

**UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO**

FLÁVIO DE OLIVEIRA VALENTIM

**INTEGRANDO TECNOLOGIAS NA CONSTRUÇÃO DE UM
SISTEMA DE AUTOMAÇÃO RESIDENCIAL SEM FIO**

VITÓRIA – ES
JULHO DE 2014

FLÁVIO DE OLIVEIRA VALENTIM

INTEGRANDO TECNOLOGIAS NA CONSTRUÇÃO DE UM SISTEMA DE AUTOMAÇÃO RESIDENCIAL SEM FIO

Parte manuscrita do Projeto de Graduação do aluno **Flávio de Oliveira Valentim**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito para obtenção do grau de Engenheiro Eletricista.

Orientador: Dr. Celso J. Munaro

VITÓRIA – ES
JULHO DE 2014

FLÁVIO DE OLIVEIRA VALENTIM

INTEGRANDO TECNOLOGIAS NA CONSTRUÇÃO DE UM SISTEMA DE AUTOMAÇÃO RESIDENCIAL SEM FIO

Parte manuscrita do Projeto de Graduação do aluno **Flávio de Oliveira Valentim**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovada em 30, de Julho de 2014.

COMISSÃO EXAMINADORA:

Dr. Celso J. Munaro
Universidade Federal do Espírito Santo
Orientador

Dra. Eliete Maria de Oliveira Caldeira
Universidade Federal do Espírito Santo
Examinador

Dr. Jair Adriano Silva
Universidade Federal do Espírito Santo
Examinador

A Deus, criador de todas as coisas. A minha família, noiva e amigos pelo incentivo e suporte em todos os momentos.

AGRADECIMENTOS

Agradeço a Deus, que me permitiu vencer mais esta etapa da vida. Aos meus pais e irmã, João, Dulcinea e Karina, pelo suporte e incentivo em todos os momentos. Seus exemplos de vida me tornaram o que hoje sou.

A minha noiva Debora, pelo carinho e compreensão e por me incentivar, mesmo com lágrimas de saudade, a viver uma experiência no exterior.

Aos familiares e amigos que me proporcionaram momentos de lazer incríveis, sem os quais seria difícil chegar ao fim desta jornada.

A todos os colegas de classe pelo companheirismo em todos esses anos. Em especial ao Vinícius Catrinque e Marcus Pagum, por serem mais que amigos, verdadeiros irmãos, que estiveram presente em todos os momentos.

Ao orientador e amigo Dr. Celso J. Munaro, por acreditar no meu trabalho, demonstrando sempre entusiasmo e compromisso no desenvolvimento do mesmo.

“Não confunda derrotas com fracasso nem vitórias com sucesso. Na vida de um campeão sempre haverá algumas derrotas, assim como na vida de um perdedor sempre haverá vitórias. A diferença é que, enquanto os campeões crescem nas derrotas, os perdedores se acomodam nas vitórias.”

(Roberto Shinyashiki)

Sumário

LISTA DE FIGURAS.....	8
LISTA DE TABELAS.....	10
RESUMO	11
ABSTRACT.....	12
1 INTRODUÇÃO	13
1.1. Objetivos.....	13
1.2. Trabalhos Relacionados.....	14
1.3. Estrutura do Trabalho.....	15
2 TECNOLOGIAS UTILIZADAS.....	16
2.1. Protocolos de Comunicação.....	16
2.1.1. Bluetooth.....	16
2.1.2. Interface Periférica Serial (Serial Peripheral Interface - SPI).....	18
2.1.3. Protocolo de Comunicação RS-232.....	19
2.1.4. Protocolo de Controle de Transmissão (Transmission Control Protocol - TCP). ..	20
2.2. Microcontroladores.....	21
2.2.1. Kits de Desenvolvimento	22
2.2.2. Arduino Mega.....	22
2.2.3. Arduino Nano.....	23
2.2.4. Arduino Ethernet.....	24
2.3. A Linguagem de Programação Java.....	25
2.3.1. O Sistema Operacional Android.....	25
2.4. Transmissão de Dados Sem Fio.....	26
2.4.1. NRF24L01+.....	27
3 MODELO DE SOLUÇÃO PROPOSTA.....	29
3.1. Interface Homem Máquina - IHM.....	29
3.1.1. Comunicação via Bluetooth e Wi-Fi.....	30
3.1.2. Configuração.....	32
3.1.3. Comandos do Sistema.....	33
3.2. Central de Controle - CC.....	35
3.2.1. Consistência de Dados.....	35
3.2.2. Decodificação	36

3.2.3. Encaminhamento.....	36
3.3. Rede de Sensores Sem Fio - RSSF.....	37
3.3.1. Algoritmo de Roteamento.....	37
3.3.2. Configuração dos Sensores.....	38
3.3.3. Execução de Comandos.....	39
3.3.4. Performance da RSSF.....	39
4 PROJETO DESENVOLVIDO.....	42
4.1. Interface Homem Máquina - IHM.....	42
4.1.1. Interação com o Usuário.....	42
4.1.2. Configuração.....	46
4.1.3. Codificação das Mensagens.....	49
4.2. Central de Controle - CC.....	51
4.2.1. Formatação e Encaminhamento das Mensagens.....	52
4.3. Rede de Sensores Sem Fio - RSSF.....	53
4.3.1. Configuração do Modo de Funcionamento.....	54
4.3.2. Decodificação e Execução de Comandos.....	56
4.4. Interface com Atuadores Analógicos.....	58
4.4.1. Controle de Potência (Dimmer).....	58
4.4.2. Controle de Temperatura.....	60
5 CONCLUSÃO.....	62
REFERÊNCIAS BIBLIOGRÁFICAS.....	63

LISTA DE FIGURAS

Figura 1.1: Visão geral do projeto.	14
Figura 2.1: Esquema de uma piconet e a formação de uma scatternet.....	17
Figura 2.2: Esquema de comunicação SPI.....	18
Figura 2.3: Padrão RS-232 com três ligações.....	20
Figura 2.4: Visão superior da placa Arduino Mega2560.....	23
Figura 2.5: Visão superior e inferior da placa Arduino Nano.....	24
Figura 2.6: Visão superior e inferior da placa Arduino Ethernet.....	24
Figura 2.7: NRF24L01+ da fabricante Nordic Semiconductor.....	28
Figura 3.1: Visão topológica do projeto.....	29
Figura 3.2: Algoritmo de criação de canal de comunicação Bluetooth e Wi-Fi.....	31
Figura 3.3: Algoritmo de gerenciamento e envio de mensagens.....	32
Figura 3.4: Codificação de mensagem trocada entre a IHM e a CC.....	34
Figura 3.5: Codificação de mensagem enviada à RSSF.....	36
Figura 3.6: Modelo de roteamento da RSSF.	37
Figura 3.7: Topologia árvore adotada e sua divisão em níveis.....	38
Figura 3.8: Performance de mensagens trocadas entre nós de nível 0 e 3.....	40
Figura 3.9: Performance de mensagens trocadas entre nós de nível 0 e 1.....	41
Figura 4.1: Sistema desenvolvido.....	42
Figura 4.2: Planta baixa exibida ao usuário na IHM em plataforma Android.....	43
Figura 4.3: Opções de comando básicas disponibilizadas ao usuário.....	44
Figura 4.4: Comandos de controle de alarme e abertura de garagem.....	45
Figura 4.5: Interface do tocador de áudio.....	46
Figura 4.6: Opções de configuração gerais.....	47
Figura 4.7: Configuração de conexão TCP.....	48
Figura 4.8: Configuração de conexão Bluetooth.....	49
Figura 4.9: Protótipo da central de controle testada.....	52
Figura 4.10: Nós da rede formada e seus respectivos endereços octadecimais.....	53
Figura 4.11: Acessando a interface de configuração via comunicação serial.....	54
Figura 4.12: Modificando o modo de operação do atuador através de interface serial.....	55
Figura 4.13: Vista exterior do modelo de ambiente residencial utilizado.....	57

Figura 4.14: Vista interior do modelo de ambiente residencial utilizado.....	57
Figura 4.15: Esquemático do circuito de detecção de cruzamento de zero.....	59
Figura 4.16: Circuito de controle de potência através de um tiristor.....	59
Figura 4.17: Circuito de acionamento e controle de potência implementado.....	60
Figura 4.18: Circuito de controle de dispositivos através de emissor infravermelho.....	61

LISTA DE TABELAS

Tabela 4.1: Endereço dos sensores.....	50
Tabela 4.2: Comandos disponíveis e seus respectivos códigos.....	50
Tabela 4.3: Parâmetros configuráveis de cada nó da rede.....	56
Tabela 4.4: Modo de funcionamento dos módulos microcontrolados.....	56

RESUMO

Nos últimos anos, o desenvolvimento de tecnologias para automação residencial tem crescido de maneira acentuada, especialmente com o uso de comunicação sem fio. O problema é que o preço é raramente atrativo o que torna difícil a aceitação dos mesmos. No entanto, um grande grupo de tecnologias são disponibilizadas nos aparelhos móveis, o que torna o seu uso uma boa prática em termos de aceitação e diminuição de custo. Este projeto descreve a implementação de um sistema de automação residencial composto por uma interface desenvolvida em sistema Android, um módulo de controle central para gerenciamento de dados e uma rede de sensores que utiliza o circuito radiotransmissor NRF24L01. Os nós que compõem a rede e o módulo central são comandados por microcontrolador da família AVR. Testes de performance comprovaram a eficiência da topologia e do *hardware* utilizado na rede de sensores. Além disso, um protótipo foi construído para validação do projeto. Futuras melhorias são também discutidas.

Palavras-chaves: Sistema de Automação Residencial, Rede de Sensores sem Fio, Interface Homem Máquina, Microcontroladores, Sistemas Embarcados.

