Centro Universitário de Brasília – UniCEUB

Faculdade de Tecnologia e Ciências Sociais Aplicadas – FATECS

Controle à Distância de Dispositivos de Iluminação Utilizando o Padrão ZigBee

Paulo Arthur Venturi

RA: 2046598/3

Monografia de Conclusão do Curso de Engenharia de Computação

Orientador: José Julimá Bezerra Junior

Brasília – DF, 1º **semestre de 2009**

Paulo Arthur Venturi

Controle à Distância de Dispositivos de Iluminação Utilizando o Padrão ZigBee

Trabalho de conclusão do curso apresentado ao Centro Universitário de Brasília - UNICEUB a fim de obter o título de Bacharelado em Engenharia de Computação

Orientador: José Julimá Bezerra Junior

Brasília – DF, 1º semestre de 2009

Dedico este trabalho à minha mãe, ao meu pai, aos meus irmãos, a minha namorada e aos meus amigos.

AGRADECIMENTOS

Agradeço, primeiramente, a Deus e aos meus pais, Paulo Venturi e Terezinha Folador, pelo amor, carinho e dedicação na minha criação e pelo incentivo e apoio ao longo dos anos. Aos meus irmãos, Pedro Henrique e Michele Regina, pelo apoio, compreensão e ajuda ao longo da minha vida. Ao professor orientador José Julimá pela dedicação em solucionar minhas dúvidas, orientação e cobrança ao longo do projeto. A todos os professores pelos ensinamentos durante o curso. Aos meus amigos da Engenharia, Leonardo Conde, João Henrique, Fernando Carvalho, Paulo Alexandre, Thiago Costa, Thiago de Almeida e Rodrigo Souza, com quem estudo desde o primeiro semestre e tiveram muita paciência em certos momentos. Aos meus colegas de trabalho pelo apoio e idéias. A minha namorada, Camilla Bueno, que esteve ao meu lado em praticamente todo o curso e me deu total apoio e incentivo, além de me ajudar a superar as dificuldades encontradas ao longo do curso.

RESUMO

Este projeto surgiu a partir da observação das necessidades do dia a dia de algumas pessoas com relação a iluminação de suas residências. Com o intuito de proporcionar maior conforto, é proposto um sistema de gerenciamento de dispositivos de iluminação de ambientes de uma residência, utilizando a comunicação wireless para acionar tais dispositivos, fazendo uso da tecnologia ZigBee. O sistema contém funcionalidades responsáveis por manter o cadastro desses ambientes e dispositivos, além de efetuar o acionamento dos dispositivos cadastrados. O programa computacional interage com o módulo coordenador que envia as informações à placa XBee-Pro cuja finalidade é encaminhar as solicitações do sistema para o módulo remoto. O módulo remoto também contém um módulo XBee-Pro, além de um microcontrolador e dois relês responsáveis pelo acionamento dos dispositivos de iluminação.

Palavras-Chave: ZigBee; Domótica; java; flex; acionamento;

ABSTRACT

This project arose from the observation of the needs of day-to-day life of people with respect to the illumination of their homes. In order to provide greater comfort, it is proposed a management system for lighting devices in a home environment, using wireless communication to trigger such devices, using the ZigBee technology. The system contains functionality responsible for maintaining the register of environments and devices, as well as trigger devices registered. The software interacts with the module coordinator who sends the information to the board XBee-Pro whose purpose is to forward the requests of the system for the remote module. The remote module also contains a board XBee-Pro, a microcontroller and two relays responsible for trigger the lighting devices.

Keywords: ZigBee; Domotics; java; flex; trigger;

LISTAS DE FIGURAS

LISTAS DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AP – *API Enable*

AS – *ActionScript*

AES – *Advanced Encryption Standard*

AMF - *ActionScript Messaging Format*

APS – Suporte a Aplicação

CE – *Coordinator Enable*

CPU – *Central Processing Unit*

CSS – *Cascading Style Sheets*

CSMA/CD – *Carrier Sense Multiple Access with Collision Detection*

DL – endereço destino

DMA – Acesso Direto à Memória

DSSS – Seqüência Direta de Espectro Estendido

ECP – *Extended Capabilities Port*

EJB – *Enterprise JavaBeans*

EPP – *Enhanced Parallel Port*

FNC – Federal Networking Council

FFD – *Full Function Device*

FTP – *File Transfer Protocol*

GIF – *Graphics Interchange Format*

GHz – Gigahertz

HTML – *HyperText Markup Language*

HTTP – *Hypertext Transfer Protocol*

HTTPS – *HyperText Transfer Protocol Secure*

IP – *Internet Protocol*

IDE – Ambiente de Desenvolvimento Integrado

IPR – *Intellectual Property Rights*

ISM - *Industrial, Scientific and Medical*

IEEE – Instituto de Engenheiros Eletricistas e Eletrônicos

JSP – *JavaServer Pages*

JDK – *Java Development Kit*

JPEG – *Joint Photographic Experts Group*

KB/s – *Kilobits* por segundo

LAN – Redes de Área Local

LSB – *Least Significant Byte*

MB/s – *Megabits* por segundo

MY – Endereço Fonte

MAC – Controle de Acesso ao Meio

MHz – Megahertz

MSB – *Most Significant Byte*

MVC – *Model-View-Controller*

NI – *Node Identifier*

NWK – Camada de Rede

PC – *Personal Computer*

PDA – *Personal Digital Assistant*

PHY – Camada Física

RIA – *Rich Internet Application*

RFD – *Reduced Function Device*

SM – *Sleep Mode*

SAR – Sistema de Automação Residencial

SQL - Structured Query Language

SSP - Standard Parallel Port

SWF - Shockwave Flash File

TCP – *Transmission Control Protocol*

USB – *Universal Serial Bus*

XML – *Extensible Markup Language*

WWW - World Wide Web

WAP – *Wireless Access Protocol*

WPAN – *Wireless Personal Area Network*

WLAN – *Wireless Local Area Network*

WMAN – *Wireless Metropolitan Area Network*

WWAN – *Wireless Wide Area Networ*

SUMÁRIO

1. INTRODUÇÃO

Os avanços da internet e da comunicação móvel ajudaram a impulsionar o crescimento e interesse pela área de automação residencial. Tais avanços permitiram que o controle da residência agora fosse feito à distância. Além disso, a comunicação sem fio veio para eliminar a necessidade de se montar uma nova infra-estrutura que suporte esse tipo aplicação e ganhar ainda mais espaço nessa área tão promissora. Sendo assim, este projeto consiste em um sistema de gerenciamento e acionamento de dispositivos de iluminação de ambientes de uma residência, além de um módulo coordenador e um módulo remoto.

1.1. Motivação

A partir da observação dos crescentes avanços na área de automação, seja industrial, predial ou residencial, surgiu a idéia de um projeto para facilitar o dia a dia, evitar gastos desnecessários de energia e até mesmo garantir certo nível de segurança. Baseando-se nessas idéias, a solução é um sistema WEB inovador e de fácil utilização para gerenciar e acionar dispositivos de iluminação de ambientes de uma residência.

1.2. Objetivos

A utilidade desse projeto é proporcionar uma melhoria para a sociedade através do desenvolvimento de um sistema de controle de automação sem fio de dispositivos de iluminação em uma residência. A melhoria ocorre devido a facilidade e praticidade em se acionar os dispositivos remotamente utilizando para tanto a comunicação ZigBee.

Zigbee é um padrão de comunicação wireless, baseado na norma IEEE 802.15.4, que surgiu, basicamente, com o propósito de controlar remotamente equipamentos domésticos. A comunicação é feita nas faixas de freqüência de 2.4

GHz (Global), 915Mhz (América) e 868Mhz (Europa), tem alcance de até 100 metros em áreas urbanas e velocidades de até 250kbps.

Sendo assim, esse projeto é uma alternativa a alguns sistemas atuais que utilizam a porta serial ou paralela para controlar dispositivos. Uma das limitações da utilização das portas citadas é a pequena quantidade de dispositivos que podem ser controlados e também o curto alcance da transmissão dos dados por essas vias. Ademais, há a necessidade de alteração na infra-estrutura da residência, afim de adquar toda a fiação utilizada nos projeto.

Já o projeto proposto proporciona facilidades no dia a dia. Tanto para evitar gastos desnecessários de energia ao desligar uma lâmpada remotamente que tenha ficado ligada ao sair para uma viagem, quanto para ligar lâmpadas durante a noite para evitar a ação de assaltantes ou ,ainda, para proporcionar maior comodidade para os portadores de necessidades especiais. Além da comunicação wireless não necessitar de alterações de infra-estrutura da residência.

1.3. Visão Geral do Projeto

O Projeto é composto por um módulo coordenador, um módulo remoto, ambos possuem placa XBee-Pro, e um sistema capaz de gerenciar e acionar dispositivos de uma residência. A transmissão a partir do módulo coordenador é feita através do padrão ZigBee e o módulo remoto é responsável por acionar os dispositivos. O diagrama geral do projeto é apresentado na figura 1.

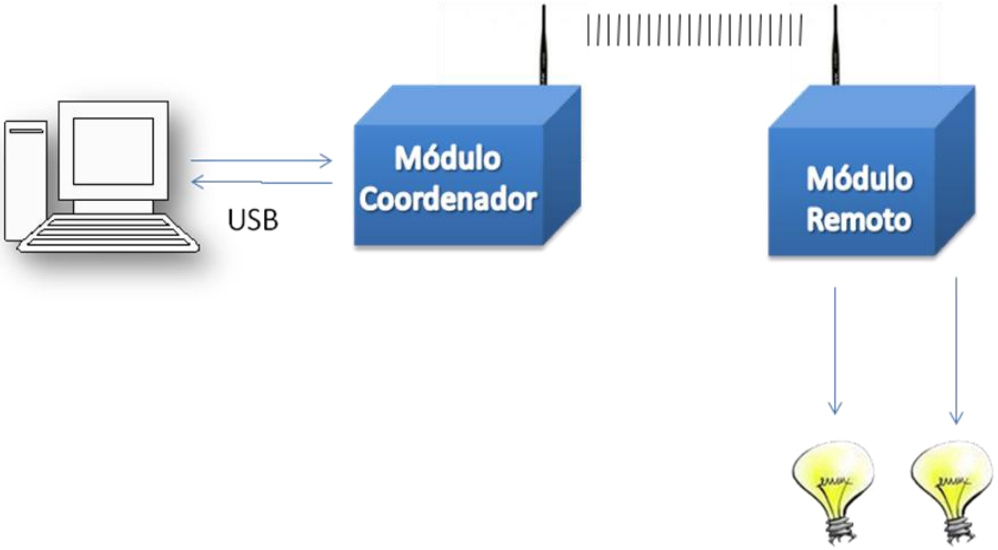


Figura 1: Diagrama geral do projeto.

O sistema desenvolvido é composto por quatro funcionalidades. A primeira é a responsável por manter o cadastro de controladores que são os módulos remotos. Já a funcionalidade de manter ambientes tem como principal função a escolha de uma imagem que represente o ambiente e um controlador. A terceira funcionalidade corresponde ao cadastro de dispositivos para os ambientes. Nesse cadastro, os dispositivos são associados aos relês disponíveis em cada controlador. A quarta e última funcionalidade é onde a comunicação com o hardware será efetuada e os dispositivos serão acionados remotamente. Primeiramente uma solicitação de acionamento será feita e o módulo coordenador irá encaminhá-la, via ZigBee, para o controlador. Este, por sua vez, acionará o dispositivo solicitado por meio de um relê, de contato seco, o qual funciona como uma chave liga/desliga.

1.4. Estrutura da Monografia

Além deste capítulo introdutório, esta monografia está dividida em outros três.

O Capítulo 2 contém o referencial tecnológico com as definições das tecnologias mais relevantes e suas respectivas aplicações ao longo do projeto.

O Capítulo 3 aborda sobre o desenvolvimento do projeto, descrevendo a implementação do software, a integração do sistema com o hardware e apresenta uma série de testes e resultados obtidos.

O Capítulo 4 apresenta a conclusão do projeto, as dificuldades encontradas durante seu desenvolvimento e sugestões para futuras evoluções do mesmo.

2. REFERENCIAL TECNOLÓGICO

2.1. Tipos de redes de comunicação

2.1.1. Internet

Em 24 de outubro de 1995, a *Federal Networking Council* (FNC) criou com unanimidade o termo Internet. Esta definição foi desenvolvida em conjunto com a *Internet and Intellectual Property Rights* (IPR) *Communities*. (Resolução, FNC, 1995)

De acordo com a FNC, o seguinte texto reflete a definição do termo "Internet”

> A Internet refere-se ao sistema de informação global que:
>
> É logicamente ligada entre si por um único espaço de endereço global baseado no protocolo IP ou nas extensões subsequentes,
>
> É habilitada a suportar comunicação usando TCP/IP ou extensões subsequentes e/ou outros protocolos IP compatíveis,
>
> Provê, usa ou torna acessível, tanto público quanto privado, serviços em camadas de alto nível de comunicação e infra-estruturas relacionadas aqui descritas. (Resolução, FNC, 1995)

Atualmente, a Internet forma um grande conjunto de computadores interligados de forma integrada. Por utilizar o protocolo de comunicação TCP/IP a integração independe da máquina do usuário. (MÜLLER, N. 2008)

O surgimento da *World Wide Web* (WWW), em 1989, facilitou ainda mais a integração entre os computadores ligados à Internet. A utilização de hipertexto estabeleceu vínculo entre as informações, não havendo mais limitações geográficas em sua localização. (HOMMERDING, N. M. S. 2001)

Com efeito, a Internet tornou-se um dos maiores meios de comunicação e a sua grande capacidade de transmissão e alcance faz com que tenha milhares de adeptos no mundo todo. Texto, imagem, som, vídeo, jogos são exemplos do que pode ser transmitido e dissipado em segundos.

2.1.2. Intranet

“A palavra Intranet começou a ser usada em meados de 1995 por fornecedores de produtos de redes, para se referirem ao uso dentro das empresas privadas de tecnologias projetadas para a comunicação por computador entre empresas”. (BENETT, G. 1997)

A Intranet é uma idéia que surgiu a partir da Internet e que utiliza a tecnologia web para centralizar comunicados, notícias, formulários, procedimentos, processos de uma empresa, etc. Funciona como um mega portal no ambiente privativo da empresa. Além disso, a troca de informações e dados entre os funcionários é feita em altas taxas de transmissão, pois a rede local é utilizada como meio de comunicação. A utilização da rede local possibilita o controle de usuários e permissões, assim as trocas de informações sigilosas entre os usuários podem ser feitas de forma segura. (TRINDADE, O. 2009)

2.2. Interfaces hardware e software do cliente

2.2.1. *Personal Computer* (PC)

O Computador Pessoal foi lançado em 1971, tinha 256 bytes de memória, não possuía CPU e foi projetado com finalidades educativas. A expressão PC – do inglês, *Personal Computer* – é utilizada para denominar um computador de mesa (*Desktop*), *laptop*, *notebook*, *netbook* ou *tablet pcs*, todos de uso individual e baixo custo. (WIKIPEDIA, 2009)

Desde seu lançamento, o computador vem sofrendo alterações arquiteturais e seus componentes ganhando cada vez mais capacidade de processamento, armazenamento de memória e qualidade de imagem. Com tais mudanças, surgiram, ainda, diversos sistemas operacionais. A grande diversidade e o baixo custo dos computadores tornaram-se grandes aliados para sua aquisição. Desta forma, milhares de pessoas utilizam o PC para realizar as mais diversas tarefas do cotidiano como navegar por site de noticias, ler e-mails, digitar documentos, escutar

músicas e assistir vídeos. A despeito de sua utilização cotidiana, o PC é utilizado como ferramenta de desenvolvimento de *software*, gerenciamento de projeto, controles de estoque, dentre outros.

2.2.2. Dispositivos móveis

Os equipamentos ou periféricos que possam ser transportados e estejam acessíveis a qualquer momento em qualquer lugar são denominados dispositivos móveis. Nessa categoria de dispositivos temos como exemplo o Assistente Pessoal Digital (PDA), telefones celulares, *smartphones* (celulares com funcionalidades avançadas que podem ser estendidas por meio de programas executados no seu sistema operacional), entre outros. (SENAC, 2007)

Através desses dispositivos e com a utilização da comunicação sem fio é possível conectar-se a intranet e navegar por portais coorporativos ou pessoais, usufruindo de todas as funcionalidades disponíveis. (TEAM WORK, 2009)

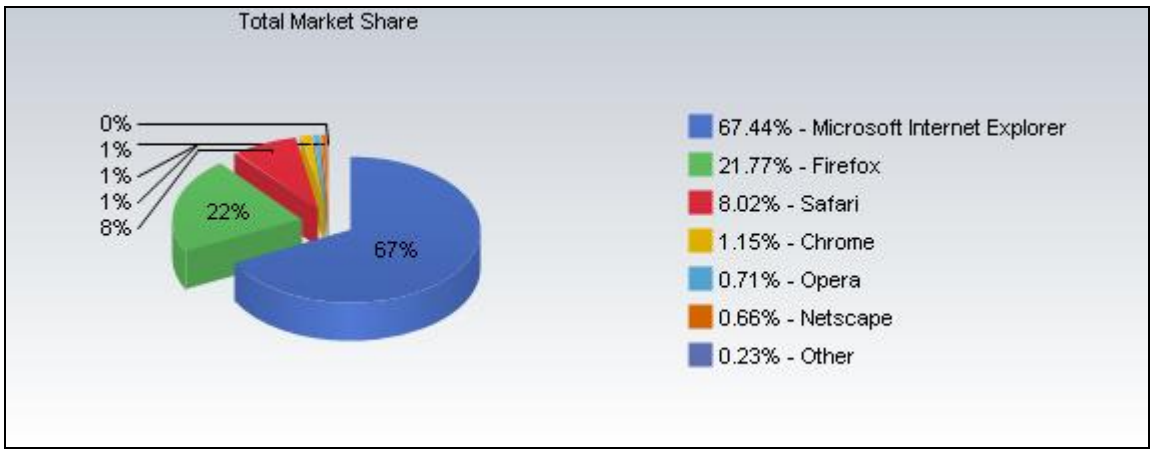
2.2.3. Navegador

Navegador ou *browser*, como é comumente chamado, é um programa utilizado para navegar por páginas na Internet, copiar programas, enviar e-mail, e é baseado no modelo cliente/servidor. O navegador atua como o cliente requisitando informações a um servidor. O servidor por sua vez processa as informações e devolve uma resposta que será interpretada pelo navegador. A comunicação entre cliente e servidor é feita, principalmente, através do Protocolo de Transferência de Hipertexto (HTTP). Os navegadores mais recentes também são capazes de utilizar outros protocolos como o Protocolo de Transferência Segura de Hipertexto (HTTPS) e o Protocolo de Transferência de Arquivos (FTP). (WIKIPEDIA, 2009)

Os navegadores tem a capacidade de interpretar informações dos mais variados tipos. Alguns desses tipos são nativos como, por exemplo, Linguagem de Marcação de Hipertexto (HTML), Linguagem de Marcação Extensível (XML), Formato de Intercâmbio Gráfico (GIF), *Joint Photographic Experts Group* (JPEG) e o

Cascading Style Sheets (CSS). Outros necessitam da utilização de *plugins* específicos para sua interpretação. É o caso do FLASH e do JAVA. (WIKIPEDIA, 2009)

Atualmente, existem diversos navegadores disponíveis no mercado. De acordo com a figura 2, dados de fevereiro de 2009 mostram que o Internet Explorer lidera, com 67,44% do mercado, seguido do FireFox, detentor de 21,77%. O restante está distribuído entre Safari, Opera, Chrome, entre outros.



Browser	Total Market Share
Microsoft Internet Explorer	67.44%
Firefox	21.77%
Safari	8.02%
Chrome	1.15%
Opera	0.71%
Netscape	0.66%
Mozilla	0.07%
Opera Mini	0.07%
Playstation	0.04%
ACCESS NetFront	0.02%
Blazer	0.02%
Microsoft Pocket Internet Explorer	0.01%
ANT Galio	0.00%

WebTV	0.00%
iCab	0.00%
Konqueror	0.00%
Lotus Notes	0.00%
BlackBerry	0.00%

Figura 2: Divisão do mercado de navegadores.

Fonte: http://marketshare.hitslink.com/report.aspx?qprid=0&qptimeframe=M&qpsp=121&qpmr=75&qpdt=1&qpct=3&qpob=UV%20DESC

A grande novidade é o desenvolvimento de navegadores voltados para dispositivos móveis. A maior mobilidade e facilidade de acesso às informações através desses dispositivos juntamente com a popularização do acesso a Internet veio para abrir novas possibilidades de pesquisas e desenvolvimento. Existem, atualmente, navegadores que são denominados micro-*browser*. Estes são desenvolvidos principalmente para *smartphones* e PDAs. (BRASIL ESCOLA. 2008)

2.3. Linguagens para web

2.3.1. HTML

HTML é uma linguagem de programação e significa Linguagem de Marcação de Hiper Texto, do inglês *HyperText Mark-up Language*. Foi inventada na década de 90, pelo cientista Tim Berners-Lee, cuja idéia inicial era criar um meio em que fosse possível o acesso e a troca de informações entre cientistas de universidades. (HTML.net. Tradução de Maurício Samy Silva)

A utilização do HTML tornou possível a visualização das informações na Internet por meio de um navegador. Através das marcações (*tags*), é possível formatar textos, inserir imagens, sons e vídeos e descrever a estrutura de um documento HTML, sendo, o navegador, responsável por interpretar essas marcações e montar a página visual. (ALVAREZ, M. A. O que é HTML, 2004)

2.3.2. CSS

A falta de um padrão na formatação dos conteúdos de uma página web impulsionou a criação do CSS, cujo objetivo era padrozinar e disponibilizar meios mais sofisticados de formatação, sendo, ainda, suportado pela maioria dos navegadores. (VANDRÉ, P. 2007)

Cascading Style Sheets (CSS), ou Folhas de Estilo em Cascata, é uma linguagem que controla a formatação, quais sejam, fontes de texto, margens, bordas, cores e imagens de fundos, de documentos HTML. (VANDRÉ, P. 2007)

A linguagem HTML também pode ser usada para controlar formatação e a diferença na sua utilização para o CSS é que, enquanto aquele é utilizado para formatar apenas conteúdos estruturados, esse é utilizado para formatar qualquer tipo de conteúdo. (HTML.net. Tradução de Maurício Samy Silva)

2.3.3. FLASH

Flash é uma tecnologia mormente utilizada para a criação de páginas, componentes gráficos e interativos para web. Baseia-se em gráficos vetoriais, o que permite criar animações que demoram menos a ser carregadas, ou seja, são mais leves. (ALVAREZ, R. 2004)

Os arquivos de animações têm como extensão o *Shockwave Flash File* (SWF), são executados nos navegadores, bastando para isso, a utilização de *plugins* específicos, sendo usualmente utilizados como propagandas animadas (*banners*). Além disso, há diversos jogos e apresentações que se utilizam dessa tecnologia. Como exemplo, tem-se o crescimento da interação com o usuário por meio de formulários. (WIKIPEDIA, 2009)

2.3.4. FLEX

Flex é o nome de um conjunto de tecnologias baseadas na plataforma do Macromedia Flash que possui uma estrutura de código aberto para a criação de RIAs. (EAGLE, M. 2004)

Rich Internet Application (RIA) é uma aplicação Web que contém características e funcionalidades de uma aplicação desktop tradicional, transferindo a responsabilidade de processamento do cliente (cliente/servidor) para o navegador. Porém, o processamento de *back-end* continua sendo feito no servidor de aplicação. (EAGLE, M. 2004)

O desenvolvimento de RIAS, utilizando a linguagem de programação Flex, é baseado em várias tecnologias integradas. A utilização do MXML, uma linguagem declarativa baseada em XML, juntamente com o ActionScript, uma linguagem de programação orientada a objetos, dá origem ao *front-end* (interface com o usuário) e a utilização de outras linguagens, como por exemplo Java, PHP, ColdFusion, ASP .Net dá origem ao *back-end* que roda nos servidores de aplicação. (EAGLE, M. 2004)

Qualquer editor de texto pode ser utilizado para desenvolver códigos tanto para arquivos MXML como para arquivos ActionScript. Existem ferramentas que facilitam o desenvolvimento. Atualmente a mais conhecida é o Adobe Flex Builder 3. Um ambiente de desenvolvimento baseado no Eclipse que está disponível em duas versões: Padrão e Professional. (ADOBE, Visão geral do Flex)

Flex é uma linguagem que provê vários meios de comunicação com os servidores de aplicação. Pode-se destacar a comunicação HTTP, web service e o *ActionScript Messaging Format* (AMF). Esse último, por usar um formato de dados binários em série, garante um rápido e eficiente meio de transporte dos servidores de aplicação para as RIAs, acelerando o desempenho. (ADOBE, Visão geral do Flex)

2.3.5. JAVA

Java é uma linguagem de programação orientada a objetos que surgiu na década de 90, com o principal objetivo de ser um ponto de convergência dos computadores com equipamentos e eletrodomésticos do dia a dia, e não uma mera linguagem. Porém, a idéia não era economicamente viável e não havia boas espectativas para o crescimento de produtos que suportassem essa linguagem. (SILVEIRA, I. F. 2003)

Foi então que, com a rápida evolução da Internet e a grande interatividade que estava se estabelecendo na web, novas oportunidades surgiram para a linguagem Java. Em 1995, foi lançada uma nova versão projetada para se mover por meio das redes heterogenias, como a Internet. Agora os aplicativos Java podiam ser rodados dentro dos navegadores e tudo estaria disponível através da Internet. Os avanços não pararam e em 1999 surgiu a plataforma de desenvolvimento e distribuição corporativa (J2EE). (DEITEL, H. M.; DEITEL, P. J. JAVA Como Programar, 2006)

O surgimento das páginas JavaServer Pages (JSP) facilitou o desenvolvimento de páginas *web* dinâmicas. Para tanto, código HTML estático, juntamente com código Java dinâmico, era escrito em um mesmo arquivo. Essa e outras tantas facilidades da linguagem, como suporte a restrições de execução via rede, independência de plataforma, facilidade de internacionalização, desalocação de memória automática, fizeram com que milhares de desenvolvedores construíssem novos aplicativos, além de evoluir os já existentes, ajudando no melhoramento da linguagem. (JAVA.com)

2.4. Servidores de Aplicação e Banco de dados

2.4.1. Apache Tomcat

O Tomcat é um servidor web Java (*web server*) desenvolvido pela Apache Foundation Software, além de ser a implementação oficial para as tecnologias Java Servlet e JavaServer Pages. É desenvolvido em um ambiente aberto e participativo e distribuído sob a licença Apache Software. Conta com a colaboração dos melhores desenvolvedores do mundo para manter sua evolução. O Apache Tomcat domina uma numerosa escala das aplicações web criticas nas mais diversas organizações e indústrias. (The Apache Software Foundation, 2009)

Embora possua características de um servidor de aplicação, não pode ser considerado como tal, uma vez que não preenche todos os requisitos da especificação J2EE, como o suporte a *Enterprise JavaBeans* (EJB).

2.4.2. JBOSS

JBoss é uma plataforma certificada J2EE para aplicações Java coorporativas, aplicações web e portais, implementado completamente através da linguagem de programação Java e possui seu código fonte aberto, motivo pelo qual diversos programadores são incentivados, pela instituição JBoss, a contribuir com o desenvolvimento de novas funcionalidades. Por ser baseado totalmente em Java, o JBoss pode ser utilizado em qualquer sistema operacional que tenha suporte à essa tecnologia. (FLEURY, M.; STARK, S.; NORMAN, R. JBoss 4.0, 2005)

O servidor de aplicação JBoss oferece uma gama completa de funcionalidades J2EE, contando com serviços estendidos incluindo *clustering*, *caching*, persistência, segurança, *web services* e servidor web. Diversos elementos contribuem para que seja o líder do mercado, quais sejam, a sua tecnologia estável, código fonte aberto, grande comunidade de desenvolvedores ativos, suporte técnico especializado e rede de parceiros autorizados e certificados. (FLEURY, M.; STARK, S.; NORMAN, R. JBoss 4.0, 2005)

2.4.3. Firebird

Caracteriza-se como sendo um banco de dados relacional de código-fonte aberto que oferece muitas características ANSI SQL-99, além de excelente gerenciamento para acessos concorrentes, desempenho elevado e poderoso suporte de idioma para procedimentos armazenados (*Store Procedure*) e *triggers*. Ademais, tem um eficaz funcionamento no Linux, Windows e Mac OS e em diversas plataformas Unix. Justifica-se, logo, o seu uso por inúmeros sistemas de produção entre grandes empresas desde 1985. (BORRIE, H. 2006)

2.4.4. MySql

MySql é o sistema de gerenciamento de banco de dados relacional de código-fonte aberto mais popular do mundo. A MySql AB, empresa comercial, fundada pelos desenvolvedores do MySql, cujo principal negócio é o fornecimento de serviços relacionados ao sistema de gerenciamento de banco de dados, sendo ainda, a responsável pelo desenvolvimento e distribuição do MySql, além de seu suporte. (MySql, 2009)

Seu sucesso se deve ao funcionamento em diversos sistemas operacionais, à consistência, ao alto desempenho, à confiabilidade e à velocidade, além de ser multi-tarefa, multi-usuário e altamente adaptável ao acesso a banco de dados na internet. (MySql, 2009)

2.5. Padrões de comunicação e Interação com Componentes Físicos

2.5.1. Serial

A comunicação serial é o método adotado para a comunicação de diversos periféricos de computadores, entre eles, modems, mouses e scanners. Para efetuar a comunicação serial, é necessário que uma mensagem longa de dados digitais seja quebrada em partes menores para serem enviadas sequencialmente, ou seja, cada

bit é transmitido por vez. Com isso, os *bits* individuais devem ser reorganizados no destino para compor a mensagem original. Geralmente, os *bits* não são enviados de maneira uniforme. O receptor deve saber o momento apropriado para ler os *bits*, saber quando um pacote começa e quanto tempo decorre entre *bits* recebidos. Quando esse tempo é conhecido, o receptor está sincronizado com o transmissor. É possível que falhas ocorram durante a transmissão dos *bits* ocasionando perda de dados ou dados corrompidos.

A porta serial pode se comunicar tanto no sistema síncrono, como no assíncrono. No primeiro, há dois canais, um para transmitir dados e outro para o sincronismo, o qual é utilizado pelo transmissor para o envio de *clocks.* O receptor, ao identificar o *clock*, lê o canal de dados, armazena as informações e aguarda um novo sinal de *clock*. O sincronismo fica garantido com o envio de dados e do clock pelo transmissor. A figura 3 ilustra o sinal serial síncrono. Já no segundo, as informações são enviadas em um único canal, contendo, normalmente, 10 ou 11 bits. O primeiro *bit* é denominado *start bit*, seguido de 8 *bits* de dados da mensagem e finalizando com 1 *bit* de paridade e 1 de parada (*stop bit*), conforme a figura 4. (CANZIAN, E. MINICURSO: Comunicação Serial - RS232)

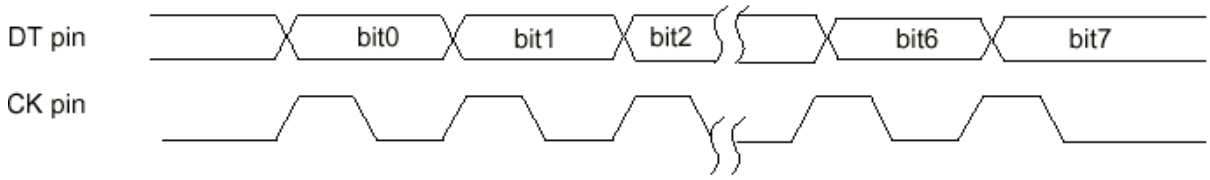


Figura 3: Sinal serial síncrono.

Fonte: http://www.coinfo.cefetpb.edu.br/professor/leonidas/irc/apostilas/comun_serial.pdf

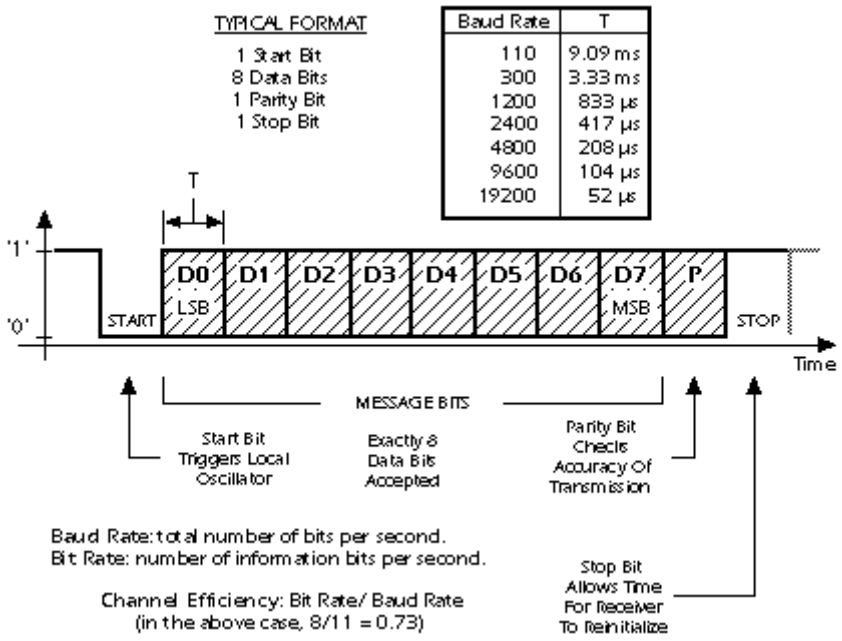


Figura 4: Sinal serial assíncrono.

Fonte: http://www.coinfo.cefetpb.edu.br/professor/leonidas/irc/apostilas/comun_serial.pdf

2.5.2. Paralela

A porta paralela é uma interface de comunicação entre computadores e periféricos. A idéia inicial para a sua utilização era a conexão entre o computador a uma impressora, para tanto, a transmissão era unidirecional. Sua evolução ocasionou na transmissão bidirecional, fazendo com que diversos periféricos de entrada e saída de dados passassem a utilizar essa porta para transmitir suas informações. (AXELSON, J. 1997)

Na transmissão em paralelo, várias vias condutoras de sinais são utilizadas para transferir grupos de *bits* simultâneos. A grande vantagem é a alta taxa de transmissão alcançada, entretanto, ela só se torna útil quando a distância até o receptor é pequena, por sofrer grande influência de ruídos. A porta paralela pode funcionar de formas distintas, são elas: (AXELSON, J. 1997)

Standard Parallel Port (SPP): A porta paralela original foi baseada em uma interface de impressora já existente, Centronics. No entando, o computador apresenta algumas diferenças dos outros sistemas que utilizavam essa interface. O padrão SPP pode transferir 8 *bits* de uma só vez a uma taxa de 150 KB/s a um

periférico, utilizando um protocolo semelhante às originais interfaces Centronic. (AXELSON, J. 1997)

Enhanced Parallel Port (EPP): O modo EPP possibilita uma capacidade de transmissão de 32 *bits* entre o processador e a porta, sendo quebrado em grupos de 8 *bits* na hora da transmissão, a uma taxa de 2MB/s. Por ser bidirecional, o modo EPP pode mudar rapidamente de sentido, por isso é muito eficaz quando utilizado para a comunicação com disco rígido, unidades de fita e outros dispositivos de transferência de dados em ambas as direções. (AXELSON, J. 1997)

Extended Capabilities Port (ECP): O modo ECP foi proposto pela primeira vez pela HP (Hewlett Packard) e Microsoft, além de ser bidirecional, possui sua capacidade de transmissão aumentada pela utilização de um algoritmo de compressão e por ter Acesso Direto à Memória (DMA). A tabela 1 mostra uma comparação entre os três tipos de funcionamento da porta paralela. (AXELSON, J. 1997)

Tabela 1: Comparação dos modos de funcionamento da porta palalela.

	SPP	EPP	ECP
Data de introdução	1981	1994	1994
Fabricante	IBM	Intel e Xircom	Hewlett Packard e Microsoft
Bidirecional	Não	Sim	Sim
DMA	Não	Não	Sim
Velocidade	150 KB/s	2 MB/s	2 MB/s

2.5.3. Ethernet

O termo *Ethernet* refere-se à família de produtos de redes locais abrangidos pelo padrão Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE) 802.3 que define o que é geralmente conhecido como protocolo *Carrier Sense Multiple Access with Collision Detection* (CSMA/CD). As redes que utilizam a *ethernet* são normalmente de pequeno alcance, sendo utilizadas para conectar dispositivos próximos. Porém, a utilização da *ethernet* pode conectar dispositivos a centenas de metros ou até mesmo a dezenas de quilômetros. (Cisco Systems, Inc.)

