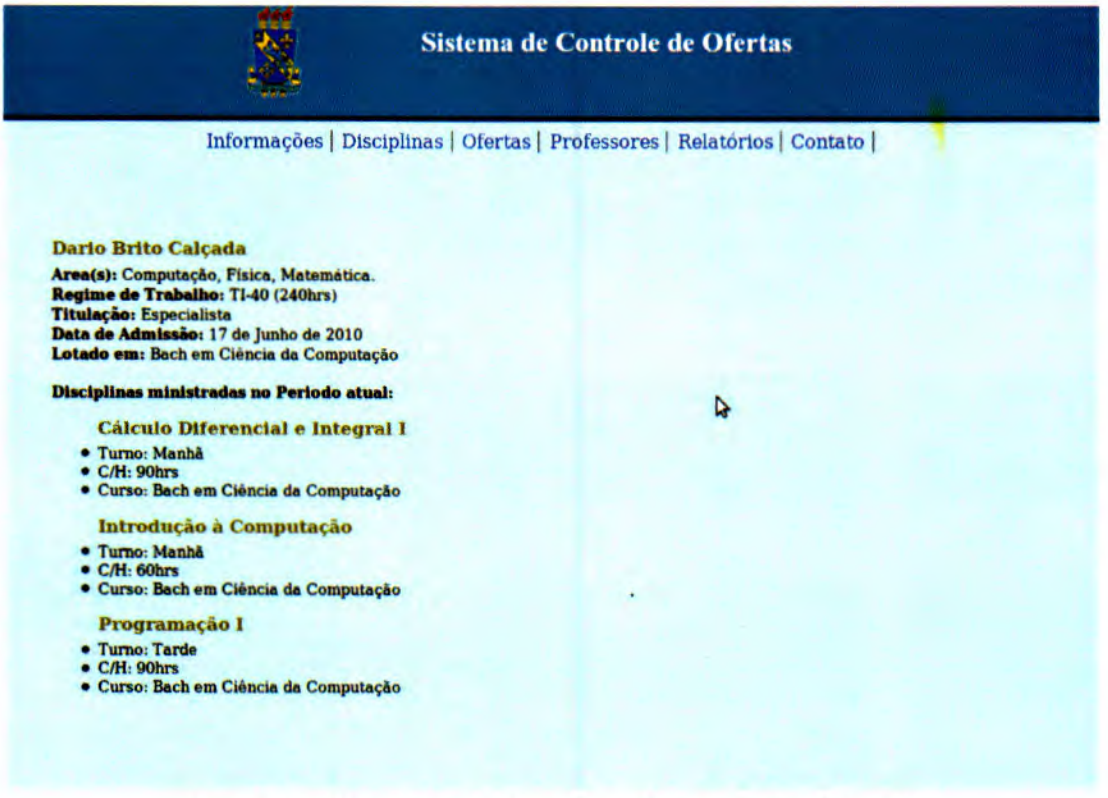

Figura 19: Página de detalhes de Professor *Fonte: Primária*

No item "Informações", é possível visualizar informações do período corrente.