ABSTRACT

In recent years, the development of technologies for automating homes and buildings has highly increased, especially with the use of wireless communication. The problem is that the price is hardly attractive which makes difficult their acceptance. However, a large group of technologies are available on mobile devices, which makes its use a good practice in terms of acceptance in addition to decreasing costs. This project describes the implementation of a Building Automation System composed by an interface developed on Android system, a central control module for handling data and a Sensor Network based on the NRF24L01 radio transceiver. The Network's nodes and the central module are commanded by AVR's family microcontroller. Several tests have been proved the efficiency of the network topology and hardware. Furthermore, a prototype has been made for validating the project. Future improvements are also discussed.

Keywords: Building Automation System, Wireless Sensor Network, Human Machine Interface, Android, Microcontroller, Embedded Systems.

1 INTRODUÇÃO

O desenvolvimento de um Sistema de Automação Residencial (SAR) envolve as seguintes variáveis: confiabilidade, fácil instalação, interface simplificada e custo. O problema é que essas variáveis não estão relacionadas umas com as outras. Por exemplo, um sistema de automação tradicional cabeado, baseado em um Controlador Lógico Programável (CLP) é altamente confiável, mas sua instalação é complicada devido sobretudo aos cabos que são necessários utilizar e seu custo é alto.

No entanto, existe um grande grupo de tecnologias de comunicação atualmente disponíveis na maior parte dos dispositivos de telefonia móvel que podem ser utilizados na implementação de sistemas de automação: *Bluetooth*, *Wi-Fi*, Navegadores de Internet e outros. Portanto, a chave para o desenvolvimento de SARs está na integração dessas tecnologias aliada ao uso de microcontroladores de baixo custo. Existem diversas soluções propostas no mercado, mas nenhuma delas está consolidada. Algumas por não serem soluções completas (*hardware* e *software* de interface) e outras, mesmo sendo completas, têm preços raramente atrativos.

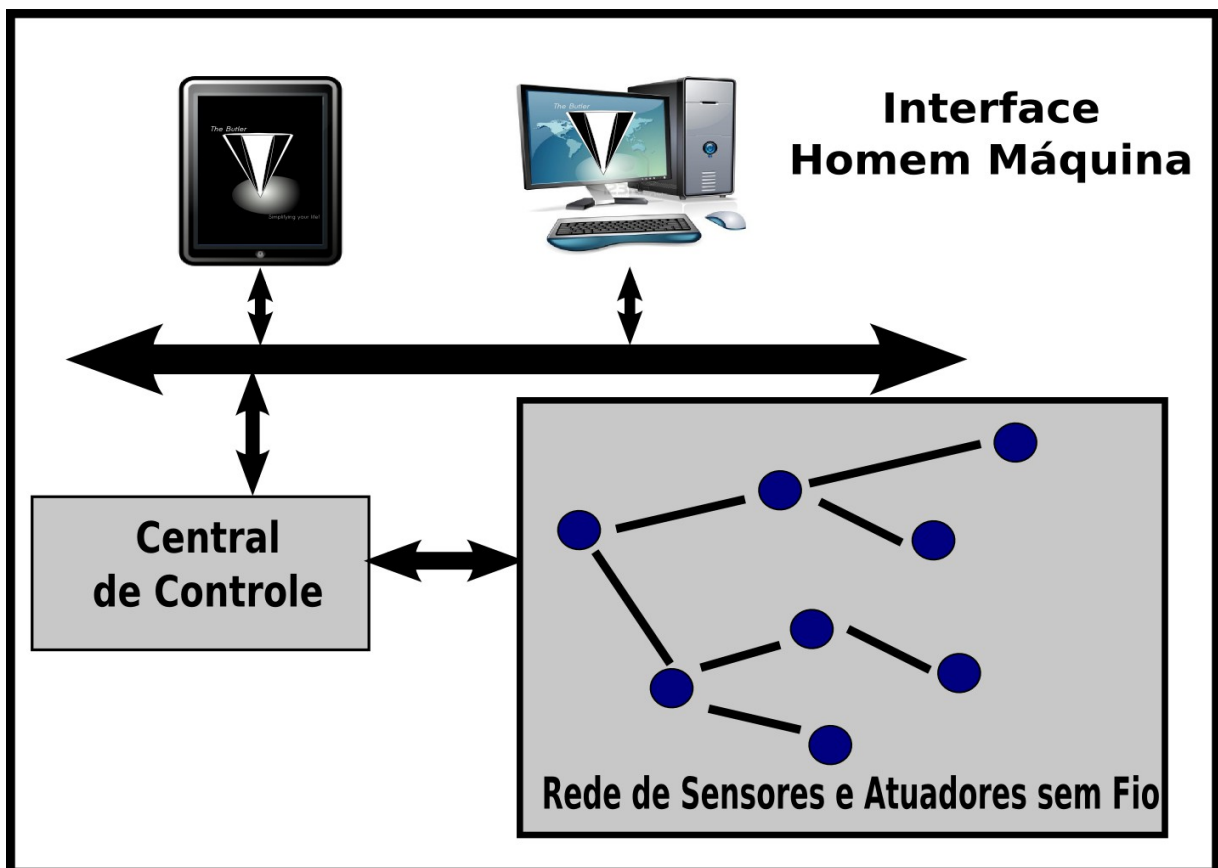
Este projeto propõe uma solução para as questões acima através de um sistema composto de uma Interface Homem Máquina (IHM), uma Central de Controle (CC) e uma Rede de Sensores Sem Fio (RSSF). Essa arquitetura é uma hierarquia de três camadas (Figura 1.1). A IHM está na camada de gerenciamento e permite ao usuário comandar o sistema. A CC se encontra na camada de automação e consiste em uma ponte de conexão entre a IHM e a RSSF. Finalmente, a RSSF fica na camada de campo, onde cada nó consiste em um sensor ou atuador que executa as ações de comando.

1.1 Objetivos

Este projeto tem por objetivo o desenvolvimento de um SAR que proporcione ao usuário uma experiência agradável em sua interação com o ambiente doméstico em que vive, aumentando substancialmente o conforto e comodidade transmitidos pelo mesmo, sendo acessível a partir de um simples toque na tela de um *smartphone*.

O sistema deverá ser modular, para que se possa expandi-lo conforme a necessidade. Os principais parâmetros controlados serão: luminosidade, temperatura e música ambiente. Esse controle deverá ser feito através de uma IHM amigável cuja instalação e configuração seja simplificada podendo ser feita pelo próprio usuário e a um custo viável, tornando-o assim potencialmente comercializável.

Figura 1.1: Visão geral do projeto.



Fonte: Produção do próprio autor.

1.2 Trabalhos Relacionados

Existem atualmente alguns SARs disponíveis no mercado como por exemplo a IHM proposta por (WISER, 2014), que permite controlar um certo grupo de dispositivos da fabricante francesa Schneider Electric. Esta, por sua vez, comercializa o sistema (MYSEHOME, 2014) que visa reduzir o consumo de energia. Numa linha que visa um sistema de instalação simplificada e modular, a marca (Z-WAVE, 2014) tem se destacado nesse mercado.

Muitos estudos estão sendo feitos em busca verificar a viabilidade de novas tecnologias na construção de SARs. Dentre os quais se destaca o trabalho de (HSIAO et al., 2012) que propõe a integração entre PLCs e uma RSSF baseada em ZigBee através de um conversor de protocolos. O trabalho de (BARAKA et al., 2013) que propõe um sistema híbrido utilizando os protocolos ZigBee e X10. Além de (SHANI et al., 2012), que implementa uma rede sem fio baseada no dispositivo NRF24L01 da fabricante NORDIC Semiconductor.

Este projeto integra o uso de tecnologias consolidadas, como o *Bluetooth* e *Wi-Fi* com uma RSSF baseada no circuito NRF24L01 e microcontroladores da família AVR disponíveis nas plataformas Arduino. Dessa forma, os custos de projeto são largamente reduzidos.

1.3 Estrutura do Trabalho

Este trabalho apresenta no Capítulo 2 as tecnologias utilizadas, com o princípio teórico das mesmas. Na sequência, o Capítulo 3 apresenta a visão topológica do modelo de solução proposta. Em seguida, o Capítulo 4 trata da escolha das tecnologias e implementação do sistema proposto, acrescido da construção de um protótipo para validação do sistema. Por fim, no Capítulo 5 estão a conclusão do trabalho, a análise dos resultados e futuras melhorias.

2 TECNOLOGIAS UTILIZADAS

Para que seja possível desenvolver um modelo que atenda ao propósito de criar um ambiente residencial que possa interagir com o usuário, será necessária a utilização de um amplo conjunto de tecnologias que terão seus aspectos técnicos abordados no decorrer deste capítulo.

2.1 Protocolos de Comunicação

Um microcontrolador, após executar uma série de instruções, normalmente deverá trocar informações com seus periféricos, que em muitos casos trabalham em diferentes frequências e não compartilham o barramento de dados do microcontrolador, sendo necessário um acordo entre as partes para que se defina como será feita a troca desses dados. Dessa necessidade surgiu o protocolo que é, por definição, um conjunto de regras que estabelecem como a comunicação entre duas partes será feita.

Existem diversos protocolos de comunicação hoje disponíveis, atendendo à uma vasta gama de aplicações e cujas performances variam conforme o sistema no qual ele é aplicado. A escolha de um desses protocolos deve levar em conta uma série de fatores, como confiabilidade, taxa de transmissão, distância máxima permitida para os cabos, ou no caso de comunicação sem fio, o alcance, consumo de energia e tantos outros quanto se necessite para garantir o bom desempenho do sistema.

A seguir, serão abordados os aspectos de funcionamento dos protocolos de comunicação utilizados no desenvolvimento deste projeto.

