UNIVERSIDADE FEDERAL DE CAMPINA GRANDE

CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA

DEPARTAMENTO DE SISTEMAS E COMPUTAÇÃO

RELATÓRIO DE ESTÁGIO

SIGCENTER – SISTEMA PARA IMPLANTAÇÃO E MANUTENÇÃO DE TELECENTROS

FELIPE LEAL COUTINHO
Estagiário

HYGGO OLIVEIRA DE ALMEIDA
Orientador Acadêmico

KYLLER COSTA GORGÔNIO
Supervisor Técnico

Campina Grande – PB

Dezembro de 2009

SIGCENTER – SISTEMA PARA IMPLANTAÇÃO E MANUTENÇÃO DE TELECENTROS

APROVADO EM _____

BANCA EXAMINADORA

_____

Prof. Dr. Hyggo Oliveira de Almeida

ORIENTADOR ACADÊMICO

_____

Prof[a]. Dr[a]. Joseana Macêdo Fechine

MEMBRO DA BANCA

_____

Prof. Me. Camilo de Lélis Gondim Medeiros

MEMBRO DA BANCA

Biblioteca Setorial do CDSA. Maio de 2021.

Sumé - PB

AGRADECIMENTOS

Gostaria de agradecer a todos os sócios da Signove e ao professor Hyggo Almeida por me darem essa oportunidade. Em especial, gostaria de agradecer a Kyller, Marcos Fábio, Felipe, Cristóvão, Andressa e a todos os colegas de trabalho pelos ensinamentos passados e pelas experiências compartilhadas.

APRESENTAÇÃO

Como parte das exigências do curso de Ciência da Computação, da Universidade Federal de Campina Grande, para cumprimento da disciplina de estágio integrado, apresenta-se o relatório de estágio de Felipe Leal Coutinho, cujo foco foi o desenvolvimento de uma aplicação para implantação e manutenção de telecentros.

O estágio foi realizado na Signove S/A, empresa fundada a partir de um *spin-off* acadêmico.

O conteúdo do relatório está distribuído conforme descrição a seguir:

Seção 1 – Introdução.

Seção 2 – Ambiente de Estágio.

Seção 3 – Fundamentação Teórica e Tecnologias Utilizadas.

Seção 4 – Atividades do Estágio.

Seção 5 – Considerações Finais.

Referências Bibliográficas.

Apêndices.

RESUMO

O estágio foi realizado na Signove S/A e teve como objetivo o desenvolvimento de um sistema Web que possibilite a criação e manutenção sustentável de telecentros pela exibição de propagandas. As atividades do estágio consistiam na criação do sistema para gerenciamento de telecentros e propagandas, do sistema de cobrança de propagandas e do sistema de exibição dessas propagandas nos telecentros cadastrados.

SUMÁRIO

LISTA DE SIGLAS E ABREVIATURAS

AJAX ..... *Asynchronous JavaScript and XML*
API ..... Application Programming Interface
CRUD ..... *Create, Read, Update and Delete*
HTML ..... *Hypertext Markup Language*
i18n ..... Internacionalização
JSON ..... *JavaScript Object Notation*
JVM ..... Java *Virtual Machine*
LBS ..... *location-based service*
RSS ..... *Really Simple Syndication*
SOAP ..... *Simple Object Access Protocol*
TLC ..... Telecentro
US ..... *User Story*
XML ..... *Extensible Markup Language*
ORM ..... *Object-Relational Mapping*

LISTA DE FIGURAS

LISTA DE QUADROS

SEÇÃO I

INTRODUÇÃO

1. INTRODUÇÃO

Com a crescente disseminação das tecnologias da informação, vê-se aumentar o interesse das forças políticas e econômicas em promover a inclusão digital entre as camadas mais pobres da sociedade. Para que isto ocorra, é necessária uma solução sustentável que permita fácil implantação e manutenção de centros de inclusão digital, os chamados telecentros (TLC). A aplicação desenvolvida nesse estágio, o SigCenter, é uma proposta para tal solução sustentável, que utiliza clientes leves para baratear a implantação e propagandas para financiar a manutenção dos telecentros.

O SigCenter é composto por um servidor central, em que serão cadastrados os TLC de acordo com o posicionamento geográfico, os clientes, as propagandas por TLC e, por fim, gerará a conta a ser paga por cada cliente. Além disso, cada TLC terá um servidor leve, em que serão armazenados os dados do TLC, como, por exemplo, as informações das contas dos usuários. Por fim, o TLC será composto por vários clientes leves, que são os computadores que os usuários vão utilizar. Nos clientes leves serão implantados programas exibidores de propagandas de modo a exibir as propagandas cadastradas pelos clientes do SigCenter.

O objetivo desse estágio foi o desenvolvimento de um protótipo do SigCenter. Entre os objetivos específicos, pode-se citar a implementação do servidor central e a implementação do cliente exibidor de propagandas.

Neste relatório, são apresentadas as atividades realizadas no estágio entre Agosto e Dezembro de 2009 na empresa Signove[1]. A carga horária de trabalho foi de 30 horas semanais pelo período de cinco meses, cobrindo assim a carga horária mínima de 300 horas exigidas pela disciplina de Estágio Integrado da UFCG.

Por fim, pode-se destacar a importância da disciplina de Estágio Integrado para o currículo e a formação profissional de todos os estudantes de computação, pois possibilita a aplicação dos conceitos estudados em sala de

[1] http://www.signove.com/

aula e a vivência de um ambiente empresarial real. Particularmente, esse estágio possibilitou o aprendizado de diversas novas tecnologias e a troca de experiências.