A *Ethernet* original foi desenvolvida como uma rede de cabo coaxial experimental na década de 1970 pela *Xerox Corporation* para operar com uma taxa de dados de 3 Mbps. Um esquema conhecido como CSMA/CD organizava a forma pela qual os nós de uma rede se comunicavam. Ao longo do tempo várias outras tecnologias e protocolos surgiram como prováveis substitutos, mas a *Ethernet* tem sobrevivido como a principal tecnologia de redes de área local (LAN) sendo utilizada por 85% dos computadores com ligação LAN. A facilidade de implementação e manutenção, além de seu baixo custo e ampla flexibilidade de topologias de redes, garante a interconexão e funcionamento bem sucedido de produtos compatíveis com a norma independentemente do fabricante. Ademais, as altas taxas de transmissão de dados influenciam na sua escolha. Atualmente existem 3 taxas de transmissão de dados: 10 Mbps - 10Base-T Ethernet, 100 Mbps - Fast Ethernet e 1000 Mbps - Gigabit Ethernet. (Cisco Systems, Inc.)

2.5.4. Wireless

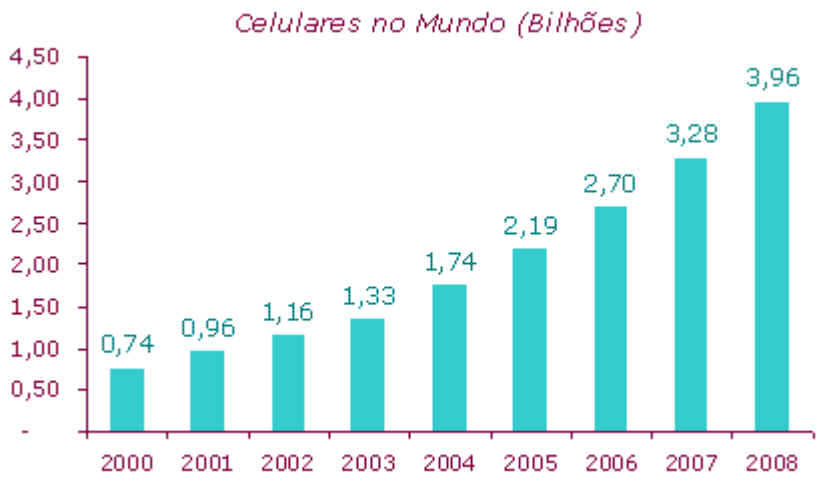


Figura 5: Gráfico Celulares no Mundo.

Fonte: Fonte: UIT, Wireless Intelligence e GSA/Informa

O crescimento da comunicação *Wireless* se deve ao aumento exponencial do número de aparelhos celulares, alcançando cerca de 3,96 bilhões de usuários (figura 5), tendo em vista que tornaram-se uma ferramenta empresarial e parte da vida cotidiana da maioria das pessoas. Outro fator importante é a substituição das redes locais com fios por redes sem fio. Isto já é realidade em muitas casas, bares, restaurantes, aeroportos, empresas e *campus* de universidades. As novas aplicações, incluindo redes sem fio de sensoriamento, automação de fábricas, casas e prédios inteligentes, são alguns dos novos projetos que auxiliam esse rápido crescimento. (GOLDSMITH, A. 2005)

As redes Wireless podem ser classificadas em pessoais ou curta distância (WPAN), locais (WLAN), metropolitanas (WMAN) e geograficamente distribuídas ou de longa distância (WWAN).

*Wireless Personal Area Network (*WPAN*)* ou rede pessoal sem fio é utilizada para interligar dispositivos fisicamente próximos. Utilizados para interligar teclados, impressoras, telefones móveis, mouses e outros. Nos equipamentos mais recentes é utilizado o padrão Bluetooth para estabelecer esta comunicação. (RIGONATTI, T. 2005)

*Wireless Local Area Network (*WLAN*)* ou rede local sem fio é utilizada para fazer conexão com a internet ou entre outras redes. São compostas por computadores com modens de radio que acessam um ponde de acesso (*Acess Point*). Esse tipo de rede atende pelo padrão IEEE 802.11. (RIGONATTI, T. 2005)

Wireless Metropolitan Area Network (WMAN) ou rede de área metropolitana refere-se a redes de uso corporativo que atravessam cidades e estados. Essa conexão é utilizada na prática entre os provedores de acesso e seus pontos de distribuição. Esse tipo de rede utiliza um dos últimos padrões de banda larga para rede MAN definido pela IEEE 802.16. (RIGONATTI, T. 2005)

Wireless Wide Area Network (WWAN) ou rede de longa distancia é uma rede de computadores que abrange uma grande área geográfica, geralmente um país ou continente. Esta cobertura normalmente é oferecida por empresas de telefonia celular para usuários móveis que necessitem do acesso à internet. (RIGONATTI, T. 2005)

2.5.5. Bluetooth

O *Bluetooth* surgiu da idéia de conectar telefones móveis a outros dispositivos sem a utilização de cabos. Foi então que, em 1994, um consórcio de empresas (IBM, Nokia, Intel, Toshiba e L. M. Ericsson) se formou com o objetivo de desenvolver um padrão sem fio para interconectar dispositivos de computação, comunicação e ainda acessórios utilizando rádios sem fio de curto alcance, baixa potência e baixo custo. Em 1999, o *Bluetooth* foi formalizado e uma especificação foi emitida pelo consórcio para o comitê de padrões IEEE. Pouco tempo depois, foi padronizado e hoje atende pela camada 802.15.2 do padrão IEEE. Atualmente, possui três classes de potência e diferentes taxas de transmissão conforme tabelas abaixo. Pode-se observar os vários tipos de classes de *Bluetooth* pela tabela 2. Já a tabela 3 mostra as taxas de transmissão para as suas diferentes versões. (TANENBAUM, A. S. 2003)

Tabela 2: Tipos de classes Bluetooth.

	Potência máxima permitida (mW/dBm)	Alcance (Aproximadamente)
Classe 1	100 mW (20 dBm)	~ 100 metros
Classe 2	2.5 mW (4 dBm)	~ 10 metros
Classe 3	1 mW (0 dBm)	~ 1 metros

Tabela 3: Taxas de transmissão Bluetooth.

	Taxa de transmissão
Versão 1.2	1 Mbps
Versão 2.0 + EDR	3 Mbps

A arquitetura atual consiste em um nó mestre e sete nós escravos. Essa arquitetura básica é denominada de *piconet*. A interconexão de *piconets* forma uma *scatternet*. Além dos sete nós escravos ativos, pode haver até 255 nós inativos. No estado inativo, o nó somente responde a comandos de ativação ou baliza. O nó mestre é central e responsável por determinar qual nó escravo irá se comunicar em cada *slot* de tempo. Sendo assim, não é permitida a comunicação entre dois nós escravos. (TANENBAUM, A. S. 2003)

2.5.6. ZigBee

O padrão ZigBee foi desenvolvido como alternativa de comunicação para as redes que necessitem de simplicidade para seu controle tornando o custo de aquisição, instalação e manutenção mais baratos. Também foi projetado para ter um baixo consumo de potência, confiabilidade e possibilidade de escalar milhares de dispositivos. Além disso, a utilização de um protocolo de pacotes de dados com

características específicas oferece maior flexibilidade aos tipos de dispositivos que pode controlar. (IEEE Standard 802.15.4)

Atualmente, a tecnologia ZigBee, apesar de ser uma das mais recentes nesse grupo de aplicações sem fio, é uma especificação baseada no padrão IEEE 802.15.4, homologado em maio de 2003, sendo utilizada para automação industrial e residencial, sensoriamente de áreas rurais, controle de periféricos para PCs e controle remoto de produtos eletrônicos. (TEIXEIRA, L. M. 2006)

2.5.6.1. Topologias de rede

A especificação ZigBee permite que três topologias de redes diferente sejam implementadas de acordo com a necessidade de cada aplicação. Logo abaixo, serão apresentados os detalhes das topologias Estrela, Árvore e Malha. A figura 6 apresenta um esquemático dos diferentes tipos de topologis da rede ZigBee.

- Estrela

A topologia Estrela é uma das mais simples de ser implementada. É composta por um coordenador de funções completas (FFD – *Full Function Device*) que deve ser capaz de se comunicar com qualquer dispositivo de funções reduzidas (RFD - *Reduced Function Device*) que esteja na rede. Os ambientes ideais para esse tipo de rede são locais com poucos obstáculos para facilitar a transmissão e recepção dos sinais.

- Árvore

A topologia Árvore é semelhante à Estrela. Para formar uma rede em Árvore basta substituir um ou mais dispositivos RFDs por dispositivos FFDs e deste outros dispositivos RFD e FFDs poderão ser conectados aumentando a área de cobertura da rede.

- Malha

Por último, a topologia Malha contém um coordenador FFD onde vários outros dispositivos FFDs estarão conectados e vários dispositivos RFD estarão conectados aos FFDs. Porém cada RFD somente estará conectado à um único dispositivo FFD. Nessa topologia, a rede pode se auto-organizar para melhorar o trafego na rede já que existem diversos caminhos possíveis para a comunicação. (MONSIGNORE, F. 2007)

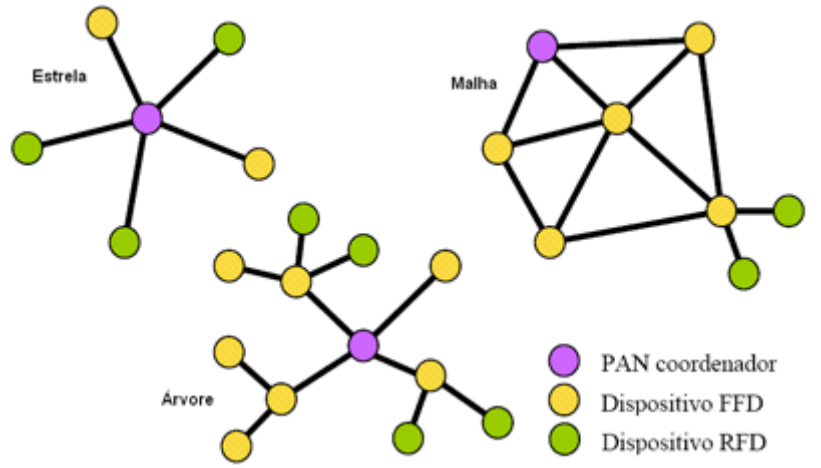


Figura 6: Topologias de rede ZigBee.

Fonte: UFRJ: Thiago S. de Carvalho e Fernando S. P. dos Anjos

2.5.6.2. Tipos de Dispositivos

Dispositivo de função completa (FFD – *Full-Function Device*): Esse tipo de dispositivo possui maior capacidade computacional e são os mais complexos. Além disso, funcionam em qualquer uma das topologias e podem assumir o papel de coordenador, roteador ou dispositivo final da rede.

Dispositivos de função reduzida (RFD – *Reduced-Function Device*): São dispositivos com capacidade computacional reduzida, não podendo atuar como coordenador ou roteadores da rede. Na prática podem ser interruptores, sensores, controles de relês, entre outros.

Também há três classes de dispositivos lógicos: Coordenador, roteador e *end device* (dispositivo final).

Coordenador: é o principal dispositivo da rede. É o responsável pela inicialização, distribuição de endereços, manutenção da Rede, reconhecimento de todos os Nós, entre outras funções podendo servir como ponte entre várias outras Redes ZigBee. Deve ser utilizado um dispositivo do tipo FFD.

Roteador: esse dispositivo é do tipo FFD e tem a principal função de ampliar a área de alcance de determinados *end device*.

End device (Dispositivo final): para esse dispositivo pode ser utilizado um FFD ou RFD. É nesse dispositivo que um sensor ou controle de relê será conectado. (MONSIGNORE, F. 2007)

2.5.6.3. Camadas de protocolos

A arquitetura do ZigBee foi projetada em camadas. A camada física (PHY) e Controle de Acesso ao Meio (MAC) foram definidas pelo padrão IEEE 802.15.4. Já a camada de rede (NWK) e de suporte a aplicação (APS) são definidas pela ZigBee Alliance e por último ficando a cargo do usuário, a camada de aplicação do usuário. A figura 7 representa um esquemático das várias camadas da arquitetura ZigBee. (MONSIGNORE, F. 2007)

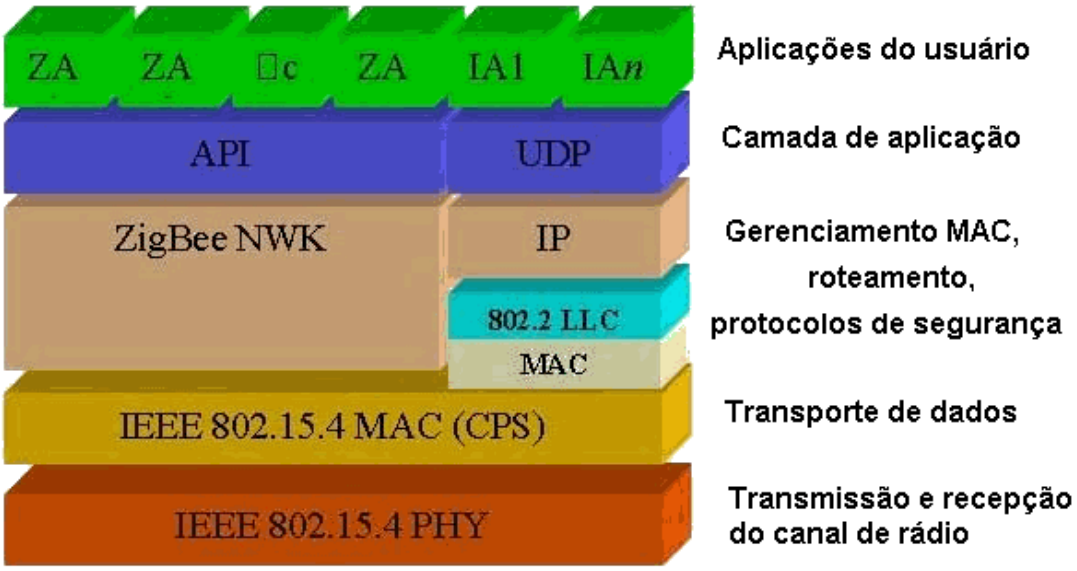


Figura 7: Camadas de protocolos.

Fonte: As Redes com ZigBee, José Mauricio Santos Pinheiro em 27/07/2004

Camada física (PHY): foi projetada para atender as necessidades de interface de baixo custo de implementação pela utilização da Seqüência Direta de Espectro Estendido (DSSS), permitindo que os dispositivos sejam mais simples. (IEEE Standard 802.15.4)

Camada Media Access Control (MAC): foi projetada para permitir múltiplas topologias de baixa complexidade. Dispositivos RDF podem operar na rede sem a necessidade de grandes quantidades de memória disponível. Além disso, a MAC pode controlar vários dispositivos sem a necessidade de colocá-los em modo de espera. É também responsável por prover um mecanismo confiável de transmissão, pelo transporte de dados e sincronização. (IEEE Standard 802.15.4)

Camada de Rede (NWK): algumas das responsabilidades dessa camada são a descoberta e manutenção de rotas entre os dispositivos envolvidos, segurança dos dados e operação de grandes quantidades de nós com latência relativamente baixa. Além disso, há a possibilidade de aumentar a rede sem precisar de equipamentos de transmissão com potências muito altas. (MONSIGNORE, F. 2007)

Camada de Suporte a Aplicação (APS): têm a responsabilidade de rotear as mensagens através dos diferentes pontos que não estão habilitados para se comunicar diretamente, descobrir pontos ativos na região de alcance do dispositivo e efetuar a comunicação com a camada AF. É também nessa camada que há a definição quanto ao tipo de dispositivo (FFD ou RFD), as funções de segurança da rede, respostas e atos para cada evento do sistema. (MONSIGNORE, F. 2007)

Camada de Aplicação (AF): nesta camada estão os softwares responsáveis por executar operações nos *end devices*. (MONSIGNORE, F. 2007)

2.5.6.4. Características Gerais

Os dispositivos baseados na tecnologia Zigbee operam nas faixas "*Industrial, Scientific and Medical*" ISM que não requer licença para funcionamento. Estão disponíveis as faixas 2,4 GHz (Global), 915 MHz (América) e 868 MHz (Europa) e suas taxas de transferência variam de acorda com a faixa ISM, podendo ser de 250 kbps, 40 kbps ou 20 kbps respectivamente. Pode-se observar pela tabela 4 as diferentes faixas e taxas de transmissão suportadas nessas redes. (TEIXEIRA, L. M. 2006)

Tabela 4: Faixa e taxas de transmissão

Padrão	Freqüência	N° de canais	Técnica de modulação	Taxa de dados
802.15.4	2.4 – 2.4835 GHz	16 (11 a 26)	DSSS, O-QPSK	250 Kbit/s
	868 – 870 MHz	1 (0)	DSSS, BPSK	20 Kbit/s
	902 – 928 MHz	10 (1 a 10)	DSSS, BPSK	40 Kbit/s

O trafego de dados na rede varia conforme a necessidade de cada aplicação. Algumas aplicações têm a necessidade de receber dados periódicos de sensores. Neste caso, o sensor ficará no modo *standby* até acordar em determinado momento, procurar por uma sinalização de um coordenador da rede, solicitar a entrada na rede e caso seja aceito enviar os dados e então voltar ao modo *standby*. Outra aplicação é a utilização de dados provenientes de interruptores e chaves. O dispositivo ficará desconectado. Ele somente se juntará a rede quando for necessário efetuar a comunicação. Esse tipo de trafego é denominado intermitente. Por último, os dados provenientes de dispositivos repetitivos de baixa latência, por exemplo, *mouse* é um método que permite cada dispositivo ter uma duração de tempo específica definida pelo coordenador da rede. (TEIXEIRA, L. M. 2006)

Na especificação 802.15.4 também há a utilização de algoritmo de criptografia para prover confidencialidade, integridade e autenticidade dos dados quando

necessário. Para tanto a camada MAC utiliza o padrão *Advanced Encryption Standard* (AES). O trafego de dados pode ter três níveis de segurança: Sem segurança, controle de acesso aos dados ou ainda segurança com chave simétrica (128 *bits* AES). (TEIXEIRA, L. M. 2006)

O IEEE 802.15.4 ainda define o endereçamento dos dispositivos através de um identificador de 64 *bits* para o nó e um identificador de 16 *bits* da rede. O trafego de dados é feito em pacotes. O pacote de dados possui tamanho variável e é utilizado para enviar mensagens a um único nó ou para múltiplos nós. Cada pacote de dados possui um *flag* para indicar o tipo do pacote, habilitação de segurança, modo de endereçamento utilizado e reconhecimento de pacote requisitado. (TEIXEIRA, L. M. 2006)

3. DESENVOLVIMENTO DO PROJETO

Este capítulo tem como objetivo apresentar a análise e desenvolvimento de um Sistema de Automação Residencial (SAR) capaz de gerenciar e executar comandos para ligar e desligar dispositivos de iluminação.

Será dividido em conceitos iniciais, custos de implantação, módulos do sistema e interação software/hardware.

3.1. Conceitos Iniciais

O SAR é um sistema de gerenciamento de automação residencial que tem como principal função o acionamento de dispositivos de iluminação. A partir de computadores remotos ou dispositivos móveis conectados a rede local é possível acessar o sistema, via navegador, e manter o cadastro de controladores, ambientes e dispositivos, bem como acionar os respectivos dispositivos.

O sistema é baseado em tecnologia web e foi desenvolvido com o intuito de proporcionar facilidade aos moradores de uma residência que necessitem ligar ou desligar lâmpadas através de um computador ou dispositivo móvel. Para proporcionar ainda mais conforto aos moradores, a tecnologia de comunicação ZigBee foi utilizado. Assim, a comunicação do sistema com os módulos controladores dos dispositivos de iluminação é totalmente sem fio.

Cada módulo controlador de dispositivos de iluminação é denominado no sistema de Controlador. O Controlador é uma placa RCOM-HOMEBEE (Figura 8). Esta placa contém dois relês, associados aos dispositivos de iluminação, um módulo XBee-Pro da Maxstream (Figura 9) e um microcontrolador PIC16F688, capaz de controlar as funções dos relês.

Figura 8: Figura placa RCOM-HOMEBEE.

Fonte: http://www.rogercom.com/

Figura 9: Módulo X-Bee Pro da Maxstream.

Fonte: http://www.trossenrobotics.com/store/i/is.aspx?path=/images/PImages/XbeePro60Wire.jpg

No cadastro de ambientes será possível cadastrar imagens dos ambientes da residência e associar um Controlador. Por exemplo, será feito o cadastro da sala de jantar. No módulo Ambiente do sistema será escolhida a imagem da sala de jantar e um Controlador para associar a este Ambiente.

O cadastro de dispositivos é feito na funcionalidade Dispositivo. Esta funcionalidade foi desenvolvida para associar os dispositivos de iluminação, em cada Ambiente, aos respectivos relês disponíveis no Controlador.

Por fim, o sistema conta com uma funcionalidade capaz de acionar os Dispositivos de cada ambiente. Esta funcionalidade é chamada de Controle Remoto.

3.1.1. Interações com os usuários

O sistema possui quatro funcionalidades que possuem restrições de acesso. O controle de acesso é feito com base em permissões pré-cadastradas: ENGENHEIRO, ADMINISTRADOR e MORADOR. O usuário com permissão de ENGENHEIRO tem acesso total ao sistema. Já o usuário com permissão de ADMINISTRADOR tem acesso às funcionalidades: Ambiente, Dispositivos e Controle Remoto e o usuário com permissão de MORADOR tem acesso somente à funcionalidade de Controle Remoto.

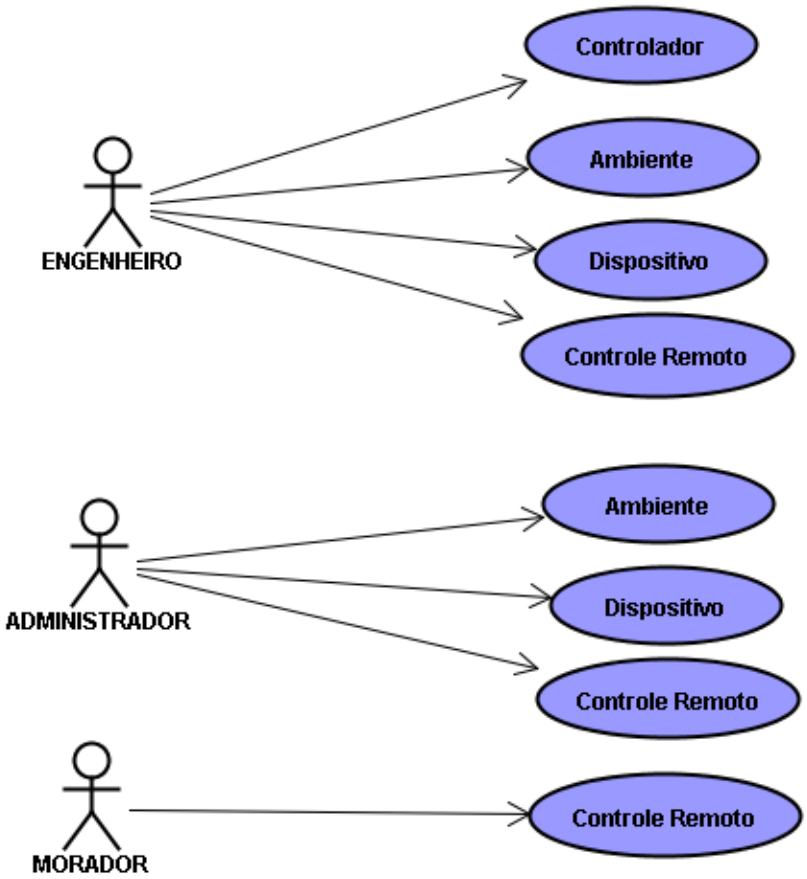


Figura 10: Diagrama de caso de uso.

Ao analisar a figura 10, pode-se notar que a funcionalidade de Controlador somente é acessível para usuários que tenham permissão de ENGENHEIRO. Isso porque ele, normalmente, é o único capaz de conhecer as especificações físicas e lógicas das placas RCOM-HOMEEBEE. As funcionalidades Ambiente e Dispositivo podem ser acessadas pelos usuários com permissão de ENGENHEIRO e ADMINISTRADOR. Já a funcionalidade Controle Remoto pode ser acessada por todos os usuários do sistema.

A funcionalidade Controlador é utilizada para manter o cadastro de módulos de controladores de iluminação. Esta fica disponível quando o login é realizado por um usuário com permissão de ENGENHEIRO, conforme figura 11. Ao

acessar a funcionalidade, o usuário poderá pesquisar Controladores já cadastrados, alterar ou excluir os existentes ou ainda incluir novos.

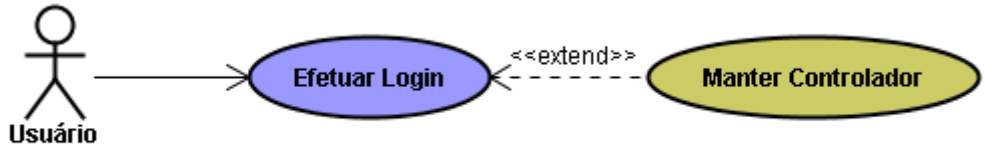


Figura 11: Diagrama de caso de uso: Controlador.

A funcionalidade Ambiente é utilizada para manter o cadastro de ambientes de uma casa. Esta fica disponível quando o login é realizado por um usuário com permissão de ENGENHEIRO ou ADMINISTRADOR, conforme figura 12. Ao acessar a funcionalidades, o usuário poderá pesquisar Ambientes já cadastrados, alterar ou excluir os existentes ou ainda incluir novos. Para a inclusão ou alteração de novos ambientes é necessário que se tenha uma imagem digitalizada do ambiente no formato jpg, jpeg, bmp ou gif.

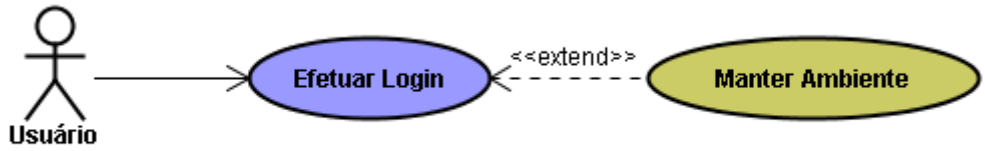


Figura 12: Diagrama de caso de uso: Ambiente.

A funcionalidade Dispositivo é utilizada para manter o cadastro de dispositivos de um ambiente. Esta fica disponível quando o login é realizado por um usuário com permissão de ENGENHEIRO ou ADMINISTRADOR, conforme figura 13. Ao acessar a funcionalidade, o usuário poderá pesquisar Ambientes já cadastrados para os quais se deseja incluir, alterar ou excluir dispositivos. Na inclusão de um novo dispositivo é necessário que ao menos um relê esteja disponível no Controlador associado ao Ambiente.

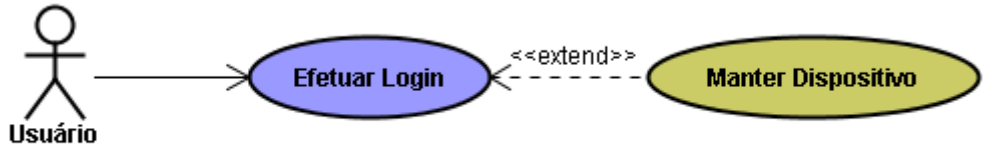


Figura 13: Diagrama de caso de uso: Dispositivo.

A funcionalidade Controle Remoto é utilizada para acionar os dispositivos cadastrados. Esta fica disponível quando o login é realizado por qualquer usuário do sistema, de acordo com a figura 14. Ao acessar a funcionalidade, o usuário poderá pesquisar o Ambiente onde o Dispositivo está cadastrado e assim ligar ou desligar os dispositivos.

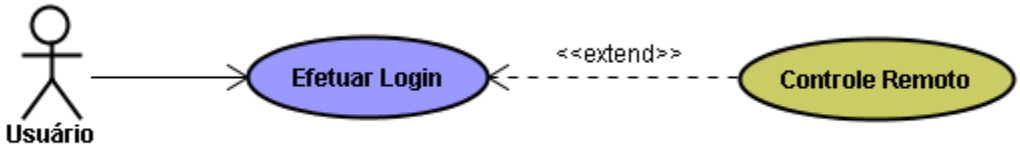


Figura 14: Diagrama de caso de uso: Controle Remoto.

3.1.2. Modelagem do Sistema

O desenvolvimento do sistema foi baseado na criação de camadas e a interação entre as camadas foi baseada no padrão de projeto *Model-View-Controller* (MVC). Esse padrão de projeto é utilizado para que haja a separação entre os dados (*Model*) e o layout (*View*). A vantagem da utilização do MVC é que alterações no layout não interferem na manipulação de dados e estes poderão ser reorganizados sem que haja a necessidade de alterar o layout.

O sistema foi dividido em três camadas: Apresentação, Negócio e Persistência.

A camada de apresentação representa o *View* no padrão MVC. Esta contém os *layouts* ou telas do sistema. Essas telas são desenvolvidas utilizando a linguagem de programação flex e são armazenadas em arquivos mxml. O *Controller* são arquivos *ActionScript (***AS***)* que recebem os dados inseridos na tela pelo usuário e os encaminham para a camada de negócio.

A camada de negócio é desenvolvida utilizando a linguagem de programação java e ,de acordo com a figura 15, observa-se que os dados enviados pelo *Controller* são recebidos por esta camada e processados de acordo com as regras de negócio do sistema. Em seguida, os dados, que deverão ser armazenados no banco de dados, são encaminhados para a camada de persistência. Logo após o processamento na camada de persistência, uma resposta é enviada para a camada de negócio. A resposta pode ser processada e encaminha para o *Controller* e este, por sua vez, processa-a e envia-a para View. É nesta camada, também, que as informações de solicitação de acionamento de dispositivos são processadas e encaminhadas para o hardware através dos módulos XBee-Pro.

A camada de persistência juntamente com a camada de negócio representa o *Model* no padrão MVC. É nesta camada que há a comunicação com o banco de dados. Assim, ao receber uma solicitação, esta camada poderá realizar operações de consultas, inserções, alterações ou exclusões.

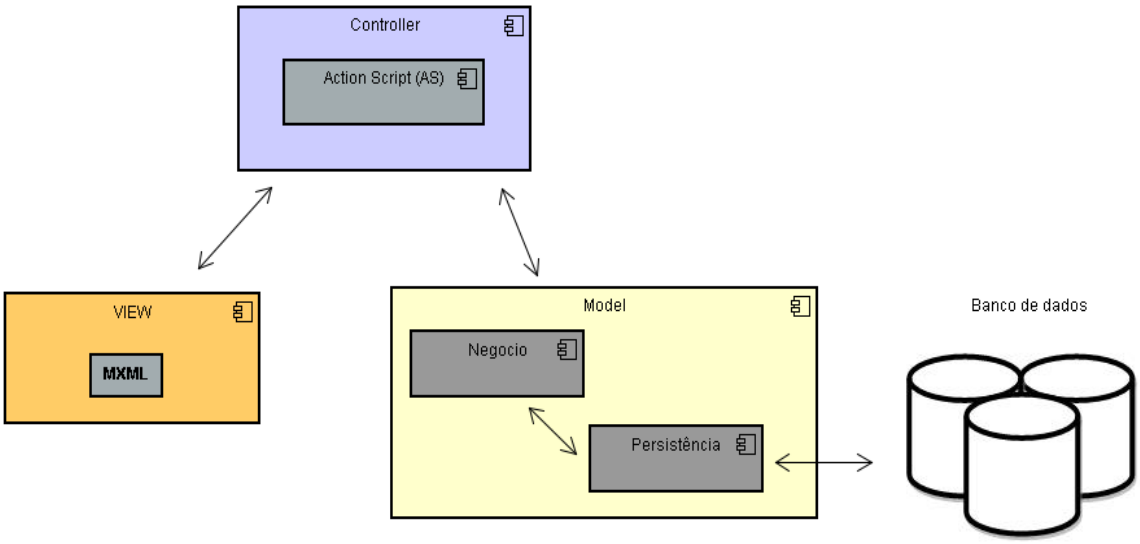


Figura 15: Modelagem do sistema SAR.

3.1.3. Modelagem de distribuição

O sistema será distribuído para funcionar como uma intranet na rede local de uma residência. Os computadores ou dispositivos móveis que tiverem acesso a rede local estarão aptos a acessar o sistema. Um computador que tenha uma porta *Universal Serial Bus* (USB) será utilizado como servidor web e servidor de banco de dados. A placa CON-USBBEE, figura 16, será conectada a porta USB. O sistema envia dados para a porta USB e através do módulo ZigBee coordenador, os dados são transmitidos a um módulo dispositivo especifico.

É importante lembrar que os navegadores utilizados, tanto dos computadores como dos dispositivos móveis conectados a rede local, devem ter suporte ao flash player 9 ou superior, pois a camada *View* é totalmente desenvolvida utilizando a linguagem de programação flex que é executada no flash player.

Figura 16: Placa CON-USBBEE.

A interação entre cada um dos componentes do sistema pode ser observada na figura 17.

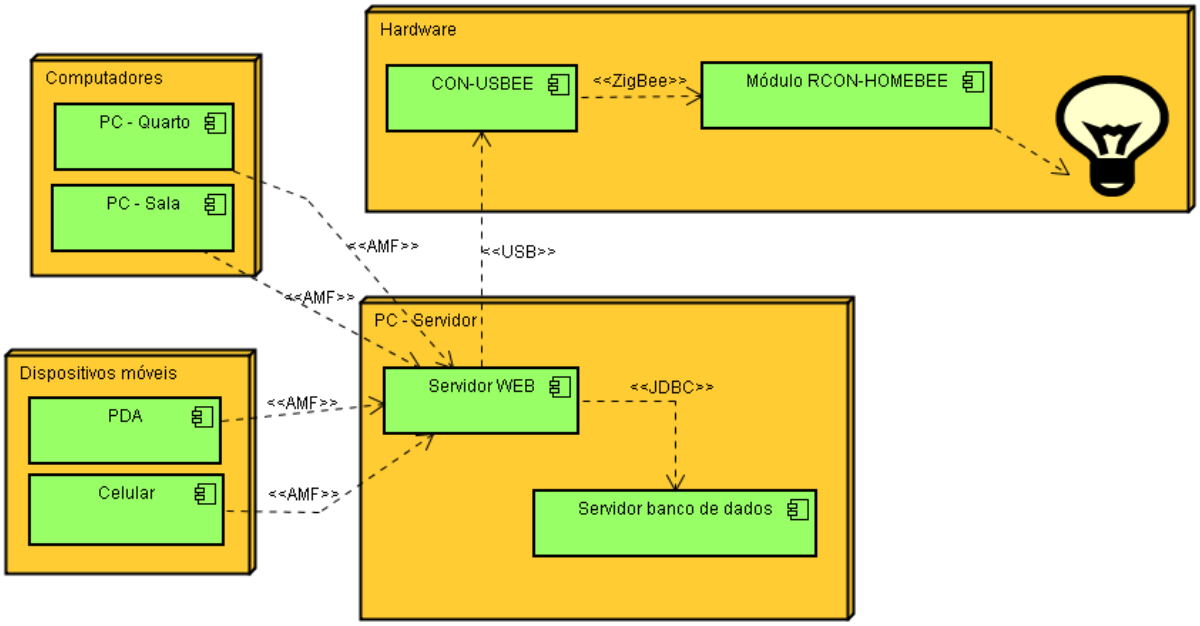


Figura 17: Diagrama de deployment.

3.1.4. Modelagem do Banco de Dados

O modelo de banco de dados foi desenvolvido com base na modelagem do sistema e nas regras de negócio apresentadas nos conceitos iniciais, modelagem do sistema e modelagem de distribuição.

O MySql Server 5.0 foi utilizado por ser um sistema de banco de dados livre, pela sua facilidade de uso, ter pouca exigência de recursos de hardware e compatibilidade com a linguagem de programação java.

Na modelagem do banco de dados existem 8 tabelas no total. As tabelas Menu, Permissao, RelMenuPermissao e Usuario fazem parte do controle de acesso ao sistema, logo, foram colocadas no *schema* CTA. As tabelas Ambiente, Controlador, Dispositivo e Porta são referentes ao sistema de automação residencial, logo, foram colocadas no *schema* SAR. As informações que constam nas tabelas do CTA são constantes. Essas informações são armazenadas no momento da implantação do sistema. Já as informações do SAR são variáveis sendo armazenadas no momento da execução do sistema.

Primeiramente, pela análise do *schema* CTA (figura 18), pode-se observar que um Usuario tem somente uma Permissao e vários Usuarios diferentes podem ter uma mesma Permissao. Portanto a tabela Usuario tem um relacionamento de **Muitos-para-Um** com a tabela Permissao. A tabela Menu representa cada um dos módulos do sistema. Um item Menu pode ter várias Permissoes e vários itens Menu podem ter uma mesma Permissao. Portanto a tabela Menu tem um relacionamento de **Muitos-para-Muitos** com a tabela Permissao. Para efetuar o relacionamento entre Menu e Permissao foi criada a tabela RelMenuPermissao. (Apêndice C)

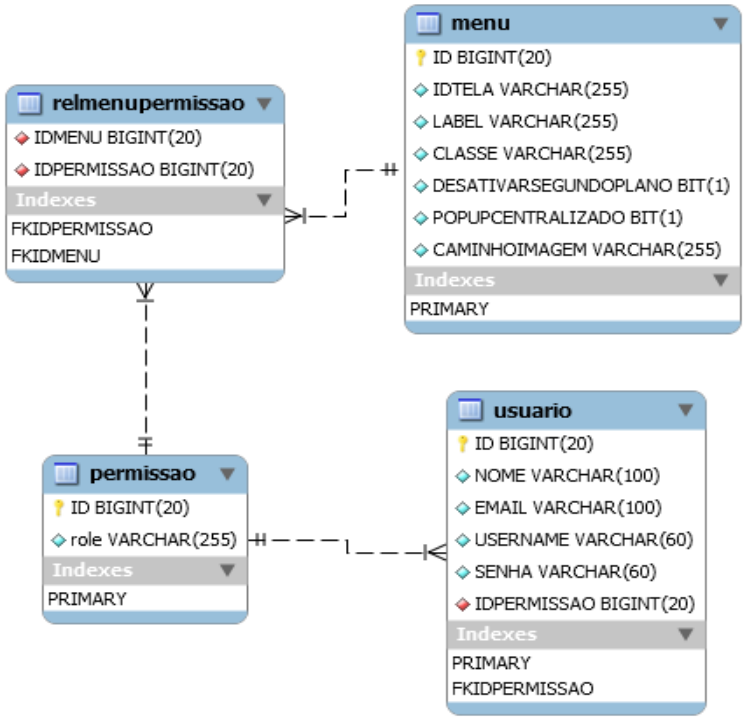


Figura 18: MER – Modelo de Entidade Relacionamento: schema CTA.