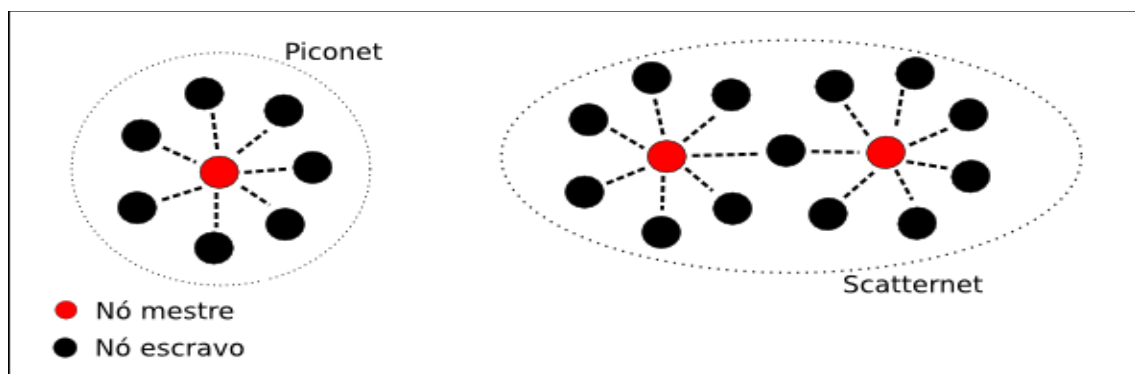
2.1.1 *Bluetooth*

Com sua primeira versão oficial divulgada em julho de 1999 por um consórcio de cinco empresas (L.M. Ericsson, IBM, Intel, Nokia e Toshiba), o padrão de comunicações sem fio *Bluetooth* tem por objetivo permitir a interconexão entre dispositivos de computação, comunicação e acessórios, utilizando rádio sem fio de curto alcance e baixo consumo. Em 2002, o IEEE - “*The Institute of Electrical and*

Electronics Engineers, Inc.” - tomou por base o *Bluetooth* para aprovar o primeiro padrão para redes sem fio de área pessoal (PAN – *Personal Area Network*), o IEEE 802.15.1 (TANENBAUM, 2003).

O funcionamento do sistema *Bluetooth* se dá através de conexão ponto à ponto e ponto à multiponto formada por unidades base compostas por um nó mestre e até sete nós escravos ativos, o que forma uma *piconet*. Duas ou mais *piconets* podem ainda se comunicar através de um nó de ponte, ao que se denomina *scatternet*, como ilustrado na Figura 2.1.

Figura 2.1: Esquema de uma piconet e a formação de uma scatternet.



Fonte: Produção do próprio autor.

O *Bluetooth* opera na banda ISM (*Industrial Scientific Medical*) de 2.4 GHz, tendo sua banda dividida em 79 canais de 1 MHz cada. No intuito de ser um sistema resistente a ambientes ruidosos, sua camada de rádio utiliza modulação chaveada por deslocamento de frequência, onde os *piconets* saltam de um canal para outro, obedecendo à sequência ditada pelo mestre (TANENBAUM, 2003).

A primeira versão oficial desse padrão trabalha com uma taxa de transferência de 1 Mbps, tendo evoluído para versões que, atualmente, ultrapassam 24 Mbps com baixo consumo de energia e alcance de até 100 metros. Sua taxa de transferência associada ao baixo consumo, torna-o extremamente indicado para aparelhos *smartphones*, que no entanto, devido às limitações físicas de tamanho da antena, reduzem o seu alcance a cerca de 10 metros.

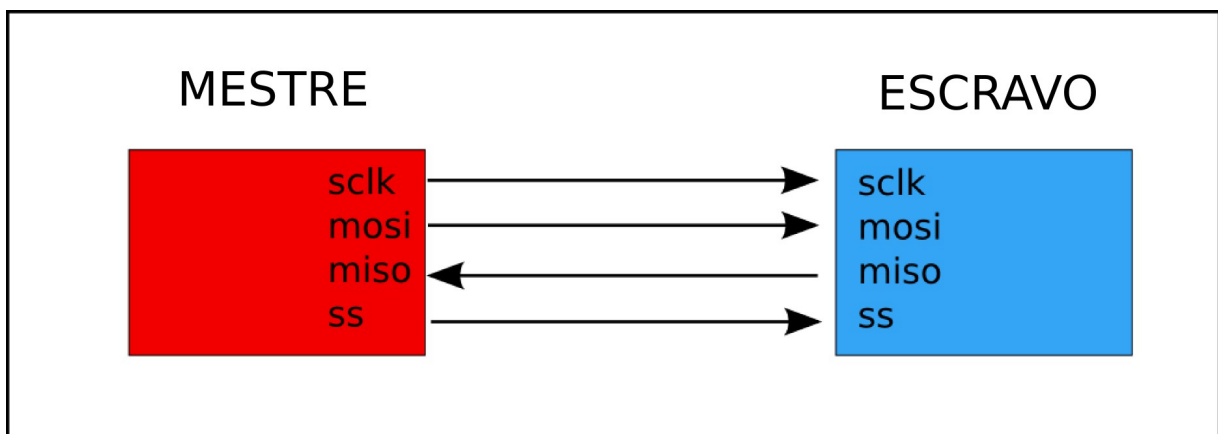
2.1.2 Interface Periférica Serial (*Serial Peripheral Interface - SPI*)

O padrão SPI (*Serial Peripheral Interface*) é um protocolo de comunicação síncrono, serial e que opera em modo *full-duplex*, através de uma estrutura composta por um mestre e ao menos um escravo. Consistem em registradores de deslocamento, um nó mestre e um em cada um dos escravos, cujo o relógio é gerado no mestre e enviado via uma linha de dados aos escravos (NAIMI MAZIDI, 2011).

Esse tipo de comunicação utiliza ao menos quatro sinais lógicos na transferência de dados (Figura 2.2):

- SCLK – relógio serial (*serial clock*);
- MOSI – saída mestre (*master output*);
- MISO – entrada mestre (*master input*);
- SS – seleção de escravo (*slave select*);

Figura 2.2: Esquema de comunicação SPI.



Fonte: Produção do próprio autor.

Para iniciar a comunicação, o mestre configura o relógio para um valor de frequência inferior ou igual ao máximo suportado pelo escravo, normalmente entre 10 kHz e 100 MHz, e então transmite o valor lógico '0' através do sinal 'SS' – ativa em zero – começando então a transmissão de dados. Além disso, para cada escravo

utiliza-se um sinal 'SS' único, ou seja, serão necessários tantos sinais 'SS' quanto a quantidade de escravos configurados.

Esse tipo de comunicação é normalmente utilizada entre microcontroladores e seus periféricos, como por exemplo:

- Sensores: temperatura, pressão, aceleração;
- Memória: *flash* e EEPROM;
- Periféricos de Comunicação: *Ethernet*, USB, IEEE 802.15.4;
- Dispositivos de Imagem: LCD;

As principais vantagens de seu uso estão na comunicação *full-duplex*, na alta taxa de rendimento quando comparado com os protocolos I²C e *SMBus* e ainda a flexibilidade quanto ao tamanho dos pacotes de dados. Em contrapartida, permite o uso de apenas um dispositivo mestre, não possui um sistema de checagem de erro definido e é aplicável apenas em curtas distâncias de cabos, especialmente se comparados com os protocolos RS-232, RS-485 ou CAN.

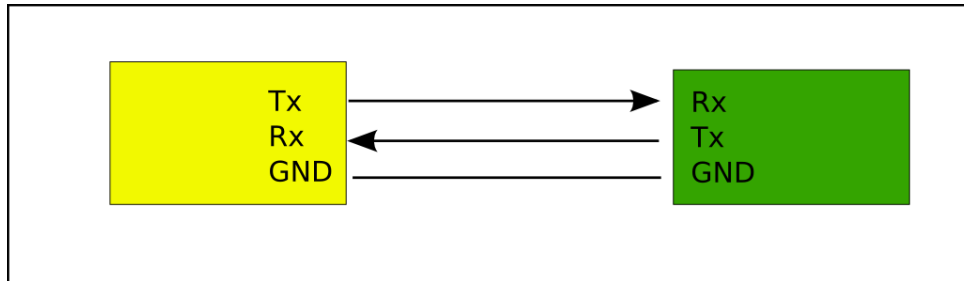
2.1.3 Protocolo de Comunicação RS-232

O RS-232 consiste em um padrão serial de transmissão de dados assíncrono que surgiu na década de 1960 com o objetivo de permitir a interconexão entre Equipamentos de Dados Terminal (DTE – *Data Terminal Equipment*) e Equipamentos de Comunicação de Dados (DCE – *Data Circuit-Terminating Equipment*).

Até a década de 1990, o RS-232 foi utilizado em larga escala por computadores, porém ele vem sendo gradualmente substituído pelo padrão USB (*Universal Serial Bus*) devido, sobretudo, a simplicidade de seu conector e a velocidade de transmissão.

O padrão RS-232 (Figura 2.3), define as características elétricas, como por exemplo os níveis de tensão e taxa de sinalização, além das características mecânicas dos conectores e identificação dos seus pinos.

Figura 2.3: Padrão RS-232 com três ligações.



Fonte: Produção do próprio autor.

A codificação mais comumente usada consiste no envio de um bit de início, seguido por oito bits de dados úteis, até dois bits de paridade e, por fim, até dois bits de parada. Além disso, sua taxa de transmissão deve ser definida previamente, variando de 300 à 115.200 bits/s. Devido a suas características construtivas e a taxa de transmissão de dados, trata-se de um padrão para conexões de até quinze metros.

2.1.4 Protocolo de Controle de Transmissão (*Transmission Control Protocol - TCP*)

Em redes de computadores a transmissão de dados é descrita através de uma pilha de protocolos, que consiste num modelo de camadas, em que cada camada é responsável por um certo grupo de tarefas.

Criado em 1970, o modelo OSI, cujo nome se deve à Organização Internacional para Padronização (*International Organization for Standardization - ISO*), foi o primeiro padrão para protocolos de comunicação, servindo de base para o desenvolvimento e normatização do modelo TCP/IP (TANENBAUM, 2003).