SEÇÃO II

AMBIENTE DE ESTÁGIO

2. AMBIENTE DE ESTÁGIO

A Signove S/A é uma empresa formada a partir de um *spin-off* acadêmico. A maioria dos sócios eram alunos de Mestrado e Doutorado da UFCG, que trabalhavam no Laboratório de Sistemas Embarcados e Computação Pervasiva – Embedded[2].

Basicamente, a Signove está dividida em dois setores: o setor administrativo e o setor de desenvolvimento. O setor administrativo é composto por Kyller Gorgônio, Adma Costa e Marcos Procópio. O setor de desenvolvimento é composto por Kyller, pelos outros sócios, pelos estagiários e pelos bolsistas. Atualmente, a Signove conta com mais de 20 pessoas no setor de desenvolvimento.

Na Figura 1 pode-se ver a sala onde o desenvolvimento dos produtos da Signove é feito.

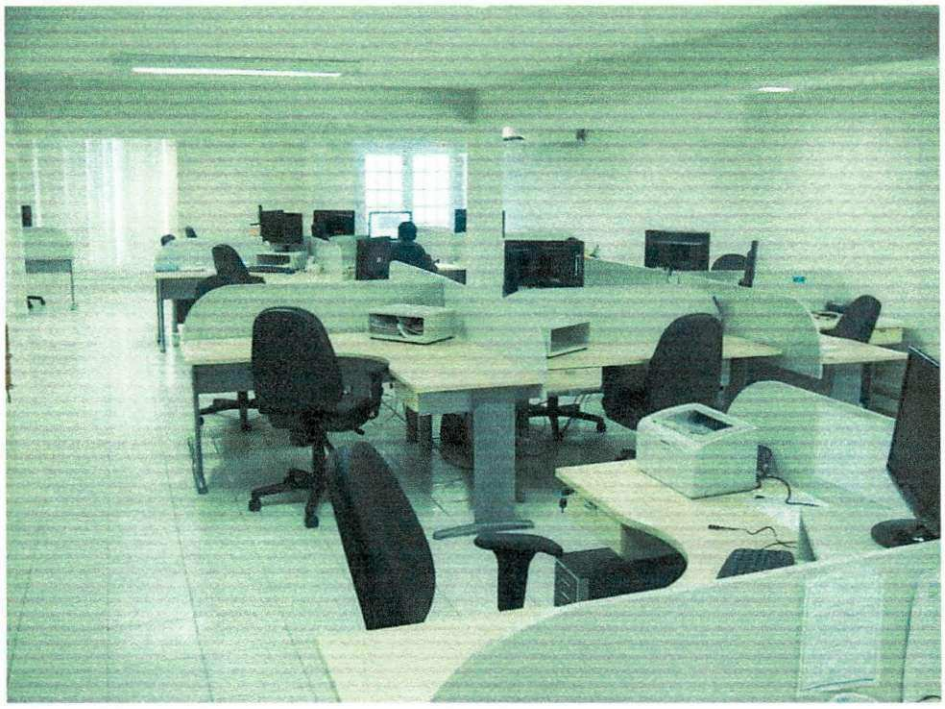

Figura 1 - Sala de Desenvolvimento

[2] http://www.embedded.ufcg.edu.br/

2.1. *Infraestrutura*

A Signove tem uma grande preocupação em relação às licenças de software. Por isso, ela tem as licenças de todos os softwares comerciais que ela utiliza.

No projeto foram utilizados os seguintes softwares: Eclipse[3], Grails [SPRINGSOURCE, 2009], Java[4], Ubuntu[5], API (*Application Programming Interface*) do Google Maps [GOOGLE, 2009], SVN[6] e PostgreSQL[7].

Quando ao hardware utilizado, o projeto dispunha de 4 computadores para desenvolvimento (um por desenvolvedor) com a seguinte configuração:

- Intel core 2 duo E7400 2.80GHz
- DDRII 4 GB de memória principal
- 320 GB de memória secundária
- Placa de rede de 10/100/1000 Mbps

Pode-se ver na Figura 2 a estação de trabalho utilizada pelo estagiário.

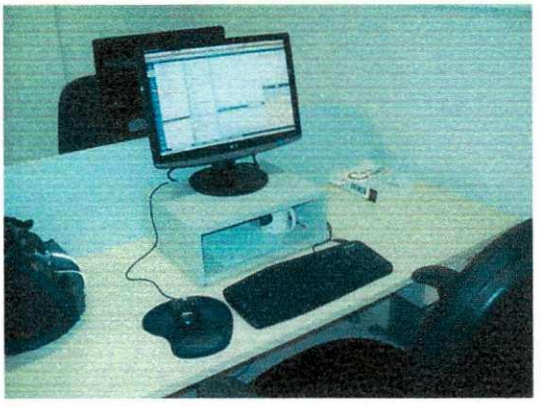

Figura 2 - Estação de Trabalho

---

[3] http://www.eclipse.org/

[4] http://java.sun.com/

[5] http://www.ubuntu-br.org/

[6] http://subversion.tigris.org/

[7] http://www.postgresql.org/

O projeto também dispunha de um cliente leve para os testes da distribuição linux embarcada. Por fim, também estava disponível um servidor para testes e implantação da aplicação. Ele era uma máquina virtual hospedada em um dos servidores da Signove, ao qual os desenvolvedores do projeto não tinham acesso.

2.2. *Equipe*

A equipe de desenvolvimento era composta por quatro desenvolvedores, sendo três bolsistas e um estagiário. Os membros da equipe eram:

- Andressa Bezerra: bolsista, cursa o 5º período do curso de Ciência da Computação da UFCG e trabalhou no desenvolvimento do servidor central;
- André Fiúza: bolsista, cursa o 7º período do curso de Engenharia Elétrica da UFCG e trabalhou no desenvolvimento de uma distribuição Linux personalizada para os clientes leves;
- Cristóvão Dutra: bolsista, cursa o 5º período do curso de Ciência da Computação da UFCG e trabalhou no desenvolvimento do servidor central e do exibidor de propagandas;
- Felipe Coutinho: estagiário, cursa o último período do curso de Ciência da Computação da UFCG e trabalhou no desenvolvimento do servidor central e do exibidor de propagandas.