Logo em seguida, pela análise do *schema* SAR (figura 19), pode-se observar que para cada Controlador existem várias Portas. Portanto a tabela Controlador tem um relacionamento de **Um-para-Muitos** com a tabela Porta. Isso porque cada Controlador pode ter dois relês disponíveis para utilização. A tabela Ambiente tem somente um Controlador e cada Controlador pode controlar mais de um Ambiente. Portanto, a tabela Ambiente tem um relacionamento de **Muitos-para-Um** com a tabela Controlador. Por último, um Dispositivo deve estar associado a uma porta e

uma porta somente pode ter um Dispositivo. Portanto, a tabela Dispositivo tem um relacionamento de **Um-para-Um** com a tabela Porta. (Apêndice C)

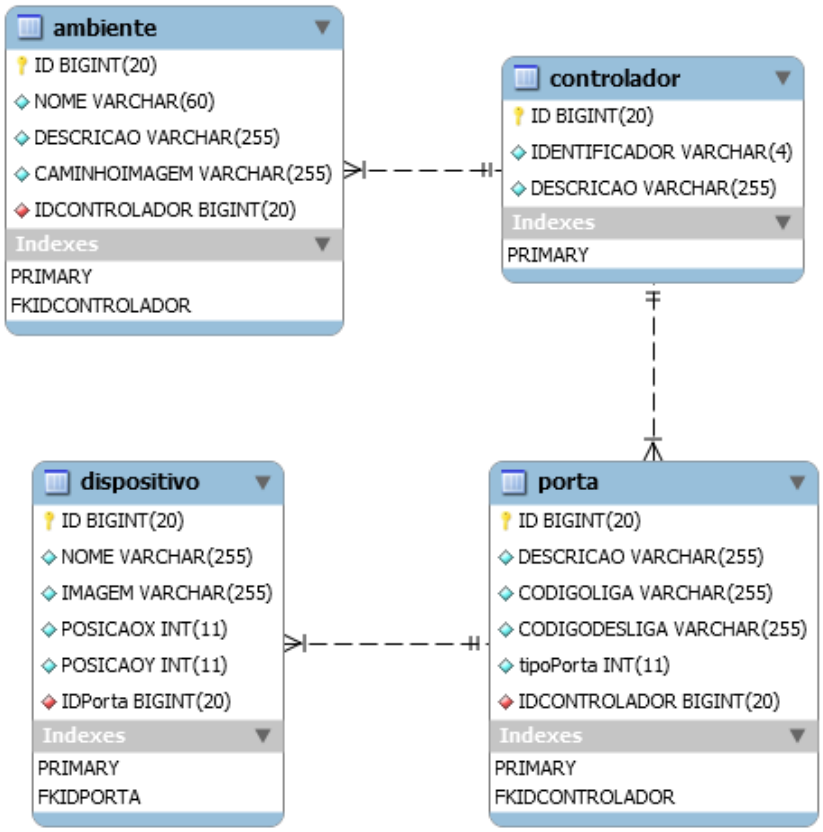


Figura 19: MER – Modelo de Entidade Relacionamento: schema SAR.

Para a modelagem do banco de dados foi utilizada a ferramenta MySql Workbench 5.0 que é uma ferramenta de design visual para banco de dados, open-source e que foi desenvolvida pela empresa MySql.

3.1.5. Implementação do Sistema

Para o desenvolvimento do sistema, alguns pré-requisitos devem ser observados. É necessária a instalação dos seguintes softwares: ambiente de desenvolvimento integrado (IDE) Eclipse 3.4 para a codificação java, Adobe Flex Builder 3: IDE baseada no Eclipse para codificação flex, servidor de aplicações JBoss 4.2.2 GA para dar suporte à aplicação web, banco de dados MySql 5 e o *Java Development kit* (JDK), composto por bibliotecas e pelo compilador java.

O desenvolvimento do sistema foi dividido em projetos flex e projetos java. Os projetos flex são responsáveis por conter as telas, componentes, imagens e classes de manipulação das telas. Já os projetos java são responsáveis por conter as classes de regras de negócio, classes de entidades do banco de dados, classes de acesso ao banco de dados, classes responsáveis pelo tratamento de erros e classes responsáveis pela comunicação do flex com o java.

Abaixo, segue o detalhamento de cada projeto utilizado no desenvolvimento do sistema.

- Projetos Flex
    - Componentes

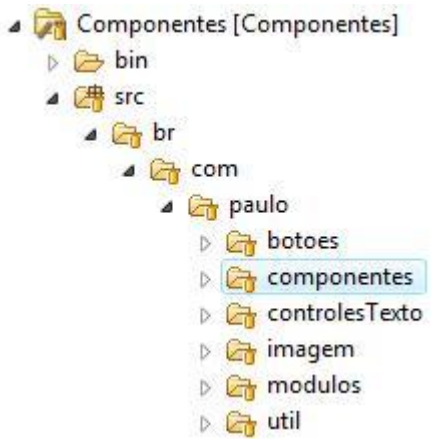


Figura 20: Detalhamento do projeto Componentes.

De acordo com a figura 20, a pasta “bin” é onde o arquivo compilado, contendo todos os itens desenvolvidos nesse projeto, se encontra. Dentro da pasta “src” segue uma estrutura de subpastas que foi escolhida como padrão para o sistema. Na subpasta “src.br.com.paulo.botoes” encontram-se arquivos responsáveis pelos componentes de botões utilizados no menu e nas telas do sistema. Na subpasta “src.br.com.paulo.componentes” encontram-se componentes de mensagens de alerta, bloqueio de tela, *upload* de arquivos, caixa de listagem de registros e componentes de validação de dados. A subpasta “src.br.com.paulo.controlesTexto” ficou reservada para os campos de texto, campos numéricos e áreas de texto. A subpasta “src.br.com.paulo.imagem” é responsável por agrupar todas as imagens utilizadas nesse projeto. Na subpasta

“src.br.com.paulo.modulos” encontram-se componentes de maior robustez. Esses componentes são responsáveis por definir o *layout* e funcionalidades padrão de todo o sistema. Por fim, na subpasta “src.br.com.paulo.util” encontram-se arquivos para realização de efeitos em imagens e botões, criação de janelas *popup*, *upload* de arquivo e serviços com chamadas aos projetos java.

- SARFlex

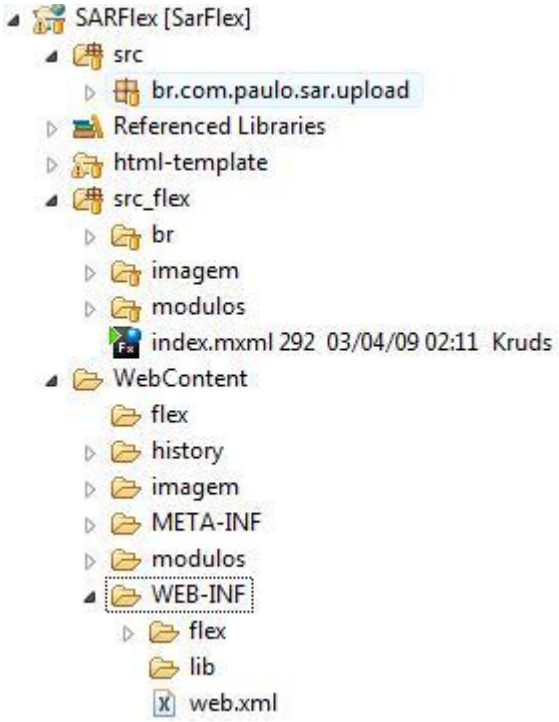


Figura 21: Detalhamento do projeto SARFlex.

De acordo com a figura 21, dentro da pasta “src”, há a subpasta “br.com.paulo.sar.upload” e nela encontra-se a classe responsável por receber e processar as requisições da funcionalidade de *upload* de imagens.

Dentro de “Referenced Libraries”, encontram-se arquivos de bibliotecas necessários para o funcionamento do sistema.

Na pasta “html-template” encontram-se alguns arquivos de configuração do projeto de uma aplicação flex.

A pasta “src-flex” é a mais importante do projeto. Na subpasta “br” encontram-se os arquivos de definição das telas do sistema, classes responsáveis pela

montagem do menu da aplicação, classes contendo os atributos de cada tabela do banco de dados e ainda classes responsáveis pelo armazenamento das informações que serão enviadas e recebidas quando efetuada a comunicação com o java. Na subpasta “imagem” encontram-se todas as imagens utilizadas nesse projeto. Na subpasta “modulos” encontram-se todas as classes responsáveis pelo controle das ações realizadas nas telas, inclusive manipular e encaminhar dados inseridos pelo usuário para os projetos java, assim como processar respostas encaminhadas pelos projetos java. O arquivo index.mxml é um arquivo de configuração de *layout* da tela inicial do sistema.

É na pasta “WebContent” que encontram-se todos os arquivos compilados e também arquivos de configuração da comunicação do flex com o java e de ambiente web. O arquivo web.xml, da subpasta “WEB-INF”, contém a configuração de qual classe deve ser executada primeiramente quando uma requisição for solicitada.

```
<servlet-class>flex.messaging.MessageBrokerServlet</servlet-class>
```

Contém também a informação de parâmetros necessário para identificadar quais classes java serão solicitadas durante a execução do sistema.

```
<init-param>
      <param-name>services.configuration.file</param-name>
      <param-value>/WEB-INF/flex/services-config.xml</param-value>
</init-param>
```

Alem de conter a classe que deverá ser executada ao realizar uma solicitação de *upload*.

```
<servlet-class>br.com.paulo.sar.upload.ServletUploadImagem</servlet-class>
```

Dentro da subpasta “WEB-INF/flex”, o arquivo remoting-config.xml é responsável por conter os identificadores e as classes java que serão solicitadas pelas requisições flex. Por exemplo:

```
<destination id="cadastrarControladoresFachada">
      <properties>
            <source>
                  br.com.paulo.sar.fachada.CadastrarControladoresFachada
```

```
            </source>
        </properties>
</destination>
```

- Projetos JAVA

    - Infra

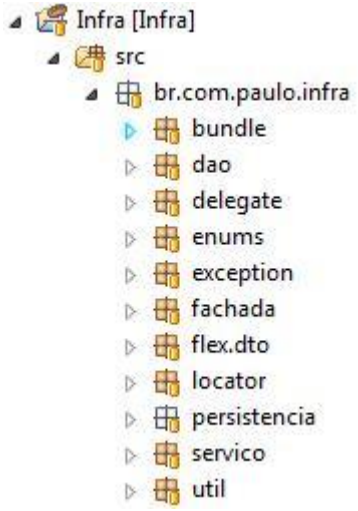


Figura 22: Detalhamento do projeto Infra.

O projeto Infra, figura 22, é responsável por prover aos outros projetos java a estrutura, desde a classe receptora das requisições do flex até a comunicação com o banco de dados, necessária para o desenvolvimento. A figura 23 ilustra um diagrama de classe contendo a estrutura base do sistema.

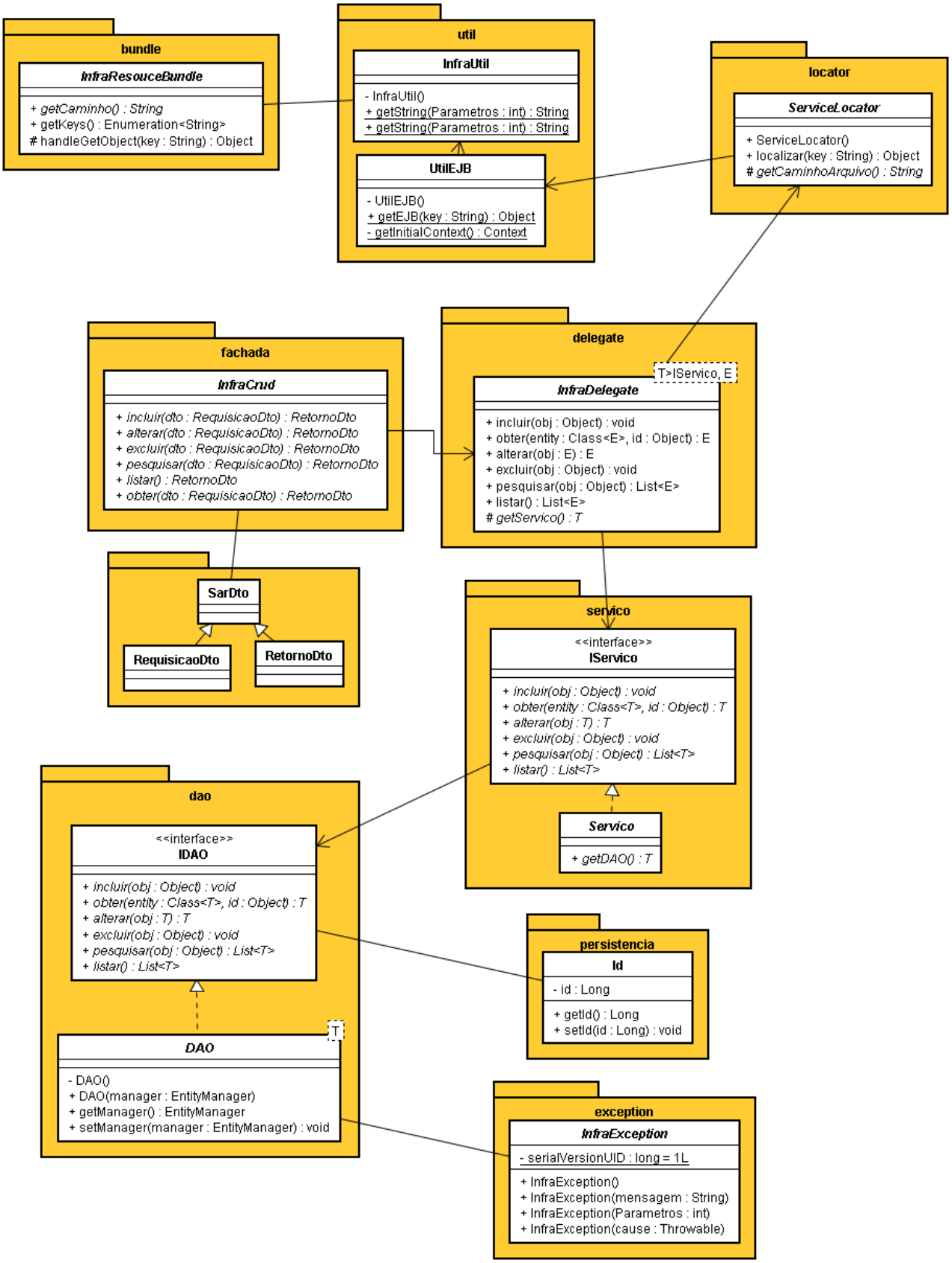


Figura 23: Diagrama de classes: Estrutura base do sistema.

O pacote “bundle” contém classes estruturais para a recuperação de mensagens que estão em arquivos de propriedades. O arquivo de propriedades contém, para cada mensagem, uma chave. A mensagem será recuperada a partir de sua respectiva chave. Para auxiliar no processo de recuperação de mensagem, a classe InfraUtil, localizada no pacote “util”, foi criada. A classe InfraResouceBundle, localizada no pacote “bundle” define um método abstrato responsável pela definição do caminho do arquivo da mensagem. Já a classe InfraUtil, defini um método para recuperar a mensagem a partir de uma chave e de um caminho de arquivo de mensagens específico.

O pacote “dao” contém um conjunto de classes estruturais que são utilizadas para a realização de pesquisas, inclusões, alterações e exclusões de dados no banco de dados.

O pacote “delegate” contém uma classe responsável por disponibilizar métodos que terão a funcionalidade de delegar uma requisição para as classes de serviços.

O pacote “enum” contém classes que representam listas de itens pré-definidos que poderão ser utilizados no desenvolvimento desse sistema ou em futuras melhorias.

O pacote “exception” contém um conjunto de classes responsáveis pelo tratamento de exceções lançadas pelas classes de acesso ao banco de dados e pelas classes de regras de negócio. Além das classes, esse pacote tem um subpacote, resources, contendo um arquivo de mensagens: mensagem.properties.

O pacote “fachada” contém uma classe utilizada para disponibilizar métodos que receberão as requisições do flex. Além disso, esses métodos terão a responsabilidade de executar algumas manipulações de dados, quando necessário, e então encaminhar uma solicitação para os “delegates”.

O pacote “flex.dto” contém classes responsáveis por encapsular os dados que serão enviados para o flex ou recebidos.

O pacote “locator” contém uma classe que tem a função de localizar um objeto do tipo Serviço. É realizada uma busca em um arquivo de mensagens, onde o conteúdo é do tipo chave = valor, e através de uma chave especifica é localizado o respectivo valor. Assim, este valor é utilizado para recuperar o objeto Serviço que se encontra no contexto da aplicação.

O pacote “persistencia” contém uma classe “Id” que é responsável por ter uma propriedade mapeada para as colunas PRIMARY KEY das tabelas no banco de dados. Todas as classes do sistema, que tenham mapeamento com o banco de dados, estendem essa classe “Id”.

O pacote “servico” é onde estão as classes estruturais para a realização de regras de negócios e é partir desses serviços que as classes responsáveis pelo acesso ao banco de dados serão chamadas. A classe “Servico” contém alguns métodos de funcionalidade padrão. Esses métodos realizam chamadas às classes DAO para a realização de consultas, inserções, alterações e exclusões.

O pacote “util” contém classes de utilidade padrão no sistema. Essas classes são responsáveis por localizar arquivos de mensagens e recuperar um valor a partir de uma chave específica ou ainda localizar um objeto que esteja no contexto da aplicação a partir de um valor específico.

- SegurancaEJB e SarEJB

O detalhamento desses projetos será feito juntamente com os módulos do sistema na seção 3.3 deste capítulo.

3.2. Custo para implementação

Tendo em vista a redução no custo do desenvolvimento do projeto, algumas medidas foram adotadas para que o produto final tivesse a qualidade desejada, a consistência esperada e um baixo custo.

Para a codificação java, o eclipse para desenvolvedores JEE foi o ambiente de desenvolvimento adoto por ser um software livre, possuir ferramentas para criação de projetos web e facilidade para depuração de códigos. Além disso, a utilização do ambiente de desenvolvimento flex, o Flex Builder 3, facilita a integração do código java com flex, pois, é um ambiente baseado no eclipse, além de ser distribuído gratuitamente para todos os clientes educacionais. Para criação dos diagramas, o software, *open source*, JUDE Community 5.5 foi utilizado. O servidor de aplicações open source JBoss foi adotado para dar suporte à aplicação web.

O banco de dados MySql Server 5.0 foi adotado por ser *open source* e possuir facilidade de integração com a linguagem java. Também foi utilizado o MySql Workbench. Um software *open source* para criação de modelo entidade-relacionamento.

Para a comunicação com o hardware foram adquiridos 2 módulos XBee-Pro no valor de R$ 177,00 cada um, 1 placa CON-USBBEE no valor de R$ 100,00 e 1 placa RCOM-HOMEBEE no valor de R$ 150,00. Também foram adquiridos alguns materiais como fios, fita isolante, bocais para lâmpadas e madeira no valor de R$ 20,00. A tabela 5 mostra o custo unitário de cada produto e o custo total do protótipo.

Tabela 5: Tabela de preço de produtos adquiridos.

Item	**Valor Unitário**	**Valor Total**	**Data Aquisição**
1 Placa CON-USBBEE	R$ 100,00	R$ 100,00	12/11/2008
1 Placa RCOM-HOMEBEE	R$ 150,00	R$ 150,00	12/11/2008
2 Módulos XBee-Pro	R$ 177,00	R$ 354,00	12/11/2008
2 Bocais para lâmpada	R$ 2,50	R$ 5,00	23/05/2009
2 Plugs macho	R$ 1,50	R$ 3,00	23/05/2009

2 Lâmpadas de 40W	R$ 2,00	R$ 4,00	23/05/2009
2 Metros de fio paralelo	R$ 1,50 / m	R$ 3,00	23/05/2009
Fita isolante	R$ 3,00	R$ 3,00	23/05/2009
Madeira	R$ 2,00	R$ 2,00	23/05/2009
TOTAL R$ 624,00			

Portanto, a qualidade e a consistência desejada foram alcançadas pela utilização de ferramentas de grande aceitação no mercado e no mundo do software livre. Além disso, o preço do hardware adquirido é acessível às pessoas que queiram dispor de um serviço de automação residencial.

3.3. Funcionalidades do sistema

Todo o sistema depende primeiramente da realização do login. A tela de Login (Figura 24) é solicitada sempre que um usuário iniciar o sistema.

Figura 24: Tela de Login.

Na tabela de Usuario, há três usuários pré-cadastrados. Cada usuário possui uma permissão diferente. É a permissão do usuário que defini quais funcionalidades estará disponível. O username deverá ser informado com sua respectiva senha. Caso o usuário seja inexistente ou a senha estiver errada, a mensagem de alerta “Username/Senha inválidos!” será mostrada. Ao efetuar o login no sistema, o usuário poderá acessar as funcionalidades, a ele disponibilizadas, através do menu. (Apêndice A)

3.3.1. Controlador

A funcionalidade **Controlador** é considerada crucial para o funcionamento de todo o projeto. Ela é a responsável por manter o cadastro dos módulos controladores de dispositivos de iluminação.

Para acessar essa funcionalidade**,** um usuário com a permissão “ENGENHEIRO” deverá estar logado no sistema.

Acessando a funcionalidade Controlador (Figura 25), o usuário poderá incluir novos controladores, pesquisar controladores já cadastrados ou fechar a tela. Os botões “Visualizar”, “Alterar” e “Excluir” estarão desabilitados enquanto um controlador já cadastrado não for selecionado na sessão “RESULTADO”.

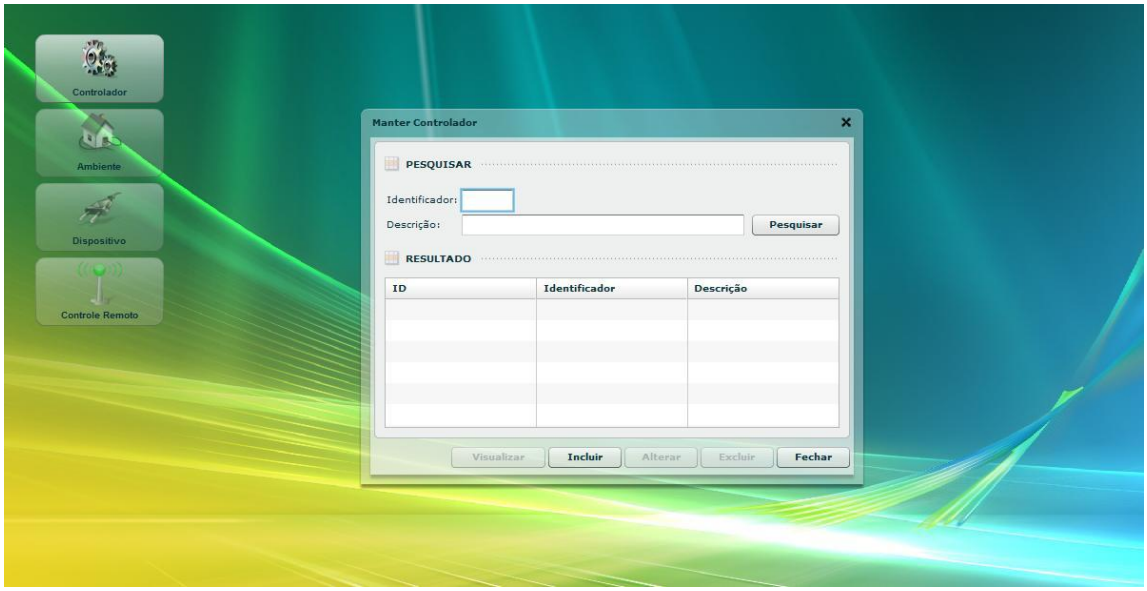


Figura 25: Tela Manter Controlador.

Se o usuário desejar incluir um novo controlador, deverá clicar no botão “Incluir” e a tela Incluir Controlador (Figura 26) será exibida.

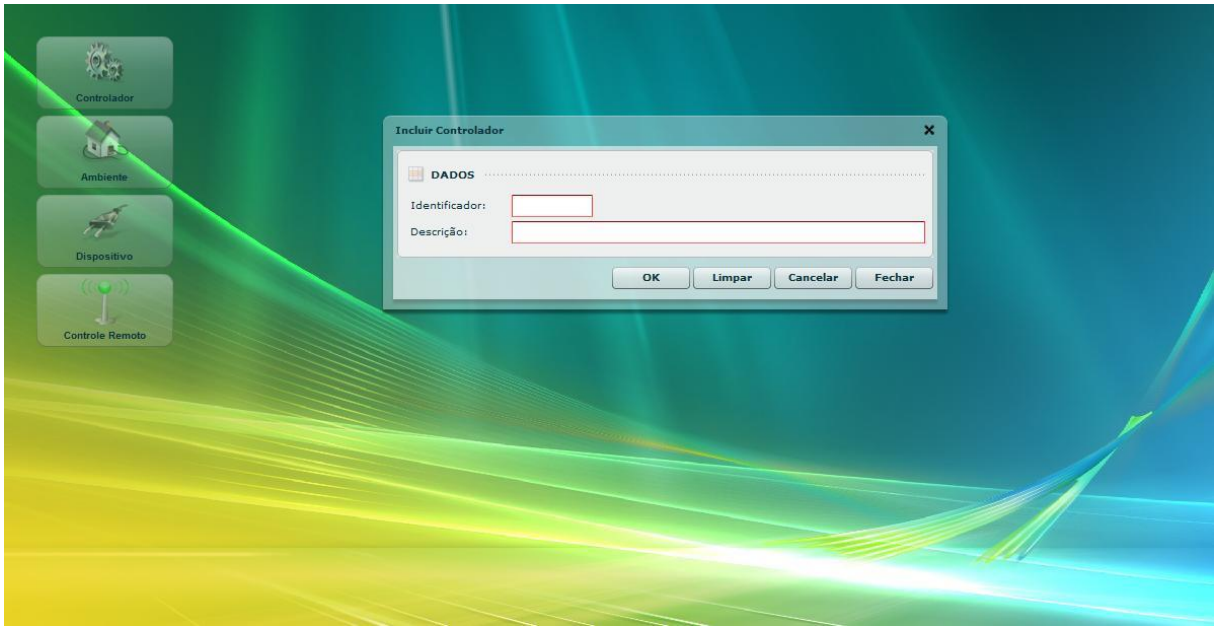


Figura 26: Tela Incluir Controlador.

Na sessão “DADOS”, o campo “Identificador” é obrigatório e deve ser preenchido de acordo com o identificador cadastrado para o módulo ZigBee acoplado a placa RCOM-HOMEBEE. O campo “Descrição” também é obrigatório e deve ser preenchido com uma descrição para identificar facilmente o controlador. Ao clicar no botão “OK”, um novo controlador com os dados informados será cadastrado no banco de dados. Nessa operação, também serão incluídas duas portas vinculadas ao controlador na tabela “Porta”. Isso porque cada placa RCOM-HOMEBEE possui dois relês. Ao clicar no botão “Limpar”, todos os dados informados serão apagados da tela. Ao clicar no botão “Cancelar”, a tela volta ao estado inicial e ao clicar em “Fechar” a tela é finalizada.

Caso o usuário queira visualizar, alterar ou excluir algum controlador, será necessário primeiramente pesquisar os controladores já cadastrados. Na sessão “FILTRO”, poderá ser informado o identificador e a descrição do controlador para filtrar a pesquisa. O resultado da pesquisa será mostrado na sessão “RESULTADO”, conforme figura 27. Ao selecionar um controlador, os botões “Visualizar”, “Alterar” e “Excluir” automaticamente serão habilitados.

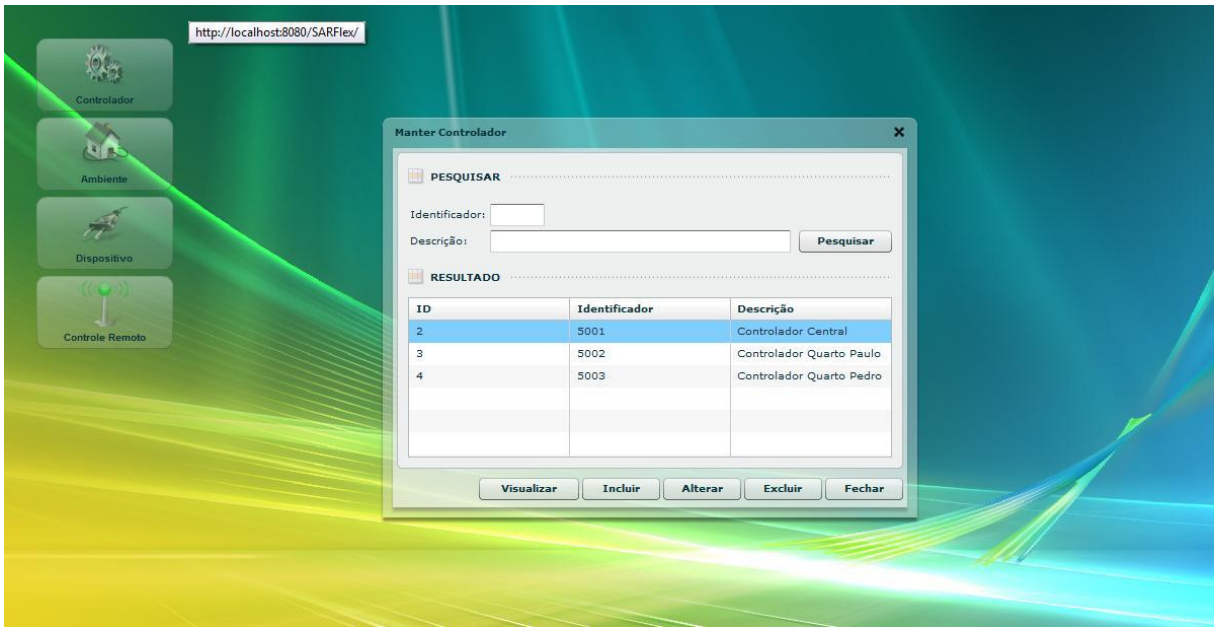


Figura 27: Tela Manter Controlador: Pesquisa de controlador.

Se o usuário desejar visualizar os dados de um controlador, deverá selecionar o controlador e clicar no botão “Visualizar” e a tela Visualizar Controlador (figura 28) será exibida com todos os campos desabilitados. O único botão disponível nessa tela é o botão “Fechar”.

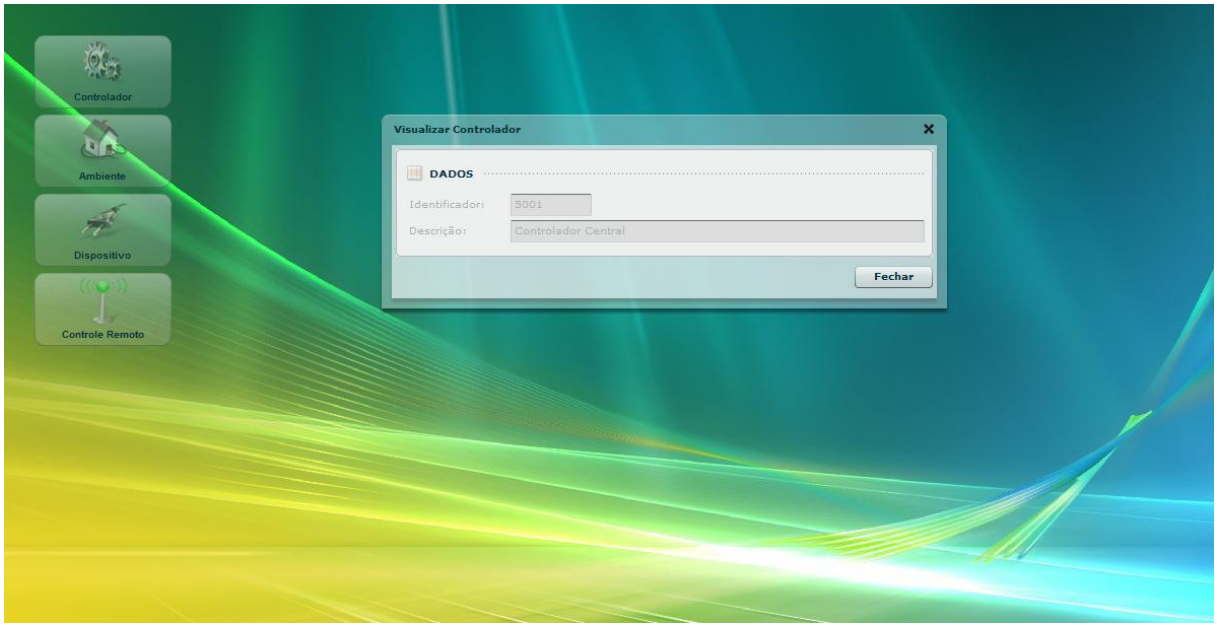


Figura 28: Tela Visualizar Controlador.

Se o usuário desejar alterar os dados de um controlador, deverá selecionar o controlador e clicar no botão “Alterar” e a tela Alterar Controlador (figura 29) será

exibida com os campos preenchidos de acordo com o controlador selecionado. Os botões dessa tela terão as mesmas funcionalidades da tela de Incluir Controlador.

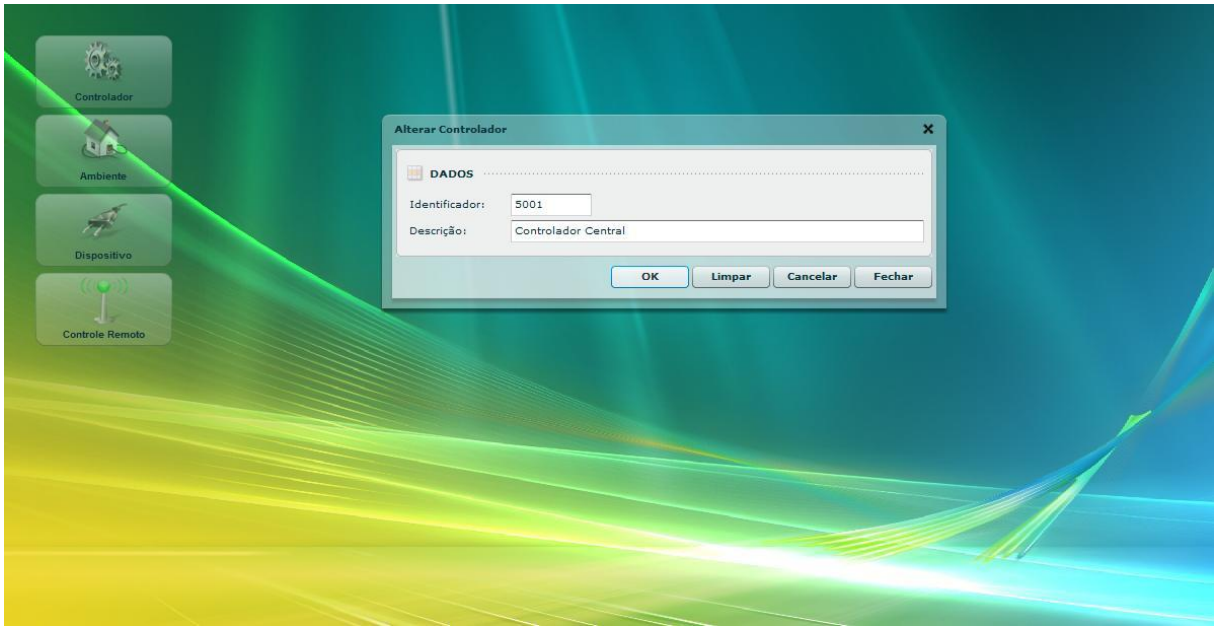


Figura 29: Tela Alterar Controlador.

Se o usuário desejar excluir um controlador, deverá selecionar o controlador e clicar no botão “Excluir”. O controlador será excluído automaticamente e a sessão “RESULTADO” será atualizada.

Para finalizar a funcionalidade **Controlador**, basta clicar no botão “Fechar”.

O detalhamento do código-fonte dessa funcionalidade está no Apêndice B.