O modelo TCP/IP, se organiza em cinco camadas:

1. **Acesso:** Trata das tecnologias de transmissão de dados usadas, tanto em termos de *hardware* (equivalente à Camada Física no modelo OSI), quanto em *software*, controlando o fluxo de dados e aplicando algoritmos de detecção e correção de erros;

2. **Internet:** Estabelece a conexão entre duas ou mais redes, estando portanto relacionado ao roteamento de pacotes. Um dos principais protocolos desta camada é o IP (*Internet Protocol*) que permite a transmissão de dados para camadas mais altas identificando-os por um endereço de IP.
3. **Transporte:** Como o nome sugere, esta camada cuida da transferência dos dados, transportando-o de sua origem ao seu destino de maneira eficiente, provendo mecanismos que garantam a sequência correta e a ausência de erros na transmissão. Dois de seus principais protocolos são o UDP (*User Datagram Protocol*), que consiste no transporte de pacotes de dados sem que se aguarde a confirmação de chegada dos mesmos, e o TCP (*Transmission Control Protocol*) que inclui o controle de fluxo e erro.
4. **Aplicação:** Uma aplicação possui, normalmente, um formato de dados característico, ou seja, específico para ela. Assim, a camada de aplicação é responsável por formatar os dados fornecidos pela camada de transporte, tornando-os compreensíveis para a aplicação a que se destina. Os principais protocolos que o integram são o HTTP (*Hypertext Transfer Protocol*), o FTP (*File Transfer Protocol*) e o SMTP (*Simple Mail Transfer Protocol*).

Cada camada é responsável por prestar serviços à sua camada imediatamente superior, abstraindo, dessa maneira, os serviços das camadas inferiores. O protocolo TCP, utilizado na solução proposta no capítulo a seguir, garante o transporte de dados com controle de fluxo e erro, sendo toda a parte de transmissão realizada pelas camadas inferiores.

2.2 Microcontroladores

Um microcontrolador é um conjunto composto essencialmente por um microprocessador, memória e um conjunto de periféricos de entrada e saída, integrados em um único circuito, sendo o espaço final ocupado e o consumo de energia, dois dos principais parâmetros que os tornam ideais para a utilização na indústria automobilística, aeroespacial e todo o tipo de dispositivos e sistemas de controle automático.

2.2.1 Kits de Desenvolvimento

Um dos principais fatores que dificultam o estudo e desenvolvimento de sistemas microcontrolados é a complexidade de configuração e montagem da placa de circuito a partir do circuito integrado do microcontrolador, o que requer um conhecimento técnico aprofundado do mesmo.

Por esse motivo os fabricantes comercializam *kits* de desenvolvimento, que consistem em uma placa de circuito composta pelo microcontrolador em questão, adjunto de toda a parte analógica necessária ao seu funcionamento, como por exemplo o circuito responsável pela criação do relógio que o comandará, além de um conjunto de periféricos acoplados às saídas do sistema, permitindo o imediato teste da lógica implementada pelo desenvolvedor.

Apesar de facilitar a tarefa do desenvolvedor, esses kits ainda requerem um bom nível de conhecimento técnico em termos de arquitetura do sistema, sendo esse um obstáculo que, quando rompido, permitiria por certo, a acentuada difusão dessa tecnologia. Pensando nisto, um grupo italiano, iniciou na cidade de Ivrea, em 2005, um projeto que implementava uma placa de desenvolvimento microcontrolada, cuja memória continha um programa que realizava toda a configuração necessária (*bootloader*) e a partir de uma IHM simplificada, o desenvolvedor poderia carregar diretamente a lógica de funcionamento do sistema descrito em linguagem C e C++, abstraindo portanto, as questões técnicas relativas à configuração.

A esse projeto, deu-se o nome de Arduino, que tem se espalhado rapidamente pelo mundo, sobretudo devido à sua proposta de sistema de código e dispositivo aberto – *Open-Source* e *Open-Hardware* – portanto, ser reproduzido por qualquer um que o desejar, sob certas regras que regem esse tipo de projeto.

2.2.2 Arduino Mega

O Arduino Mega é uma placa microcontrolada, baseada no controlador ATmega2560 da Atmel. Composta de 54 pinos de entradas e saídas digitais – dos quais 15 podem ser usados como uma saída PWM – 16 entradas analógicas, 4 UARTs, um cristal oscilador de 16MHz, uma porta de conexão USB, uma entrada de

alimentação, pinos para programação direta do tipo ICSP (*In Circuito Serial Programming*) e um botão de *reset* (MEGA, 2013).

O Arduino Mega2560 (Figura 2.4) se destaca sobretudo pela quantidade de portas gerais de entrada e saída (GPIOs – *General Purpose Input/Output*) e pela memória do tipo Flash que é de 256 kB, sendo portanto capaz de executar uma lógica com uma quantidade razoavelmente grande de instruções, uma vez que a mesma é armazenada nesta memória.

Figura 2.4: Visão superior da placa Arduino Mega2560.



Fonte: MEGA, 2013.

2.2.3 Arduino Nano

O Arduino Nano se encontra disponível em duas versões, a primeira utiliza o microcontrolador Atmega328 e a segunda o ATmega168 com 32 kB e 16 kB de memória flash disponível, respectivamente. Dentre suas principais características, é importante citar que ambas as versões disponibilizam um total 14 pinos de entradas e saídas digitais – dos quais 6 podem ser usados como saída PWM – 8 entradas analógicas, um cristal oscilador de 16MHz. Além disso, suporta comunicação serial, e SPI (NANO, 2013).

Esta placa (Figura 2.5) tem 0,70 de largura por 1,70 de comprimento em polegadas, o que a torna um dispositivo muito interessante sobretudo para aplicações que requerem a utilização de pouco espaço. Em contrapartida, sua memória reduzida faz necessário o uso de uma lógica razoavelmente simples.

Figura 2.5: Visão superior e inferior da placa Arduino Nano.



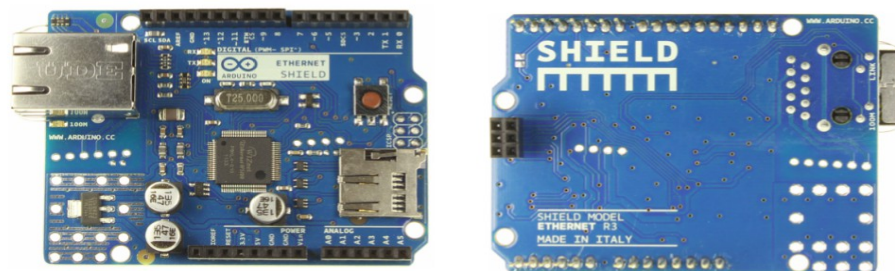
Fonte: NANO, 2013.

2.2.4 Arduino Ethernet

Um ponto forte do projeto Arduino está no fato de ser modular, ou seja, pode-se acrescentar periféricos com as mais diversas funcionalidades a partir da junção de uma das extensões disponíveis à placa base que contém um microcontrolador. À essas extensões deu-se o nome de “*Shield*”.

O *shield* de *Ethernet* (Figura 2.6) permite conectar-se à internet através de um cabo de rede, utilizando por base o chip Wiznet W5100. Ele é capaz de estabelecer conexão IP e suporta os protocolos de transporte TCP e UDP, além disso este *shield* se comunica com a placa Arduino através de uma conexão do tipo SPI (ETHERNET, 2013).

Figura 2.6: Visão superior e inferior da placa Arduino *Ethernet*.



Fonte: ETHERNET, 2013.

Por fim, a biblioteca base utilizada para controle deste *shield* é disponibilizada de maneira gratuita e em código aberto, permitindo ao desenvolvedor a plena compreensão de seu funcionamento bem como a adaptação do projeto, caso se faça necessário.

2.3 A Linguagem de Programação Java

Um microprocessador executa instruções descritas em linguagem máquina que é a composição de zeros e uns (código binário). No entanto, além de complexo, seria extremamente tedioso e repetitivo utilizar este tipo de linguagem para a descrição de um programa. Por este motivo, foram desenvolvidas linguagens que permitem maior performance por parte do desenvolvedor por serem de mais fácil compreensão, e que, antes de serem executadas, são traduzidas em linguagem máquina por programas tradutores, os chamados *assemblers* (DEITEL, 2010).

Desenvolvida pela Sun Microsystems em 1995, Java é uma linguagem de programação classificada como de alto nível devido ao fato de uma única instrução poder realizar um conjunto de tarefas, o que acelera enormemente o processo de programação.

O desenvolvimento de um programa em Java passa por cinco etapas:

1. Edição – O programa é escrito em um arquivo com a extensão '.java'.
2. Compilação – O programa é compilado, ou seja, convertido em linguagem *bytecode*, que é uma forma intermediária entre a linguagem Java e a linguagem de máquina.
3. Carregamento – O carregador armazena o programa na memória da máquina.
4. Verificação – O código é verificado no intuito de se assegurar que o mesmo não viola restrições de segurança da linguagem.
5. Execução – A máquina virtual Java (JVM – *Java Virtual Machine*) lê e traduz o código *bytecode* para linguagem de máquina, permitindo a sua execução.

2.3.1 O Sistema Operacional Android

Financiado pela Google, o Android é um sistema operacional direcionado especialmente a aparelhos *smartphones* e *tablets*. Ele foi utilizado oficialmente pela primeira vez em 2008 pela fabricante chinesa HTC – *High Tech Computer Corporation* – e, a partir daí, se difundiu e evoluiu muito rapidamente, sendo hoje o principal nome em termos de sistema operacional, no mercado de dispositivos

móveis de telefonia e tecnologia.

Um dos pontos fortes deste sistema está no fato de o mesmo ser de código aberto – *Open-Source* – podendo ser, desta forma, explorado por qualquer um que assim o queira e permitindo aos fabricantes de dispositivos que desejem utilizá-lo, fazê-lo sem que seja necessário o pagamento da licença, reduzindo o custo e, conseqüentemente, o valor de venda desses produtos.