2.3. *Orientador Acadêmico*

Nome: Hyggo Oliveira de Almeida

Endereço Profissional: Universidade Federal de Campina Grande, Departamento de Sistemas e Computação, Laboratório de Sistemas Embarcados e Computação Pervasiva – Embedded

Av. Aprígio Veloso, 882. CEP 58429-900, Bodocongó, Campina Grande, PB, Brasil.

Telefone: (83) 3310-1692

E-mail: hyggo@dsc.ufcg.edu.br

2.4. Supervisor Técnico

Nome: Kyller Costa Gorgônio

Endereço Profissional: Signove S/A - R. Dom Pedro II, 675, Prata, CEP 58400-565, Campina Grande, Paraíba, Brasil.

Telefone: (83) 3321-6367

E-mail: kyller@signove.com

SEÇÃO III

FUNDAMENTAÇÃO TEÓRICA

3. FUNDAMENTAÇÃO TEÓRICA

Nesse projeto, foi possível aprofundar o contato com programação para Web, mas também com sistemas baseados em localização, hardware, etc. Logo, isso possibilitou o aprendizado de diversos novos conceitos.

3.1. Grails

Grails é um *framework open source* para desenvolvimento web. Ele possibilita a criação de aplicações web de modo fácil e ágil.

Grails é similar em alguns pontos a Ruby on Rails[8], porém utiliza Groovy [CODEHAUS FOUNDATION, 2009] e Java como linguagens de programação ao invés de Ruby[9].

3.2. Groovy

Groovy é uma linguagem dinâmica que é interpretada pela máquina virtual de Java (JVM; do inglês Java *Virtual Machine*) e tem uma sintaxe similar a Java. Ela é produtiva como Ruby, mas leva algumas vantagens devido à integração com a plataforma Java.

3.3. Web Services

Web services é, atualmente, uma das soluções mais utilizadas para interligar diferentes sistemas, escritos em linguagens diferentes e até

---

[8] http://rubyonrails.org/

[9] http://www.ruby-lang.org/

executando em plataformas diferentes. No caso, as aplicações se comunicam a partir da troca de mensagens XML[10] (*eXtensible Markup Language*).

3.4. Desenvolvimento Web

O desenvolvimento web por meio de Grails foi complementado, quando necessário, com a utilização da linguagem de programação JavaScript[11] e AJAX[12] (*Asynchronous Javascript And XML*) para criar páginas mais dinâmicas, interativas e atrativas para os usuários.

Deve-se observar também que o mecanismo de *plugins*[13] do Grails auxiliou no desenvolvimento, pois traz soluções para as mais diversas situações como, por exemplo, implementação de mecanismos de autenticação e autorização.

3.5. Sistemas Baseados em Localização

O SigCenter faz uso de sistema de propagandas baseado na localização geográfica dos TLC. Dessa forma, foi necessário o uso de uma API de localização geográfica. A API escolhida foi a do Google Maps.

Por meio da utilização de JavaScript, a API do Google Maps possibilita a inserção de mapas nas páginas HTML [REFSNES DATA, 2009] (*HyperText Markup Language*), assim como a marcação de pontos de interesse no mapa, representando as localizações dos TLC.

---

[10] http://www.w3.org/XML/

[11] http://en.wikipedia.org/wiki/Javascript

[12] http://en.wikipedia.org/wiki/Ajax_(programming)

[13] http://www.grails.org/plugin/home

3.6. Selenium

Selenium [OPENQA, 2009] é um *framework* para realizar testes de interface em páginas web. Ele realiza quase todas as ações que um usuário normal poderia realizar nas páginas web, como clicar em um link, digitar textos em caixas de texto, etc. Dessa forma, é possível realizar testes completos de interface.

3.7. RSS

RSS[14] (*Really Simple Syndication*) é um formato de arquivo XML que serve para agregar conteúdo, sendo geralmente utilizado em sites de notícias.

No SigCenter, o RSS foi utilizado para tornar disponível para os clientes de um TLC as propagandas a ele associadas. Para gerar o RSS das propagandas associadas a um TLC, foi utilizado um *plugin*[15] do Grails para geração desses tipos de arquivos.

3.8. Mapeamento Objeto Relacional

O Grails já oferece integração ao *framework* Hibernate[16], ferramenta que realiza o mapeamento objeto relacional (ORM - Object-Relational Mapping) em Java, provendo suporte a diversos bancos de dados, como o Oracle[17], MySQL,[18] PostgreSQL, etc. Além disso, Grails torna ainda mais fácil a

---

[14] http://pt.wikipedia.org/wiki/RSS

[15] http://www.grails.org/Feeds+Plugin

[16] https://www.hibernate.org/

[17] http://www.oracle.com/br/index.htm

[18] http://www.mysql.com/

utilização de Hibernate já que abstrai as configurações necessárias para utilizar o *framework* com Java.

3.9. Desenvolvimento Desktop com QT e C++

Para o desenvolvimento para *desktop* foi utilizada programação em C++ junto com o *framework* QT [NOKIA CORPORATION, 2009]. QT é uma aplicação multiplataforma e um *framework* para desenvolvimento de interface gráfica que permite que o mesmo código fonte feito com QT e C++ execute em diversos ambientes diferentes.