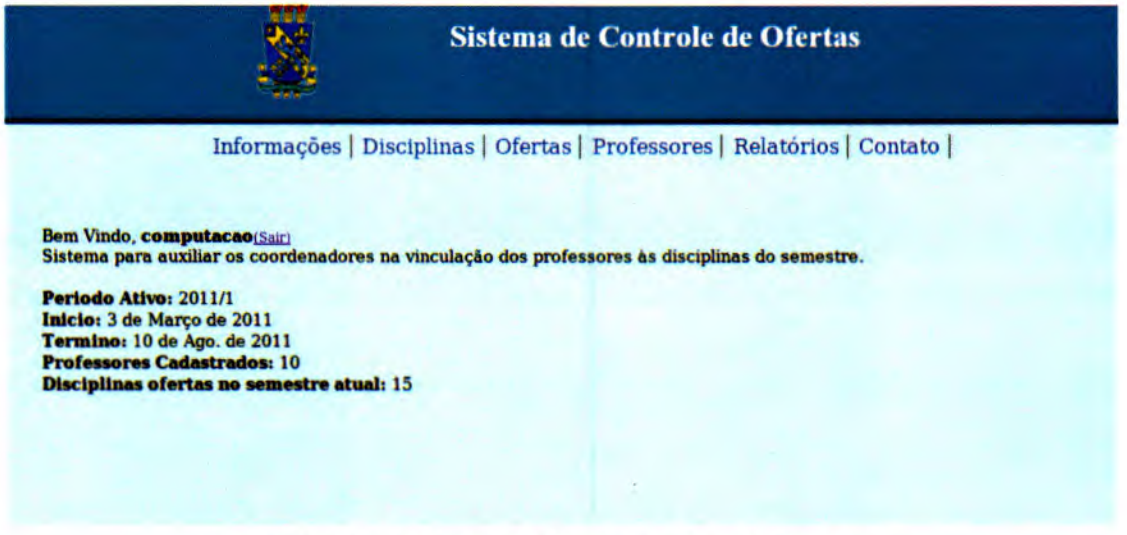

Figura 20: Página de Informações *Fonte: Primária*

O sistema fornece dois tipos de relatórios (que pode ser acessado no menu, no item "Relatórios"): Relatório de todas as ofertas do curso no semestre e Relatório de todas as ofertas dos professores lotados no curso. Vejamos a tela na figura a seguir:

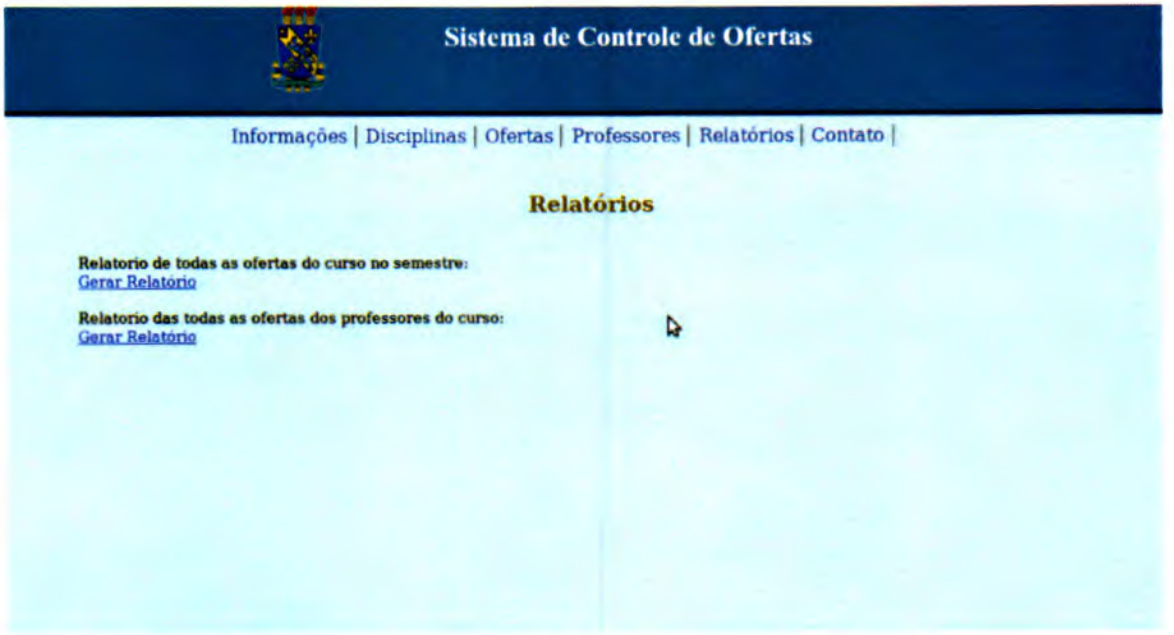

Figura 21: Página de Relatórios *Fonte: Primária*

Um exemplo de cada um dos relatórios está disponível nos apêndices.

5 CONSIDERAÇÕES FINAIS

Este trabalho mostrou as etapas e conceitos para a implementação de um sistema web de controle de oferta de disciplina para a UESPI - Campus Parnaíba. No trabalho, foram mostradas as ferramentas de programação, o conceito de sistemas web, a metodologia de desenvolvimento e a análise do sistema.

Em relação às ferramentas de programação, as escolhas se mostraram satisfatórias. O framework Django mostrou-se bastante eficiente permitindo uma implementação ágil e organizada. Além disso, a ferramenta oferece diversas soluções de implementação já disponíveis como, por exemplo, controle de acesso e página de administração, todas elas editáveis, deixando assim o programador livre para fazer as alterações que julgar necessárias. A linguagem Python, se mostrou poderosa e de fácil entendimento e aprendizado, com uma documentação extensa e detalhada. O PostgreSQL, já consolidado no mercado, se mostrou mais do que suficiente para suprir as necessidade do sistema.

No tocante ao sistema proposto, este foi desenvolvido atendendo todos os requisitos inicialmente solicitados. O fato de ter sido desenvolvido para web acaba agregando-lhe valor, pois facilita a manutenção e possibilita sua utilização em qualquer lugar que tenha acesso à internet.