3.3.2. Ambiente

A funcionalidade **Ambiente** é a responsável por manter o cadastro dos ambientes de uma residência. É nessa funcionalidade que um controlador será associado ao ambiente e também será feito o *upload* da imagem do ambiente.

Para acessar essa funcionalidade, um usuário com permissão de “ENGENHEIRO” ou “ADMINISTRADOR” deverá estar logado.

Acessando a funcionalidade Ambiente (Figura 30), o usuário poderá incluir novos ambientes, pesquisar ambientes já cadastrados ou fechar a tela. Os botões “Visualizar”, “Alterar” e “Excluir” estarão desabilitados enquanto um ambiente já cadastrado não for selecionado na sessão “RESULTADO”.

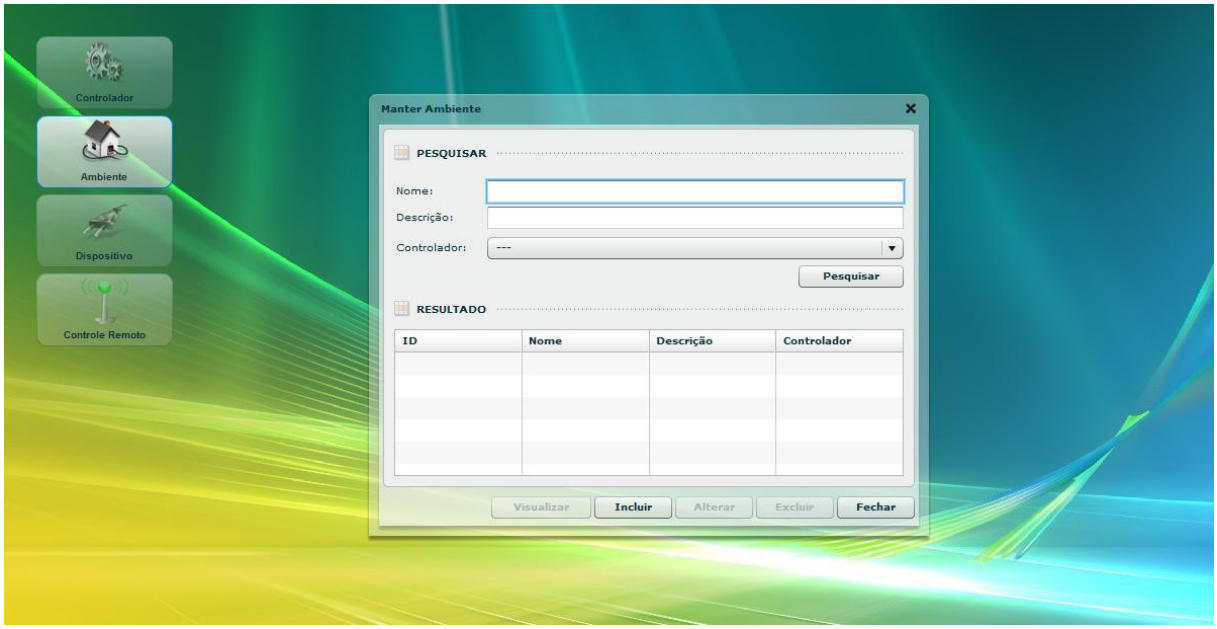


Figura 30: Tela Manter Ambiente.

Se o usuário desejar incluir um novo ambiente, deverá clicar no botão “Incluir” e a tela Incluir Ambiente (Figura 31) será exibida.

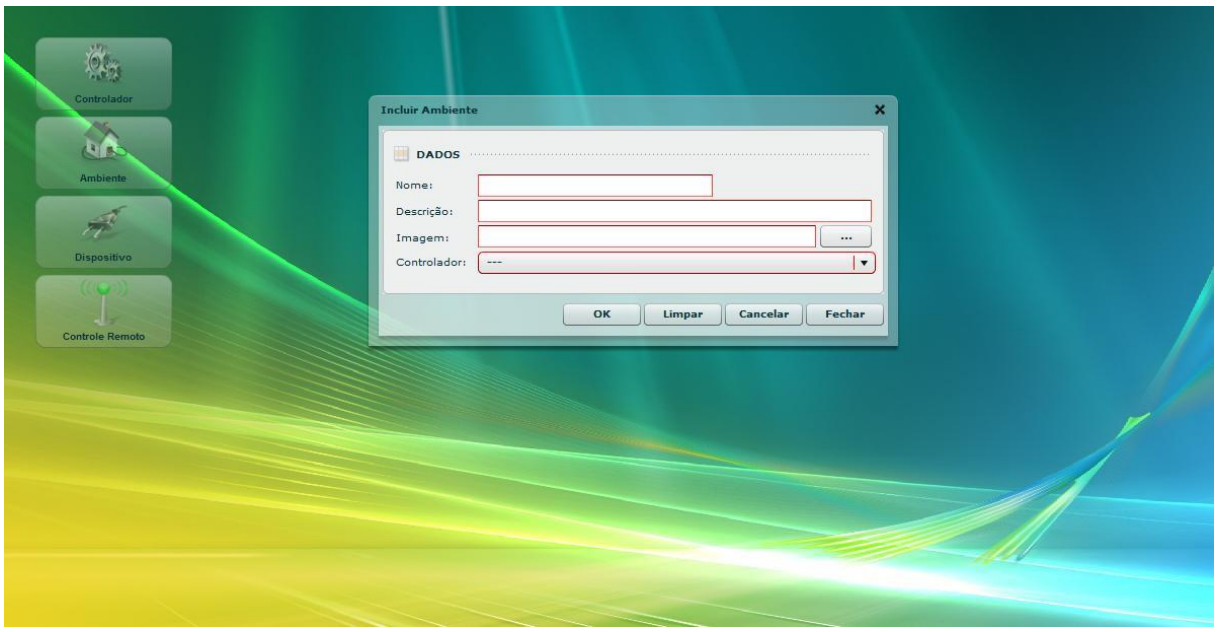


Figura 31: Tela Incluir Ambiente.

Na sessão “DADOS”, todos os campos são de preenchimento obrigatório. O campo “Nome” deve ser preenchido com um nome que identifique o ambiente, o campo “Descrição” é uma descrição resumida para o ambiente. O campo “Imagem” representa o nome da imagem do ambiente que será feito o *upload*, é por padrão desabilitado e será preenchido com o nome do arquivo que o usuário selecionar. Ao clicar no botão “...” uma caixa de seleção será exibida e o usuário poderá navegar pelas pastas do seu computador para selecionar a imagem do ambiente desejada. Ao iniciar a tela de inclusão, uma consulta à tabela Controlador do banco de dados é efetuada e todos os controladores disponíveis são recuperados. Os controladores recuperados serão dispostos no campo de seleção “Controlador”.

Ao clicar no botão “OK”, um novo ambiente com os dados informados e associado ao controlador selecionado será cadastrado no banco de dados. Ao clicar no botão “Limpar”, todos os dados informados serão apagados da tela. Ao clicar no botão “Cancelar”, a tela volta ao estado inicial e ao clicar em “Fechar” a tela é finalizada.

Caso o usuário queira visualizar, alterar ou excluir algum ambiente, será necessário primeiramente pesquisar os ambientes já cadastrados. Na sessão “FILTRO”, poderá ser informado o nome, descrição e o controlador associado ao ambiente para filtrar a pesquisa. O resultado da pesquisa será mostrado na sessão “RESULTADO”, conforme figura 32. Ao selecionar um ambiente, os botões “Visualizar”, “Alterar” e “Excluir” automaticamente serão habilitados.

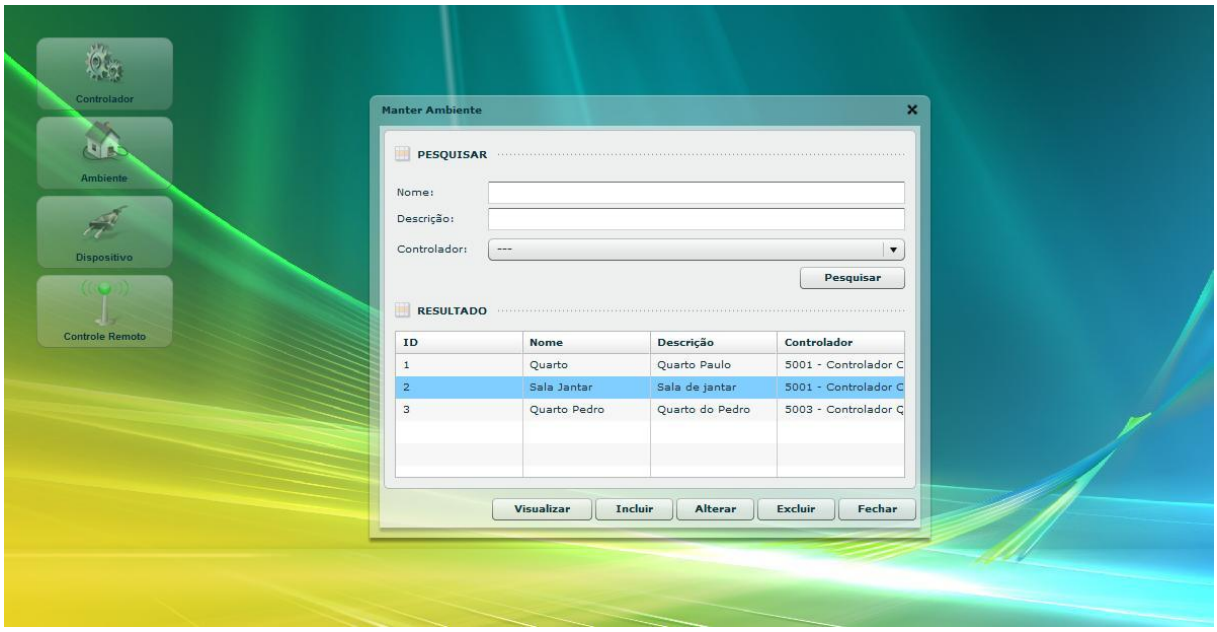


Figura 32: Tela Manter Ambiente: Pesquisa de ambiente.

Se o usuário desejar visualizar os dados de um ambiente, deverá selecionar o ambiente e clicar no botão “Visualizar” e a tela Visualizar Ambiente (figura 33) será exibida com todos os campos desabilitados. O único botão disponível nessa tela é o botão “Fechar”.

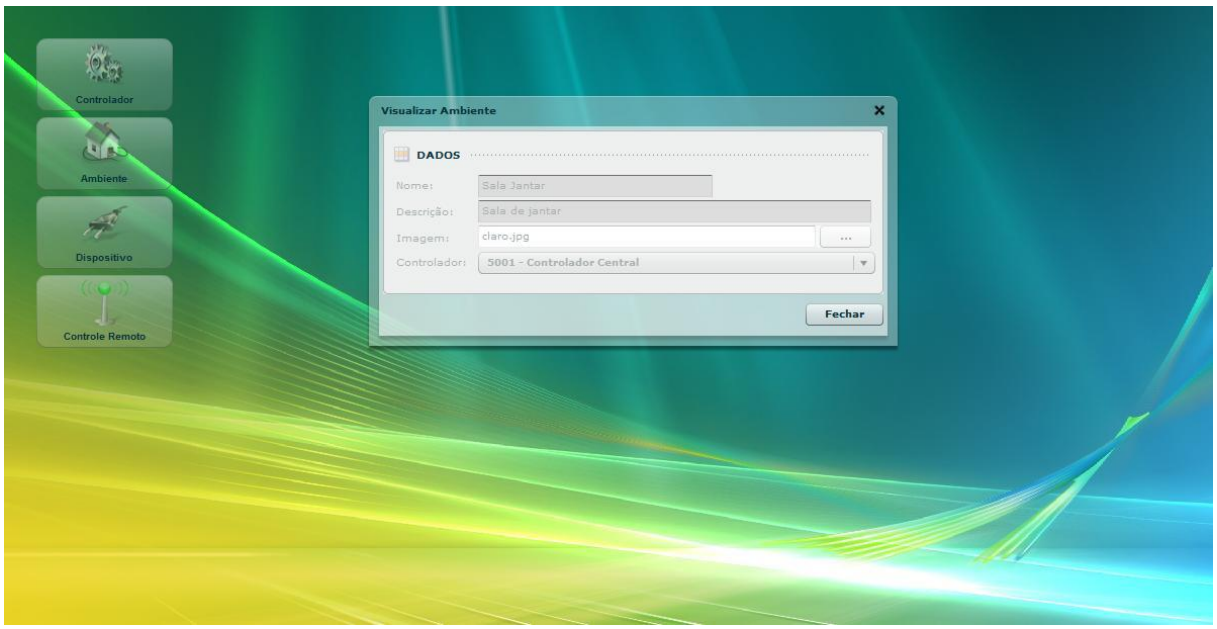


Figura 33: Tela Visualizar Ambiente.

Se o usuário desejar alterar os dados de um ambiente, deverá selecionar o ambiente e clicar no botão “Alterar” e a tela Alterar Ambiente (figura 34) será exibida com os campos preenchidos de acordo com o ambiente selecionado. Na alteração,

somente se a imagem do ambiente for alterada é que será feito o *upload* da imagem, assim evitando processamentos desnecessários. Os botões dessa tela terão as mesmas funcionalidades da tela de Incluir Ambiente.

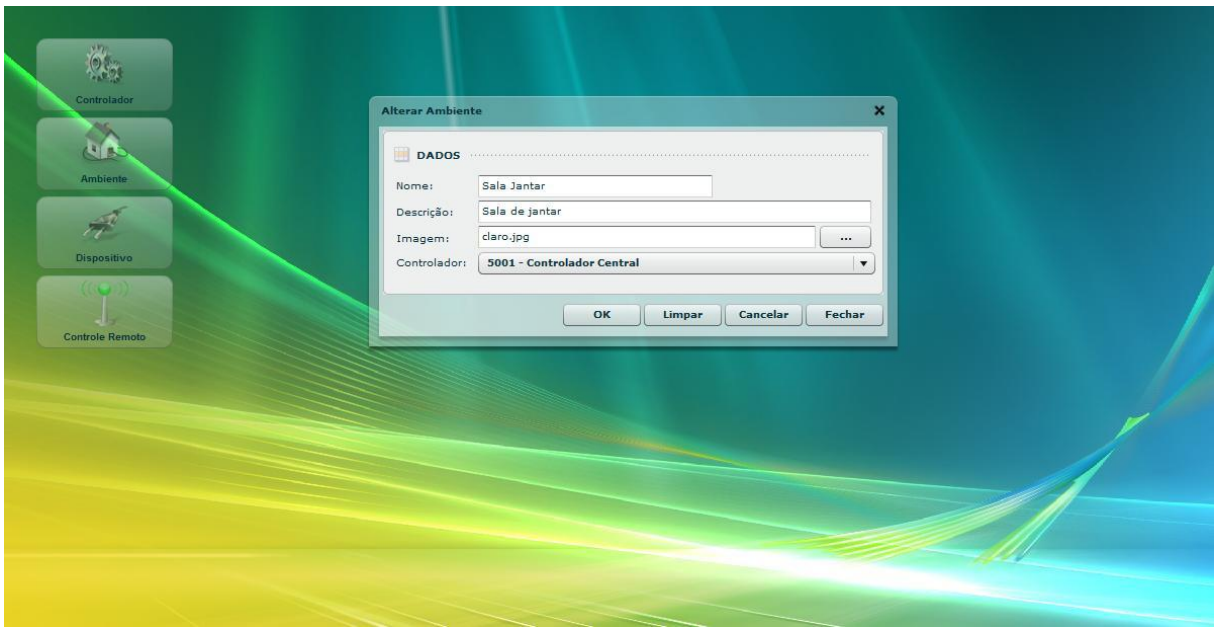


Figura 34: Tela Alterar Ambiente.

Se o usuário desejar excluir um ambiente, deverá selecionar o ambiente e clicar no botão “Excluir”. O ambiente será excluído automaticamente e a sessão “RESULTADO” será atualizada.

Para finalizar a funcionalidade **Ambiente**, basta clicar no botão “Fechar”.

O detalhamento do código-fonte dessa funcionalidade está no Apêndice B.

3.3.3. Dispositivo

A funcionalidade **Dispositivo** é a responsável por manter o cadastro dos dispositivos de um ambiente. É nessa funcionalidade que os dispositivos serão associados a um ambiente e a um relê específico do controlador.

Para acessar essa funcionalidade, um usuário com permissão de “ENGENHEIRO” ou “ADMINISTRADOR” deverá estar logado.

Acessando a funcionalidade Dispositivo (Figura 35), o usuário deverá primeiramente pesquisar o ambiente já cadastrado em que se deseja manipular os dispositivos. Após a seleção do ambiente na sessão “RESULTADO”, o usuário poderá incluir novos dispositivos, alterar a posição dos dispositivos ou ainda excluir dispositivos já cadastrados.

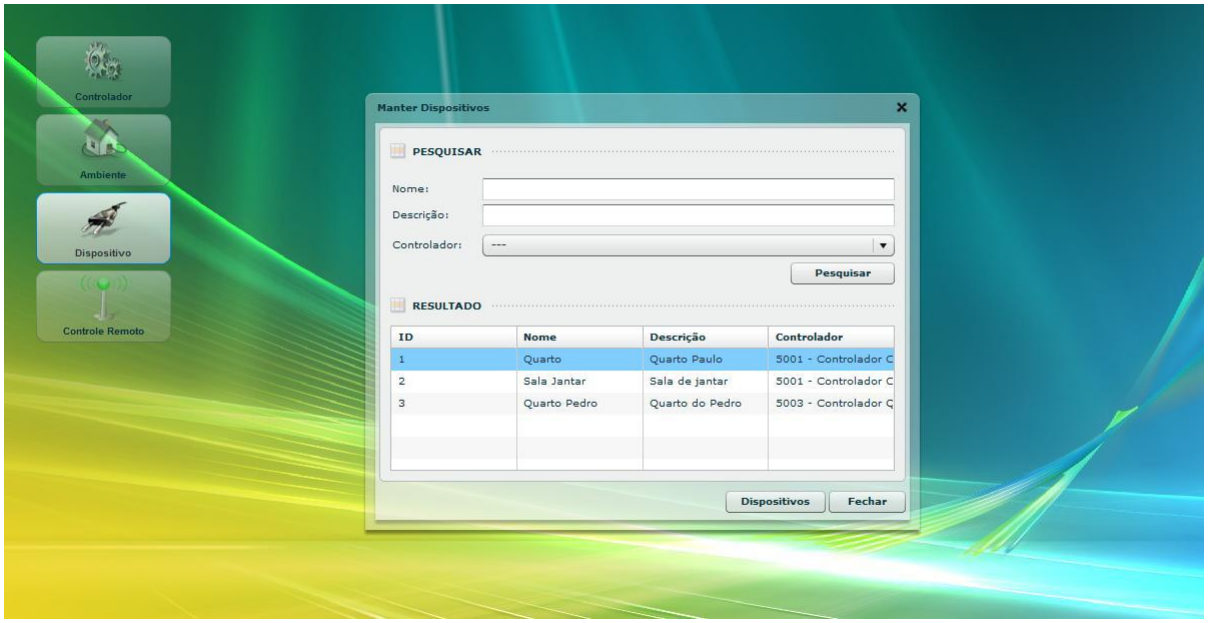


Figura 35: Tela Manter Dispositivos: Selecionar Ambiente.

Ao clicar no botão “Dispositivos”, a tela com o ambiente selecionado será exibida (Figura 36). Caso haja dispositivos cadastrados para o ambiente, eles estarão dispostos na tela. No lado direito da tela contém um menu com imagens de dispositivos de iluminação. Para incluir uma dessas imagens no ambiente, o evento *drag-andd-drop* (arrastar e soltar) deverá ser realizado (Figura 37). Ao finalizar o evento, a mensagem de aviso “É necessário adicionar uma porta para o dispositivo” será exibida (Figura 38). Neste momento, somente a sessão “DADOS DO DISPOSITIVO” estará habilitada. Uma porta deverá ser escolhida na caixa de seleção para ser associada ao dispositivo. Ao clicar no botão “Adicionar” a operação é realizada. Se o botão “Cancelar” for pressionado, a operação de inclusão do dispositivo é desfeita (Figura 39). Depois que um dispositivo for incluído, somente será possível a alteração da sua posição (Figura 40). Se desejar excluir, deverá utilizar o evento *drag-and-drop*. Ao arrastar o dispositivo, uma lixeira no canto direito inferior será exibida (Figura 41). O dispositivo deve ser solto nessa lixeira para a exclusão do mesmo (Figura 42).

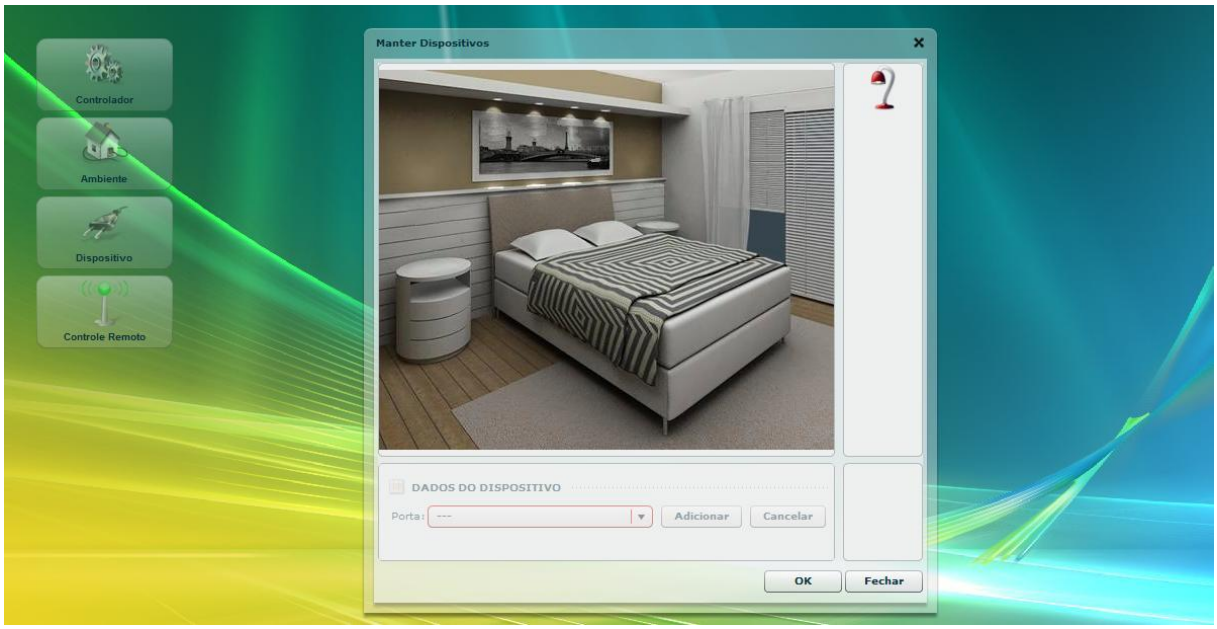


Figura 36: Tela Manter Dispositivos.

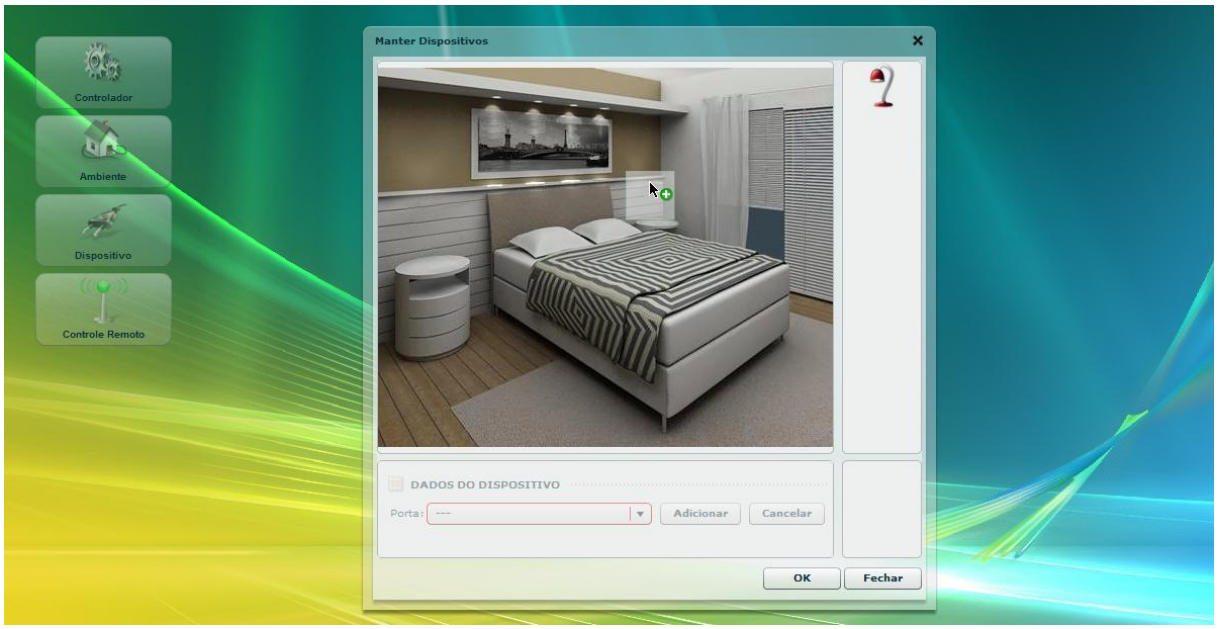


Figura 37: Tela Manter Dispositivos: Incluir Dispositivo.

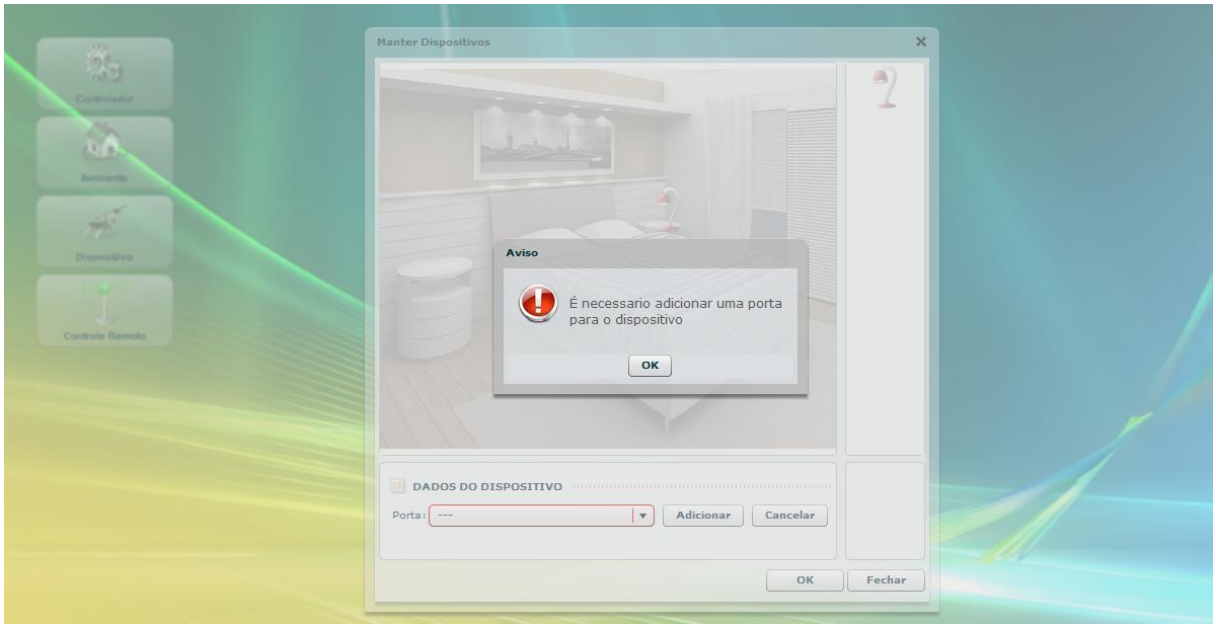


Figura 38: Tela Manter Dispositivos: Mensagem de aviso.

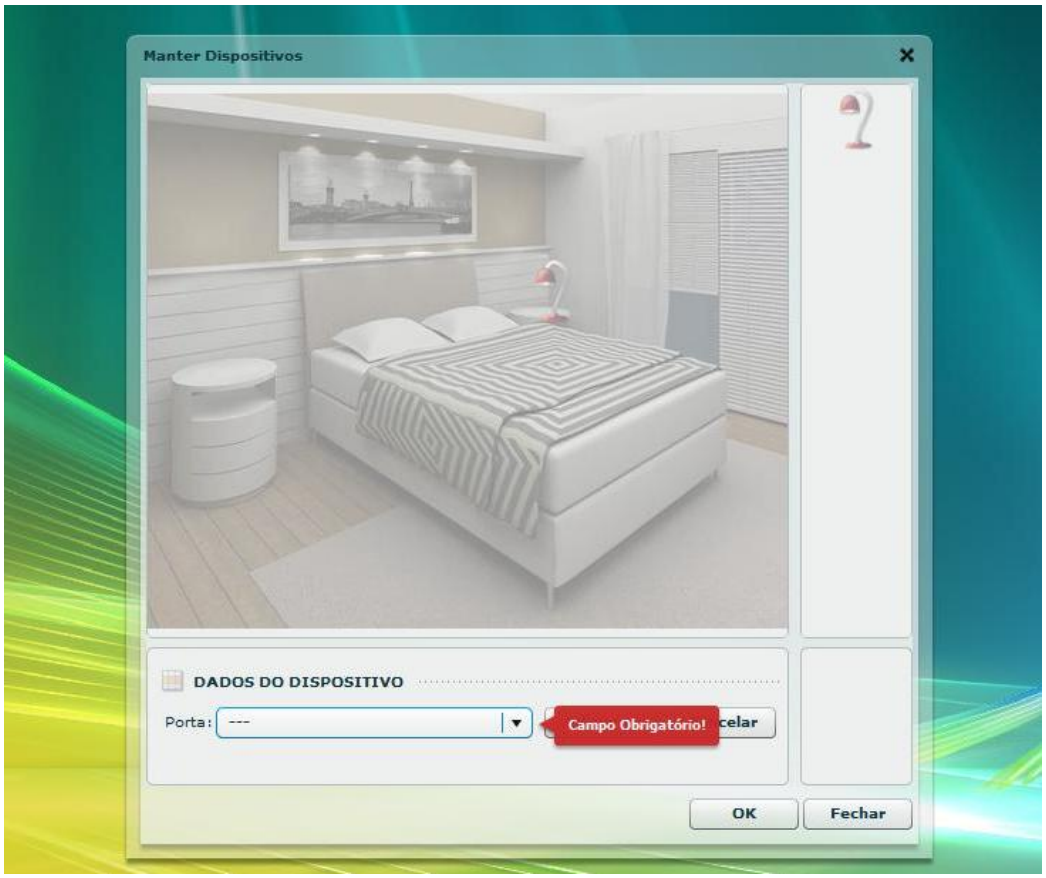


Figura 39: Tela Manter Dispositivos: Dados do Dispositivo.

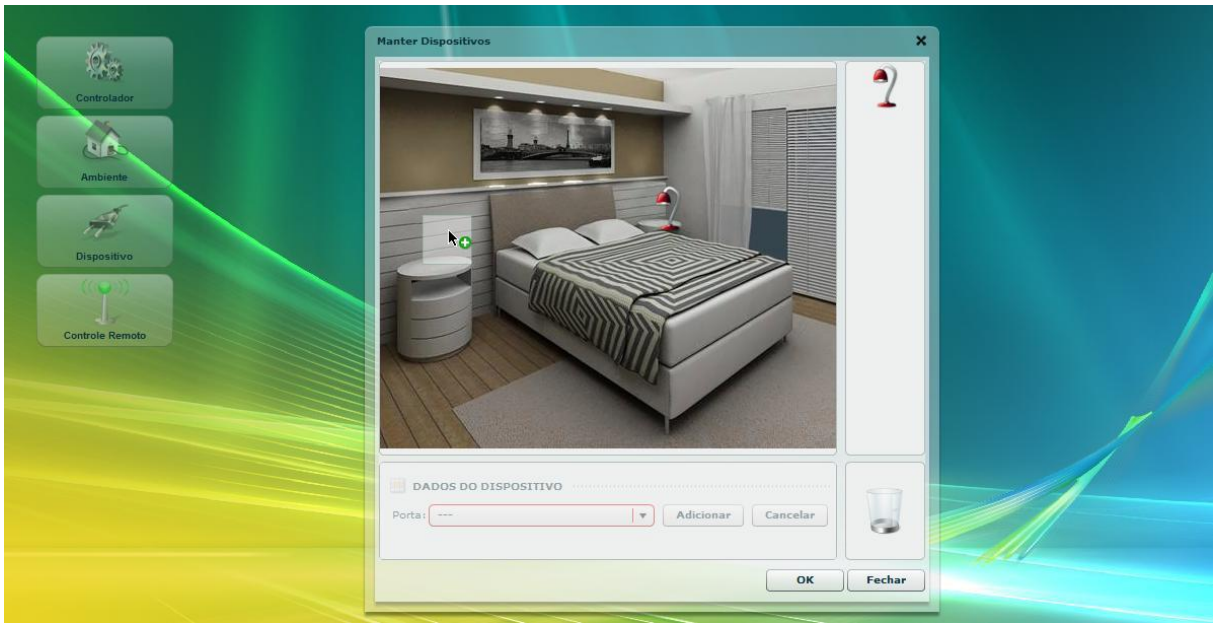


Figura 40: Tela Manter Dispositivos: Alterar posição do Dispositivo.

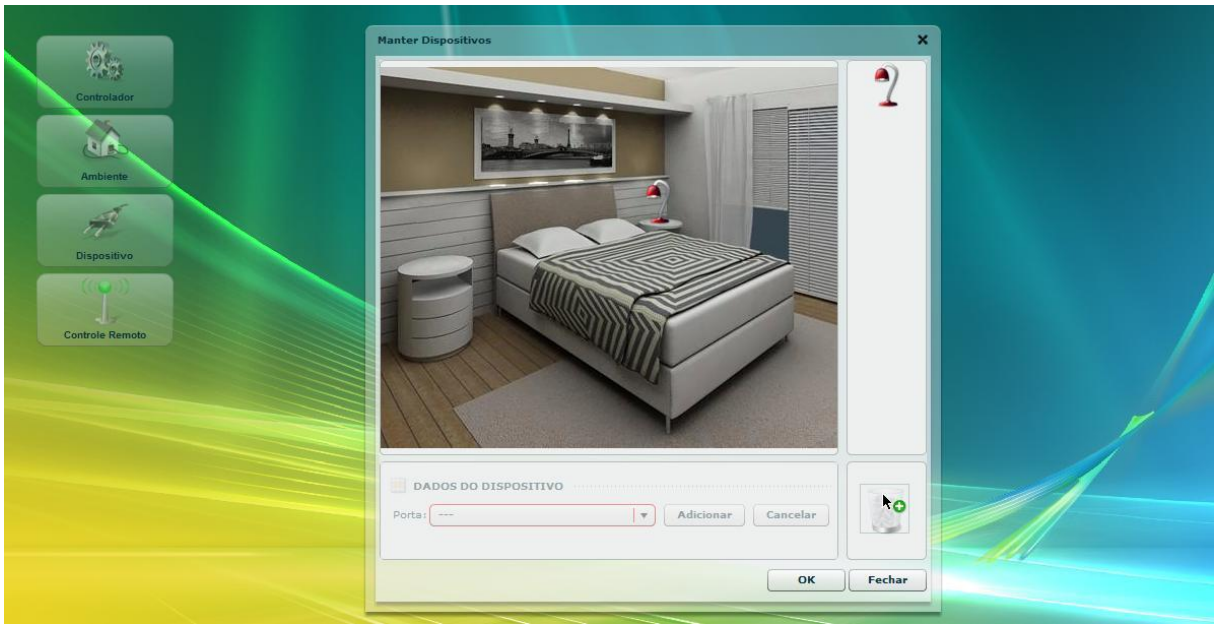


Figura 41: Tela Manter Dispositivos: Excluir Dispositivo.