3.10. SCRUM

A maioria dos projetos desenvolvidos pela Signove utilizam SCRUM [SCRUM ALLIANCE, 2009], assim como o SigCenter. SCRUM é um processo iterativo incremental para desenvolvimento ágil de projetos. Os três atores principais envolvidos no processo são:

- **Product Owner:** É quem responde pelo cliente do projeto e quem define o que deve ser implementado.
- **Scrum Master:** Responsável por acompanhar a equipe, resolver impedimentos ao avanço do trabalho, manter a produtividade.
- **Team:** São os desenvolvedores do projeto. Eles têm autonomia para organizar e distribuir atividades entre seus membros.

Em SCRUM, o *product backlog* é um documento que contém as funcionalidades a serem desenvolvidas ao longo do projeto. Os *sprints* (outro nome para iterações) têm duração de duas a quatro semanas. Em cada começo de *sprint* é feita uma reunião para escolher que itens do *product backlog* que devem ser implementados naquela iteração. Ao final de cada sprint, se tem uma versão incrementada do produto. O cliente deve ver essa

versão do produto e dizer se o *backlog* para aquele *sprint* foi cumprido a seu ver.

Uma das grandes vantagens do SCRUM é a reunião diária. Essas reuniões diárias são de curta duração (no máximo 15 minutos) e permitem que os membros da equipe discutam o que fizeram, o que ainda falta fazer e as dificuldades encontradas ao longo do desenvolvimento.

SEÇÃO IV

ATIVIDADES REALIZADAS

4. ATIVIDADES REALIZADAS

Neste capítulo, encontram-se as atividades realizadas no período de estágio. Para facilitar o entendimento, elas serão divididas por *user stories*. É importante observar, que aqui serão descritas somente as atividades realizadas pelo estagiário. Porém, cada *user story* (US) pode ter tido mais atividades, as quais foram realizadas por outros membros da equipe.

Agrupando o cronograma das atividades realizadas pelas *user stories* tem-se o Quadro 1.

Quadro 1 - Cronograma de Atividades

US	AGO				SET				OUT				NOV				DEZ			
4.1	X	X																		
4.2		X	X																	
4.3				X	X															
4.4						X	X													
4.5								X												
4.6								X												
4.8									X	X										
4.9											X	X								
4.10												X	X							
4.11														X	X					
Elaboração do relatório									X	X	X	X	X	X	X	X				
Elaboração da defesa																X	X			

É importante observar, que o cronograma final exibe muitas diferenças em relação ao cronograma do plano de estágio. Vide Quadro 2. Isso se deve ao fato do cronograma inicial ter sido elaborado sem o contato com o *backlog*

do projeto. O *backlog* só foi apresentado ao time de desenvolvimento na primeira reunião de planejamento, que só ocorreu na metade do mês de Agosto.

Quadro 2 - Cronograma de Atividades do Plano de Trabalho

Atividade	AGO	SET	OUT	NOV	DEZ
Estudo das tecnologias	X				
Desenvolvimento do sistema de gerenciamento do TLC	X	X			
Desenvolvimento do sistema de propagandas		X	X		
Desenvolvimento da distribuição Linux			X	X	
Testes				X	X
Elaboração do relatório					X

Os detalhes de cada atividade realizada estão agrupados por *user stories* e são apresentados nas subseções seguintes.

4.1. Criar aplicação Web no Grails

4.1.1. Configurar Ambiente de Desenvolvimento

Esta atividade inclui a instalação e configuração de todo o ambiente de desenvolvimento inicial. Ou seja:

- Instalação das máquinas do projeto;
- Instalação do Ubuntu 9.04
- Instalação e configuração do banco de dados PostgreSQL;
- Instalação do JDK 6.0;
- Instalação e configuração do Grails;

- Instalação do eclipse e dos *plugins* para trabalhar com SVN[19], Groovy[20] e Grails[21];
- Configuração do servidor de testes.

4.1.2. Estudar Grails

Esta atividade se iniciou nessa US, porém se entendeu por várias outras quando necessário. Nos estudos foram utilizados um livro [DICKINSON, 2009], a própria documentação de Grails[22] (que é bem completa) e tutoriais da internet.

4.1.3. Criar Aplicação SigCenter

Constituiu em criar uma aplicação simples no Grails, ou seja, sem funcionalidades. Ela seria o ponto de partida para o desenvolvimento do SigCenter.

Essa aplicação foi colocada no SVN da Signove para garantir *backup*, controle de versões e facilitar o trabalho em equipe. Além disso, também foi feito o *deploy* dela no servidor para validar a configuração do mesmo.

4.2. Implementar Suporte à Internacionalização

4.2.1. Estudar Processo de Internacionalização no Grails

---

[19] http://subclipse.tigris.org/servlets/ProjectProcess?pageID=p4wYuA

[20] http://groovy.codehaus.org/Eclipse+Plugin

[21] http://www.grails.org/Eclipse+IDE+Integration

[22] http://www.grails.org/Documentation

Como o Grails já coloca automaticamente a aplicação para utilizar internacionalização (i18n; do inglês *internationalization*), só foi necessário estudar como criar os textos internacionalizados. Para tal, utilizou-se a documentação do Grails para i18n[23].

4.3. *Implementar Autenticação e CRUD de Usuários Baseada no SigAccounts*

O SigAccounts é uma aplicação da Signove que tem a intenção de gerenciar as contas dos usuários de todas as aplicações da Signove. Ela implementa um modelo parecido com o do Google Accounts[24], em que o usuário tem um único login e senha para todas as aplicações do Google.

4.3.1. Implementar CRUD de Usuários no SigCenter

Criação do CRUD (*Create, Read, Update and Delete*) de Usuários do SigCenter integrado ao SigAccounts através de SOAP[25] (*Simple Object Access Protocol*) *web service*. No banco de dados do SigCenter ficaram armazenados somente as informações do usuário exclusivas do SigCenter. Informações como login e senha, por exemplo, são armazenadas no SigAccounts, pois são comuns a todas as aplicações.