Como recomendação para trabalhos futuros a esse TCC, tem-se o desenvolvimento e/ou a integração de uma aplicação que permita que os próprios alunos possam planejar suas matrículas nas disciplinas ofertadas criadas pelos coordenadores. Nesse caso, a utilização do Framework Django é aconselhada, pois a capacidade do Django de criar aplicativos plugáveis é um dos pontos fortes da ferramenta.

REFERÊNCIAS BIBLIOGRÁFICAS

ANICETO, Jefferson. **Aplicações Web.** Apostila ASP.net. Escola Técnica da Univale (ETEIT). 2009.

BECK, K., **Extreme Programming Explained: Embrace Change**, 1st Edition, Addison-Wesley, 1999.

BORGES, Luis Eduardo. **Python para desenvolvedores.** 2ed. Rio de Janeiro: O'Reilly, 2010.

BRANDÃO, José Mário Neiva. **Aprendendo Django no Planeta Terra.** Disponível em: <www.aprendendodjango.com>. Acesso em: 29 de Dezembro de 2010.

_____, **Django Brasil,** http://www.djangobrasil.org/ Acesso em: 02 de Janeiro de 2011.

_____, **Python v2.7.1 documentation,** http://docs.python.org/faq/general#why-was-python-created-in-the-first-place Acesso em: 27 de Novembro de 2010.

FOWLER, M., BECK, K., **Agile Manifesto,** Disponível em: http://www.agilemanifesto.org>. Acesso em 7 de Janeiro de 2011

KAPPEL, Gerti; PR¨OLL, Birgit; REICH, Siegfried; RETSCHITZEGGER, Werner. **Web Engineering The Discipline of Systematic Development of Web Applications.** John Wiley & Sons Ltd., 2006.

LUTZ Mark; ASCHER David. **Aprendendo Python;** Tradução João Tortello. 2.ed. Porto Alegre: Bookman, 2007.

MANHÃES, Vinícius Teles. **Extreme Programming - Aprenda como encantar seus usuárlos desenvolvendo software com agilidade e alta qualidade,** Rio de Janeiro: Novatec Editora, 2004.

MENDES, Alexandre. **TIC – Muita gente está comentando, mas você sahe o que é?** Disponível em: <http://imasters.uol.com.br/artigo/8278>. Acesso em: 09 de fevereiro de 2011.

NAPHTA, disponível em <http://www.naphta.com.br/xpmanager/xpmanager_metodologiasageis.html>. Acesso em 21 de Janeiro de 2011.

MILANI, André. **PostgreSQL - Guia do Programador**;1ª ed. Novatec, 2008.

PEREIRA, Felipe Luiz. CPITIL: **Uma aplicação de apoio ao gerenciamento de problemas baseado na recomendação ITIL**. Trabalho de Conclusão de Curso (Graduação em Ciências da Computação) - Centro de Ciências Tecnológicas da Terra e do Mar, Universidade do Vale do Itajaí, Itajaí, 2007.

_____, **PostgreSQL 8.4.6 Documentation,** disponível em: **<http://wysanti.postgresql.org/files/documentation/pdf/8.4/postgresql-8.4.6-US.pdf>.** Acesso em 20 de Janeiro de 2011.

PRESSMAN, R. S., **Engenharia de Software,** 6ª Ed.,Makron Books, 2006.

SANDRA; ROSÂNGELA; JÚNIA; SÍLVIA. **Engenharia para Web.** Universidade Federal de São Carlos (UFSCar). Projeto e Gerência de Sistemas de Software. Interface Homem Máquina. 2002.

SOARES, Michel dos S. disponível em <http://www.dcc.ufla.br/infocomp/artigos/v3.2/art02.pdf >, Acesso em 11 de Janeiro de 2011.

SOMMERVILLE, I., **Software Engineering,** 6ª Edition, Addison-Wesley, 2004, 1ª reimpressão.

SUH, Woojong. **Web Engineering Principles and Techniques**. IGI Publishing, 2005.

_____, **The Django Book.** Disponível em: <http://www.djangobook.com/>. Acesso em: 02 de Janeiro de 2011.

WELLS, D., Disponível em <http://www.extremeprogramming.org.> Acesso em: 13 de Janeiro de 2011.

WORLD WIDE WEB CONSORTIUM, **HTTP 1.1 Specification.** Estados Unidos, 1999. Disponível em:<http://www.w3.org/protocols>. Acesso em: 02 de fevereiro de 2011.