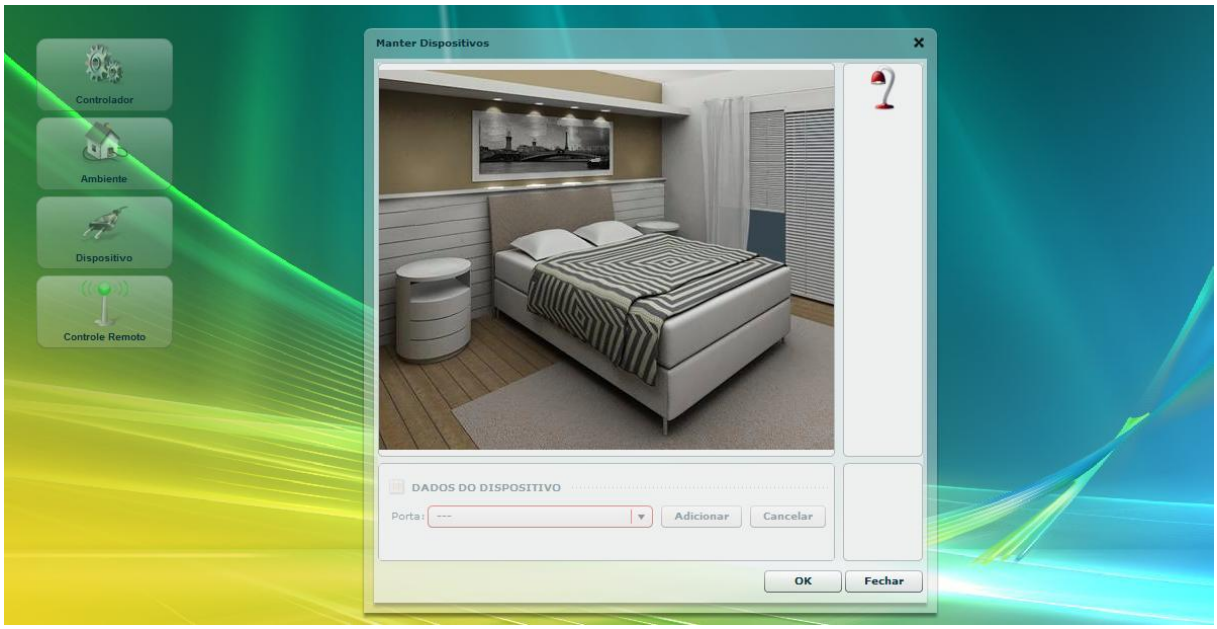


Figura 42: Tela Manter Dispositivos: Finalizar exclusão.

Ao termino da manipulação dos dispositivos, o botão “OK” deverá ser pressionado para que os dados sejam processados no banco de dados. Se o botão “Fechar” for pressionado, a funcionalidade é encerrada e os dados serão perdidos.

Para finalizar a funcionalidade **Dispositivo**, basta clicar no botão “Fechar”.

O detalhamento do código-fonte dessa funcionalidade está no Apêndice B.

3.3.4. Controle Remoto

A funcionalidade **Controle Remoto** é a responsável por acionar os dispositivos cadastrados para um ambiente. É nessa funcionalidade que haverá a interação do software com o hardware. Os dispositivos serão ligados e desligados fisicamente.

Para acessar essa funcionalidade, um usuário com permissão de “ENGENHEIRO”, “ADMINISTRADOR” ou “MORADOR” deverá estar logado.

Acessando a funcionalidade Controle Remoto (Figura 43), o usuário deverá primeiramente pesquisar o ambiente já cadastrado em que se deseja acionar os

dispositivos. Após a seleção do ambiente na sessão “RESULTADO”, o usuário poderá controlar os dispositivos cadastrados.

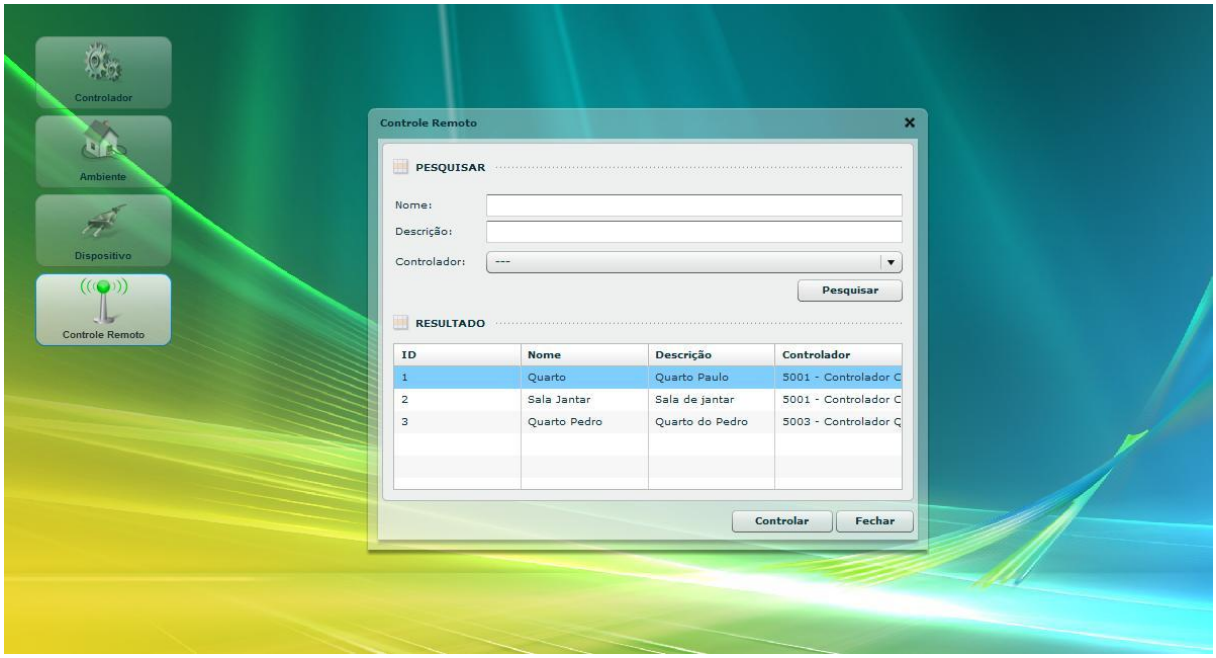


Figura 43: Tela Controle Remoto: Selecionar Ambiente.

Ao clicar no botão “Controlar”, a tela com o ambiente selecionado será exibida (Figura 44) e uma função para verificar o estado de cada dispositivo será executada. Os dispositivos cadastrados para o ambiente estarão dispostos na tela em forma de botões, contendo a imagem do dispositivo.

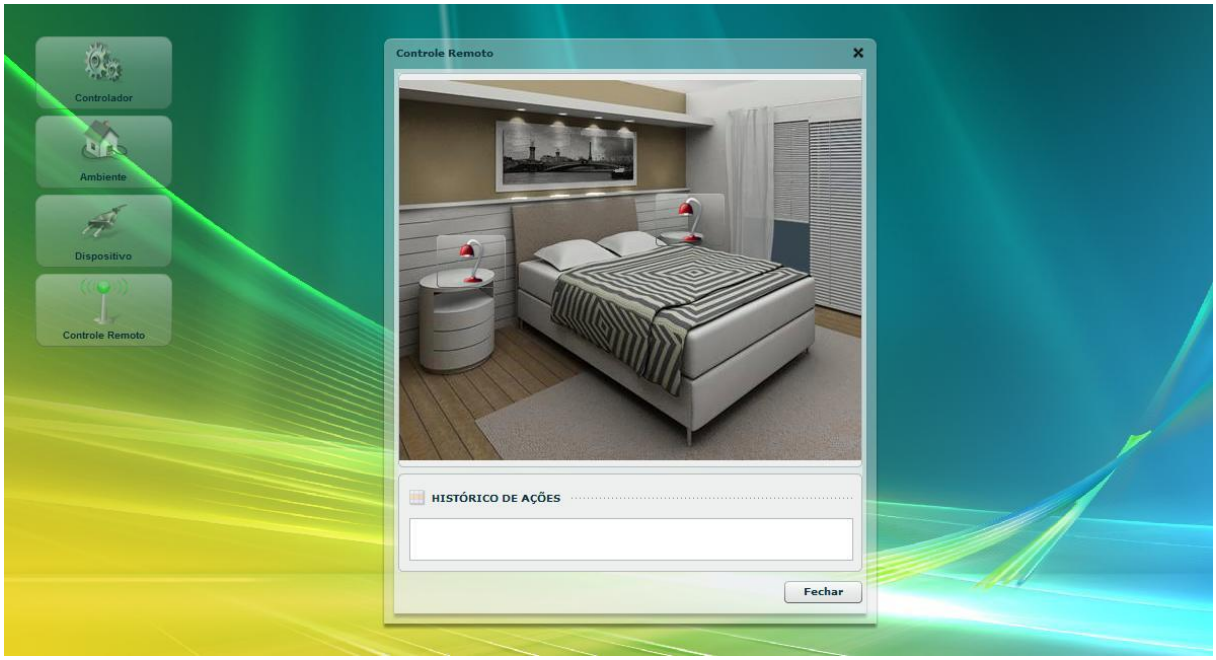


Figura 44: Tela Controle Remoto.

Para ligar ou desligar um dispositivo é necessário que o botão do dispositivo desejado seja pressionado. Será executada uma função para verificar o estado do respectivo dispositivo. Caso a função retorne o estado de “Ligado”, será executado o comando para desligar o dispositivo. Caso contrario, será executado o comando para ligar o dispositivo. Sempre que um comando for executado, o mesmo será exibido na sessão “HISTÓRICO DE AÇÕES” (Figura 45).

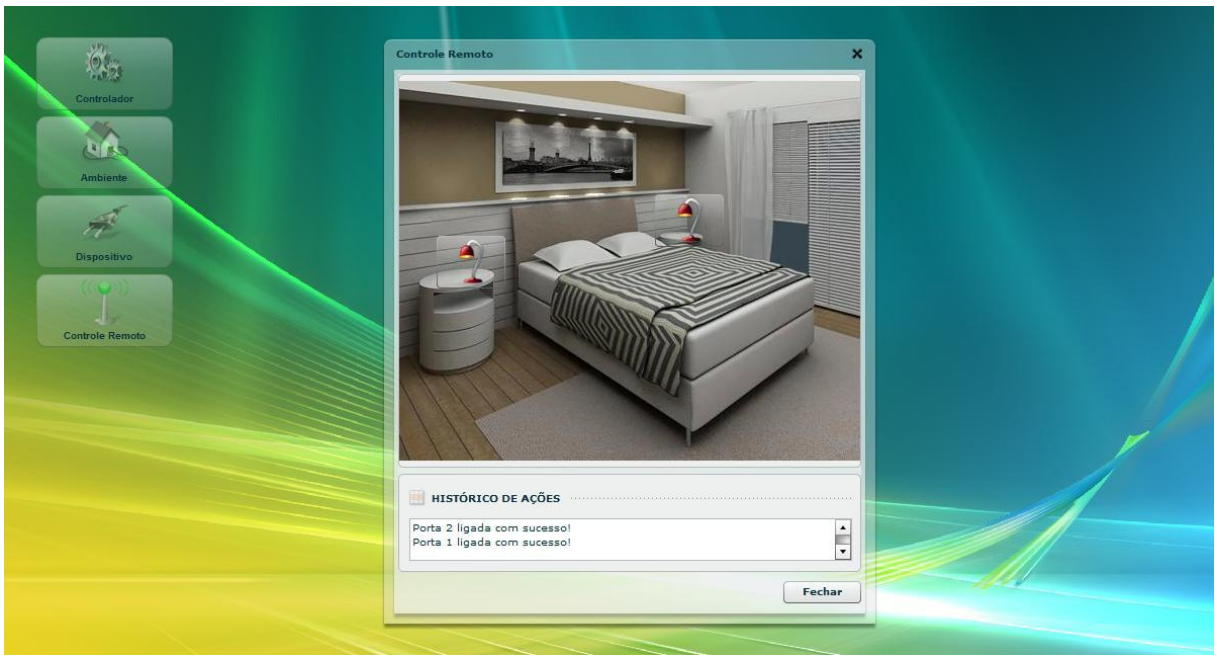


Figura 45: Tela Controle Remoto: Histórico de ações.

Para finalizar a funcionalidade **Controle Remoto**, basta clicar no botão “Fechar”.

O detalhamento do código-fonte dessa funcionalidade está no Apêndice B.

3.4. Interação software/hardware

3.4.1. Detalhamento do hardware

O hardware é composto pela placa CON-USBBEE e pela placa RCOM-HOMEE. A cada uma dessas placas é acoplado um módulo XBee-Pro. Um destes módulos funciona como o coordenador da rede ZigBee, portanto é um dispositivo FFD, enquanto o outro será o *end device* que também é um dispositivo FFD. A implementação da rede será baseada na topologia estrela, a faixa ISM utilizada será 2.4 GHz com uma taxa de transmissão de até 250kbps. As figuras 46, 47 e 48 representam uma visão do protótipo do projeto.

Figura 46: Visão do protótipo do projeto.

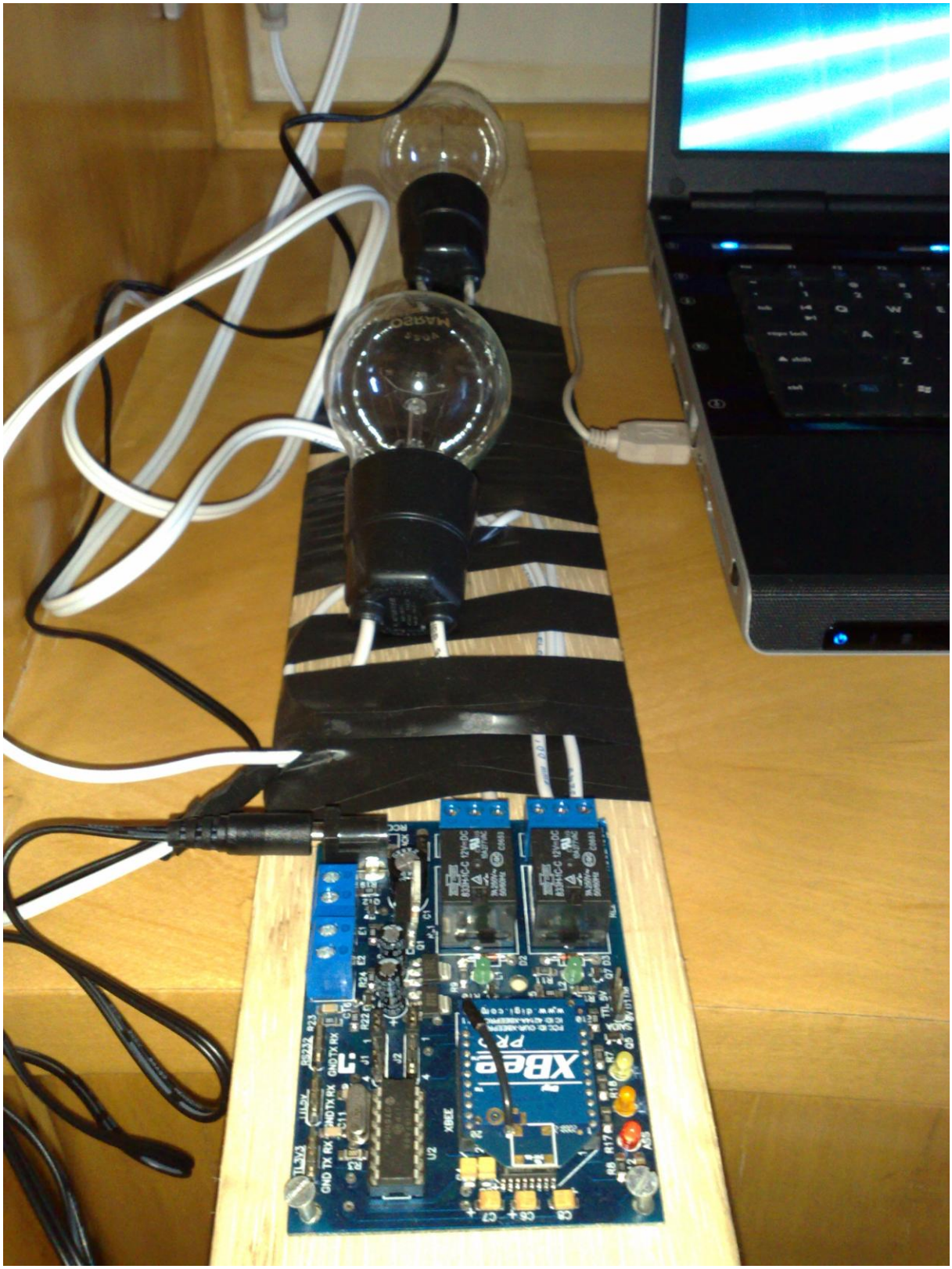


Figura 47: Visão do protótipo do projeto. Placa RCOM-HOMEBEE.

Figura 48: Visão do protótipo do projeto. Placa CON-USBBEE.

3.4.1.1. Placa CON-USBBEE + Módulo XBee-Pro

A placa CON-USBBEE foi projetada para facilitar a conexão do módulo XBee-Pro ao computador. De acordo com o datasheet, a placa CON-USBEE possui chip conversor USB/serial, regulador de tensão LDO (*Low Drop Out*), três *Light Emitting Diode* (LEDs) que estão conectados a um detector de *Received Signal Strength Indication* (RSSI) para indicação da força do sinal de *Radio frequency* (RF), um LED Indicador de transmissão (TX), um LED Indicador de recepção (RX) e um LED indicador de módulo ligado. (Anexo C)

O fornecedor da placa disponibiliza um *driver* que deve ser instalado para que quando a placa for conectada ao computador, uma porta COMx virtual será criada. Sendo assim, será possível acessar a placa como se fosse uma comunicação serial padrão RS232.

Esse módulo será o coordenador da rede ZigBee implementada nesse projeto sendo o responsável por enviar requisições para o módulo RCOM-HOMBEE, ligar ou desligar as lâmpadas associadas, e receber do módulo RCOM-HOMEBEE as respostas das operações realizadas.

3.4.1.2. Placa RCOM-HOMEBEE + Módulo XBee-Pro

Esta placa foi projetada para facilitar a automação de determinados ambientes de uma residência, indústria, escritório ou qualquer outro dispositivo sujeito a automação. De acordo com o datasheet, a placa RCOM-HOMEBEE deve ser alimentada com uma fonte de alimentação de 12-24 v / 600 mA. Ela possui duas saídas a relês, onde serão ligadas as lâmpadas, que podem ser utilizadas para ligar e desligar dispositivos com tensão de até 220v e corrente de 10A. Além disso, um microcontrolador PIC16F688 foi utilizado sendo o responsável por controlar as funções dos relês. Há na placa também dois LEDs de cor verde que quando acessos indicam que os relês estão ligados, um LED vermelho para indicar que o módulo XBee-Pro está ligado, um LED laranja para indicar TX e um LED amarelo para indicar RX. (Anexo B)

Este módulo será o *end device* da rede ZigBee implementada nesse projeto sendo o responsável por receber as informações do coordenador, processá-las e então encaminhar uma resposta para o módulo coordenador.

3.4.1.3. Configuração de parâmetros dos módulos XBee-Pro

Para configurar os parâmetros dos módulos XBee-Pro é necessário a utilização do software X-CTU (figura 49) que é fornecido pela MaxStream. Todas as configurações são feitas na aba “*Modem Configuration*”.

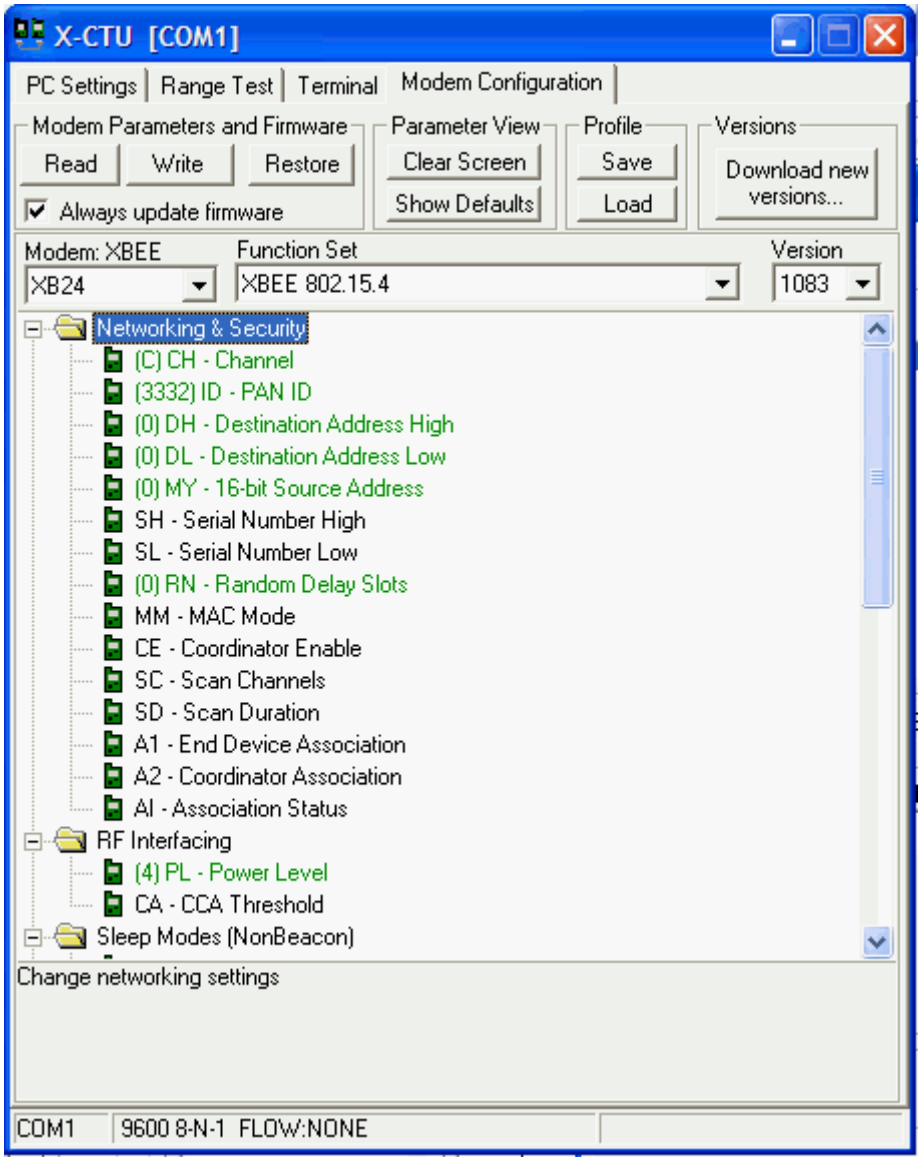


Figura 49: Programa X-CTU da MaxStream.

Primeiramente para o coordenador, é preciso definir um valor para seu endereço fonte (MY), nesse caso 5000. Depois, o endereço destino (DL) deve ser definido para broadcast, a propriedade *Coordinator Enable* (CE) deve ser definido para 1 (*Coordinator*), a propriedade *Node Identifier* (NI) é uma descrição qualquer de até 20 caracteres e a propriedade *API Enable* (AP) deve ser definida para 1 (*Enable*).

Já para o *end device*, é preciso definir um valor para seu endereço fonte (MY), nesse caso 5001. Depois, o endereço destino (DL) deve ser definido para o endereço fonte do coordenador, nesse caso 5000, a propriedade CE deve ser definido para 0 (*End Device*), a propriedade NI é uma descrição qualquer de até 20

caracteres, a propriedade AP deve ser definida para 0 e por último a propriedade *Sleep Mode* (SM) deve ser definida para 0 (*Disable*).

3.4.2. Enviando dados pela rede ZigBee

Para enviar e ler dados da porta serial, utilizando a linguagem de programação java, é necessário que um conjunto de classes disponíveis no pacote RXTXcomm.jar seja incorporado ao projeto SarEJB. Essas classes são responsáveis por recuperar uma porta serial específica, configurar parâmetros tais como velocidade da porta, quantidades de *bits* da mensagem, *bit* de parada e *bit* de paridade, além de especificar se é necessária a notificação de eventos, qual classe será responsável por coletar as notificações e o mais importante enviar e receber dados.

Na funcionalidade Controle Remoto, assim que o usuário selecionar o ambiente em que deseja controlar será enviado um pacote de dados solicitando o estado do dispositivo *end device*. A tabela 6 representa o pacote de dados de escrita.

Tabela 6: Pacote de dados de escrita.

Bytes	Descrição
1	Delimitador Inicial
2	Tamanho dos bytes
1	Identificador da API
1	API Frame ID
2	Endereço destino (DL)

1	Byte de opção
2	Pacote de dados
1	Checksum

O pacote de dados é montado com a utilização de valores hexadecimais, por isso a representação do valor dos bytes será iniciada por “0x”. A seguir serão mostrados quais os valores de cada byte do pacote de dados de escrita.

O delimitador inicial é fixo e o valor é 0x7E. O tamanho dos bytes é subdividido em dois tipos: MSB (*Most Significant Byte*) e LSB (*Least Significant Byte*). A soma da quantidade de bytes de Identificador da API, API Frame ID, endereço destino, byte de opção e pacote de dados dá o valor total do tamanho de bytes. O identificador da API será sempre 0x01 para indicar transmissão. A API Frame ID será 0x00 caso não haja necessidade de resposta de confirmação. O endereço destino nesse projeto será 0x50 0x01. O byte de opção será 0x01 para desabilitar a opção de confirmação. O pacote de dados pode ter até 100 bytes por pacote e o checksum é calculado para verificar a integridade do dado. O calculo do checksum é feito pela soma de todos os bytes, não incluindo o delimitador inicial e o tamanho de bytes, se o resultado for um número de 3 dígitos ou mais considere sempre os 2 dígitos à direita e então subtraia de 0xFF.

O pacote de dados utilizado para manipular os relês da placa RCOM-HOMBEE são: 0x7B 0x00, 0x7B 0x01, 0x7B 0x02, 0x7B 0x03. O byte 0x7B é o identificador de envio de dados para a placa. Já o byte 0x00 é utilizada para desligar os dois relês, o byte 0x01 é utilizado para ligar o relê 1 e desligar o relê 2, o byte 0x02 é utilizado para ligar o relê 2 e desligar o relê 1 e o byte 0x03 é utilizado para ligar ambos os relês.

Um exemplo de pacote de dados de escrita para uma placa RCOM-HOMEBEE com endereço destino igual a 5001 e que se deseja ligar o relê seria: 0x7E 0x00 0x07 0x01 0x00 0x50 0x01 0x00 0x7B 0x01 checksum.

Para enviar um pedido de solicitação de estado para a placa RCOM-HOMEBEE, basta substituir o exemplo acima por: 0x7E 0x00 0x07 0x01 0x00 0x50 0x01 0x00 0x7C 0x00 checksum. O byte 0x7C é o identificador de solicitação de estado.

A placa RCOM-HOMEBEE ao receber uma solicitação de estado irá processar a informação e enviar um pacote de dados de retorno. O módulo coordenador irá recepcioná-lo e encaminhá-lo para o sistema que processará a informação e identificará o estado. A tabela 7, representa o pacote de dados recebidos.

Tabela 7: Pacote de dados recebidos.

Bytes	Descrição
1	Delimitador Inicial
2	Tamanho dos Bytes
1	Identificador da API
2	Endereço fonte (MY)
1	RSSI
1	Opções
2	Pacote de dados
1	Checksum

O pacote de dados recebidos também é montado com a utilização de valores hexadecimais, por isso a representação do valor dos bytes será iniciada por “0x”. A

seguir serão mostrados quais os valores de cada byte do pacote de dados recebidos.

O delimitador inicial é fixo e o valor é 0x7E. O tamanho dos bytes é subdividido em dois tipos: MSB (*Most Significant Byte*) e LSB (*Least Significant Byte*). A soma da quantidade de bytes de Identificador da API, endereço fonte, byte RSSI, byte de opção e pacote de dados dá o valor total do tamanho de bytes. O identificador da API será sempre 0x81 para indicar recepção. O endereço fonte será o valor da fonte transmissora dos dados. O byte de opção será 0x00 para indicar recepção reservada. O pacote de dados pode ter até 100 bytes por pacote e o checksum é calculado para verificar a integridade do dado. O calculo do checksum é feito pela soma de todos os bytes, não incluindo o delimitador inicial e o tamanho de bytes, se o resultado for um número de 3 dígitos ou mais considere sempre os 2 dígitos à direita e então subtraia de 0xFF.

Um exemplo de pacote de dados recebido da placa RCOM-HOMEBEE, como resposta ao comando de solicitação de estado, com endereço fonte igual a 5001 e que esteja com o relê 2 ligado seria: 0x7E 0x00 0x07 0x81 0x50 0x01 0x24 0x00 0x52 0x02 checksum.

O módulo coordenador, ao enviar um pacote de dados de escrita para a placa RCOM-HOMEBEE solicitando a execução de algum comando, receberá uma resposta contendo o estado da solicitação. Se o pacote de dados 0x7E 0x00 0x07 0x01 0x00 0x50 0x01 0x00 0x7B 0x01 checksum fosse solicitado uma resposta como o pacote a seguir seria recebida: 0x7E 0x00 0x07 0x81 0x50 0x01 0x24 0x00 0x57 0x01 checksum. O byte 0x57 identifica uma resposta a uma solicitação de execução de comando e o byte 0x01 identifica o estado da placa.

3.4.3. Controlando dispositivos pela funcionalidade Controle Remoto

Ao acessar a funcionalidade Controle Remoto e escolher qual ambiente se deseja controlar, uma solicitação de estado será encaminhada do módulo coordenador para o módulo *end device*. A partir da resposta recebida, os dispositivos serão dispostos na tela em forma de botões. Para ligar ou desligar um

desses dispositivos, basta pressionar o botão referente ao dispositivo desejado e um pacote de dados será enviado para o módulo *end device* solicitando a execução do comando. Logo após a execução, uma resposta será enviada para o módulo coordenador e está resposta informará se o comando foi executado ou não.

3.5. Aplicação da solução com resultados

Após a fase de implementação, constatou-se que o projeto estava funcionando como esperado. Dessa forma, várias medições foram feitas com o intuito de verificar a máxima distância que os módulos XBee-Pro seriam capazes de se comunicar, além de verificar o sinal RSSI em cada transmissão. Para tanto, simulações de distância, obstáculos (parede, porta, janela) e interferência pela utilização de celular foram realizadas a fim de se obter um resultado condizente para uma real utilização do projeto e a análise dos pacotes de dados recebidos como resposta das solicitações de execução tornou possível a verificação do sinal RSSI.

A primeira medição foi realizada a uma distância de 1 metro, sem obstáculos e sem a interferência de celular. Nessas condições o projeto funcionou corretamente e o nível de sinal recebido foi de - 36 dBm.

Em seguida, a distância de 1 metro foi mantida, porém havia uma parede separando o módulo coordenador do *end device* e não havia interferência de celular. O projeto continuou funcionando perfeitamente e o nível de sinal permaneceu em - 36 dBm.

Utilizando a mesma distância, a medição foi realizada com a presença de interferência de celular e uma parede como obstáculo, porém nenhuma alteração foi verificada. O sinal continuou em - 36 dBm.

Foram realizadas medições para 5 metros e 10 metros. Nessas distâncias, em linha reta, o nível do sinal foi o mesmo: - 44 dBm. Ainda com essas distâncias, mas com várias paredes separando o módulo coordenador do *end device*, o projeto

funcionou corretamente e o nível do sinal medido foi de - 65 dBm. A utilização de celular não interferiu nos resultados dessas medições.

A uma distância de 30 metros, tendo portas e paredes como obstáculos, o projeto funcionou normalmente. Porém o nível do sinal medido foi para - 78 dBm. Este valor também foi obtido para as medições onde havia interferência do uso de celular.

O módulo *end device* estava dentro de uma residência, enquanto o módulo coordenador estava na rua a 100 metros de distância e o projeto ainda assim funcionou corretamente. O nível de sinal medido foi de - 92 dBm. Logo depois, a distância do módulo coordenador foi alterada para 150 metros, porém o projeto não mais funcionou. Vale ressaltar que nessa ultima medição ainda havia a presença de obstáculos no caminho.

4. CONCLUSÃO

O SAR é um sistema projetado para atender as necessidades de gerenciamento de dispositivos de iluminação utilizando o padrão de comunicação sem fio ZigBee. A partir de computadores conectados a rede local de uma residência é possível acessar o sistema e acionar remotamente lâmpadas por meio de contato seco de relês. Durante a implementação desse projeto várias dificuldades foram encontradas, algumas delas foram superadas e outras não.

Uma das dificuldades superadas foi a pequena quantidade de materiais referentes à tecnologia ZigBee devido a sua pouca disseminação e pouca utilização. Através do estudo do *datasheet* do módulo XBee-Pro e de monografias e teses de mestrado foi possível conhecer as características dessa tecnologia e algumas de suas aplicações. (Anexo A)

Uma questão não superada foi a limitação de dois relês na placa RCOM-HOMEBEE adquirida. Como o foco do projeto é o sistema SAR e o padrão de comunicação ZigBee, o hardware não foi alterado e o projeto limitou-se a controlar somente duas lâmpadas.

O projeto foi desenvolvido seguindo o conhecimento e apoio de diversas disciplinas do curso de Engenharia de Computação. Primeiramente, foi a disciplina de Engenharia de Programas que forneceu o conhecimento para entender e elaborar todas as etapas de evolução de um projeto. As disciplinas de Linguagens e Técnicas de Programação I e II forneceram um conhecimento sólido em lógica de programação e base para o aprendizado das linguagens de programação java e flex. A disciplina de Banco de Dados I forneceu o conhecimento necessário para entender e criar modelos de dados relacionais e utilizar de forma correta instruções de seleção, inserção, alteração e exclusão. Para entender o funcionamento e conhecer os diferentes tipos de redes de computadores, as disciplinas de Redes de Computadores I e II e Tópicos Avançados de Rede facilitaram bastante. Já as disciplinas Circuitos e Máquinas Elétricas e Instalações Elétricas foram essenciais para o entendimento de tensões e correntes elétricas juntamente com a utilização de relês.

A proposta do projeto é bem interessante quando da análise da realidade do mundo atual. Cada vez mais a tecnologia está presente no dia a dia e nada melhor como proporcionar mais segurança e comodidade nas residências. Com o custo de hardware e software relativamente baixo é possível implantar essa tecnologia em residências de classe baixa até as de classe alta. Além disso, conforme item 3.5, os testes realizados comprovam que a comunicação via ZigBee pode ser feita mesmo com a presença de obstáculos e à distâncias de até 100 metros, tornando viável a utilização em um ambiente real de uma residência.

A realização do projeto alcançou o resultado esperado em todos os aspectos. Inovação em sistemas de gerenciamento de dispositivos de iluminação, utilização de tecnologia em que não seriam necessárias mudanças de infra-estrutura pela utilização de comunicação sem fio e custo relativamente baixo puderam ser observados em todas as etapas de seu desenvolvimento. Sendo assim, o aprendizado no decorrer do curso foi comprovado e o projeto concluído com sucesso.

4.1. Projetos futuros

As melhorias aqui sugeridas para o projeto podem ser feitas tantos por alunos de Ciências da Computação quanto de Engenharia de Computação.

- Adicionar um motor de passo ao hardware para controle de cortinas;

- Adicionar sensor de luminosidade ao hardware para controle de um motor de passo;

- Alterar funcionalidade **Dispositivo** do sistema para suportar vários tipos e não somente lâmpadas.

- Implementar funcionalidade para manter *log* de usuários cadastrados e operações realizadas, a fim de aumentar a segurança e evitar acessos indevido.

REFERÊNCIAS BIBLIOGRÁFICAS

Resolução, FNC. Definition of "Internet". Disponível em: <http://www.itrd.gov/fnc/Internet_res.html>. Acesso em: 24 de mar. 2009.

MÜLLER, N. Internet, intranet e extranet o que são, e quais as diferenças?. Disponível em: <http://www.oficinadanet.com.br/artigo/1276/internet_intranet_e_extranet_o_que_sao_e_quais_as_diferencas>. Acesso em: 24 de mar. 2009.

HOMMERDING, N. M. S. O Profissional da Informação e a Gestão do Conhecimento nas Empresas: Um Novo Espaço Para Atuação, com Ênfase no Processo de Mapeamento do Conhecimento e Disponibilização Por Meio da Internet. 2001. 221 f. Dissertação (Mestrado em Ciências da Comunicação) – Escola de Comunicacoes e Artes. Universidade de São Paulo. São Paulo. 2001. Disponível em: <http://www.terraforum.com.br/sites/terraforum/Biblioteca/libdoc00000140v001Tese%20Nadia%20Hommerding.pdf>. Acesso em: 24 de mar. 2009.

BENETT, G. Intranets Como Implantar Com Sucesso na Sua Empresa. Rio de Janeiro: Campus, 1997.

TRINDADE, O. Intranets na USP: avaliação da tecnologia e recomendações. Disponível em: <http://www.ime.usp.br/~is/infousp/onofre.htm>. Acesso em: 24 de mar. 2009.

WIKIPEDIA. Computador pessoal. Disponível em: <http://pt.wikipedia.org/wiki/Computador_pessoal>. Acesso em: 25 de mar. 2009.

SENAC. M-leaning: a educação com mobilidade. Disponível em: <http://www.ead.sp.senac.br/newsletter/outubro07/ead.asp?nome=tecnologia>. Acesso em: 26 de mar. 2009.

TEAM WORK. Desenvolvimento Sistemas Móveis. Disponível em: <http://www.twic.com.br/site/index.php?option=com_content&task=view&id=13&Itemid=1>. Acesso em: 26 de mar. 2009.

WIKIPEDIA. Navegador. Disponível em: <http://pt.wikipedia.org/wiki/Navegador>. Acesso em: 29 de mar. 2009.

BRASIL ESCOLA. Navegador. Disponível em: <http://www.brasilescola.com/informatica/navegador.htm>. Acesso em: 29 de mar. 2009.

HTML.net. HTML Tutorial: O que é HTML?. Disponível em: <http://www.pt-br.html.net/tutorials/html/lesson2.asp>. Acesso em: 29 de mar. 2009.

ALVAREZ, M. A. O que é HTML. Disponível em: <http://www.criarweb.com/artigos/7.php>. Acesso em: 29 de mar. 2009.