4.3.2. Implementar Autenticação de Usuários no SigCenter

Para realizar a autenticação, foi necessário implementar a integração do SigCenter com o módulo do SigAccounts que faz a autenticação de um usuário para todas as aplicações da Signove.

---

[23] http://grails.org/Internationalization

[24] http://www.google.com/support/accounts/bin/answer.py?hl=en&answer=27439

[25] http://www.w3schools.com/soap/default.asp

4.4. *Implementar CRUD de TLC Ciente de Localização e Baseado no LBS do TaskIt*

O TaskIt[26] é outra aplicação da Signove. Ele utiliza um sistema baseado em localização geográfica (LBS; do inglês *location-based service*) para indicar pontos de interesse. Para isso, ele utiliza a API do Google Maps e programação em JavaScript.

Desse modo, esta US consistia em criar o CRUD de TLC para armazenar o nome do TLC, o telefone, o endereço e a localização geográfica dada através da API do Google Maps. Boa parte do que foi desenvolvido no TaskIt foi reutilizado nessa US.

4.4.1. Estudar API do Google Maps

Foi necessário, por parte dos integrantes da equipe, o estudo da API do Google Maps para saber como mostrar mapas nas páginas HTML e como marcar pontos neles representando os TLC.

4.4.2. Estudar Código do TaskIt

O estudo da forma como o TaskIt manipula os mapas permitiu uma economia de tempo, pois boa parte do código JavaScript dele foi reaproveitado no SigCenter.

4.4.3. Implementar CRUD de TLC

Implementação do CRUD de TLC incluindo a capacidade do usuário escolher o ponto do mapa em que seu TLC está localizado.

---

[26] http://www.taskit.com

4.5. Implementar CRUD de Propagandas e o Vínculo com TLC

Com a implementação dessa US foi possível criar propagandas no sistema e vincular a elas os TLC em que iriam ser exibidas. Uma propaganda é composta por título, link do RSS onde é possível encontrar os produtos da propaganda e o conjunto de TLC que a exibirão. Cada TLC pode exibir mais de uma propaganda.

Ao final dessa US, já era possível encolher quais os TLC em que uma propaganda iria ser apresentada.

4.5.1. Implementar CRUD de Propagandas

Foi implementado o CRUD de propagandas e feito a relação com os TLC. Ficou definida uma relação muitos para muitos entre propagandas e TLC.

4.6. Implementar Melhorias na UI de Propagandas

Para que o cliente que publicou sua propaganda no SigCenter escolha melhor os TLC que ela deva ser veiculada, foi criado a possibilidade dele escolher os TLC a partir do mapa.

Dessa forma, ele pode, por exemplo, escolher somente os TLC que se encontram mais perto do seu negócio.

4.6.1. Implementar Seleção por Clique

O usuário poderá clicar duas vezes em um TLC no mapa e ele ficará pré-selecionado. Essa atividade foi realiza utilizando os recursos da API do Google Maps.

4.6.2. Implementar Inserção de Marcos de Cores Diferentes no Mapa com Balão de Informações do TLC

Nessa atividade, foram inseridas cores diferentes para o marco de um TLC dependendo da sua seleção. Caso ele não esteja selecionado, o marco tem cor verde. Caso ele seja pré-selecionado, a cor muda para amarelo. Por fim, quando ele é selecionado para ser um dos TLC onde a propaganda será exibida a sua cor passa para vermelho.

O balão de informações do TLC foi inserido para facilitar a identificação do TLC pelo usuário.

A tela final da seleção do TLC em que uma propaganda será veiculada ficou como na Figura 3.

Figura 3 - Tela de Seleção do TLC

4.7. Implementar Gerador de RSS, por TLC, com Base nas Informações de Propagandas

Esta US consistia na criação de um gerador de RSS para gerar periodicamente um RSS por TLC contendo os itens de propagandas a serem veiculados naquele TLC. Essa US foi desenvolvida pelos outros membros da equipe.

4.8. Implementar Contador de Cliques nos Itens de Propaganda por TLC e por Propaganda

Cada vez que um item for clicado em um TLC, esse clique deverá ser contabilizado para o TLC que o exibiu e para a propaganda desse item. Dessa forma, o clique poderá gerar recursos para a manutenção do TLC e gerará um custo para o anunciante.

Além disso, a página do item clicado deverá abrir no navegador padrão.

4.8.1. Implementar Contador de Clique

O contador de cliques foi implementado de forma simples a partir de um contador inteiro em cada entidade que se desejava contar os cliques. Cada vez que o sistema recebe uma requisição HTTP dizendo que um item foi clicado, os respectivos contadores são incrementados.

4.8.2. Redirecionamento para Propaganda que Permita Contagem do Clique

Para permitir a contagem do clique e posterior redirecionamento para a página de propaganda, foi definido um formato para a url do item. Nos RSS gerados pelo SigCenter, o formato do link do item era o seguinte:

```
http://localhost:8080/sigcenter/restful/clicksCounter?
ad=1&center=2&item=http://path/to/item
```

Dessa forma, é possível identificar a qual propaganda o item pertence (através do parâmetro "ad"), de qual TLC veio o clique (parâmetro "center") e qual o endereço do item clicado (parâmetro "item"). Assim, é realizada a contagem e depois o SigCenter redireciona para a página de propaganda do item.

4.9. Implementar Web Service de View Counter

4.9.1. Estudar *Web Service* com REST

REST[27] foi escolhido para implementar o serviço de contagem de visualizações por ser leve e simples, ideal para integrar com aplicações feitas para dispositivos com recursos reduzidos.