O desenvolvimento de aplicativos para o sistema Android pode ser feito a partir da linguagem de programação C++ ou Java, sendo esta a mais utilizada e até mesmo incentivada pela Google, que fornece diversas ferramentas, artigos, tutoriais e documentação necessárias ao programador. Destaca-se o conjunto de APIs (*Application Programming Interfaces*) que consiste em bibliotecas de classes prontas para utilização, facilitando e agilizando enormemente o desenvolvimento de aplicativos, uma vez que permite a abstração de muitos detalhes técnicos de utilização das funções disponibilizadas pelo sistema, como por exemplo as funções de mídia (música, vídeo e outros), de transmissão de dados via *Bluetooth* entre outros.

2.4 Transmissão de Dados Sem Fio

Na era da comunicação, a substituição de sistemas de transmissão de dados cabeados por novas tecnologias de transmissão sem fio, tem sido uma alternativa essencial na integração entre os sistemas. Um exemplo cotidiano desta integração, está na possibilidade de, utilizando um dispositivo móvel, solicitar a impressão de um arquivo em uma impressora remota através da tecnologia *Bluetooth*.

Dois dos principais parâmetros que são avaliados em um projeto que vise utilizar transmissão de dados sem fio são: a taxa de transmissão, que deve estar de acordo com a necessidade do projeto e a potência consumida, que deverá ser sempre minimizada. Isso porque, o objetivo de uma transmissão sem fio é exatamente de não haver necessidade de cabos para comunicação (transporte de dados) e, se possível, não utilizar alimentação elétrica cabeada, sendo este um fator um pouco mais complexo e que muitas vezes abre-se mão.

Em projetos de automação sem fio, os sensores necessitam normalmente de baixa taxa de transmissão de dados, já que sua comunicação com o microcontrolador ocorre em intervalos longos de tempo, ao menos em relação à frequência de operação do mesmo. Desta maneira, sistemas de transmissão como o *Bluetooth* e *Wi-Fi* não são os mais indicados, pois por trabalharem com taxas de 3 Mbps e 100 Mbps respectivamente, sua aplicação acarretaria na subutilização da tecnologia.

Dentre as principais alternativas disponíveis no mercado, destacam-se os padrões: 802.15.4 normatizado pela IEEE. O padrão ZigBee que junto com o 802.15.4 formam uma das tecnologias de radiotransmissão mais difundidas atualmente. Além do dispositivo NRF24L01+ da fabricante Nordic que foi o escolhido para este projeto e terá seus aspectos técnicos abordados no tópico a seguir.

2.4.1 NRF24L01+

O NRF24L01+ do fabricante Nordic Semiconductor[®], representado na Figura 2.7, é um dispositivo de transmissão em 2.4GHz em baixa potência com taxas de transmissão configuráveis em 250kbps, 1Mbps e 2Mbps (NORDIC, 2013). A seguir algumas características fornecidas pela documentação técnica do fabricante:

- Consumo:
 - Modo transmissão: 11.3 mA a 0dBm de potência de saída;
 - Modo recepção: 13.5 mA a 2Mbps;
 - Modo espera: 26uA.
- Tensão de entrada de 1.9 a 3.6V;
- Aplicações:
 - Periféricos sem fio para computadores;
 - Sensores de Baixa Potência;
 - Automação Residencial;

- Controles de Jogos;
- Brinquedos;

A modulação utilizada é a *Gaussian Frequency-Shift Keying* (GFSK). Ela utiliza duas frequências distintas para envio e recebimento de pacotes, além de melhorar o espectro do sinal devido ao uso de um filtro de Gaus.

A principal vantagem deste circuito está na sua interface de programação (API) que permite ao desenvolvedor configurar de maneira simples, através de um conjunto de comandos via comunicação SPI. Além disso, encontra-se disponível em diversos fóruns de desenvolvimento, códigos que permitem pilotar o dispositivo e que implementam inclusive algoritmos de roteamento de redes.

Figura 2.7: NRF24L01+ da fabricante Nordic Semiconductor.



Fonte: NORDIC, 2013.

Em contrapartida, têm-se, dentre outros problemas, o fato de não haver uma camada que forneça serviço de segurança na troca de mensagens e de não seguir um padrão reconhecido pela IEEE, o que, em um produto final, implica muitas vezes em sua credibilidade e aceitação no mercado.

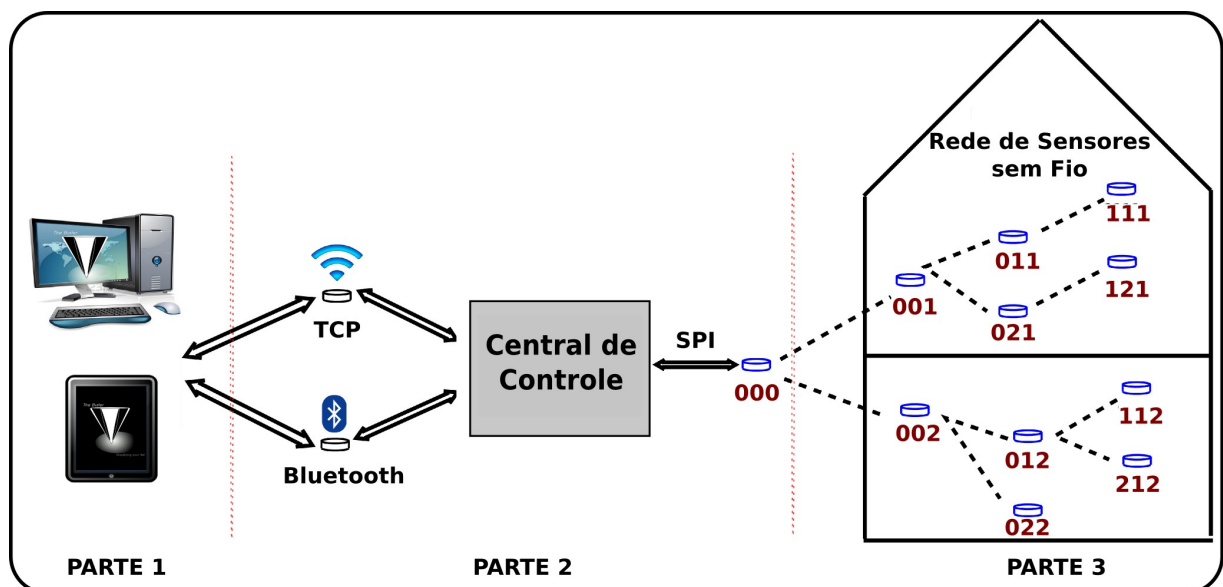
3 MODELO DE SOLUÇÃO PROPOSTA

Com a base teórica abordada no capítulo anterior, é possível propor uma solução que ao integrar as tecnologias citadas permita a criação de um ambiente residencial capaz de interagir com o seu usuário por meio de uma IHM intuitiva.

O modelo proposto, ilustrado na Figura 3.1, organiza a solução em três macro etapas, as quais serão abordadas em detalhe no decorrer deste capítulo:

- Parte 1: Desenvolvimento da IHM;
- Parte 2: Lógica implementada no microcontrolador responsável por interpretar os comandos do usuário e os protocolos de comunicação utilizados;
- Parte 3: Implementação e configuração de uma RSSF.

Figura 3.1: Visão topológica do projeto.



Fonte: Produção do próprio autor.

3.1 Interface Homem Máquina - IHM

Em termos gerais, uma IHM pode ser vista como uma ponte de ligação entre um sistema e seu utilizador, uma vez que é através dela que são trocadas todas as

informações. Além disso, é ela quem possibilita abstrair as características técnicas de funcionamento, facilitando enormemente a operação do sistema.

O projeto em questão visa um produto potencialmente comercializável, sendo portanto essencial que a IHM seja tão intuitiva quanto possível e ao mesmo tempo possua uma estrutura interna que permita a armazenagem e utilização de dados de funcionamento do sistema.

Os próximos tópicos deste item tratarão dos aspectos funcionais da IHM proposta.