VANDRÉ, P. Revista PHP/CSS: O que é CSS?. Disponível em: <http://www.revistaphp.com.br/artigo.php?id=119>. Acesso em: 29 de mar. 2009.

HTML.net. CSS Tutorial: O que é CSS?. Disponível em: < http://www.pt-br.html.net/tutorials/css/lesson1.asp>. Acesso em: 29 de mar. 2009.

WIKIPEDIA. Adobe Flash. Disponível em: <http://pt.wikipedia.org/wiki/Adobe_Flash>. Acesso em: 29 de mar. 2009.

ALVAREZ, R. O que é Flash. Disponível em: <http://www.criarweb.com/artigos/282.php>. Acesso em: 29 de mar. 2009.

EAGLE, M. 2004. Integrating Macromedia Flex with Java. Disponível em: <http://www.onjava.com/pub/a/onjava/2004/12/01/flexjava.html?page=1>. Acesso em: 30 de mar. 2009.

ADOBE. Visão geral do Flex. Disponível em: <http://www.adobe.com/br/products/flex/overview/>. Acesso em: 30 de mar. 2009.

SILVEIRA, I. F. Linguagem JAVA. Disponível em: <http://www.infowester.com/lingjava.php>. Acesso em: 31 de mar. 2009.

DEITEL, H. M.; DEITEL, P. J. JAVA Como Programar. São Paulo: Pearson, 2006. 1110 p.

JAVA.com. Saiba mais sobre a tecnologia Java. Disponível em: <http://java.com/pt_BR/about/>. Acesso em: 31 de mar. 2009.

The Apache Software Foundation. Apache Tomcat. Disponível em: <http://tomcat.apache.org/>. Acesso em: 31 de mar. 2009.

FLEURY, M.; STARK, S.; NORMAN, R. JBoss 4.0 The Official Guide. Sams Publishing, 2006. 648 p.

BORRIE, H. Firebird Databases as the Back-end to Enterprise Software Systems. Disponível em: <http://www.firebirdsql.org/devel/doc/papers/html/paper-fbent.html#paper-fbent-intro/>. Acesso em: 04 de abr. 2009.

MySql. Manual de Referência do MySQL 4.1. Disponível em: <http://dev.mysql.com/doc/refman/4.1/pt/index.html>. Acesso em: 04 de abr. 2009.

CANZIAN, E. MINICURSO: Comunicação Serial - RS232. Disponível em: <http://www.coinfo.cefetpb.edu.br/professor/leonidas/irc/apostilas/comun_serial.pdf>. Acesso em: 04 de abr. 2009.

AXELSON, J. Parallel Port Complete: Programming, Interfacing, & Using the PC's Parallel Printer Port. Lakeview Research; Pap/Dis edition, 1997. 343 p.

Cisco Systems, Inc. Internetworking Technology Handbook: Ethernet Technologies. Disponível em: <http://www.cisco.com/en/US/docs/internetworking/technology/handbook/Ethernet.pdf>. Acesso em: 07 de abr. 2009.

GOLDSMITH, A. Wireless Communications. United States of America: Cambridge University Press, 2005. 673 p.

RIGONATTI, T. Introdução ao Mundo Wireless. Disponível em: <http://www.mobilelife.com.br/artigos/introducao-ao-mundo-wireless>. Acesso em: 05 de abr. 2009.

TANENBAUM, A. S. Redes De Computadores. Campus, 2003. 945 p.

Institute of Electrical and Electronics Engineers (IEEE), Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-PANs), IEEE Standard 802.15.4, Outubro de 2003. Disponível em: <http://standards.ieee.org>. Acesso em: 20 de mai. 2009.

TEIXEIRA, L. M. Desenvolvimento de uma aplicação com o protocolo ZigBee aplicado em Instrumentação de Ensaio em Vôo. 2006. 163 f. Instituto Tecnologico de Aeronautica, São José dos Campos. Disponível em: <http://www.bd.bibl.ita.br/tde_busca/arquivo.php?codArquivo=851>. Acesso em: 20 de mai. 2009.

MONSIGNORE, F. Sensoriamento de ambiente utilizando o padrão ZigBee. 2007. 92 f. Dissertação (Mestrado em Engenharia Elétrica) – Escola de Engenharia de São Carlos. Universidade de São Paulo. São Paulo. 2007. Disponível em: <http://www.teses.usp.br/teses/disponiveis/18/18152/tde-27042007-102640/>. Acesso em: 20 de mai. 2009.

Datasheet dos módulos XBee-Pro. Disponível em: http://www.digi.com/products/wireless/point-multipoint/xbee-series1-moduledocs.jsp. Acesso em: 20 de mar. de 2009.

Datasheet da placa CON-USBBEE. Disponível em: http://www.rogercom.com/ManualUsbBee.pdf. Acesso em: 20 de mar. de 2009.

Datasheet da placa RCOM-HOMEBEE. Disponível em: http://www.rogercom.com/ManualHomeBee.pdf. Acesso em: 20 de mar. de 2009.

APÊNDICE A

index.mxml

```
<?xml version="1.0" encoding="utf-8"?>

<mx:Application
	xmlns:mx="http://www.adobe.com/2006/mxml"
	layout="vertical"
	verticalAlign="middle"
	horizontalAlign="center"
	xmlns:screen="flex.*"
	width="100%"
	height="100%"
	creationComplete="{init()}"
	xmlns:login="br.com.paulo.projetos.sar.seguranca.*">

	<mx:Style>

		Application {
		  background-image: Embed(source="/imagem/desktop.jpg");
		}

	</mx:Style>

			<mx:Script>
	    <![CDATA[
		import mx.automation.delegates.controls.AlertFormAutomationImpl;
		import modulos.seguranca.Login;
		import mx.core.IFlexDisplayObject;
		import br.com.paulo.botoes.MenuBotao;
		import mx.managers.PopUpManagerChildList;
		import mx.containers.TitleWindow;
		import mx.managers.PopUpManager;
		import mx.collections.ArrayCollection;
		import mx.events.FlexEvent;

				public function init(obj: Object = null): void {
```

```
                                        var popUp: IFlexDisplayObject =
PopUpManager.createPopUp(Application.application as DisplayObject, Login, false);
                                        PopUpManager.bringToFront(popUp);
                                }
        ]]>
        </mx:Script>

                <mx:Canvas width="100%" height="100%">

                        <mx:VBox id="menu" width="135" height="80%" x="10" y="10">

                        </mx:VBox>

                </mx:Canvas>

</mx:Application>
```

Login.as

```
package modulos.seguranca
{
        import br.com.paulo.botoes.MenuBotao;
        import br.com.paulo.componentes.Alerta;
        import br.com.paulo.componentes.RealizarValidacao;
        import br.com.paulo.dto.RequisicaoDto;
        import br.com.paulo.infra.Index;
        import br.com.paulo.projetos.sar.seguranca.LoginView;
        import br.com.paulo.util.ServicoJava;
        import br.com.paulo.vo.sar.seguranca.MenuVO;
        import br.com.paulo.vo.sar.seguranca.UsuarioVO;

        import flash.events.MouseEvent;
        import flash.utils.getDefinitionByName;

        import modulos.ambiente.CadastrarAmbienteSelecao;
        import modulos.controlador.CadastrarControladoresSelecao;
        import modulos.controleRemoto.ListarAmbienteControleRemoto;
        import modulos.dispositivo.ListarAmbienteSelecao;

        import mx.events.FlexEvent;
        import mx.managers.PopUpManager;
```

```
import mx.rpc.events.ResultEvent;

public class Login extends LoginView
{

	private var cadastrarControladores: CadastrarControladoresSelecao;
	private var cadastrarAmbientes: CadastrarAmbienteSelecao;
	private var listarAmbientes: ListarAmbienteSelecao;
	private var controleRemoto: ListarAmbienteControleRemoto;

	[Embed (source = "/imagem/local_network_48.png")]
	private var iconeCasa: Class;

	[Embed (source = "/imagem/engrenagem_48.png")]
	private var iconeEngrenagem: Class;

	[Embed (source = "/imagem/plug_48.png")]
	private var iconeDispositivo: Class;

	[Embed (source = "/imagem/sinal.png")]
	private var iconeControleRemoto: Class;

	private static var CAMINHO_CASA: String = "/imagem/local_network_48.png";
	private static var CAMINHO_ENGRENAGEM: String = "/imagem/engrenagem_48.png";
	private static var CAMINHO_DISPOSITIVO: String = "/imagem/plug_48.png";
	private static var CAMINHO_CONTROLE_REMOTO: String = "/imagem/sinal.png";

	public function Login()
	{
		this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
	}

	private function init(event: FlexEvent): void {
		this.btnLogar.addEventListener(MouseEvent.CLICK, clickLogar);
	}

	private function clickLogar(event: MouseEvent): void {

		var validacao: RealizarValidacao = qdDados.realizarValidacao();
```

```
            if (!validacao.valido) {
                Alerta.showCampoObrigatorio(validacao.campoFoco);
            } else {
                var dto: RequisicaoDto = new RequisicaoDto();

                var usuario: UsuarioVO = new UsuarioVO();
                usuario.userName = txtUsername.text;
                usuario.senha = txtSenha.text;

                dto.mapa.usuario = usuario;

                ServicoJava.enviar("segurancaFachada", "login", retornoLogin, null, dto);
            }
        }

        private function retornoLogin(event: ResultEvent): void {

            PopUpManager.removePopUp(this);

            var index: Index = new Index();

            var usuario: UsuarioVO = event.result.mapa.usuario as UsuarioVO;

            var menu: MenuVO;
            for each(menu in usuario.permissao.menus) {
                index.addBotao(getBtnMenu(menu));
            }
            index.montarMenu();
        }


        private function getBtnMenu(menu: MenuVO): MenuBotao {
            var botao: MenuBotao = new MenuBotao;
            botao.id = menu.idTela;
            botao.label = menu.label;

    var className: String = menu.classe;
    var clazz: Class = Class(getDefinitionByName(className));

            botao.classe = clazz;
```

```
                botao.ativarSegundoPlano = menu.isAtivarSegundoPlano;
                botao.isPopUpCentralizado = menu.isPopUpCentralizado;

                var classeIcone: Class;
                switch(menu.caminhoImagem) {

                        case CAMINHO_CASA: {
                                classeIcone = iconeCasa;
                                break;
                        }

                        case CAMINHO_ENGRENAGEM: {
                                classeIcone = iconeEngrenagem;
                                break;
                        }

                        case CAMINHO_DISPOSITIVO: {
                                classeIcone = iconeDispositivo;
                                break;
                        }

                        case CAMINHO_CONTROLE_REMOTO: {
                                classeIcone = iconeControleRemoto;
                                break;
                        }

                        default: {
                                classeIcone = iconeCasa;
                        }
                }

                botao.icone = classeIcone;

                return botao;
        }
    }
}
```

LoginView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<modulos:JanelaView
	xmlns:mx="http://www.adobe.com/2006/mxml"
	xmlns:modulos="br.com.paulo.modulos.*"
	xmlns:controlexTexto="br.com.paulo.controlesTexto.*"
	width="346" height="218"
	cornerRadius="10"
	xmlns:ns1="br.com.paulo.botoes.*"
	title="Efetuar Login"
	showCloseButton="false"
	xmlns:infra="br.com.paulo.infra.*"
	x="460" y="100">

	<modulos:CanvasKruds id="qdTitulo" width="100%" height="68" backgroundColor="#FFFFFF" cornerRadius="10" left="5" top="5" right="5">

		<mx:Image id="iconCadeado" source="./imagem/user_48.png"  x="10" height="63" width="60" alpha="1.0" themeColor="#E6EFF5" verticalCenter="1"/>
		<controlexTexto:Rotulo id="lblLogin" height="24" text="Sistema de Automação Residencial" textAlign="center" fontFamily="Arial" fontSize="12" color="#46A7BC" textDecoration="normal" fontStyle="normal" fontWeight="bold" top="34" left="78" right="0"/>
		<controlexTexto:Rotulo x="68" y="10" width="58" text="SAR" fontSize="25" fontFamily="Arial" textAlign="center" color="#356999"/>

	</modulos:CanvasKruds>

	<modulos:CanvasKruds id="qdDados" left="5" top="72" height="100%" width="100%" bottom="5" right="5" backgroundColor="#FFFFFF" cornerRadius="10" borderStyle="solid">

		<controlexTexto:Rotulo x="21" y="10" width="72" text="Username:" textAlign="left"/>
		<controlexTexto:Texto id="txtUsername" x="89" y="8" campoObrigatorio="true" width="209"/>

		<controlexTexto:Rotulo x="21" y="36" width="72" text="Senha:" textAlign="left"/>
		<controlexTexto:Texto id="txtSenha" displayAsPassword="true" campoObrigatorio="true" x="89" y="34" width="209"/>

		<ns1:Botao id="btnLogar" x="222" y="64" width="76" label="Logar"/>

	</modulos:CanvasKruds>

</modulos:JanelaView>
```

Menu.java

```
package br.com.paulo.seguranca.entidades;

@Entity(name = "MENU")
@Table(name = "MENU", schema = "CTA")
public class Menu extends Id {

	private static final long serialVersionUID = 1L;

	@Column(name = "IDTELA", nullable = false, length = 255)
	private String idTela;

	@Column(name = "LABEL", nullable = false, length = 255)
	private String label;

	@Column(name = "CLASSE", nullable = false, length = 255)
	private String classe;

	@Column(name = "DESATIVARSEGUNDOPLANO", nullable = false)
	private Boolean isAtivarSegundoPlano;

	@Column(name = "POPUPCENTRALIZADO", nullable = false)
	private Boolean isPopUpCentralizado;

	@Column(name = "CAMINHOIMAGEM", nullable = false, length = 255)
	private String caminhoImagem;

	@ManyToMany(fetch = FetchType.EAGER)
	@JoinTable(name = "RELMENUPERMISSAO", schema = "CTA",
			joinColumns = {@JoinColumn(name = "IDMENU", referencedColumnName =
"ID")},
			inverseJoinColumns = {@JoinColumn(name = "IDPERMISSAO",
referencedColumnName = "ID")})
	private List<Permissao> permissoes;
}
```

Usuario.java

```
package br.com.paulo.seguranca.entidades;
```

```
@Entity(name = "USUARIO")
@Table(name = "USUARIO", schema = "CTA")

@NamedQueries({
	@NamedQuery(name = "RecuperarUsuarioPorUserName",
				query = "FROM br.com.paulo.seguranca.entidades.Usuario as usuario where usuario.userName = :userName")
	})
public class Usuario extends Id {

	private static final long serialVersionUID = 1L;

	@Column(name = "NOME", nullable = false, length = 100)
	private String nome;

	@Column(name = "EMAIL", nullable = true, length = 100)
	private String email;

	@Column(name = "USERNAME", nullable = false, length = 60)
	private String userName;

	@Column(name = "SENHA", nullable = false, length = 60)
	private String senha;

	@ManyToOne()
	@JoinColumn(name = "IDPERMISSAO", referencedColumnName = "ID")
	private Permissao permissao;
}
```

Permissao.java

```
package br.com.paulo.seguranca.entidades;

@Entity(name = "PERMISSAO")
@Table(name = "PERMISSAO", schema = "CTA")
public class Permissao extends Id {

	private static final long serialVersionUID = 1L;

	@Enumerated(EnumType.STRING)
	private Role role;
```

```
	@ManyToMany(mappedBy = "permissoes")
	private List<Menu> menus;

	@ManyToMany(mappedBy = "permissao")
	private List<Usuario> usuarios;
}
```

Role.java

```
package br.com.paulo.seguranca.tipos;

public enum Role {

	ADMIN("Administrador"),
	ENGENHEIRO("Engenheiro"),
	MORADOR("Morador");

	Role(String descricao) {
		this.key = descricao;
	}

	private String key;

	public String getKey() {
		return key;
	}
}
```

SegurancaFachada.java

```
package br.com.paulo.seguranca.fachada;

public class SegurancaFachada {

	private SegurancaDelegate segurancaDelegate =
SegurancaDelegateFactory.getInstance().getSegurancaDelegate();

	public RetornoDto login(RequisicaoDto dto) throws SegurancaException {

		Usuario usuario = (Usuario) dto.getMapa().get("usuario");
```

```
                usuario = segurancaDelegate.login(usuario);

                System.out.println(usuario.getUserName());
                System.out.println(usuario.getSenha());

                RetornoDto retornoDto = new RetornoDto();
                retornoDto.getMapa().put("usuario", usuario);

                return retornoDto;
        }
}
```

SegurancaDelegate.java

```
package br.com.paulo.seguranca.delegate;

public class SegurancaDelegate extends InfraDelegate<ISegurancaServico, Usuario> {

        public Usuario login(Usuario usuario) throws SegurancaException {
                return getServico().login(usuario);
        }

        @Override
        protected ISegurancaServico getServico() throws SegurancaException {
                try {
                        return SegurancaServiceLocator.getInstance().recuperarSegurancaServico();
                } catch (SystemException e) {
                        throw new SegurancaException(e);
                }
        }
}
```

SegurancaServico.java

```
package br.com.paulo.seguranca.servico.impl;

@Stateless()
public class SegurancaServico extends Servico<Usuario> implements ISegurancaServico {

        @PersistenceContext(unitName = "emCTA")
        private EntityManager entityManager;
```

```
	@SuppressWarnings("unchecked")
	@Override
	public IUsuarioDAO getDAO() throws DAOException {
		return DAOFactory.getInstance().getDAO(UsuarioDAO.class, entityManager);
	}

	public Usuario login(Usuario usuario) throws SegurancaException {
		String senha = usuario.getSenha();
		try {
			usuario = getDAO().login(usuario);

			if (!SegurancaUtil.criptografarSenha(senha).equals(usuario.getSenha())) {
				throw new SegurancaException("seguranca.senha.invalida");
			}

			inicializaColecoes(usuario);
		} catch (DAOException e) {
			throw new SegurancaException("seguranca.login.invalido");
		}
		return usuario;
	}

	private void inicializaColecoes(Usuario usuario) {
		usuario.getPermissao().getMenus().size();
		usuario.getPermissao().getUsuarios().size();
		for (Menu menu : usuario.getPermissao().getMenus()) {
			for (Permissao p : menu.getPermissoes()) {
				p.getUsuarios().size();
				p.getMenus().size();
			}
		}
	}
}
```

UsuarioDAO.java

```
package br.com.paulo.seguranca.persistencia.impl;

@SuppressWarnings("unchecked")
```

```
public class UsuarioDAO extends DAO<Usuario> implements IUsuarioDAO {

	public UsuarioDAO(EntityManager manager) {
		super(manager);
	}

	public Usuario login(Usuario usuario) throws DAOException {
		Query query = getManager().createNamedQuery("RecuperarUsuarioPorUserName");
		query.setParameter("userName", usuario.getUserName());

		try {
			return (Usuario) query.getSingleResult();

		} catch(NoResultException e) {
			throw new DAOException(e);
		}
	}
}
```

APÊNDICE B

CadastrarControladoresSelecao.as

```
package modulos.controlador
{
	import br.com.paulo.componentes.Alerta;
	import br.com.paulo.componentes.RealizarValidacao;
	import br.com.paulo.dto.RequisicaoDto;
	import br.com.paulo.dto.SarDto;
	import br.com.paulo.projetos.sar.controlador.CadastrarControladoresView;
	import br.com.paulo.util.ServicoJava;
	import br.com.paulo.vo.sar.entidades.ControladorVO;
	import br.com.paulo.vo.sar.entidades.PortaVO;

	import flash.events.MouseEvent;

	import mx.collections.ArrayCollection;
```

```
import mx.events.FlexEvent;
import mx.events.ListEvent;

public class CadastrarControladores extends CadastrarControladoresView
{
	private var _controlador: ControladorVO;

	public function CadastrarControladores()
	{
		this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
	}

	private function init(event: FlexEvent): void {

	}

	override public function botaoOKPressionado(event: MouseEvent): void {

		var validacao: RealizarValidacao = realizarValidacao();
		if (!validacao.valido) {
			Alerta.showCampoObrigatorio(validacao.campoFoco);
		} else {
			var dto: RequisicaoDto = new RequisicaoDto();
			if (modo == MODO_INCLUIR) {

				_controlador = new ControladorVO();
				_controlador.identificador = txtIdentificador.text;
				_controlador.descricao = txtDescricao.text;

				dto.mapa[SarDto.OBJETO_INCLUIDO] = _controlador;

				ServicoJava.incluir("cadastrarControladoresFachada", dto, this);
			} else {

				_controlador.identificador = txtIdentificador.text;
				_controlador.descricao = txtDescricao.text;

				dto.mapa[SarDto.OBJETO_ALTERADO] = _controlador;

				ServicoJava.alterar("cadastrarControladoresFachada", dto, this);
```

```
                        }
                    }
                }

                override public function botaoLimparPressionado(event: MouseEvent): void {
                    txtIdentificador.text = "";
                    txtDescricao.text = "";
                }

                override public function preencherCampos(): void {
                    if (modo == MODO_VISUALIZAR || modo == MODO_ALTERAR) {

                        _controlador = objetoSelecionado as ControladorVO;

                        txtIdentificador.text = _controlador.identificador;
                        txtDescricao.text = _controlador.descricao;
                    }
                }
            }
}
```

CadastrarControladoresSelecaoView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<modulos:ListaCadastroView
    xmlns:mx="http://www.adobe.com/2006/mxml"
    xmlns:modulos="br.com.paulo.modulos.*"
    width="528" height="394"
    xmlns:ns1="br.com.paulo.controlesTexto.*"
    xmlns:ns2="br.com.paulo.componentes.*"
    xmlns:ns3="br.com.paulo.botoes.*"
    title="Manter Controlador"
    xmlns:controlador="br.com.paulo.projetos.sar.controlador.*"
xmlns:controlador1="modulos.controlador.*">

    <modulos:CanvasKruds id="telaSelecao" width="100%" height="100%" cornerRadius="5"
borderStyle="solid" backgroundColor="#EEEFEF">

        <ns2:SubtituloView texto="Pesquisar" x="10" y="10" width="478">
        </ns2:SubtituloView>
        <ns1:Rotulo x="10" y="52" width="83" text="Identificador:" />
```

```
		<ns1:Numerico id="txtIdentificador" y="50" width="54" left="91" maxChars="4" />

		<ns1:Rotulo x="10" y="78" width="83" text="Descrição:"/>
		<ns1:Texto id="txtDescricao" x="91" y="76" width="297" maxChars="250" />
		<ns3:Botao id="btnPesquisar" x="396" y="76" width="92" label="Pesquisar"/>


		<ns2:SubtituloView texto="Resultado" x="10" y="109" width="478"/>
		<modulos:DataGridKruds id="tabela" x="10" y="142" width="478" numeroLinhas="20"
height="158">
			<modulos:columns>
	  <modulos:ColumnGrid dataField="id" headerText="ID" />
	  <modulos:ColumnGrid dataField="identificador" headerText="Identificador" />
	  <modulos:ColumnGrid dataField="descricao" headerText="Descrição" />
	</modulos:columns>
		</modulos:DataGridKruds>

	</modulos:CanvasKruds>

	<modulos:formularioCadastro>
		<controlador1:CadastrarControladores width="100%" />
	</modulos:formularioCadastro>

	<modulos:BarraBotoes id="barraBotoes" width="100%"></modulos:BarraBotoes>

</modulos:ListaCadastroView>
```

CadastrarControladores.as

```
package modulos.controlador
{
	import br.com.paulo.componentes.Alerta;
	import br.com.paulo.componentes.RealizarValidacao;
	import br.com.paulo.dto.RequisicaoDto;
	import br.com.paulo.dto.SarDto;
	import br.com.paulo.projetos.sar.controlador.CadastrarControladoresView;
	import br.com.paulo.util.ServicoJava;
	import br.com.paulo.vo.sar.entidades.ControladorVO;
	import br.com.paulo.vo.sar.entidades.PortaVO;

	import flash.events.MouseEvent;
```

```
import mx.collections.ArrayCollection;
import mx.events.FlexEvent;
import mx.events.ListEvent;

public class CadastrarControladores extends CadastrarControladoresView
{
	private var _controlador: ControladorVO;

	public function CadastrarControladores()
	{
		this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
	}

	private function init(event: FlexEvent): void {

	}

	override public function botaoOKPressionado(event: MouseEvent): void {

		var validacao: RealizarValidacao = realizarValidacao();
		if (!validacao.valido) {
			Alerta.showCampoObrigatorio(validacao.campoFoco);
		} else {
			var dto: RequisicaoDto = new RequisicaoDto();
			if (modo == MODO_INCLUIR) {

				_controlador = new ControladorVO();
				_controlador.identificador = txtIdentificador.text;
				_controlador.descricao = txtDescricao.text;

				dto.mapa[SarDto.OBJETO_INCLUIDO] = _controlador;

				ServicoJava.incluir("cadastrarControladoresFachada", dto, this);
			} else {

				_controlador.identificador = txtIdentificador.text;
				_controlador.descricao = txtDescricao.text;

				dto.mapa[SarDto.OBJETO_ALTERADO] = _controlador;
```

```
                                        ServicoJava.alterar("cadastrarControladoresFachada", dto, this);
                                }
                        }
                }

                override public function botaoLimparPressionado(event: MouseEvent): void {
                        txtIdentificador.text = "";
                        txtDescricao.text = "";
                }

                override public function preencherCampos(): void {
                        if (modo == MODO_VISUALIZAR || modo == MODO_ALTERAR) {

                                _controlador = objetoSelecionado as ControladorVO;

                                txtIdentificador.text = _controlador.identificador;
                                txtDescricao.text = _controlador.descricao;
                        }
                }
        }
}
```

CadastrarControladoresView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<modulos:FormularioCadastroView
        xmlns:mx="http://www.adobe.com/2006/mxml"
        xmlns:modulos="br.com.paulo.modulos.*"
        xmlns:controlador="modulos.controlador.*"
        width="532" height="106"
        xmlns:ns1="br.com.paulo.controlesTexto.*"
        xmlns:ns2="br.com.paulo.componentes.*"
        xmlns:ns3="br.com.paulo.botoes.*" >

        <modulos:CanvasKruds id="telaEdicao" width="100%" height="100%" cornerRadius="5"
borderStyle="solid" backgroundColor="#EEEFEF">

                <ns2:SubtituloView texto="Dados" x="10" y="10" width="510" />

                <ns1:Rotulo id="lblIdentificador" x="10" y="45" text="Identificador:" width="94"/>
```

```
		<ns1:Numerico id="txtIdentificador" x="112" y="43" width="80" maxChars="4" campoObrigatorio="true"/>

		<ns1:Rotulo id="lblDescricao" x="10" y="71" text="Descrição:" width="94"/>
		<ns1:Texto id="txtDescricao" x="112" y="69" width="408" maxChars="255" campoObrigatorio="true"/>

	</modulos:CanvasKruds>

	<modulos:barraBotoes>
		<modulos:BarraBotoesFormularioCadastroView id="barraBotoes" width="100%" />
	</modulos:barraBotoes>

</modulos:FormularioCadastroView>
```

Controlador.java

```
package br.com.paulo.sar.entidades;

@Entity(name = "Controlador")
@Table(name = "Controlador", schema = "SAR")
public class Controlador extends Id {

	/**
	 *
	 */
	private static final long serialVersionUID = 1L;

	@Column(name = "IDENTIFICADOR", nullable = false, length = 4)
	private String identificador;

	@Column(name = "DESCRICAO", nullable = false, length = 255)
	private String descricao;

	@OneToMany(fetch = FetchType.EAGER, mappedBy = "controlador", cascade = {CascadeType.ALL})
	private List<Porta> portas;
}
```

Porta.java

```
package br.com.paulo.sar.entidades;
```

```
@Entity(name = "Porta")
@Table(name = "Porta", schema = "SAR")
public class Porta extends Id {

	/**
	 *
	 */
	private static final long serialVersionUID = 1L;

	@Column(name = "DESCRICAO", nullable = false, length = 255)
	private String descricao;

	@Enumerated(value = EnumType.ORDINAL)
	private TipoPorta tipoPorta;

	@ManyToOne(optional = true)
	@JoinColumn(name = "IDCONTROLADOR", referencedColumnName = "ID")
	private Controlador controlador;

	@OneToOne(mappedBy = "porta", cascade = {CascadeType.REMOVE})
	private Dispositivo dispositivo;
}
```

TipoPorta.java

```
package br.com.paulo.sar.tipos;

public enum TipoPorta {

	PORTA1(0),
	PORTA2(1);

	TipoPorta(Integer descricao) {
		this.key = descricao;
	}

	private Integer key;

	public Integer getKey() {
```

```
		return key;
	}
}
```

CadastrarControladoresFachada.java

```
package br.com.paulo.sar.fachada;

public class CadastrarControladoresFachada extends InfraCrud {

	private ControladorDelegate controladorDelegate =
SarDelegateFactory.getInstance().getControladorDelegate();

	@SuppressWarnings("unchecked")
	@Override
	public RetornoDto alterar(RequisicaoDto dto) throws SystemException {
		Controlador controlador = (Controlador) dto.getMapa().get(OBJETO_ALTERADO);

		controlador = controladorDelegate.alterar(controlador);

		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put(OBJETO_ALTERADO, controlador);
		return retornoDto;
	}

	@Override
	public RetornoDto excluir(RequisicaoDto dto) throws SystemException {
		Controlador controlador = (Controlador) dto.getMapa().get(FILTRO);

		controladorDelegate.excluir(controlador);

		RetornoDto retornoDto = new RetornoDto();
		return retornoDto;
	}

	@SuppressWarnings("unchecked")
	@Override
	public RetornoDto incluir(RequisicaoDto dto) throws SystemException {
		Controlador controlador = (Controlador) dto.getMapa().get(OBJETO_INCLUIDO);
		incluirPortas(controlador);
```

```
		controladorDelegate.incluir(controlador);

		RetornoDto retornoDto = new RetornoDto();
		return retornoDto;
	}

	private void incluirPortas(Controlador controlador) throws SystemException {
		Porta porta = new Porta();
		porta.setDescricao("Porta relê 1");
		porta.setTipoPorta(TipoPorta.PORTA1);
		porta.setControlador(controlador);

		controlador.getPortas().add(porta);

		porta = new Porta();
		porta.setDescricao("Porta relê 2");
		porta.setTipoPorta(TipoPorta.PORTA2);
		porta.setControlador(controlador);

		controlador.getPortas().add(porta);
	}

	@Override
	public RetornoDto obter(RequisicaoDto dto) throws SystemException {
		Controlador controlador = (Controlador) dto.getMapa().get(OBJETO_SELECIONADO);

		controlador = controladorDelegate.obter(Controlador.class, controlador.getId());

		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put("controlador", controlador);
		return retornoDto;
	}

	@Override
	public RetornoDto pesquisar(RequisicaoDto dto) throws SystemException {
		Controlador controlador = (Controlador) dto.getMapa().get(FILTRO);

		List<Controlador> resultado = controladorDelegate.pesquisar(controlador);
```

```
		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put(RESULTADO, resultado);
		return retornoDto;
	}

	@Override
	public RetornoDto listar() throws SystemException {
		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put(RESULTADO, controladorDelegate.listar());
		return retornoDto;
	}
}
```

ControladorDAO.java

```
package br.com.paulo.sar.persistencia.impl;

@SuppressWarnings("unchecked")
public class ControladorDAO extends DAO<Controlador> implements IControladorDAO {

	public ControladorDAO(EntityManager manager) {
		super(manager);
	}

	@Override
	public List<Controlador> pesquisar(Controlador controlador) throws DAOException {

		StringBuffer sb = new StringBuffer();
		sb.append("SELECT c FROM Controlador as c ");

		if (controlador != null) {
			sb.append("WHERE 1 = 1 ");

			if (controlador.getDescricao() != null) {
				sb.append("and c.descricao like :descricao ");
			}
			if (controlador.getIdentificador() != null &&
!controlador.getIdentificador().equals("")) {
				sb.append("and c.identificador = :identificador ");
			}
		}
```

```
		Query query = getManager().createQuery(sb.toString());
		configurarParametros(query, controlador);
		return query.getResultList();
	}

	public void configurarParametros(Query query, Controlador controlador) {
		if (controlador != null) {
			if (controlador.getDescricao() != null) {
				query.setParameter("descricao", "%" +controlador.getDescricao()+ "%");
			}
			if (controlador.getIdentificador() != null &&
!controlador.getIdentificador().equals("")) {
				query.setParameter("identificador", controlador.getIdentificador());
			}
		}
	}

	public List<Controlador> listarControladoresNaoUtilizados(Long id) throws DAOException {
		StringBuffer sb = new StringBuffer();
		sb.append("SELECT c FROM Controlador as c ");
		sb.append("WHERE c.id NOT IN (");
		sb.append("SELECT a.controlador.id FROM Ambiente as a ");

		if (id != null) {
			sb.append("WHERE a.controlador.id != :id)");
		} else {
			sb.append(")");
		}

		Query query = getManager().createQuery(sb.toString());

		if (id != null) {
			query.setParameter("id", id);
		}

		return query.getResultList();
	}
}
```

CadastrarAmbienteSelecao.as

```
package modulos.ambiente
{
	import br.com.paulo.dto.RequisicaoDto;
	import br.com.paulo.dto.SarDto;
	import br.com.paulo.projetos.sar.ambiente.CadastrarAmbienteSelecaoView;
	import br.com.paulo.util.ServicoJava;
	import br.com.paulo.vo.sar.entidades.AmbienteVO;
	import br.com.paulo.vo.sar.entidades.ControladorVO;

	import flash.events.MouseEvent;

	import mx.events.FlexEvent;
	import mx.rpc.events.ResultEvent;

	public class CadastrarAmbienteSelecao extends CadastrarAmbienteSelecaoView
	{
		public function CadastrarAmbienteSelecao()
		{
			this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
		}

		private function init(event: FlexEvent): void {
			alterarTituloJanela = true;
			this.btnPesquisar.addEventListener(MouseEvent.CLICK,
botaoPesquisarPressionado);

			ServicoJava.enviar("cadastrarAmbientesFachada", "obterDadosSelecao",
retornoObterDadosSelecao);
		}

		private function retornoObterDadosSelecao(event: ResultEvent): void {
			cbxControlador.dataProvider = event.result.mapa.listaControladores;
		}

		override public function botaoPesquisarPressionado(event: MouseEvent): void {
			super.botaoPesquisarPressionado(event);
			var dto: RequisicaoDto = new RequisicaoDto();

			var ambiente: AmbienteVO = new AmbienteVO;
```

```
                ambiente.nome = txtNome.text;
                ambiente.descricao = txtDescricao.text;
                ambiente.controlador = cbxControlador.selectedItem as  ControladorVO;

                dto.mapa[SarDto.FILTRO] = ambiente;
                ServicoJava.pesquisar("cadastrarAmbientesFachada", dto, this);
            }

            override public function botaoExcluirPressionado(event: MouseEvent): void {
                var dto: RequisicaoDto = new RequisicaoDto();

                var ambiente: AmbienteVO = objetoSelecionado as  AmbienteVO;

                dto.mapa[SarDto.FILTRO] = ambiente;
                ServicoJava.excluir("cadastrarAmbientesFachada", dto, this);
            }
        }
}
```

CadastrarAmbienteSelecaoView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<modulos:ListaCadastroView
        xmlns:mx="http://www.adobe.com/2006/mxml"
        xmlns:modulos="br.com.paulo.modulos.*"
        width="554" height="436"
        title="Manter Ambiente"
        xmlns:componentes="br.com.paulo.componentes.*"
        xmlns:controlesTexto="br.com.paulo.controlesTexto.*"
        xmlns:ambiente="modulos.ambiente.*"
        xmlns:ns1="br.com.paulo.botoes.*">

        <modulos:CanvasKruds id="telaSelecao" width="100%" height="100%" cornerRadius="5"
borderStyle="solid" backgroundColor="#EEEFEF">

            <componentes:SubtituloView texto="Pesquisar" x="10" y="10" width="504" />

            <controlesTexto:Rotulo x="10" y="52" width="84" text="Nome:" />
            <controlesTexto:Texto id="txtNome" x="102" y="50" width="412"/>

            <controlesTexto:Rotulo x="10" y="78" width="84" text="Descrição:" />
```

```
		<controlesTexto:Texto id="txtDescricao" x="102" y="76" width="412"/>

		<controlesTexto:Rotulo x="10" y="108" width="84" text="Controlador:"/>
		<componentes:ComboBoxKruds id="cbxControlador" labelField="textoComboBox"
itemOpcional="true" width="412" x="102" y="106"/>

		<ns1:Botao id="btnPesquisar" x="410" y="134" width="104" label="Pesquisar"/>

		<componentes:SubtituloView texto="Resultado" x="10" y="164" width="504"/>
		<modulos:DataGridKruds id="tabela" x="10" y="197" width="504" numeroLinhas="20"
height="145">
			<modulos:columns>
	  <modulos:ColumnGrid dataField="id" headerText="ID" />
	  <modulos:ColumnGrid dataField="nome" headerText="Nome" />
	  <modulos:ColumnGrid dataField="descricao" headerText="Descrição" />
	  <modulos:ColumnGrid dataField="descricaoControlador" headerText="Controlador" />
	</modulos:columns>
		</modulos:DataGridKruds>

	</modulos:CanvasKruds>

	<modulos:formularioCadastro>
		<ambiente:CadastrarAmbiente width="100%"/>
	</modulos:formularioCadastro>

	<modulos:BarraBotoes id="barraBotoes" width="100%"></modulos:BarraBotoes>

</modulos:ListaCadastroView>
```