Em serviços REST, as mensagens são trocadas utilizando o corpo de mensagens HTTP. Logo, optou-se por passar as mensagens do corpo no formato de troca de dados *JavaScript Object Notation*[28] (JSON). Esse é um formato leve, fácil de gerar, fácil de fazer *parser* e independente de linguagem.

4.9.2. Implementar *Action* de *View Counter*

Uma *action* em Grails nada mais é do que o código Java que recebe as requisições das páginas Web, as processa e responde para o navegador. Nesse caso, a *action* era quem fornecia o serviço REST.

---

[27] http://en.wikipedia.org/wiki/Representational_State_Transfer

[28] http://json.org/

Esse serviço é responsável por receber as informações dos TLC com as propagandas que foram exibidas neles. Dessa forma, é possível contabilizar as visualizações (além dos cliques) e cobrar pelo serviço.

Esse serviço recebe uma tabela no formato JSON indicando as informações das propagandas visualizadas e salvando no banco de dados do servidor para futuras consultas.

4.10. Desenvolver Leitor de RSS para o Cliente Leve

Segundo a arquitetura do sistema (ver Figura 4), um TLC é composto por vários clientes leves, que são computadores com recursos limitados. Esses computadores deverão ter um leitor de RSS responsável por baixar do servidor as propagandas a serem exibidas, exibi-las aos usuários do TLC e enviar periodicamente ao serviço *view counter* as atualizações. Esse leitor de RSS foi desenvolvido utilizando Qt e C++.

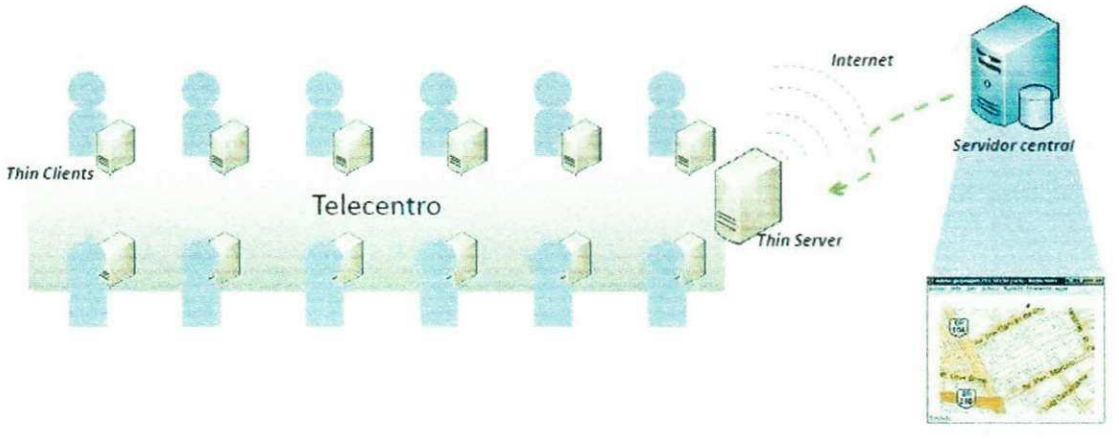

Figura 4 - Arquitetura do Sistema

4.10.1. Estudo das Linguagens e Ferramentas

Essa atividade envolveu uma revisão nos conceitos de programação em C++ (foi utilizado o livro C++ Primer [LIPPMAN; LAJOIE; MOO, 2005]), o

estudo de Qt e da ferramenta Qt Creator[29] (ambiente de desenvolvimento para Qt). Os estudos relacionados à Qt foram feitos a partir do próprio site e documentação[30] do *framework*.

Foram feitos pequenos exemplos de programas com interface gráfica para validar os estudos. Por exemplo, foi feito um programa leitor de RSS para assimilar os *widgets* de Qt estudados.

4.10.2. Implementar Leitor de RSS

Consistiu na criação do *parser* do RSS do TLC, na geração da interface gráfica que mostra periodicamente diferentes propagandas para o usuário e na implementação de um mecanismo que envia periodicamente as informações de visualização para o serviço *view counter*. Pode-se ver, na Figura 5, a tela final do leitor.

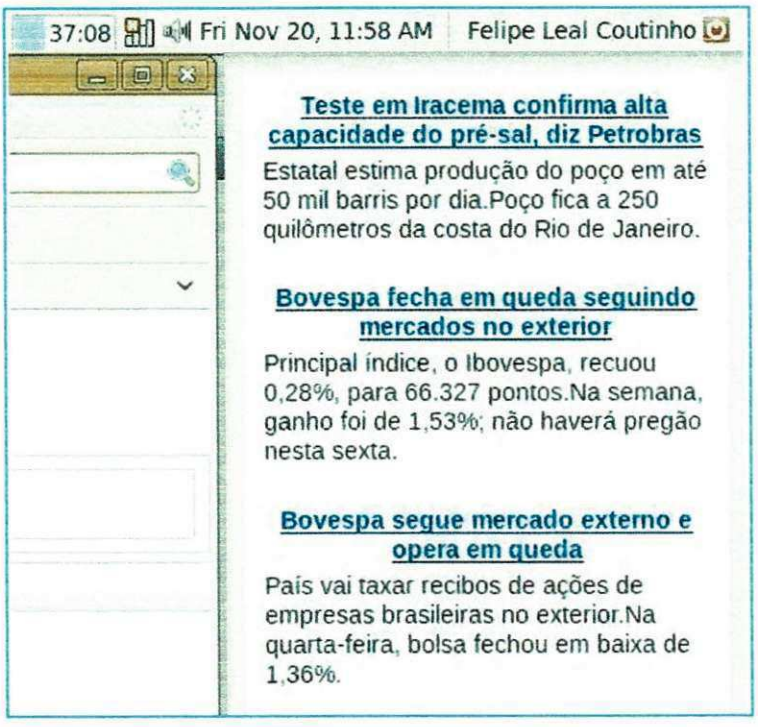

Figura 5 - Tela do Leitor de RSS

[29] http://qt.nokia.com/products/developer-tools

[30] http://qt.nokia.com/doc/4.5/index.html

Como a finalização do leitor de RSS pode-se ter a primeira versão funcional do protótipo.