3.1.1 Comunicação via *Bluetooth* e *Wi-Fi*

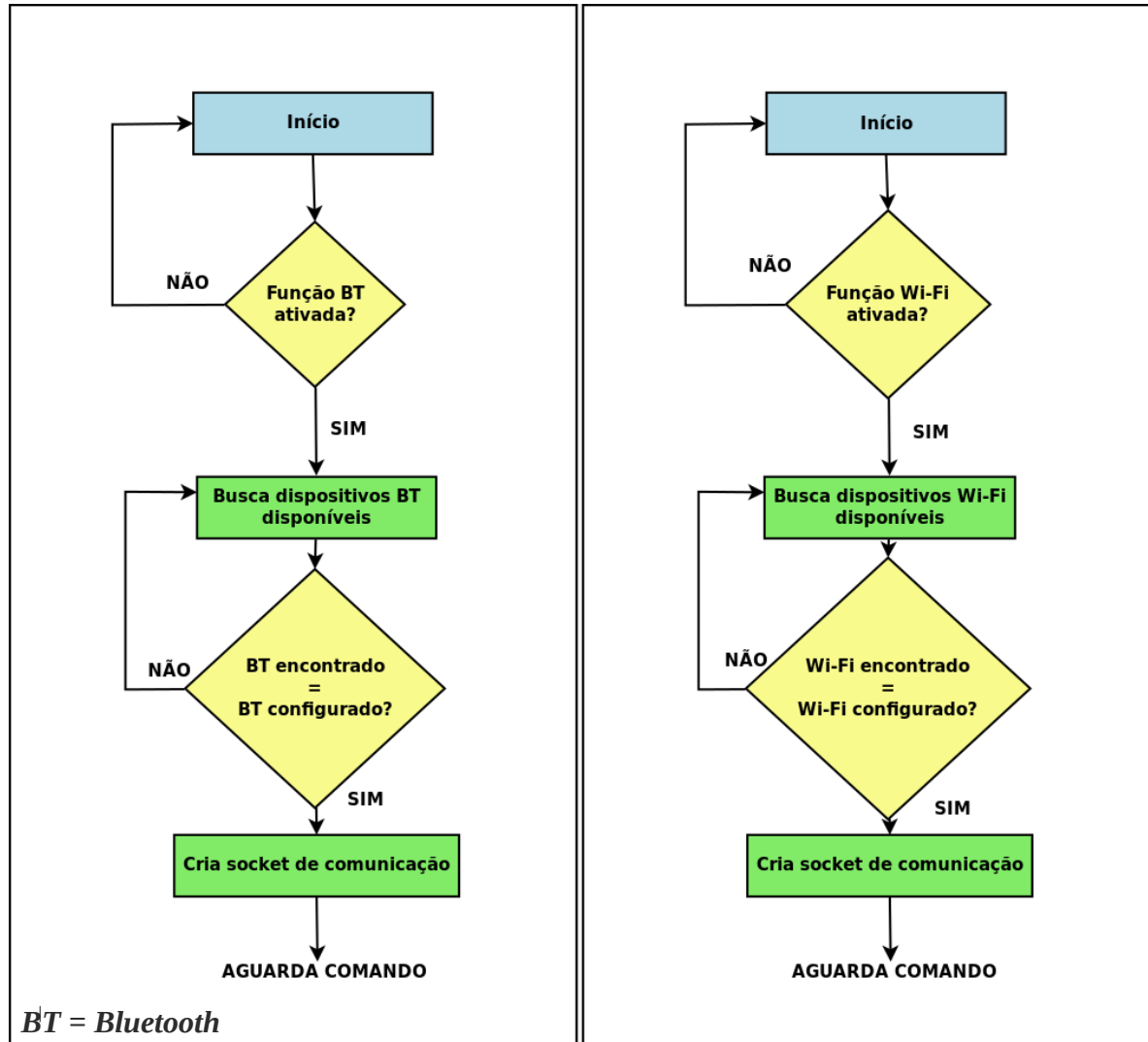
A troca de informações entre a IHM e a CC será feita a partir de duas tecnologias de comunicação sem fio:

- ***Bluetooth***: Por permitir conexões de curto alcance, num raio de 10 metros, e por se tratar de uma conexão ponto a ponto garante uma forma de acesso local e segura ao sistema.
- ***Wi-Fi***: Possibilita a utilização da mesma infraestrutura de rede utilizada pelos dados de internet, permitindo acesso local ou remoto.

Para ambos os casos, o algoritmo base de criação de um canal de comunicação proposto consiste em: uma vez que se faça necessária a troca de dados com a CC, é verificado se a funcionalidade da tecnologia que deseja-se utilizar está ativa. Em seguida, é realizada uma busca dos dispositivos disponíveis naquele ambiente comparando-os com aqueles previamente configurados. Por fim, é estabelecido um canal de comunicação. Este algoritmo pode ser visualizado na Figura 3.2.

Com o canal de comunicação pronto para a troca de mensagens, faz-se necessária uma forma de gerenciamento das demandas de mensagens, ou seja, quando há uma série de comandos sequenciais a serem enviados, é preciso garantir que todos serão transmitidos.

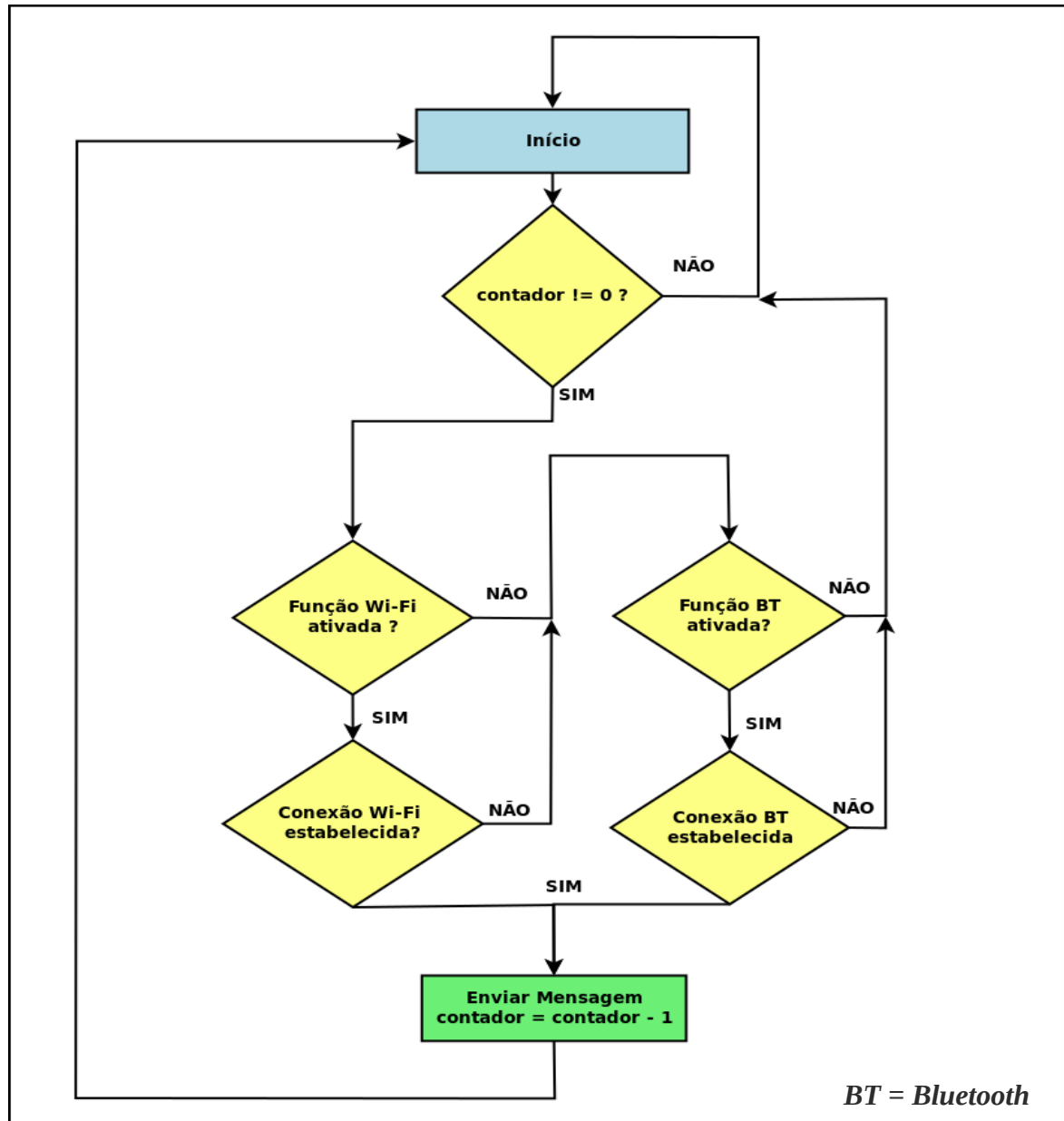
Figura 3.2: Algoritmo de criação de canal de comunicação *Bluetooth* e *Wi-Fi*.



Fonte: Produção do próprio autor.

O algoritmo proposto para gerenciar a situação supracitada consiste em um esquema de armazenamento e encaminhamento das mensagens que funciona da seguinte forma: utilizando um contador que indica a quantidade de mensagens que ainda não foram enviadas, é verificado se há mensagens pendentes. Em caso positivo é averiguada a possibilidade de envio através da rede *Wi-Fi*. Caso a mesma esteja indisponível, utiliza-se a conexão *Bluetooth*, conforme ilustrado na Figura 3.3.

Figura 3.3: Algoritmo de gerenciamento e envio de mensagens.



Fonte: Produção do próprio autor.

3.1.2 Configuração

É desejável que o usuário final tenha o mínimo de trabalho em termos de configuração, pois como dito, para que a plataforma tenha maior potencial de comercialização é preciso que ela seja de fácil operação, abrangendo um maior público-alvo. Desta forma, as únicas configurações realmente necessárias serão

relativas à porta de saída de dados:

- *Bluetooth*: deverá ser informado o endereço MAC (*Media Access Control*) do transceptor da CC;
- *Wi-Fi*: será necessário informar o endereço MAC do dispositivo, endereço IP e a porta utilizada pelo módulo *Ethernet* da CC.

Além disso, propõe-se que seja possível criar contas de usuários para que se possa gerar um banco de dados com informações do perfil de cada usuário, tendo-se então uma estrutura que permita a implementação de um certo nível de inteligência que seja capaz de, por exemplo, com base nesses dados prever os comandos do usuário, se antecipando aos mesmos.

3.1.3 Comandos do Sistema

Com a base operacional pronta, resta definir de que maneira serão mostrados os comandos disponíveis e como serão codificadas as mensagens. Nelas deverão estar contidas todas as informações referentes ao tipo de ação que será executada e os dados que permitam identificar o sensor para o qual a mesma deverá ser encaminhada.