CadastrarAmbiente.as

```
package modulos.ambiente
{
	import br.com.paulo.componentes.Alerta;
	import br.com.paulo.componentes.RealizarValidacao;
	import br.com.paulo.componentes.UploadArquivo;
	import br.com.paulo.dto.RequisicaoDto;
	import br.com.paulo.dto.SarDto;
	import br.com.paulo.projetos.sar.ambiente.CadastrarAmbienteView;
	import br.com.paulo.util.ServicoJava;
	import br.com.paulo.vo.sar.entidades.AmbienteVO;
```

```
import br.com.paulo.vo.sar.entidades.ControladorVO;

import flash.events.Event;
import flash.events.MouseEvent;

import mx.events.FlexEvent;
import mx.rpc.events.ResultEvent;

public class CadastrarAmbiente extends CadastrarAmbienteView
{

	private var _ambiente: AmbienteVO;

	public function CadastrarAmbiente()
	{
		this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
	}

	private function init(event: FlexEvent): void {
		uploadImagem.urlUpload = "http://192.168.0.100:8080/SARFlex/UploadImagem";
		uploadImagem.addEventListener(UploadArquivo.UPLOAD_FINALIZADO,
retornoUploadFinalizado);
	}

	private function retornoObterDadosSelecao(event: ResultEvent): void {
		cbxControlador.dataProvider = event.result.mapa.listaControladores;

		if (modo == MODO_ALTERAR || modo == MODO_VISUALIZAR) {
			preencherCombo();
		}
	}

	private function retornoUploadFinalizado(event: Event): void {
		if (modo == MODO_ALTERAR) {
			alterarAmbiente();
		} else {
			incluirAmbiente();
		}
	}
```

```
		private function obterDadosSelecao(): void {
			ServicoJava.enviar("cadastrarAmbientesFachada", "obterDadosSelecaoManter",
retornoObterDadosSelecao);
		}

		private function incluirAmbiente(): void {
			ServicoJava.incluir("cadastrarAmbientesFachada", montarDto(), this);
		}

		private function alterarAmbiente(): void {
			ServicoJava.alterar("cadastrarAmbientesFachada", montarDto(), this);
		}

		private function montarDto(): RequisicaoDto {
			var parametro: String;

			parametro = SarDto.OBJETO_ALTERADO;

			if (modo == MODO_INCLUIR) {
				_ambiente = new AmbienteVO;
				parametro = SarDto.OBJETO_INCLUIDO;
			}

			_ambiente.nome = txtNome.text;
			_ambiente.descricao = txtDescricao.text;
			_ambiente.controlador = cbxControlador.selectedItem as ControladorVO;
			_ambiente.caminhoImagem = uploadImagem.txtNomeArquivo.text;

			var dto: RequisicaoDto = new RequisicaoDto();
			dto.mapa[parametro.toString()] = _ambiente;
			return dto;
		}

		private function preencherCombo(): void {
			var controlador: Object;
			for each(controlador in cbxControlador.dataProvider) {
				if (controlador.id == _ambiente.controlador.id) {
					cbxControlador.selectedItem = controlador;
					break;
				}
```

```
                }
            }

            override public function botaoOKPressionado(event: MouseEvent): void {
                var validacao: RealizarValidacao = realizarValidacao();
                if (!validacao.valido) {
                    Alerta.showCampoObrigatorio(validacao.campoFoco);
                } else {
                    if (modo == MODO_ALTERAR && uploadImagem.txtNomeArquivo.text
== _ambiente.caminhoImagem) {
                    alterarAmbiente();

                    } else {
                        uploadImagem.enviarArquivo(new
MouseEvent(MouseEvent.CLICK));
                    }
                }
            }

            override public function botaoLimparPressionado(event: MouseEvent): void {
                txtNome.text = "";
                txtDescricao.text = "";
                uploadImagem.txtNomeArquivo.text = "";
                cbxControlador.selectedIndex = 0;
            }

            override public function preencherCampos(): void {
                var dto: RequisicaoDto = new RequisicaoDto;
                if (modo == MODO_VISUALIZAR || modo == MODO_ALTERAR) {

                    _ambiente = objetoSelecionado as AmbienteVO;

                    txtNome.text = _ambiente.nome;
                    txtDescricao.text = _ambiente.descricao;
                    uploadImagem.txtNomeArquivo.text = _ambiente.caminhoImagem;

                    dto.mapa.idSelecionado = _ambiente.controlador.id;
                }
                ServicoJava.enviar("cadastrarAmbientesFachada", "obterDadosSelecaoManter",
retornoObterDadosSelecao, null, dto);
            }
```

```
        }
}
```

CadastrarAmbienteView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<modulos:FormularioCadastroView
        xmlns:mx="http://www.adobe.com/2006/mxml"
        xmlns:modulos="br.com.paulo.modulos.*"
        width="498" height="162"
        xmlns:componentes="br.com.paulo.componentes.*"
        xmlns:controlesTexto="br.com.paulo.controlesTexto.*">

        <modulos:CanvasKruds id="telaEdicao" width="100%" height="100%" cornerRadius="5" borderStyle="solid" backgroundColor="#EEEFEF">

                <componentes:SubtituloView texto="Dados" x="10" y="10" width="472" />

                <controlesTexto:Rotulo id="lblNome" x="10" y="45" text="Nome:" width="60"/>
                <controlesTexto:Texto id="txtNome" x="93" y="43" width="232" maxChars="240" campoObrigatorio="true"/>

                <controlesTexto:Rotulo id="lblDescricao" x="10" y="71" text="Descrição:" width="79"/>
                <controlesTexto:Texto id="txtDescricao" x="93" y="68" width="389" maxChars="240" campoObrigatorio="true"/>

                <controlesTexto:Rotulo id="lblUpload" x="10" y="97" text="Imagem:" width="79"/>
                <componentes:UploadArquivo id="uploadImagem" width="389" x="93" y="93" />

                <controlesTexto:Rotulo x="10" y="121" width="79" text="Controlador:"/>
                <componentes:ComboBoxKruds id="cbxControlador" labelField="textoComboBox" campoObrigatorio="true" itemOpcional="true" width="393" x="93" y="119"/>

        </modulos:CanvasKruds>

        <modulos:barraBotoes>
                <modulos:BarraBotoesFormularioCadastroView id="barraBotoes" width="100%" />
        </modulos:barraBotoes>

</modulos:FormularioCadastroView>
```

ServletUploadImagem.java

```
package br.com.paulo.sar.upload;


public class ServletUploadImagem extends HttpServlet {


	private static final long serialVersionUID = 1L;


	@Override
	public void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
		doPost(req, resp);
	}


	@SuppressWarnings("unchecked")
	@Override
	public void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
		System.out.println(req.getServerName());
		System.out.println(req.getContextPath());



		DiskFileItemFactory factory = new DiskFileItemFactory();
		ServletFileUpload upload = new ServletFileUpload(factory);


		try {
			List<FileItem> items = upload.parseRequest(req);
			for (FileItem fileItem : items) {
				if (fileItem.getName() != null) {
					File arquivo = new File(getServletContext().getRealPath("/")+
"imagem\\ambiente\\" +fileItem.getName());
					fileItem.write(arquivo);
				}
			}
		} catch (FileUploadException ex) {
			ex.getMessage().toString();
		} catch (Exception ex) {
			ex.getMessage().toString();
		}


		resp.getOutputStream().print("arquivo:");
		resp.getOutputStream().flush();
	}
```

}

Ambiente.java

```
package br.com.paulo.sar.entidades;

@Entity(name = "Ambiente")
@Table(name = "Ambiente", schema = "SAR")
public class Ambiente extends Id {

	private static final long serialVersionUID = 1L;

	@Column(name = "NOME", nullable = false, length = 60)
	private String nome;

	@Column(name = "DESCRICAO", nullable = false, length = 255)
	private String descricao;

	@Column(name = "CAMINHOIMAGEM", nullable = false, length = 255)
	private String caminhoImagem;

	@ManyToOne(optional = false)
	@JoinColumn(name = "IDCONTROLADOR", referencedColumnName = "ID")
	private Controlador controlador;
}
```

CadastrarAmbienteFachada.java

```
public class CadastrarAmbientesFachada extends InfraCrud {

	private AmbienteDelegate ambienteDelegate =
SarDelegateFactory.getInstance().getAmbienteDelegate();
	private ControladorDelegate controladorDelegate =
SarDelegateFactory.getInstance().getControladorDelegate();
	private DispositivoDelegate dispositivoDelegate =
SarDelegateFactory.getInstance().getDispositivoDelegate();

	public RetornoDto obterDadosSelecao() throws SystemException {
		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put("listaControladores", controladorDelegate.listar());
		return retornoDto;
	}
```

```
	public RetornoDto obterDadosSelecaoManter(RequisicaoDto dto) throws SystemException {

		String idSelecionado = (String) dto.getMapa().get("idSelecionado");

		Long id = null;
		if (idSelecionado != null) {
			id = new Long(idSelecionado);
		}

		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put("listaControladores",
controladorDelegate.listarControladoresNaoUtilizados(id));
		return retornoDto;
	}

	@Override
	public RetornoDto alterar(RequisicaoDto dto) throws SystemException {
		Ambiente ambiente = (Ambiente) dto.getMapa().get(OBJETO_ALTERADO);

		ambiente = ambienteDelegate.alterar(ambiente);

		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put(OBJETO_ALTERADO, ambiente);
		return retornoDto;
	}

	@Override
	public RetornoDto excluir(RequisicaoDto dto) throws SystemException {
		Ambiente ambiente = (Ambiente) dto.getMapa().get(FILTRO);

		for (Porta porta : ambiente.getControlador().getPortas()) {
			if (porta.getDispositivo() != null) {
				dispositivoDelegate.excluir(porta.getDispositivo());
			}
		}

		ambienteDelegate.excluir(ambiente);

		RetornoDto retornoDto = new RetornoDto();
```

```
        return retornoDto;
}

@Override
public RetornoDto incluir(RequisicaoDto dto) throws SystemException {
        Ambiente ambiente = (Ambiente) dto.getMapa().get(OBJETO_INCLUIDO);

        ambienteDelegate.incluir(ambiente);

        RetornoDto retornoDto = new RetornoDto();
        return retornoDto;
}

@Override
public RetornoDto obter(RequisicaoDto dto) throws SystemException {
        Ambiente ambiente = (Ambiente) dto.getMapa().get(OBJETO_SELECIONADO);

        ambiente = ambienteDelegate.obter(Ambiente.class, ambiente.getId());

        RetornoDto retornoDto = new RetornoDto();
        retornoDto.getMapa().put("ambiente", ambiente);
        return retornoDto;
}

@Override
public RetornoDto pesquisar(RequisicaoDto dto) throws SystemException {
        Ambiente ambiente = (Ambiente) dto.getMapa().get(FILTRO);

        List<Ambiente> resultado = ambienteDelegate.pesquisar(ambiente);

        RetornoDto retornoDto = new RetornoDto();
        retornoDto.getMapa().put(RESULTADO, resultado);
        return retornoDto;
}

@Override
public RetornoDto listar() throws SystemException {
        RetornoDto retornoDto = new RetornoDto();
        retornoDto.getMapa().put(RESULTADO, ambienteDelegate.listar());
        return retornoDto;
```

```
	}
}
```

AmbienteDAO.java

```
package br.com.paulo.sar.persistencia.impl;

@SuppressWarnings("unchecked")
public class AmbienteDAO extends DAO<Ambiente> implements IAmbienteDAO {

	public AmbienteDAO(EntityManager manager) {
		super(manager);
	}

	@Override
	public List<Ambiente> pesquisar(Ambiente ambiente) throws DAOException {
		StringBuffer sb = new StringBuffer();
		sb.append("SELECT a FROM Ambiente as a ");

		if (ambiente != null) {
			sb.append("WHERE 1 = 1 ");

			if (ambiente.getNome() != null) {
				sb.append("and a.nome like :nome ");
			}
			if (ambiente.getDescricao() != null) {
				sb.append("and a.descricao like :descricao ");
			}
			if (ambiente.getControlador() != null && ambiente.getControlador().getId() != null) {
				sb.append("and a.controlador.id = :idControlador ");
			}
		}

		Query query = getManager().createQuery(sb.toString());
		configurarParametros(query, ambiente);
		return query.getResultList();
	}

	private void configurarParametros(Query query, Ambiente ambiente) {
		if (ambiente != null) {
```

```
                        if (ambiente.getNome() != null) {
                                query.setParameter("nome", "%" +ambiente.getNome()+ "%");
                        }
                        if (ambiente.getDescricao() != null) {
                                query.setParameter("descricao", "%" +ambiente.getDescricao()+ "%");
                        }
                        if (ambiente.getControlador() != null && ambiente.getControlador().getId() != null) {
                                query.setParameter("idControlador", ambiente.getControlador().getId());
                        }
                }
        }
}
```

ListarAmbienteSelecao.as

```
package modulos.dispositivo
{
        import br.com.paulo.dto.RequisicaoDto;
        import br.com.paulo.dto.SarDto;
        import br.com.paulo.projetos.sar.dispositivo.ListarAmbienteSelecaoView;
        import br.com.paulo.util.ServicoJava;
        import br.com.paulo.vo.sar.entidades.AmbienteVO;
        import br.com.paulo.vo.sar.entidades.ControladorVO;

        import flash.events.MouseEvent;

        import mx.events.FlexEvent;
        import mx.rpc.events.ResultEvent;

        public class ListarAmbienteSelecao extends ListarAmbienteSelecaoView
        {
                public function ListarAmbienteSelecao()
                {
                        this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
                }

                private function init(event: FlexEvent): void {
                        this.btnPesquisar.addEventListener(MouseEvent.CLICK,
botaoPesquisarPressionado);

                        barraBotoes.document.btnIncluir.visible = false;
```

```
                barraBotoes.document.btnVisualizar.visible = false;
                barraBotoes.document.btnExcluir.visible = false;
                barraBotoes.document.btnAlterar.width = 100;
                barraBotoes.document.btnAlterar.label = "Dispositivos";
                barraBotoes.swapChildren(barraBotoes.document.btnAlterar,
barraBotoes.document.btnExcluir);
                barraBotoes.validateNow();

                ServicoJava.enviar("cadastrarAmbientesFachada", "obterDadosSelecao",
retornoObterDadosSelecao);
        }

        private function retornoObterDadosSelecao(event: ResultEvent): void {
                cbxControlador.dataProvider = event.result.mapa.listaControladores;
        }

        override public function botaoPesquisarPressionado(event: MouseEvent): void {
                super.botaoPesquisarPressionado(event);
                var dto: RequisicaoDto = new RequisicaoDto();

                var ambiente: AmbienteVO = new AmbienteVO;
                ambiente.nome = txtNome.text;
                ambiente.descricao = txtDescricao.text;
                ambiente.controlador = cbxControlador.selectedItem as  ControladorVO;

                dto.mapa[SarDto.FILTRO] = ambiente;
                ServicoJava.pesquisar("cadastrarAmbientesFachada", dto, this);
        }
    }
}
```

ListarAmbienteSelecaoView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<modulos:ListaCadastroView
    xmlns:mx="http://www.adobe.com/2006/mxml"
    xmlns:modulos="br.com.paulo.modulos.*"
    width="554" height="436"
    title="Manter Dispositivos"
    xmlns:componentes="br.com.paulo.componentes.*"
    xmlns:controlesTexto="br.com.paulo.controlesTexto.*"
```

```
        xmlns:dispositivo="modulos.dispositivo.*"
        xmlns:ns1="br.com.paulo.botoes.*">

        <modulos:CanvasKruds id="telaSelecao" width="100%" height="100%" cornerRadius="5"
borderStyle="solid" backgroundColor="#EEEFEF">

                <componentes:SubtituloView texto="Pesquisar" x="10" y="10" width="504" />

                <controlesTexto:Rotulo x="10" y="52" width="84" text="Nome:" />
                <controlesTexto:Texto id="txtNome" x="102" y="50" width="412"/>

                <controlesTexto:Rotulo x="10" y="78" width="84" text="Descrição:" />
                <controlesTexto:Texto id="txtDescricao" x="102" y="76" width="412"/>

                <controlesTexto:Rotulo x="10" y="108" width="84" text="Controlador:"/>
                <componentes:ComboBoxKruds id="cbxControlador" labelField="textoComboBox"
itemOpcional="true" width="412" x="102" y="106"/>

                <ns1:Botao id="btnPesquisar" x="410" y="134" width="104" label="Pesquisar"/>

                <componentes:SubtituloView texto="Resultado" x="10" y="164" width="504"/>
                <modulos:DataGridKruds id="tabela" x="10" y="197" width="504" numeroLinhas="20"
height="145">
                        <modulos:columns>
          <modulos:ColumnGrid dataField="id" headerText="ID" />
          <modulos:ColumnGrid dataField="nome" headerText="Nome" />
          <modulos:ColumnGrid dataField="descricao" headerText="Descrição" />
          <modulos:ColumnGrid dataField="descricaoControlador" headerText="Controlador" />
        </modulos:columns>
                </modulos:DataGridKruds>

        </modulos:CanvasKruds>

        <modulos:formularioCadastro>
                <dispositivo:CadastrarDispositivo width="100%"/>
        </modulos:formularioCadastro>

        <modulos:BarraBotoes id="barraBotoes" width="100%"></modulos:BarraBotoes>

</modulos:ListaCadastroView>
```

CadastrarDispositivo.as

```
package modulos.dispositivo
{
	import br.com.paulo.componentes.Alerta;
	import br.com.paulo.componentes.RealizarValidacao;
	import br.com.paulo.dto.RequisicaoDto;
	import br.com.paulo.projetos.sar.dispositivo.CadastrarDispositivoView;
	import br.com.paulo.util.ServicoJava;
	import br.com.paulo.vo.sar.entidades.AmbienteVO;
	import br.com.paulo.vo.sar.entidades.DispositivoVO;
	import br.com.paulo.vo.sar.entidades.PortaVO;

	import flash.events.Event;
	import flash.events.MouseEvent;

	import mx.collections.ArrayCollection;
	import mx.controls.Image;
	import mx.events.FlexEvent;
	import mx.rpc.events.ResultEvent;

	public class CadastrarDispositivo extends CadastrarDispositivoView
	{

		public function CadastrarDispositivo() {
			this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
		}

		private var _ambienteSelecionado: AmbienteVO;

		[Bindable]
		private var _listaComboPortas: ArrayCollection = new ArrayCollection;

		private var _listaDispositivosIncluidos: ArrayCollection = new ArrayCollection;
		private var _listaDispositivosExcluidos: ArrayCollection = new ArrayCollection;
		private var _listaDispositivosAlterados: ArrayCollection = new ArrayCollection;

		//RegExp para substituir todos os valores numericos encontrados
		private var numeroRegExp: RegExp = /[0-9]/g;
```

```
		private function init(event: FlexEvent): void {

			barraBotoes.document.removeChild(barraBotoes.document.btnLimpar);
			barraBotoes.document.removeChild(barraBotoes.document.btnCancelar);

			telaEdicao.enabled = false;

			telaAmbiente.addEventListener(DISPOSITIVO_ADICIONADO,
dispositivoAdicionado);
			telaAmbiente.addEventListener(DISPOSITIVO_ALTERADO, dispositivoAlterado);
			telaLixeira.addEventListener(DISPOSITIVO_REMOVIDO, dispositivoRemovido);

			btnAdicionar.addEventListener(MouseEvent.CLICK, adicionarPorta);
			btnCancelar.addEventListener(MouseEvent.CLICK, cancelarPorta);

			montarListaDispostivos();
		}

		private function dispositivoAdicionado(event: Event): void {
			telaEdicao.enabled = true;
			telaAmbiente.enabled = false;
			telaDispositivos.enabled = false;
			Alerta.show("É necessario adicionar uma porta para o dispositivo", "Aviso",
Alerta.TIPO_ALERTA, cbxPortas);
		}

		private function dispositivoAlterado(event: Event): void {
			var dispositivo: DispositivoVO;
			for each(dispositivo in _listaDispositivosIncluidos) {
				if (dispositivo.nome == imagemAlterada.name) {
					break;
				}
				dispositivo = null;
			}

			if (dispositivo == null) {

				var porta: Object;
				for each(porta in _ambienteSelecionado.controlador.portas) {
					if (porta.dispositivo != null) {
```

```
                                        var dispositivoCadastrado: DispositivoVO =
porta.dispositivo;
                                        if (dispositivoCadastrado.nome == imagemAlterada.name)
{
                                            dispositivoCadastrado.posicaoX =
imagemAlterada.x;
                                            dispositivoCadastrado.posicaoY =
imagemAlterada.y;

                                            if (_listaDispositivosAlterados.length > 0) {
                                                var dispositivoLista: DispositivoVO;
                                                for each(dispositivoLista in
_listaDispositivosAlterados) {
                                                    if (dispositivoLista.nome ==
dispositivoCadastrado.nome) {
                                                        break;
                                                    }
                                                    dispositivoLista = null;
                                                }

                                                if (dispositivoLista != null) {
                                                    var posicaoLista: int =
_listaDispositivosAlterados.getItemIndex(dispositivoLista);
        _listaDispositivosAlterados.removeItemAt(posicaoLista);
        _listaDispositivosAlterados.addItemAt(dispositivoCadastrado, posicaoLista);
                                                } else {
        _listaDispositivosAlterados.addItem(dispositivoCadastrado);
                                                }

                                            } else {
        _listaDispositivosAlterados.addItem(dispositivoCadastrado);
                                            }
                                        }
                                    }
                                }
                            } else {
                                var posicao: int = _listaDispositivosIncluidos.getItemIndex(dispositivo);

                                var dispositivoAlterado: DispositivoVO = new DispositivoVO;
                                dispositivoAlterado.nome = imagemAlterada.name;
```

```
                                dispositivoAlterado.imagem =
imagemAlterada.name.replace(numeroRegExp, "");
                                dispositivoAlterado.posicaoX = imagemAlterada.x;
                                dispositivoAlterado.posicaoY = imagemAlterada.y;
                                dispositivoAlterado.porta = dispositivo.porta;

                                _listaDispositivosIncluidos.removeItemAt(posicao);
                                _listaDispositivosIncluidos.addItemAt(dispositivoAlterado, posicao);
                        }

                        imagemAlterada = null;
                }

                private function dispositivoRemovido(event: Event): void {
                        var dispositivo: DispositivoVO;
                        for each(dispositivo in _listaDispositivosIncluidos) {
                                if (dispositivo.nome == imagemRemovida.name) {
                                        break;
                                }
                                dispositivo = null;
                        }

                        if (dispositivo == null) {
                                var porta: Object;
                                for each(porta in _ambienteSelecionado.controlador.portas) {
                                        if (porta.dispositivo != null) {
                                                dispositivo = porta.dispositivo;
                                                if (dispositivo.nome == imagemRemovida.name) {
                                                        _listaDispositivosExcluidos.addItem(dispositivo);
                                                        _listaComboPortas.addItem(dispositivo.porta);
                                                }
                                        }
                                }
                        } else {

        _listaDispositivosIncluidos.removeItemAt(_listaDispositivosIncluidos.getItemIndex(dispositivo));
                                _listaComboPortas.addItem(dispositivo.porta);
                        }

                        imagemRemovida = null;
```

```
}

private function adicionarPorta(event: MouseEvent): void {
	var validacao: RealizarValidacao = telaEdicao.realizarValidacao();
	if (validacao.valido) {
		telaAmbiente.enabled = true;
		telaDispositivos.enabled = true;

		var dispositivo: DispositivoVO = new DispositivoVO;
		dispositivo.nome = imagemAdicionada.name;
		dispositivo.imagem = imagemAdicionada.name.replace(numeroRegExp, "");
		dispositivo.posicaoX = imagemAdicionada.x;
		dispositivo.posicaoY = imagemAdicionada.y;

		dispositivo.porta = cbxPortas.selectedItem as PortaVO;

		_listaComboPortas.removeItemAt(cbxPortas.selectedIndex);

		_listaDispositivosIncluidos.addItem(dispositivo);

		cbxPortas.selectedIndex = 0;
		imagemAdicionada = null;
		telaEdicao.enabled = false;
	} else {
		Alerta.showCampoObrigatorio(validacao.campoFoco);
	}
}

private function cancelarPorta(event: MouseEvent): void {
	telaAmbiente.enabled = true;
	telaDispositivos.enabled = true;

	telaAmbiente.removeChild(imagemAdicionada);

	imagemAdicionada = null;
	telaEdicao.enabled = false;
}

private function montarListaDispostivos(): void {
	var listaDispotivios: ArrayCollection = DispositivosUtil.listarDispositivos();
```

```
                    var imagem: Image;
                    for each(imagem in listaDispotivios) {
                        imagem.addEventListener(MouseEvent.MOUSE_DOWN,
mouseMoveHandler);
                        telaDispositivos.addChild(imagem);
                    }
                }

                override public function botaoOKPressionado(event: MouseEvent): void {
                    var dto: RequisicaoDto = new RequisicaoDto;
                    dto.mapa.listaDispositivosIncluidos = _listaDispositivosIncluidos;
                    dto.mapa.listaDispositivosAlterados = _listaDispositivosAlterados;
                    dto.mapa.listaDispositivosExcluidos = _listaDispositivosExcluidos;

                    dto.mapa.ambiente = _ambienteSelecionado;

                    ServicoJava.enviar("cadastrarDispositivosFachada", "manter",
retornoManterDispositivos, null, dto, "Atualizando cadastro de dispositivos");
                }

                private function retornoManterDispositivos(event: ResultEvent): void {
                    barraBotoes.document.btnFechar.dispatchEvent(new
MouseEvent(MouseEvent.CLICK));
                }

                override public function botaoLimparPressionado(event: MouseEvent): void {
                    telaAmbiente.removeAllChildren();
                    _listaDispositivosIncluidos = new ArrayCollection;
                    _listaDispositivosAlterados = new ArrayCollection;
                    _listaDispositivosExcluidos = new ArrayCollection;

                    telaEdicao.enabled = false;
                    telaAmbiente.enabled = true;
                    telaDispositivos.enabled = true;
                }

                override public function preencherCampos(): void {
                    modo = MODO_INCLUIR;
                    destino = "cadastrarAmbientesFachada";

                    _ambienteSelecionado = objetoSelecionado as AmbienteVO;
```

```
                alterarPlanoFundo();

                montarDispositivosAmbiente();
            }

            private function alterarPlanoFundo(): void {
                this.telaAmbiente.setStyle("backgroundImage",
"http://192.168.0.100:8080/SARFlex/imagem/ambiente/" +_ambienteSelecionado.caminhoImagem);
            }

            private function montarListaPortas(): void {
                cbxPortas.dataProvider = _listaComboPortas;
            }

            private function montarDispositivosAmbiente(): void {

                _listaComboPortas = new ArrayCollection;
                montarListaPortas();

                //Zera contador do nome da imagem
                contador = 0;

                var porta: Object;
                for each(porta in _ambienteSelecionado.controlador.portas) {
                    if (porta.dispositivo != null) {
                        var dispositivo: DispositivoVO = porta.dispositivo;
                        var imagem: Image = new Image;
                        imagem.name = dispositivo.nome;
                        imagem.source =
DispositivosUtil.recuperarSourceDesligados(dispositivo.imagem);
                        imagem.x = dispositivo.posicaoX;
                        imagem.y = dispositivo.posicaoY;

                        addLixeira();

                        imagem.addEventListener(MouseEvent.MOUSE_DOWN,
mouseMoveHandler);
                        imagem.addEventListener(MouseEvent.MOUSE_DOWN,
mouseHandlerLixeiraDown);
```

```
                                        imagem.addEventListener(MouseEvent.MOUSE_UP,
mouseHandlerLixeiraUp);

                                        //Incrementa o contador para adicionar no nome da imagem
                                        contador++;

                                        telaAmbiente.addChild(imagem);
                                } else {
                                        _listaComboPortas.addItem(porta);
                                }
                        }
                }
        }
}
```

CadastrarDispositivoView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<modulos:FormularioCadastroView
        xmlns:mx="http://www.adobe.com/2006/mxml"
        xmlns:modulos="br.com.paulo.modulos.*"
        width="542" height="492"
        xmlns:componentes="br.com.paulo.componentes.*"
        xmlns:controlesTexto="br.com.paulo.controlesTexto.*" xmlns:ns1="br.com.paulo.botoes.*">

        <mx:Script>
    <![CDATA[
        import br.com.paulo.componentes.Alerta;
        import mx.containers.Canvas;
        import mx.controls.Image;
        import br.com.paulo.util.FadeUtil;
        import mx.events.FlexEvent;
        import mx.effects.IEffectInstance;
        import mx.effects.Fade;
      import mx.managers.DragManager;
      import mx.core.DragSource;
      import mx.events.DragEvent;
      import flash.events.MouseEvent;

      // Embed icon image.
      [Embed(source='./imagem/lixeiraCheia.png')]
```

```
public var lixeiraCheia:Class;

// Embed icon image.
[Embed(source='./imagem/lixeira.png')]
public var imgLixeira:Class;

public static var DISPOSITIVO_ADICIONADO: String = "dispositivoAdicionado";
public static var DISPOSITIVO_REMOVIDO: String = "dispositivoRemovido";
public static var DISPOSITIVO_ALTERADO: String = "dispositivoAlterado";

public var imagemAdicionada: Image;
public var imagemRemovida: Image;
public var imagemAlterada: Image;

public var contador: int = 0;

public function mouseMoveHandler(event: MouseEvent): void {
    var dragInitiator: Image = Image(event.currentTarget);
    var ds: DragSource = new DragSource();
    ds.addData(dragInitiator, "img");

  if (dragInitiator.parent == telaDispositivos && telaLixeira.getChildren().length > 0) {
            removeLixeira();
  }

    DragManager.doDrag(dragInitiator, ds, event);
}

private function dragEnterHandler(event: DragEvent): void {
    if (event.dragSource.hasFormat("img")) {
        DragManager.showFeedback(DragManager.COPY);
        DragManager.acceptDragDrop(Canvas(event.currentTarget));
    }
}

private function dragDropHandler(event: DragEvent): void {

  addLixeira();

    event.preventDefault();
```

```
            var lixeira: Image = telaLixeira.getChildAt(0) as Image;

            var dispositivo: Image = event.dragInitiator as Image;

                                    if (telaAmbiente.getChildren().length > 0 &&
telaAmbiente.getChildByName(dispositivo.name) != null) {
                                        Image(event.dragInitiator).x =
Canvas(event.currentTarget).mouseX;
                    Image(event.dragInitiator).y = Canvas(event.currentTarget).mouseY;
                    FadeUtil.fadeSumir(lixeira);
                    lixeira.enabled = false;

                    imagemAlterada = dispositivo;
                    telaAmbiente.dispatchEvent(new Event(DISPOSITIVO_ALTERADO));
                                } else {
                                        var dispositivoAmbiente: Image = new Image();
                                        dispositivoAmbiente.name =
verificaExistenciaNome(dispositivo.name);
                                        dispositivoAmbiente.source = dispositivo.source;

            dispositivoAmbiente.addEventListener(MouseEvent.MOUSE_DOWN, mouseMoveHandler);

            dispositivoAmbiente.addEventListener(MouseEvent.MOUSE_DOWN, mouseHandlerLixeiraDown);
                                        dispositivoAmbiente.addEventListener(MouseEvent.MOUSE_UP,
mouseHandlerLixeiraUp);
                                        dispositivoAmbiente.x = Canvas(event.currentTarget).mouseX;
                                        dispositivoAmbiente.y = Canvas(event.currentTarget).mouseY;
                                        telaAmbiente.addChild(dispositivoAmbiente);

                                        imagemAdicionada = dispositivoAmbiente;
                                        telaAmbiente.dispatchEvent(new
Event(DISPOSITIVO_ADICIONADO));
                                }
        }

        public function mouseHandlerLixeiraDown(event: MouseEvent): void {
            addLixeira();

            var lixeira: Image = telaLixeira.getChildAt(0) as Image;

            lixeira.enabled = true;
```

```
  FadeUtil.fadeAparecer(lixeira);
}

public function mouseHandlerLixeiraUp(event: MouseEvent): void {
  addLixeira();
  var lixeira: Image = telaLixeira.getChildAt(0) as Image;

  FadeUtil.fadeSumir(lixeira);
  lixeira.enabled = false;
}

private function dragEnterHandlerLixeira(event: DragEvent): void {
    if (event.dragSource.hasFormat("img")) {
            var lixeira: Image = telaLixeira.getChildAt(0) as Image;

        DragManager.showFeedback(DragManager.COPY);
        lixeira.source = lixeiraCheia;
        DragManager.acceptDragDrop(Image(event.currentTarget));
    }
}

private function dragDropHandlerLixeira(event:DragEvent):void {

  var dispositivos: Image = event.dragInitiator as Image;

  imagemRemovida = dispositivos;
  telaLixeira.dispatchEvent(new Event(DISPOSITIVO_REMOVIDO));

  var lixeira: Image = telaLixeira.getChildAt(0) as Image;
            lixeira.source = imgLixeira;
  FadeUtil.fadeSumir(lixeira);
  lixeira.enabled = false;

            telaAmbiente.removeChild(dispositivos);
}

private function dragExitHandlerLixeira(event:DragEvent):void {
  var lixeira: Image = telaLixeira.getChildAt(0) as Image;
  lixeira.source = imgLixeira;
}
```

```
        public function addLixeira(): void {

          if (telaLixeira.getChildren().length == 0) {
                    var lixeira: Image = new Image();
                    lixeira.id = "lixeira";
                    lixeira.name = "lixeira";
                    lixeira.source = imgLixeira;
                    lixeira.alpha = 0.0;
                    lixeira.addEventListener(DragEvent.DRAG_DROP, dragDropHandlerLixeira);
                    lixeira.addEventListener(DragEvent.DRAG_ENTER, dragEnterHandlerLixeira);
                    lixeira.addEventListener(DragEvent.DRAG_EXIT, dragExitHandlerLixeira);

                    telaLixeira.addChild(lixeira);
          }
        }

        private function removeLixeira(): void {
          telaLixeira.removeChildAt(0);
        }

        private function verificaExistenciaNome(nomeImagem: String): String {
          var imagem: Image;
          for each(imagem in telaAmbiente.getChildren()) {
                    if (imagem.name == (nomeImagem + contador)) {
                              contador++;
                              verificaExistenciaNome(nomeImagem + contador);
                    }
          }
          return nomeImagem + contador;
        }
      ]]>
    </mx:Script>

        <modulos:HBoxKruds width="100%" height="100%">

                  <modulos:VBoxKruds width="85%" height="100%">

                            <modulos:CanvasKruds id="telaAmbiente" width="100%" height="80%"
cornerRadius="5" borderStyle="solid" backgroundColor="#EEEFEF"
```

```
                    dragDrop="dragDropHandler(event);"
            dragEnter="dragEnterHandler(event);">

                    </modulos:CanvasKruds>

                    <modulos:CanvasKruds id="telaEdicao" width="100%" height="20%"
cornerRadius="5" borderStyle="solid" backgroundColor="#EEEFEF" >
                    <componentes:SubtituloView texto="Dados do Dispositivo" x="10" y="10"
width="437" />

                    <controlesTexto:Rotulo x="10" y="43" text="Porta:"/>
                    <componentes:ComboBoxKruds id="cbxPortas" labelField="descricao"
campoObrigatorio="true" itemOpcional="true" x="48" y="41" width="223"/>

                    <ns1:Botao x="279" y="41" label="Adicionar" id="btnAdicionar"/>
                    <ns1:Botao x="367" y="41" label="Cancelar" id="btnCancelar"/>

                    </modulos:CanvasKruds>

            </modulos:VBoxKruds>

            <modulos:VBoxKruds width="15%" height="100%" >

                    <modulos:VBoxKruds id="telaDispositivos" width="100%" height="80%"
horizontalAlign="center" cornerRadius="5" borderStyle="solid" backgroundColor="#EEEFEF">

                    </modulos:VBoxKruds>

                    <modulos:VBoxKruds id="telaLixeira" width="100%" height="20%"
horizontalAlign="center" verticalAlign="middle" cornerRadius="5" borderStyle="solid"
backgroundColor="#EEEFEF">

                    </modulos:VBoxKruds>

            </modulos:VBoxKruds>

    </modulos:HBoxKruds>

    <modulos:barraBotoes>
            <modulos:BarraBotoesFormularioCadastroView id="barraBotoes" width="100%" />
    </modulos:barraBotoes>
```

</modulos:FormularioCadastroView>

DispositivoUtil.as

```
package modulos.dispositivo
{
	import mx.collections.ArrayCollection;
	import mx.controls.Image;

	public class DispositivosUtil
	{
		public function DispositivosUtil()
		{

		}

		private static var NOME_LAMP_LIGADA: String = "lampLigado";
		private static var NOME_LAMP_DESLIGADA: String = "lampDesligado";

		[Bindable]
	[Embed(source='./imagem/Lamp-48x48.png')]
	private static var lampLigada: Class;