4.11. Ajustes Finais

Esta US envolveu a criação de mais testes e a revisão dos já feitos. Além disso, criou-se um instalador para o leitor do cliente leve, facilitando assim o *deploy*. Para o servidor foi criado um manual de *deploy* (utilizando o formato que a Signove adota) que foi validado por outros membros do projeto.

Ao final dessa US, tinha-se um protótipo funcional do SigCenter validado pelo cliente. Pode-se destacar que a validação pelo cliente ocorreu de forma contínua durante todo o desenvolvimento do produto. Ele determinava critérios de aceitação para cada US, além de avaliar o SigCenter sendo executado ao final de cada *sprint*.

SEÇÃO V

CONSIDERAÇÕES FINAIS

5. CONSIDERAÇÕES FINAIS

A partir da realização desse estágio, foi possível perceber a importância da prática para facilitar a assimilação dos conceitos aprendidos no curso de Ciência da Computação. Além disso, percebeu-se que no dia-a-dia das empresas são utilizados muito mais conceitos do que os vistos em sala de aula. Dessa forma, é necessário estar sempre atualizado para poder acompanhar o mercado de trabalho.

Particularmente, esse estágio foi de grande importância profissional pois possibilitou o aprendizado de tecnologias já consolidadas e que estão tomando força no cenário da computação. Além disso, houve o contato com uma nova metodologia de desenvolvimento ágil e com o desenvolvimento de sistemas de grande porte.

Com relação às atividades do estágio, todas as atividades requeridas pelo cliente foram atendidas. As principais dificuldades enfrentadas foram relacionadas ao desenvolvimento web. Porém, essas eventuais dificuldades foram sanadas utilizando livros, Internet e consulta aos desenvolvedores mais experientes da Signove. Ao final do estágio, foi possível perceber que a equipe ficou com um bom conhecimento de programação para web.

Por fim, observa-se que o estágio é uma ótima oportunidade para o amadurecimento pessoal e profissional. Sugere-se que a academia dê cada vez mais importância ao estágio integrado e que incentive seus alunos a realizar estágios, principalmente fora do ambiente acadêmico.

REFERÊNCIAS BIBLIOGRÁFICAS

6. REFERÊNCIAS BIBLIOGRÁFICAS

CODEHAUS FOUNDATION. Groovy, 2009. Disponivel em: <http://groovy.codehaus.org/>. Acesso em: 15 nov. 2009.

DICKINSON, J. **Grails 1.1 Web Application**. 1. ed. Birmingham: Packt Publishing, 2009.

GOOGLE. API do Google Maps, 2009. Disponivel em: <http://code.google.com/intl/pt-BR/apis/maps/>. Acesso em: 10 nov. 2009.

LIPPMAN, S. B.; LAJOIE, J.; MOO, B. E. **C++ Primer**. 4. ed. Stoughton: Addison-Wesley Professional, 2005.

NOKIA CORPORATION. Qt - A cross-platform application and UI framework, 2009. Disponivel em: <http://qt.nokia.com/>. Acesso em: 15 nov. 2009.

OPENQA. Selenium web application testing system, 2009. Disponivel em: <http://seleniumhq.org/>. Acesso em: 20 Nov. 2009.

REFSNES DATA. HTML Tutorial, 2009. Disponivel em: <http://www.w3schools.com/html/default.asp>. Acesso em: 5 nov. 2009.

SCRUM ALLIANCE. Scrum Alliance, 2009. Disponivel em: <http://www.scrumalliance.org/>. Acesso em: 12 out. 2009.

SPRINGSOURCE. Grails, 2009. Disponivel em: <http://grails.org/>. Acesso em: 19 nov. 2009.

APÊNDICES

APÊNDICE A – Plano de Estágio

UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
DEPARTAMENTO DE SISTEMAS E COMPUTAÇÃO

Plano de Estágio Integrado

SigCenter – Sistema para Implantação e Manutenção de Telecentros

Signove Tecnologia S/A

Felipe Leal Coutinho
Estagiário – UFCG

Kyller Costa Gorgônio
Supervisor Técnico

Hyggo Oliveira de Almeida
Supervisor Acadêmico

Agosto 2009

1. Informações Pessoais

Nome: Felipe Leal Coutinho
Matrícula: 20511108
Endereço residencial: Rua Castro Pinto, Nº 242, bairro São José.
CEP: 58400-420, Campina Grande – PB.
E-mail: felipelcoutinho@gmail.com
Telefones: (83) 3321-3162
(83) 9992-9711

2. Ambiente de Estágio

O estágio será realizado na empresa Signove, a qual esta situada na rua Dom Pedro II, Nº 675, bairro da Prata, 58400-565, Campina Grande, Paraíba.

A Signove, fundada por ex-alunos de mestrado e doutorado da UFCG, possui um amplo laboratório de desenvolvimento e pesquisa, salas de reuniões, copa, recepção, sala de servidores, etc. A sala de desenvolvimento é equipada com computadores em rede, modernos, de uso individual e agrupado em ilhas para facilitar a interação do grupo.

3. Supervisão

Supervisor Acadêmico

Nome: Hyggo Oliveira de Almeida

Endereço: Laboratório de Sistemas Embarcados e Computação Pervasiva, Departamento de Engenharia Elétrica, Centro de Engenharia Elétrica e Informática, Universidade Federal de Campina Grande - Av. Aprígio Veloso, 882, Bodocongó, 58109-970, Campina Grande – PB, Brasil.