Como esta solução toma por base a utilização de aparelhos *smartphones*, os quais dispõem de uma vasta gama de recursos gráficos, propõe-se que a tela de comandos seja representada a partir da imagem da planta baixa da residência. Bastará ao usuário escolher qual o cômodo com o qual deseja interagir e serão mostradas as ações disponíveis, dando um caráter intuitivo à utilização do mesmo. Os demais comandos serão disponibilizados a partir de telas adicionais que contextualizem a ação, como por exemplo o controle de som que deverá mostrar um tocador que informe as músicas disponíveis. Portanto, os parâmetros básicos que poderão ser controlados são:

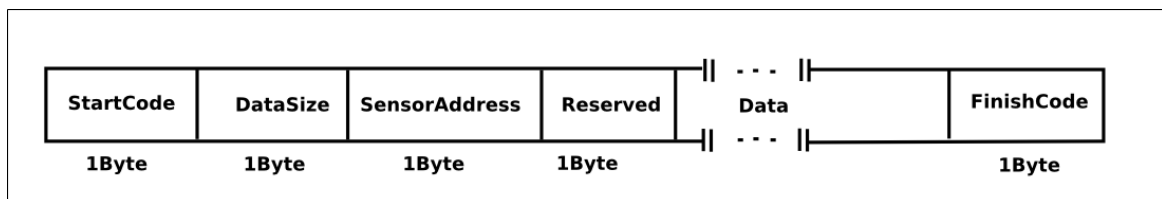
- Intensidade luminosa;
- Temperatura;
- Tocador de músicas;

- Portões de Garagem;

A proposta de codificação da mensagem (Figura 3.4) consiste em pacotes de dados de tamanho total variável, composto por:

- **StartCode** – Código de um *byte* indicando o início de um pacote. Seu valor será fixado em '0x77'.
- **DataSize** – Indica o tamanho do campo Data, permitindo calcular o tamanho total do pacote.
- **SensorAddress** – Composto de dois *bytes*, indica o endereço atribuído ao sensor com o qual deseja-se comunicar.
- **Reserved** – Reservado para futuras modificações.
- **Command** – Código do comando solicitado pelo usuário.
- **Data** – Contém as informações referentes ao comando que deverá ser executado.
- **FinishCode** – Código de um *byte* indicando o fim do pacote. Seu valor será fixado em '0x77' (base hexadecimal).

Figura 3.4: Codificação de mensagem trocada entre a IHM e a CC.



Fonte: Produção do próprio autor.

Destaca-se o fato de que o pacote será transmitido através do protocolo *IP* ou *Bluetooth*, tratando-se portanto da carga útil transmitida pelo mesmo, o que permite abstrair as questões relativas aos códigos de correção de erros já embutidos nestes protocolos.

Assim, é possível verificar se um pacote de dados recebido pela CC está

completo, para, em seguida, decodificá-lo encaminhando o comando solicitado para o destino final.

3.2 Central de Controle - CC

Os comandos enviados através da IHM devem ser processados e encaminhados ao seu destino final. Para tanto, se faz necessário algum dispositivo cujo algoritmo seja capaz de realizar tal ação, que neste projeto será chamado de Central de Controle.

Este algoritmo deverá, entre outros aspectos, ser capaz de:

- verificar a consistência dos dados recebidos;
- avaliar o tipo de ação codificada;
- montar um novo pacote e encaminhá-lo ao correto sensor;

Esta seção abordará o desenvolvimento de uma lógica que atenda as características supracitadas.

3.2.1 Consistência de Dados

Conforme abordado na seção 3.1.1, os dados são transmitidos a partir da IHM utilizando os protocolos *Bluetooth* e TCP, os quais possuem algoritmos de detecção e correção de erros que garantem a integridade dos dados. Assim, o protocolo proposto deverá verificar apenas se um pacote recebido está completo, ou seja, se o tamanho informado está condizente com o real.

Para esse procedimento, é suficiente armazenar os dados recebidos em um vetor e, então, verificar primeiramente o tamanho do campo '*Data*', o que permite determinar o tamanho total do pacote. Em seguida analisar o valor do campo '*StartCode*' e '*FinishCode*' que deverão estar de acordo com o descrito na seção 3.1.3.

3.2.2 Decodificação

Para a aplicação, os pacotes utilizados nesta fase do projeto terão seu tamanho fixado em 10 *bytes*, sendo 4 deles destinado ao campo '*Data*' e 1 ao comando à ser executado (ver Figura 3.4). Este deverá conter as informações mínimas referentes ao tipo de ação e aquele conterà um valor associado ao comando, indicando a intensidade do mesmo.

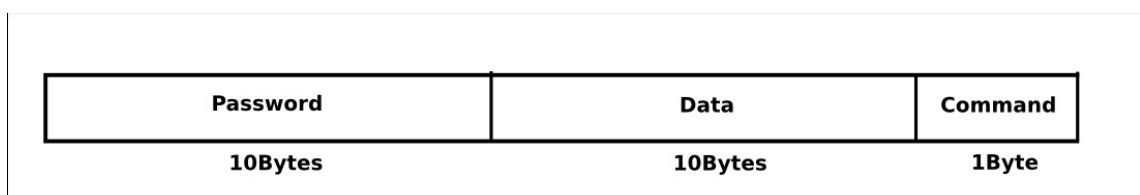
Os comandos deverão estar cadastrados na memória do microcontrolador, de tal forma que durante a decodificação eles possam ser reconhecidos como uma ação válida. Restando por fim definir de qual maneira a informação decodificada será encaminhada ao destino final, cujo endereço encontra-se no campo '*SensorAddress*'.

3.2.3 Encaminhamento

Os pacotes enviados através da RSSF serão compostos por uma estrutura como a ilustrada na Figura 3.5, cujos campos são:

- ***Password*** – senha de 10 *bytes* utilizado para autenticação dos dados recebidos;
- ***Data*** – contém a informação de magnitude da ação solicitada, como por exemplo para o controle de luz, onde se faz necessário um valor que indique qual deverá ser a sua intensidade. Seu tamanho é de 10 *bytes* pois a senha de referência poderá ser alterada através de um comando apropriado e que utiliza essa mesma estrutura de mensagem.
- ***Command*** - deverá conter o código referente ao tipo de ação solicitada.

Figura 3.5: Codificação de mensagem enviada à RSSF.



Fonte: Produção do próprio autor.

Supondo-se uma situação na qual se tenham dois sistemas distintos em operação, é possível que sensores de um sistema receba pacotes provenientes do outro, visto que seus endereços locais podem coincidir. Desta forma, o campo '*Password*' permite que se tenha um código identificador que ao ser verificado, indicará se a mensagem recebida foi encaminhada corretamente. Este código deverá ser, naturalmente, passível de alteração pelo usuário.

3.3 Rede de Sensores Sem Fio - RSSF

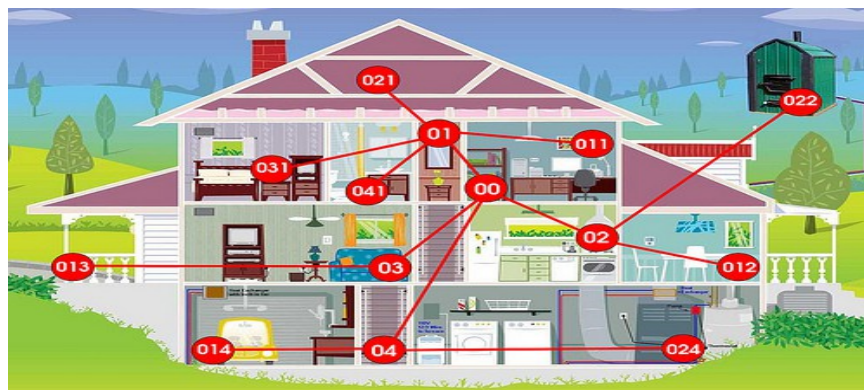
Após o processamento dos comandos provenientes da IHM e encaminhamento dos mesmos à RSSF, se faz necessário um algoritmo que permita o encaminhamento dos pacotes ao destino final, bem como a execução do comando e sua respectiva confirmação. Destaca-se o fato de que o termo 'sensores' representa aqui o conjunto formado por dispositivos de sensoriamento e de atuação (atuadores).

Nesta seção serão abordados os aspectos relacionados ao roteamento das mensagens, configuração dos sensores e execução dos comandos.

3.3.1 Algoritmo de Roteamento

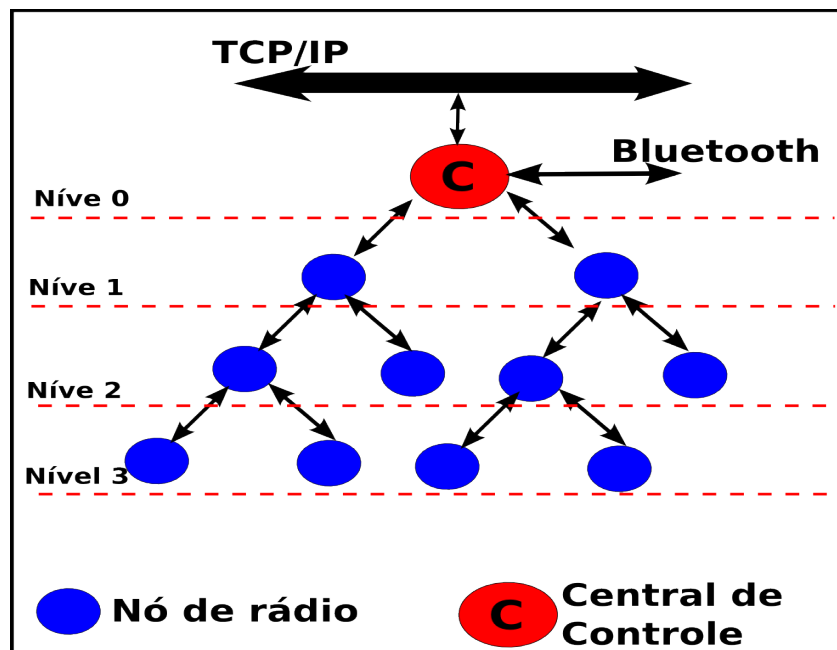
Ao receber uma mensagem, a RSSF deve verificar o seu endereço de destino e encaminhá-la ao mesmo. O algoritmo proposto neste projeto consiste numa rede de topologia do tipo árvore disponibilizada sob forma de projeto de código aberto cujo funcionamento é ilustrado na Figura 3.6.

Figura 3.6: Modelo de roteamento da RSSF.