		[Bindable]
	[Embed(source='./imagem/Lamp-48x48 desligada.png')]
	private static var lampDesligada: Class;

		private static var _listaDispositivos: ArrayCollection;

		public static function listarDispositivos(): ArrayCollection {
			_listaDispositivos = new ArrayCollection;
			_listaDispositivos.addItem(getLampDesligada());
			return _listaDispositivos;
		}

		public static function recuperarSourceDesligados(imagem: String): Class {
			var classeImagem: Class;
			switch(imagem) {

				case NOME_LAMP_DESLIGADA: {
```

```
                        classeImagem = lampDesligada;
                        break;
                }

                default: {
                        classeImagem = lampDesligada;
                }
        }
        return classeImagem;
}

public static function recuperarSourceLigados(imagem: String): Class {
        var classeImagem: Class;
        switch(imagem) {

                case NOME_LAMP_LIGADA: {
                        classeImagem = lampLigada;
                        break;
                }

                default: {
                        classeImagem = lampLigada;
                }
        }
        return classeImagem;
}

private static function getLampLigada(): Image {
        var imagem: Image = new Image();
        imagem.name = NOME_LAMP_LIGADA;
        imagem.source = lampLigada;
        return imagem;
}

private static function getLampDesligada(): Image {
        var imagem: Image = new Image();
        imagem.name = NOME_LAMP_DESLIGADA;
        imagem.source = lampDesligada;
        return imagem;
}
```

```
	}
}
```

Dispositivo.java

```
package br.com.paulo.sar.entidades;

@Entity(name = "Dispositivo")
@Table(name = "Dispositivo", schema = "SAR")
public class Dispositivo extends Id {

	private static final long serialVersionUID = 1L;

	@Column(name = "NOME", nullable = false, length = 255)
	private String nome;

	@Column(name = "IMAGEM", nullable = false, length = 255)
	private String imagem;

	@Column(name = "POSICAOX", nullable = false)
	private Integer posicaoX;

	@Column(name = "POSICAOY", nullable = false)
	private Integer posicaoY;

	@OneToOne()
	@JoinColumn(name = "IDPorta", referencedColumnName = "ID")
	private Porta porta;
}
```

CadastrarDispositivosFachada.java

```
package br.com.paulo.sar.fachada;

public class CadastrarDispositivosFachada {

	private DispositivoDelegate dispositivoDelegate =
SarDelegateFactory.getInstance().getDispositivoDelegate();
	private AmbienteDelegate ambienteDelegate =
SarDelegateFactory.getInstance().getAmbienteDelegate();

	@SuppressWarnings("unchecked")
```

```
	public RetornoDto manter(RequisicaoDto dto) throws SystemException {

		Ambiente ambiente = (Ambiente) dto.getMapa().get("ambiente");

		List<Dispositivo> listaDispositivosIncluidos = (List<Dispositivo>)
dto.getMapa().get("listaDispositivosIncluidos");
		List<Dispositivo> listaDispositivosAlterados = (List<Dispositivo>)
dto.getMapa().get("listaDispositivosAlterados");
		List<Dispositivo> listaDispositivosExcluidos = (List<Dispositivo>)
dto.getMapa().get("listaDispositivosExcluidos");

		excluirDispositivo(listaDispositivosExcluidos);
		alterarDispositivo(listaDispositivosAlterados);
		incluirDispositivo(listaDispositivosIncluidos);

		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put("ambienteAtualizado", ambienteDelegate.obter(Ambiente.class,
ambiente.getId()));
		return retornoDto;
	}

	private void incluirDispositivo(List<Dispositivo> lista) throws SystemException {
		for (Dispositivo dispositivo : lista) {
			dispositivoDelegate.incluir(dispositivo);
		}
	}

	private void alterarDispositivo(List<Dispositivo> lista) throws SystemException {
		for (Dispositivo dispositivo : lista) {
			dispositivoDelegate.alterar(dispositivo);
		}
	}

	private void excluirDispositivo(List<Dispositivo> lista) throws SystemException {
		for (Dispositivo dispositivo : lista) {
			dispositivoDelegate.excluir(dispositivo);
		}
	}
}
```

ListarAmbienteControleRemoto.as

```
package modulos.controleRemoto
{
	import br.com.paulo.dto.RequisicaoDto;
	import br.com.paulo.dto.SarDto;
	import br.com.paulo.projetos.sar.controleRemoto.ListarAmbienteControleRemotoView;
	import br.com.paulo.util.ServicoJava;
	import br.com.paulo.vo.sar.entidades.AmbienteVO;
	import br.com.paulo.vo.sar.entidades.ControladorVO;

	import flash.events.MouseEvent;

	import mx.events.FlexEvent;
	import mx.rpc.events.ResultEvent;

	public class ListarAmbienteControleRemoto extends ListarAmbienteControleRemotoView
	{
		public function ListarAmbienteControleRemoto()
		{
			this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
		}

		private function init(event: FlexEvent): void {
			this.btnPesquisar.addEventListener(MouseEvent.CLICK,
botaoPesquisarPressionado);

			barraBotoes.document.btnIncluir.visible = false;
			barraBotoes.document.btnVisualizar.visible = false;
			barraBotoes.document.btnExcluir.visible = false;
			barraBotoes.document.btnAlterar.width = 100;
			barraBotoes.document.btnAlterar.label = "Controlar";
			barraBotoes.swapChildren(barraBotoes.document.btnAlterar,
barraBotoes.document.btnExcluir);
			barraBotoes.validateNow();

			ServicoJava.enviar("cadastrarAmbientesFachada", "obterDadosSelecao",
retornoObterDadosSelecao);
		}

		private function retornoObterDadosSelecao(event: ResultEvent): void {
			cbxControlador.dataProvider = event.result.mapa.listaControladores;
		}
```

```
        override public function botaoPesquisarPressionado(event: MouseEvent): void {
            super.botaoPesquisarPressionado(event);
            var dto: RequisicaoDto = new RequisicaoDto();

            var ambiente: AmbienteVO = new AmbienteVO;
            ambiente.nome = txtNome.text;
            ambiente.descricao = txtDescricao.text;
            ambiente.controlador = cbxControlador.selectedItem as  ControladorVO;

            dto.mapa[SarDto.FILTRO] = ambiente;
            ServicoJava.pesquisar("cadastrarAmbientesFachada", dto, this);
        }
    }
}
```

ListarAmbienteControleRemotoView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<modulos:ListaCadastroView
    xmlns:mx="http://www.adobe.com/2006/mxml"
    xmlns:modulos="br.com.paulo.modulos.*"
    width="554" height="436"
    title="Controle Remoto"
    xmlns:componentes="br.com.paulo.componentes.*"
    xmlns:controlesTexto="br.com.paulo.controlesTexto.*"
    xmlns:controleRemoto="modulos.controleRemoto.*"
    xmlns:ns1="br.com.paulo.botoes.*">

    <modulos:CanvasKruds id="telaSelecao" width="100%" height="100%" cornerRadius="5"
borderStyle="solid" backgroundColor="#EEEFEF">

        <componentes:SubtituloView texto="Pesquisar" x="10" y="10" width="504" />

        <controlesTexto:Rotulo x="10" y="52" width="84" text="Nome:" />
        <controlesTexto:Texto id="txtNome" x="102" y="50" width="412"/>

        <controlesTexto:Rotulo x="10" y="78" width="84" text="Descrição:" />
        <controlesTexto:Texto id="txtDescricao" x="102" y="76" width="412"/>

        <controlesTexto:Rotulo x="10" y="108" width="84" text="Controlador:"/>
```

```
		<componentes:ComboBoxKruds id="cbxControlador" labelField="textoComboBox" itemOpcional="true" width="412" x="102" y="106"/>

		<ns1:Botao id="btnPesquisar" x="410" y="134" width="104" label="Pesquisar"/>

		<componentes:SubtituloView texto="Resultado" x="10" y="164" width="504"/>
		<modulos:DataGridKruds id="tabela" x="10" y="197" width="504" numeroLinhas="20" height="145">
			<modulos:columns>
	  <modulos:ColumnGrid dataField="id" headerText="ID" />
	  <modulos:ColumnGrid dataField="nome" headerText="Nome" />
	  <modulos:ColumnGrid dataField="descricao" headerText="Descrição" />
	  <modulos:ColumnGrid dataField="descricaoControlador" headerText="Controlador" />
	</modulos:columns>
		</modulos:DataGridKruds>

	</modulos:CanvasKruds>

	<modulos:formularioCadastro>
		<controleRemoto:ControleRemoto width="100%"/>
	</modulos:formularioCadastro>

	<modulos:BarraBotoes id="barraBotoes" width="100%"></modulos:BarraBotoes>

</modulos:ListaCadastroView>
```

ControleRemoto.as

```
package modulos.controleRemoto
{
	import br.com.paulo.botoes.Botao;
	import br.com.paulo.dto.RequisicaoDto;
	import br.com.paulo.projetos.sar.controleRemoto.ControleRemotoView;
	import br.com.paulo.util.ServicoJava;
	import br.com.paulo.vo.sar.entidades.AmbienteVO;
	import br.com.paulo.vo.sar.entidades.DispositivoVO;
	import br.com.paulo.vo.sar.entidades.PortaVO;

	import flash.events.MouseEvent;

	import modulos.dispositivo.DispositivosUtil;
```

```
import mx.events.FlexEvent;
import mx.rpc.events.ResultEvent;

public class ControleRemoto extends ControleRemotoView
{

	public function ControleRemoto() {
		this.addEventListener(FlexEvent.CREATION_COMPLETE, init);
	}

	private var _ambienteSelecionado: AmbienteVO;

	private var STATUS_LIGADA: String = "ligada";
	private var STATUS_DESLIGADA: String = "desligada";

	private function init(event: FlexEvent): void {

		barraBotoes.document.removeChild(barraBotoes.document.btnLimpar);
		barraBotoes.document.removeChild(barraBotoes.document.btnCancelar);
		barraBotoes.document.removeChild(barraBotoes.document.btnOK);

		txtHistoricoAcoes.editable = false;
	}

	private function obterStatusDispositivos(): void {
		var dto: RequisicaoDto = new RequisicaoDto();
		dto.mapa.ambiente = _ambienteSelecionado;

		ServicoJava.enviar("controleRemotoFachada", "obterStatusComponentes",
retornoObterStatus, null, dto, "Obtendo status dos dispositivos");
	}

	private function retornoObterStatus(event: ResultEvent): void {
		var status: String = event.result.mapa.status;

		var porta: PortaVO;
		for each(porta in _ambienteSelecionado.controlador.portas) {
```

```
                        if (porta.tipoPorta == PortaVO.PORTA1 && (status ==
ControleRemotoUtil.RELE_1_LIGADO_E_RELE_2_DESLIGADO || status ==
ControleRemotoUtil.RELE_1_LIGADO_E_RELE_2_LIGADO)) {
                            atualizaStatusDispositivo(porta, STATUS_LIGADA);

                        } else if (porta.tipoPorta == PortaVO.PORTA2 && (status ==
ControleRemotoUtil.RELE_1_DESLIGADO_E_RELE_2_LIGADO || status ==
ControleRemotoUtil.RELE_1_LIGADO_E_RELE_2_LIGADO)) {
                            atualizaStatusDispositivo(porta, STATUS_LIGADA);
                        }
                    }
                }

                private function executaComando(event: MouseEvent): void {
                    var dto: RequisicaoDto = new RequisicaoDto();
                    dto.mapa.ambiente = _ambienteSelecionado;

                    var botao: Botao = event.currentTarget as Botao;

                    var porta: Object;
                    for each(porta in _ambienteSelecionado.controlador.portas) {
                        if (botao.id == porta.id) {
                            dto.mapa.porta = porta;
                            break;
                        }
                    }

                    ServicoJava.enviar("controleRemotoFachada", "executarComando",
retornoExecutarComando, null, dto, "Acionando dispositivo");
                }

                private function retornoExecutarComando(event: ResultEvent): void {
                    var porta: PortaVO = event.result.mapa.porta as PortaVO;
                    var mensagem: String = event.result.mapa.mensagem;

                    var status: int = mensagem.indexOf("desligada");

                    if (status != -1) {
                        atualizaStatusDispositivo(porta, STATUS_DESLIGADA);
                    } else {
                        atualizaStatusDispositivo(porta, STATUS_LIGADA);
```

```
                }

                txtHistoricoAcoes.text += event.result.mapa.mensagem +"\n";
        }

        override public function botaoLimparPressionado(event: MouseEvent): void {
                telaAmbiente.removeAllChildren();

                txtHistoricoAcoes.text = "";
        }

        override public function preencherCampos(): void {
                _ambienteSelecionado = objetoSelecionado as AmbienteVO;

                alterarPlanoFundo();

                montarDispositivosAmbiente();

                obterStatusDispositivos();
        }

        private function alterarPlanoFundo(): void {
                this.telaAmbiente.setStyle("backgroundImage",
"http://192.168.0.100:8080/SARFlex/imagem/ambiente/" +_ambienteSelecionado.caminhoImagem);
        }

        private function montarDispositivosAmbiente(): void {
                var porta: Object;
                for each(porta in _ambienteSelecionado.controlador.portas) {
                        if (porta.dispositivo != null) {
                                var dispositivo: DispositivoVO = porta.dispositivo;

                                telaAmbiente.addChild(getBotaoLigaDesliga(porta as PortaVO));
                        }
                }
        }

        private function getBotaoLigaDesliga(porta: PortaVO): Botao {
                var botao: Botao = new Botao();
                botao.id = porta.id;
```

```
                botao.x = porta.dispositivo.posicaoX;
                botao.y = porta.dispositivo.posicaoY;

                botao.setStyle("icon",
DispositivosUtil.recuperarSourceDesligados(porta.dispositivo.imagem));

                botao.setStyle("fillAlphas", new Array(0.0, 0.0, 0.0, 0.0));
                botao.addEventListener(MouseEvent.CLICK, executaComando);

                return botao;
            }

            private function atualizaStatusDispositivo(porta: PortaVO, status: String): void {
                var nomeImagem: String = porta.dispositivo.imagem;
                var botao: Botao = recuperarDispositivo(porta);
                if (status == STATUS_LIGADA) {
                    nomeImagem = nomeImagem.substr(0,
nomeImagem.indexOf("Desligado"));
                    botao.setStyle("icon",
DispositivosUtil.recuperarSourceLigados(nomeImagem+"Ligado"));
                } else {
                    botao.setStyle("icon",
DispositivosUtil.recuperarSourceDesligados(nomeImagem));
                }
            }

            private function recuperarDispositivo(porta: PortaVO): Botao {
                var botao: Botao;
                for each(botao in telaAmbiente.getChildren()) {
                    if (botao.id == porta.id) {
                        return botao;
                    }
                }
                return null;
            }
        }
}
```

ControleRemotoView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<modulos:FormularioCadastroView
	xmlns:mx="http://www.adobe.com/2006/mxml"
	xmlns:modulos="br.com.paulo.modulos.*"
	width="461" height="492"
	xmlns:componentes="br.com.paulo.componentes.*"
	xmlns:controlesTexto="br.com.paulo.controlesTexto.*" xmlns:ns1="br.com.paulo.botoes.*">

	<modulos:VBoxKruds width="100%" height="100%">

		<modulos:HBoxKruds width="100%" height="80%">

			<modulos:CanvasKruds id="telaAmbiente" width="100%" height="100%"
cornerRadius="5" borderStyle="solid" backgroundColor="#EEEFEF" >

			</modulos:CanvasKruds>

		</modulos:HBoxKruds>

		<modulos:CanvasKruds id="telaInformacao" width="100%" height="20%" cornerRadius="5"
borderStyle="solid" backgroundColor="#EEEFEF" >

			<componentes:SubtituloView texto="Histórico de ações" x="10" y="10" width="438"
/>
			<controlesTexto:AreaTexto id="txtHistoricoAcoes" x="10" y="43" width="438"
height="42"/>

		</modulos:CanvasKruds>

	</modulos:VBoxKruds>

	<modulos:barraBotoes>
		<modulos:BarraBotoesFormularioCadastroView id="barraBotoes" width="100%" />
	</modulos:barraBotoes>

</modulos:FormularioCadastroView>
```

ControleRemotoUtil.as

```
package modulos.controleRemoto
{
	import flash.utils.Dictionary;
```

```
	public class ControleRemotoUtil
	{
		public function ControleRemotoUtil()
		{

		}

		public static var RELE_1_DESLIGADO_E_RELE_2_DESLIGADO: String = "0x00";

		public static var RELE_1_LIGADO_E_RELE_2_DESLIGADO: String = "0x01";

		public static var RELE_1_DESLIGADO_E_RELE_2_LIGADO: String = "0x02";

		public static var RELE_1_LIGADO_E_RELE_2_LIGADO: String = "0x03";

	}
}
```

ControleRemotoDelegate.java

```
package br.com.paulo.sar.delegate;

public class ControleRemotoDelegate {

	public String executarComando(Ambiente ambiente, Porta porta) throws SystemException,
ControleRemotoException {
		return getServico().executarComando(ambiente, porta);
	}

	public String verificarStatusDispositivos(Ambiente ambiente) throws SystemException,
ControleRemotoException {
		return getServico().verificarStatusDispositivos(ambiente);
	}

	private IControleRemotoServico getServico() throws SystemException {
		return SarServiceLocator.getInstance().recuperarControleRemotoServico();
	}
}
```

ControleRemotoFachada.java

```
package br.com.paulo.sar.fachada;
```

```
public class ControleRemotoFachada {

	private ControleRemotoDelegate controleRemotoDelegate =
SarDelegateFactory.getInstance().getControleRemotoDelegate();

	public RetornoDto obterStatusComponentes(RequisicaoDto dto) throws SystemException,
ControleRemotoException {
		Ambiente ambiente = (Ambiente) dto.getMapa().get("ambiente");

		String respostaStatus = controleRemotoDelegate.verificarStatusDispositivos(ambiente);

		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put("status", respostaStatus);
		return retornoDto;
	}

	@SuppressWarnings("unchecked")
	public RetornoDto executarComando(RequisicaoDto dto) throws SystemException,
ControleRemotoException {

		Ambiente ambiente = (Ambiente) dto.getMapa().get("ambiente");
		Porta porta = (Porta) dto.getMapa().get("porta");

		String resposta = controleRemotoDelegate.executarComando(ambiente, porta);

		RetornoDto retornoDto = new RetornoDto();
		retornoDto.getMapa().put("mensagem", resposta);
		retornoDto.getMapa().put("porta", porta);
		return retornoDto;
	}
}
```

ControleRemoto.java

```
package br.com.paulo.sar.util;
public abstract class ControleRemoto implements SerialPortEventListener {

	private CommPortIdentifier porta;
	private SerialPort serialPort;
	private OutputStream saida;
```

```
	private InputStream entrada;

	public ControleRemoto() {
		try {

			porta = CommPortIdentifier.getPortIdentifier("COM4");

		} catch (NoSuchPortException e) {
			e.printStackTrace();
		}
	}

	public void abrirConexao() throws ControleRemotoException {
		try {
			serialPort = (SerialPort) porta.open("ControleRemotoUtil", 2000);

			serialPort.setSerialPortParams(38400, SerialPort.DATABITS_8,
SerialPort.STOPBITS_1, SerialPort.PARITY_NONE);
			serialPort.setFlowControlMode(SerialPort.FLOWCONTROL_NONE);

			serialPort.addEventListener(this);
			serialPort.notifyOnDataAvailable(true);

			setSaida(serialPort.getOutputStream());
		} catch (IOException e) {
			e.printStackTrace();
		} catch (UnsupportedCommOperationException e) {
			e.printStackTrace();
		} catch (TooManyListenersException e) {
			e.printStackTrace();
		} catch (PortInUseException e) {
			e.printStackTrace();
		} catch (Exception e) {
			throw new ControleRemotoException("msg.erro.padrao");
		}
	}

	public void fecharConexao() {
		try {
			if (entrada == null) {
```

```
                        while(entrada == null) {

                        }
                }
                saida.close();
                entrada.close();
                serialPort.close();
        } catch (IOException e) {
                e.printStackTrace();
        }
}

        public void escreverDados(String identificador, String codigoExecucao) throws
ControleRemotoException {
                byte[] codigoSaida = montarCodigoByte(identificador, codigoExecucao);

                try {
                        saida.write(codigoSaida);
                        saida.flush();
                } catch (IOException e) {
                        throw new ControleRemotoException("msg.erro.enviarDados");
                }
        }

        public void lerStatus(String codigoIdentificador) throws ControleRemotoException {
                byte[] codigoSaida = montarCodigoByteStatus(codigoIdentificador);

                try {
                        saida.write(codigoSaida);
                        saida.flush();
                } catch (IOException e) {
                        throw new ControleRemotoException("msg.erro.enviarDadosEstado");
                }
        }

        private byte[] montarCodigoByte(String codigoIdentificador, String codigoExecucao) {
                byte[] listaCodigoByte = new byte[11];

                listaCodigoByte[0] = getByte("7E");
                listaCodigoByte[1] = getByte("00");
```

```
        listaCodigoByte[2] = getByte("07");
        listaCodigoByte[3] = getByte("01");
        listaCodigoByte[4] = getByte("01");
        listaCodigoByte[5] = getByte(codigoIdentificador.substring(0, 2));
        listaCodigoByte[6] = getByte(codigoIdentificador.substring(2, codigoIdentificador.length()));
        listaCodigoByte[7] = getByte("00");
        listaCodigoByte[8] = getByte("7B");
        listaCodigoByte[9] = getByte(codigoExecucao);

        listaCodigoByte[listaCodigoByte.length - 1] = (calcularChecksum(listaCodigoByte));
        return listaCodigoByte;
}

private byte[] montarCodigoByteStatus(String codigoIdentificador) {
        byte[] listaCodigoByte = new byte[11];

        listaCodigoByte[0] = getByte("7E");
        listaCodigoByte[1] = getByte("00");
        listaCodigoByte[2] = getByte("07");
        listaCodigoByte[3] = getByte("01");
        listaCodigoByte[4] = getByte("01");
        listaCodigoByte[5] = getByte(codigoIdentificador.substring(0, 2));
        listaCodigoByte[6] = getByte(codigoIdentificador.substring(2, codigoIdentificador.length()));
        listaCodigoByte[7] = getByte("00");
        listaCodigoByte[8] = getByte("7C");
        listaCodigoByte[9] = getByte("00");

        listaCodigoByte[listaCodigoByte.length - 1] = (calcularChecksum(listaCodigoByte));
        return listaCodigoByte;
}

private Byte getByte(String hexadecimal) {
        return Byte.valueOf(hexadecimal, 16);
}

private Byte calcularChecksum(byte[] codigo) {
        Integer checksum = 0;

        for (int i = 3; i < codigo.length; i++) {
                checksum += codigo[i];
```

```
            }

            // discard values > 1 byte
            checksum = 0xff & checksum;
            // perform 2s complement
            checksum = 0xff - checksum;

            return checksum.byteValue();
}

public void serialEvent(SerialPortEvent evt) {
            SerialPort portaSerial = (SerialPort) evt.getSource();
            try {
                        setEntrada(portaSerial.getInputStream());
                        switch (evt.getEventType()) {

                                    case SerialPortEvent.BI: {

                                    }

                                    case SerialPortEvent.OE: {

                                    }

                                    case SerialPortEvent.FE: {

                                    }

                                    case SerialPortEvent.PE: {

                                    }

                                    case SerialPortEvent.CD: {

                                    }

                                    case SerialPortEvent.CTS: {

                                    }
```

```
                        case SerialPortEvent.DSR: {

                        }

                        case SerialPortEvent.RI: {

                        }

                        case SerialPortEvent.OUTPUT_BUFFER_EMPTY: {
                                break;
                        }

                        case SerialPortEvent.DATA_AVAILABLE: {
                                while (entrada.available() > 0) {
                                        handleEventoDataAvailable();
                                }
                        }
                }
        } catch (IOException e) {
                e.printStackTrace();
        }
}

public String filtrarRespostaStatus(String resposta) {
        String[] codigosResposta = resposta.split(" ");

        String codigoStatus = "";

        for (int i = 0; i < codigosResposta.length; i++) {
                if (codigosResposta[i].equals("0x52") ) {
                        codigoStatus = codigosResposta[++i];
                        break;
                }
        }

        return codigoStatus;
}

public String filtrarRespostaExecucao(String resposta) {
        String[] codigosResposta = resposta.split(" ");
```

```
		String codigoStatus = "";

		for (int i = 0; i < codigosResposta.length; i++) {
			if (codigosResposta[i].equals("0x57") ) {

				int posicaoRSSI = i - 2;

				String sinal = codigosResposta[posicaoRSSI];

				System.out.println("-----------------Pacote de dados recebido - Nível de sinal
RSSI: " +Integer.parseInt(sinal.replace("0x", ""), 16));

				codigoStatus = codigosResposta[++i];
				break;
			}
		}

		return codigoStatus;
	}

	public String controlarRele1(String status) throws ControleRemotoException {
		String codigoExecucao = "";

	if(status.equals(ControleRemotoParametro.RELE_1_DESLIGADO_E_RELE_2_DESLIGADO)) {

			codigoExecucao =
ControleRemotoParametro.RELE_1_LIGADO_E_RELE_2_DESLIGADO;

		} else
if(status.equals(ControleRemotoParametro.RELE_1_DESLIGADO_E_RELE_2_LIGADO)) {

			codigoExecucao =
ControleRemotoParametro.RELE_1_LIGADO_E_RELE_2_LIGADO;

		} else if
(status.equals(ControleRemotoParametro.RELE_1_LIGADO_E_RELE_2_DESLIGADO)) {

			codigoExecucao =
ControleRemotoParametro.RELE_1_DESLIGADO_E_RELE_2_DESLIGADO;

		} else if (status.equals(ControleRemotoParametro.RELE_1_LIGADO_E_RELE_2_LIGADO))
{
```

```
                codigoExecucao =
ControleRemotoParametro.RELE_1_DESLIGADO_E_RELE_2_LIGADO;
        } else {
                throw new ControleRemotoException("msg.erro.generico");
        }

        return codigoExecucao.substring(2, codigoExecucao.length());
    }

    public String controlarRele2(String status) throws ControleRemotoException {
        String codigoExecucao = "";

    if(status.equals(ControleRemotoParametro.RELE_1_DESLIGADO_E_RELE_2_DESLIGADO)) {

                codigoExecucao =
ControleRemotoParametro.RELE_1_DESLIGADO_E_RELE_2_LIGADO;

        } else
if(status.equals(ControleRemotoParametro.RELE_1_LIGADO_E_RELE_2_DESLIGADO)) {

                codigoExecucao =
ControleRemotoParametro.RELE_1_LIGADO_E_RELE_2_LIGADO;

        } else if
(status.equals(ControleRemotoParametro.RELE_1_DESLIGADO_E_RELE_2_LIGADO)) {

                codigoExecucao =
ControleRemotoParametro.RELE_1_DESLIGADO_E_RELE_2_DESLIGADO;

        } else if (status.equals(ControleRemotoParametro.RELE_1_LIGADO_E_RELE_2_LIGADO))
{
                codigoExecucao =
ControleRemotoParametro.RELE_1_LIGADO_E_RELE_2_DESLIGADO;
        } else {
                throw new ControleRemotoException("msg.erro.generico");
        }

        return codigoExecucao.substring(2, codigoExecucao.length());
    }

    public String toHexaString(int valor) {

        if (valor > 0xff) {
```

```
                throw new IllegalArgumentException("Valor é maior do que o byte");
        }

        if (valor < 0x10) {
                return "0x0" + Integer.toHexString(valor);
        } else {
                return "0x" + Integer.toHexString(valor);
        }
}

/**
 * @return the serialPort
 */
public SerialPort getSerialPort() {
        return serialPort;
}

/**
 * @param serialPort the serialPort to set
 */
public void setSerialPort(SerialPort serialPort) {
        this.serialPort = serialPort;
}

/**
 * @return the saida
 */
public OutputStream getSaida() {
        return saida;
}

/**
 * @param saida the saida to set
 */
public void setSaida(OutputStream saida) {
        this.saida = saida;
}

/**
 * @return the entrada
```

```
	 */
	public InputStream getEntrada() {
		return entrada;
	}

	/**
	 * @param entrada the entrada to set
	 */
	public void setEntrada(InputStream entrada) {
		this.entrada = entrada;
	}

	public abstract void handleEventoDataAvailable();
}
```

ControleRemotoServico.java

```
package br.com.paulo.sar.servico.impl;

@Stateless()
public class ControleRemotoServico extends ControleRemoto implements IControleRemotoServico, Runnable {

	private String resposta = "";

	public String executarComando(Ambiente ambiente, Porta porta) throws SystemException,
ControleRemotoException {
		String respostaExecucao = "";

		resposta = "";

		abrirConexao();
		lerStatus(ambiente.getControlador().getIdentificador());
		Thread t = new Thread(this);
		t.run();

		fecharConexao();

		String codigoExecucao = "";
		if(porta.getTipoPorta() == TipoPorta.PORTA1) {
			codigoExecucao = controlarRele1(filtrarRespostaStatus(resposta));
		} else if (porta.getTipoPorta() == TipoPorta.PORTA2) {
```

```
                codigoExecucao = controlarRele2(filtrarRespostaStatus(resposta));
            }

            resposta = "";

            abrirConexao();
            escreverDados(ambiente.getControlador().getIdentificador(), codigoExecucao);
            t = new Thread(this);
            t.run();

            fecharConexao();

            if(porta.getTipoPorta() == TipoPorta.PORTA1) {
                respostaExecucao =
ControleRemotoParametro.getStatusPorta1(filtrarRespostaExecucao(resposta));
            } else if (porta.getTipoPorta() == TipoPorta.PORTA2) {
                respostaExecucao =
ControleRemotoParametro.getStatusPorta2(filtrarRespostaExecucao(resposta));
            }

            if (respostaExecucao == null) {
                throw new ControleRemotoException("msg.erro.generico");
            }

            return respostaExecucao;
        }

        public String verificarStatusDispositivos(Ambiente ambiente) throws SystemException,
ControleRemotoException {
            resposta = "";

            abrirConexao();
            lerStatus(ambiente.getControlador().getIdentificador());
            Thread t = new Thread(this);
            t.run();

            fecharConexao();

            String respostaStatus = filtrarRespostaStatus(resposta);

            if (respostaStatus == null) {
```

```
                    throw new ControleRemotoException("msg.erro.generico");
                }
                return respostaStatus;
        }

        @Override
        public void handleEventoDataAvailable() {
                try {
                        if (getEntrada().available() > 0) {
                                resposta += toHexaString(getEntrada().read());
                                resposta += " ";
                        }
                } catch (IOException e) {
                        e.printStackTrace();
                }
        }

        public void run() {
                try {
                        Thread.sleep(1000);
                } catch (InterruptedException e) {
                        e.printStackTrace();
                }
        }
}
```

ControleRemotoParametro.java

```
package br.com.paulo.sar.util;

public class ControleRemotoParametro {

        public static String RELE_1_DESLIGADO_E_RELE_2_DESLIGADO = "0x00";

        public static String RELE_1_LIGADO_E_RELE_2_DESLIGADO = "0x01";

        public static String RELE_1_DESLIGADO_E_RELE_2_LIGADO = "0x02";

        public static String RELE_1_LIGADO_E_RELE_2_LIGADO = "0x03";
```

```
	private static Map<String, String> statusPorta1 = null;
	private static Map<String, String> statusPorta2 = null;

	public static String getStatusPorta1(String key) {
		if (statusPorta1 == null) {
			statusPorta1 = new HashMap<String, String>();

			statusPorta1.put(RELE_1_DESLIGADO_E_RELE_2_DESLIGADO, "Porta 1 desligada com sucesso!");
			statusPorta1.put(RELE_1_DESLIGADO_E_RELE_2_LIGADO, "Porta 1 desligada com sucesso!");

			statusPorta1.put(RELE_1_LIGADO_E_RELE_2_DESLIGADO, "Porta 1 ligada com sucesso!");
			statusPorta1.put(RELE_1_LIGADO_E_RELE_2_LIGADO, "Porta 1 ligada com sucesso!");
		}
		return statusPorta1.get(key);
	}

	public static String getStatusPorta2(String key) {
		if (statusPorta2 == null) {
			statusPorta2 = new HashMap<String, String>();

			statusPorta2.put(RELE_1_DESLIGADO_E_RELE_2_DESLIGADO, "Porta 2 desligada com sucesso!");
			statusPorta2.put(RELE_1_LIGADO_E_RELE_2_DESLIGADO, "Porta 2 desligada com sucesso!");

			statusPorta2.put(RELE_1_DESLIGADO_E_RELE_2_LIGADO, "Porta 2 ligada com sucesso!");
			statusPorta2.put(RELE_1_LIGADO_E_RELE_2_LIGADO, "Porta 2 ligada com sucesso!");
		}
		return statusPorta2.get(key);
	}
}
```

APÊNDICE C

```
CREATE SCHEMA IF NOT EXISTS `cta`;
USE `cta`;

CREATE TABLE IF NOT EXISTS `cta`.`menu` (
  `ID` BIGINT (20) NOT NULL AUTO_INCREMENT,
  `IDTELA` VARCHAR (255) NOT NULL,
  `LABEL` VARCHAR (255) NOT NULL,
  `CLASSE` VARCHAR (255) NOT NULL,
  `DESATIVARSEGUNDOPLANO` BIT (1) NOT NULL,
  `POPUPCENTRALIZADO` BIT (1) NOT NULL,
  `CAMINHOIMAGEM` VARCHAR (255) NOT NULL,
  PRIMARY KEY (`ID`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `cta`.`permissao` (
  `ID` BIGINT (20) NOT NULL AUTO_INCREMENT,
  `Role` VARCHAR (255) NULL DEFAULT NULL,
  PRIMARY KEY (`ID`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `cta`.`relmenupermissao` (
  `IDMENU` BIGINT (20) NOT NULL,
  `IDPERMISSAO` BIGINT (20) NOT NULL,
  CONSTRAINT `FKIDMENU`
    FOREIGN KEY (`IDMENU`)
    REFERENCES `cta`.`menu` (`ID`),
  CONSTRAINT `FKIDPERMISSAO`
    FOREIGN KEY (`IDPERMISSAO`)
    REFERENCES `cta`.`permissao` (`ID`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `cta`.`usuario` (
  `ID` BIGINT (20) NOT NULL AUTO_INCREMENT,
  `NOME` VARCHAR (100) NOT NULL,
  `EMAIL` VARCHAR (100) NULL DEFAULT NULL,
  `USERNAME` VARCHAR (60) NOT NULL,
  `SENHA` VARCHAR (60) NOT NULL,
```

```
  `IDPERMISSAO` BIGINT (20) NULL DEFAULT NULL,
  PRIMARY KEY (`ID`),
  CONSTRAINT `FKIDPERMISSAO`
    FOREIGN KEY (`IDPERMISSAO`)
    REFERENCES `cta`.`permissao` (`ID`))
ENGINE = InnoDB;

CREATE SCHEMA IF NOT EXISTS `sar`;
USE `sar`;

CREATE TABLE IF NOT EXISTS `sar`.`controlador` (
  `ID` BIGINT (20) NOT NULL AUTO_INCREMENT,
  `IDENTIFICADOR` VARCHAR (4) NOT NULL,
  `DESCRICAO` VARCHAR (255) NOT NULL,
  PRIMARY KEY (`ID`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `sar`.`ambiente` (
  `ID` BIGINT (20) NOT NULL AUTO_INCREMENT,
  `NOME` VARCHAR (60) NOT NULL,
  `DESCRICAO` VARCHAR (255) NOT NULL,
  `CAMINHOIMAGEM` VARCHAR (255) NOT NULL,
  `IDCONTROLADOR` BIGINT (20) NOT NULL,
  PRIMARY KEY (`ID`),
  CONSTRAINT `FKIDCONTROLADOR`
    FOREIGN KEY (`IDCONTROLADOR`)
    REFERENCES `sar`.`controlador` (`ID`))
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `sar`.`porta` (
  `ID` BIGINT (20) NOT NULL AUTO_INCREMENT,
  `DESCRICAO` VARCHAR (255) NOT NULL,
  `CODIGOLIGA` VARCHAR (255) NOT NULL,
  `CODIGODESLIGA` VARCHAR (255) NOT NULL,
  `TIPOPORTA` INT (11) NULL DEFAULT NULL,
  `IDCONTROLADOR` BIGINT (20) NULL DEFAULT NULL,
  PRIMARY KEY (`ID`),
  CONSTRAINT `FKIDCONTROLADOR`
    FOREIGN KEY (`IDCONTROLADOR`)
    REFERENCES `sar`.`controlador` (`ID`))
```

```
ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS `sar`.`dispositivo` (
  `ID` BIGINT (20) NOT NULL AUTO_INCREMENT,
  `NOME` VARCHAR (255) NOT NULL,
  `IMAGEM` VARCHAR (255) NOT NULL,
  `POSICAOX` INT (11) NOT NULL,
  `POSICAOY` INT (11) NOT NULL,
  `IDPorta` BIGINT (20) NULL DEFAULT NULL,
  PRIMARY KEY (`ID`),
  CONSTRAINT `FKIDPORTA`
    FOREIGN KEY (`IDPorta`)
    REFERENCES `sar`.`porta` (`ID`))
ENGINE = InnoDB;
```

ANEXO A – DATASHEET XBEE™/XBEE-PRO™ OEM RF MODULES

ANEXO B – DATASHEET PLACA RCOM-HOMEBEE

ANEXO C – DATASHEET PLACA CON-USBBEE