Email: hyggo@dsc.ufcg.edu.br

Supervisor Técnico

Nome: Kyller Costa Gorgônio

Endereço: Signove Tecnologia S/A. Rua Castro Pinto, Nº 242, bairro Centenário. CEP: 58400-420, Campina Grande – PB.

Email: kyller.gorgonio@signove.com

4. Resumo

Desenvolver uma prova de conceito de uma solução sustentável para a implantação e manutenção de telecentros. Para reduzir o custo de propriedade, serão desenvolvidos clientes leves (*thin clients*) e um sistema de propaganda (*Ads*) direcionada e baseada em localização, em que a renda dessas propagandas poderá ser direcionada para a manutenção do telecentro.

5. Objetivos

Trabalhar no desenvolvimento de um protótipo de uma solução para implantação e manutenção de telecentros em que os clientes e o Proxy sejam leves, de modo a baratear os custos. O telecentro deverá ser ligado a um servidor central que proverá o serviço de contratação de anúncios baseados em localização, permitindo assim que os anunciantes atinjam com maior eficácia seu público.

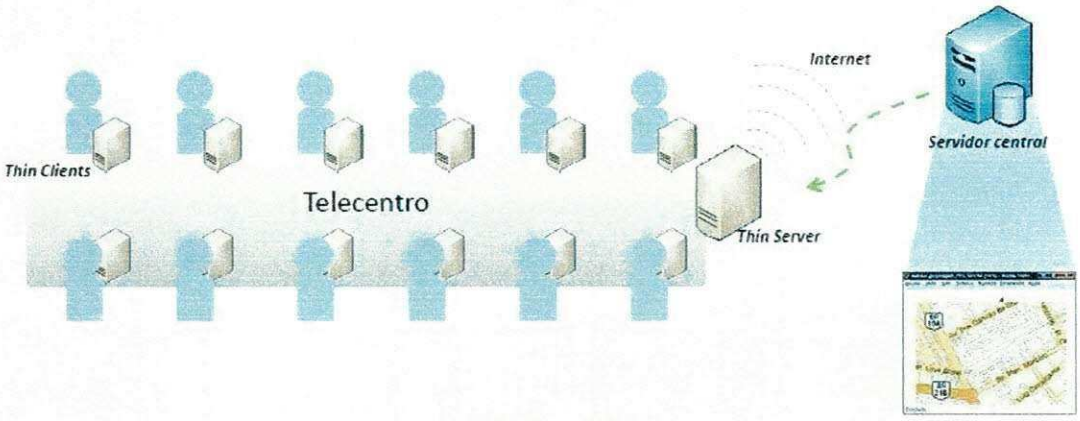

Figura 4 - Arquitetura do sistema

Entre os objetivos específicos, pode-se citar:

- Implementar software servidor de *Ads* (utilizando Grails[1]);
- Implementar o cliente exibidor de *Ads* para o *thin client* através da criação de uma distribuição Linux personalizada para o hardware simplificado dos clientes;
- Implementar web service para a sincronização de informações entre o *Thin Server* e o servidor central.

As tecnologias a ser utilizadas serão em sua maioria *open source*.

6. Resultados Esperados

Implementação completa de uma prova de conceito da solução, incluindo desenvolvimento da distribuição Linux e dos sistemas de gerenciamento do telecentro e das propagandas.

7. Metodologia

A metodologia de desenvolvimento será o SCRUM. O SCRUM apresenta-se como uma metodologia ágil e flexível, que tem por objetivo definir um processo de desenvolvimento interativo. Esta metodologia baseia-se no desenvolvimento incremental das aplicações, centrado na equipe com ciclos de iteração curto.

A metodologia SCRUM apenas estabelece conjuntos de regras e práticas de gestão que devem ser adotadas para garantir o sucesso de um projeto. Centra-se no trabalho em equipe, visando melhorar a comunicação e maximizar a cooperação. Englobando processos de engenharia, este método não requer nem fornece qualquer técnica ou método específico para a fase de desenvolvimento de software.

As fases de desenvolvimento SCRUM podem ser divididas basicamente em três. São elas:

- Planejamento: Definição de uma nova funcionalidade requerida pelo sistema baseado no conhecimento do sistema como um todo;
- Desenvolvimento: Desenvolvimento dessa nova funcionalidade respeitando o tempo previsto, requisitos exigidos e qualidade. Esses itens definem o fim do ciclo de desenvolvimento;
- Encerramento: Preparação para a entrega do produto persistindo as atividades de testes, documentação do usuário e treinamento, caso seja necessário.

8. Atividades Planejadas

O estágio integrado terá uma carga horária de 30 horas semanais. Segue as atividades planejadas:

Atividades	Horas Estimadas
1. Estudo das tecnologias	60
2. Desenvolvimento do sistema de gerenciamento do telecentro	120
3. Desenvolvimento do sistema de *ads*	120
4. Desenvolvimento da distribuição Linux	120
5. Testes	60
6. Elaboração do relatório final	50

9. Cronograma

Atividade	AGO	SET	OUT	NOV	DEZ
Estudo das tecnologias					
Desenvolvimento do sistema de gerenciamento do telecentro					
Desenvolvimento do sistema de *ads*					
Desenvolvimento da distribuição Linux					
Testes					
Elaboração do relatório final					

10. Bibliografia

[1] SpringSource. (s.d.). *Grails*. Acesso em 10 de Agosto de 2009, disponível em Grails: http://grails.org/

Kyller Costa Gorgônio
Supervisor Técnico

Hyggo Oliveira de Almeida
Supervisor Acadêmico

Joseana Macêdo Fechine
Coordenadora da Disciplina Estágio Integrado