Cada nó que compõe a rede é configurado em topologia do tipo árvore organizado em diferentes níveis (Figura 3.7), conforme o seu endereço. Neste modelo cada nó pode se comunicar diretamente apenas com o seu 'nó pai' e os seus 'nós filho'. O número máximo de nós em cada nível é: 1 no nível 0, 5 no nível 1, 25 no segundo nível e 125 no terceiro e último nível. A rede se encarregará de direcionar as mensagens enviadas para o correto endereço (MANIACBUG, 2012).

Figura 3.7: Topologia árvore adotada e sua divisão em níveis.



Fonte: Produção do próprio autor.

3.3.2 Configuração dos Sensores

Cada sensor deverá ter armazenado na memória de seu microcontrolador as informações necessárias à decodificação e encaminhamento de mensagens na rede e execução de comandos.

Tomando por base os campos que compõem a mensagem que circula na RSSF (ver seção 3.2.3), os parâmetros de configuração mínimos necessários são:

- Endereço – consiste no endereço atribuído ao sensor. Ao ser inicializado, o mesmo deverá solicitar tal endereço à rede.

- Modo de Funcionamento – Permite a configuração do sensor para diferentes modos de funcionamento:
 - Simulação de Presença;
 - Liga/Desliga;
 - Controle de Intensidade;
- Senha – Usado para autenticar uma mensagem, confirmando que a mesma é direcionada ao sensor.

3.3.3 Execução de Comandos

Após receber um pacote e autenticá-lo, o microcontrolador deverá proceder à decodificação da mensagem e execução do comando. A decodificação consistirá, neste caso, em avaliar o campo '*Command*' da mensagem recebida, comparando-o com uma lista de ações cadastradas, no intuito de reconhecê-la como uma ação executável em função do modo de funcionamento configurado previamente no sensor. E com base no valor do primeiro *byte* do campo '*Data*' ajustar, se necessário o valor da saída relacionada à ação.

As ações se resumem a colocar uma ou mais portas de saída em valor lógico '1' ou '0', ou ainda, no caso de uma saída PWM (*Pulse Width Modulation*), ajustar a sua razão cíclica.

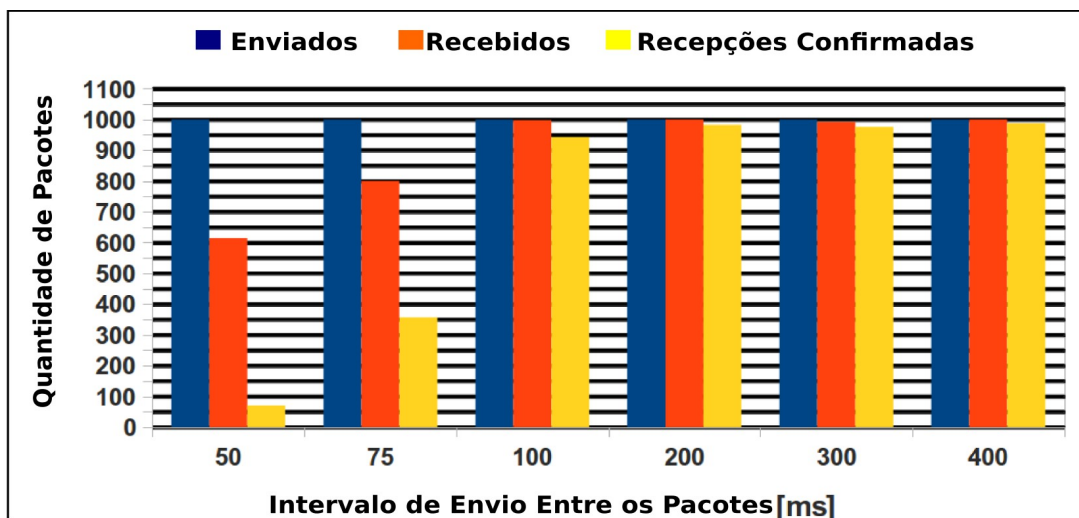
É importante ressaltar que o circuito sobre o qual o sensor deverá atuar será, em muitos casos, analógico, sendo necessária a conversão do sinal digital para analógico.

3.3.4 Performance da RSSF

Para avaliar a performance da RSSF em termos de confiabilidade e tempo de atraso dos pacotes, o seguinte teste foi realizado: um grande número de pacotes de tamanho definido foi enviado em diferentes intervalos de tempo através da RSSF, sendo de aproximadamente 10 metros a distância máxima do maior salto.

Os resultados a seguir consideram grupos de mil pacotes de 6 *Bytes* enviados de um nó nível 0 para um nó nível 3 (Figura 3.8). Para intervalos maiores que 100ms, mais de 95% dos pacotes enviados tiveram confirmação de recebimento. Além disso, o tempo de latência médio verificado foi 87,53 ms.

Figura 3.8: Performance de mensagens trocadas entre nós de nível 0 e 3.

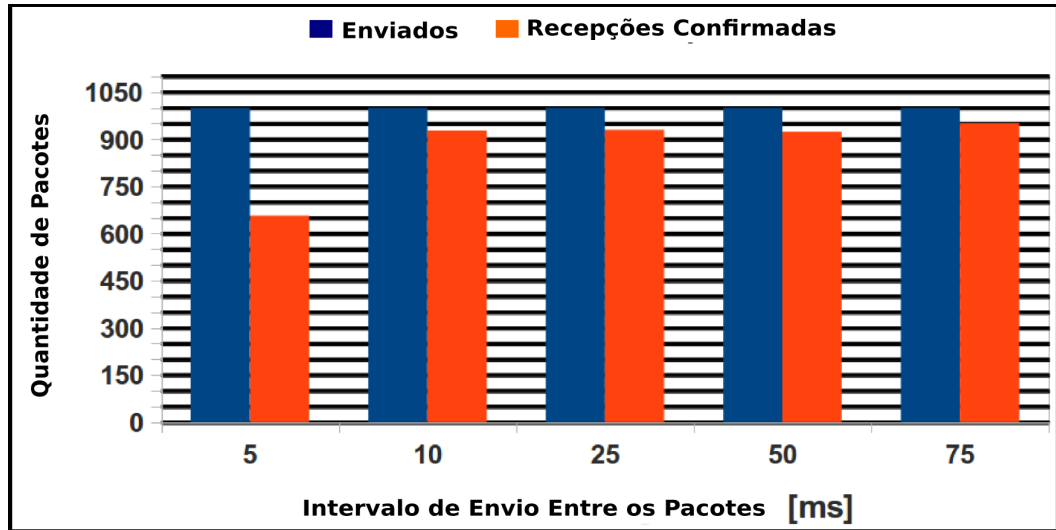


Fonte: Produção do próprio autor.

Em um segundo teste, foram enviados grupos de mil pacotes de um nó nível 0 para um nó nível 1 (Figura 3.9), que é o caso mais simples de roteamento na RSSF. Para intervalos superiores a 10 ms, mais de 90% dos pacotes enviados tiveram a confirmação de recebimento. O tempo de latência médio medido foi de 2,58 ms.

Em sistemas de automação com rede de sensores e atuadores, as informações trocadas entre os nós da rede e a IHM ocorrem, via de regra, em intervalos grandes de tempo, da ordem de segundos ou até minutos em casos de monitoramento. Assim, os resultados acima demonstram que a topologia da rede e o *hardware* utilizado atendem aos requisitos necessários à aplicação proposta neste projeto.

Figura 3.9: Performance de mensagens trocadas entre nós de nível 0 e 1.

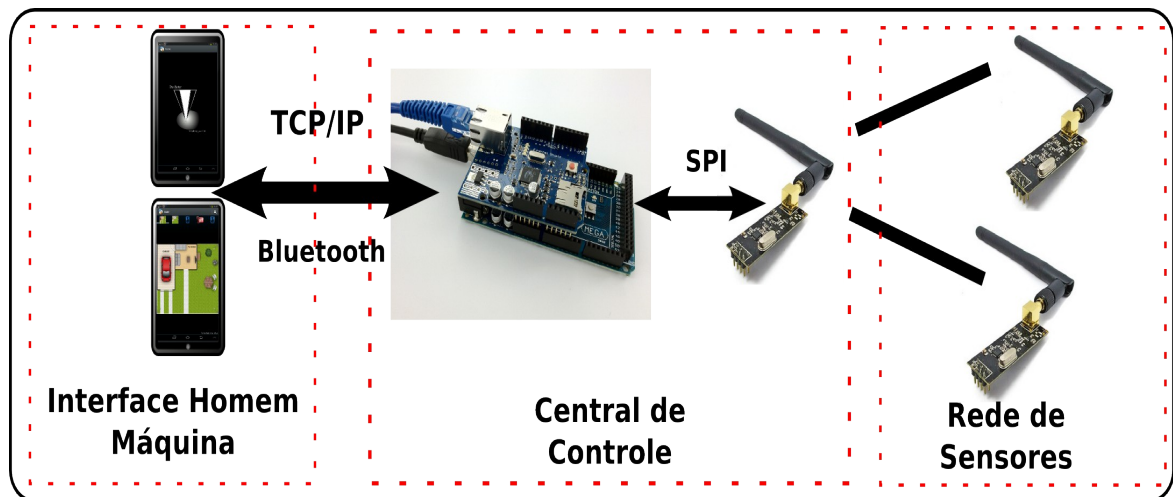


Fonte: Produção do próprio autor.

4 PROJETO DESENVOLVIDO

Proposto o modelo de solução, é preciso aplicar as tecnologias selecionadas na implementação do mesmo. A Figura 4.1 mostra a visão topológica do sistema, destacando as partes que o organizam e os respectivos dispositivos que as compõem.

Figura 4.1: Sistema desenvolvido.



Fonte: Produção do próprio autor.

4.1 Interface Homem Máquina - IHM

Para o desenvolvimento da IHM, optou-se pela utilização da tecnologia Java aplicada ao sistema Android, pois, avaliando-se os principais sistemas operacionais utilizados pelos aparelhos *smartphones*, verificou-se que em novembro de 2013, 81% desse mercado era ocupado por tal sistema (CNET, 2013).

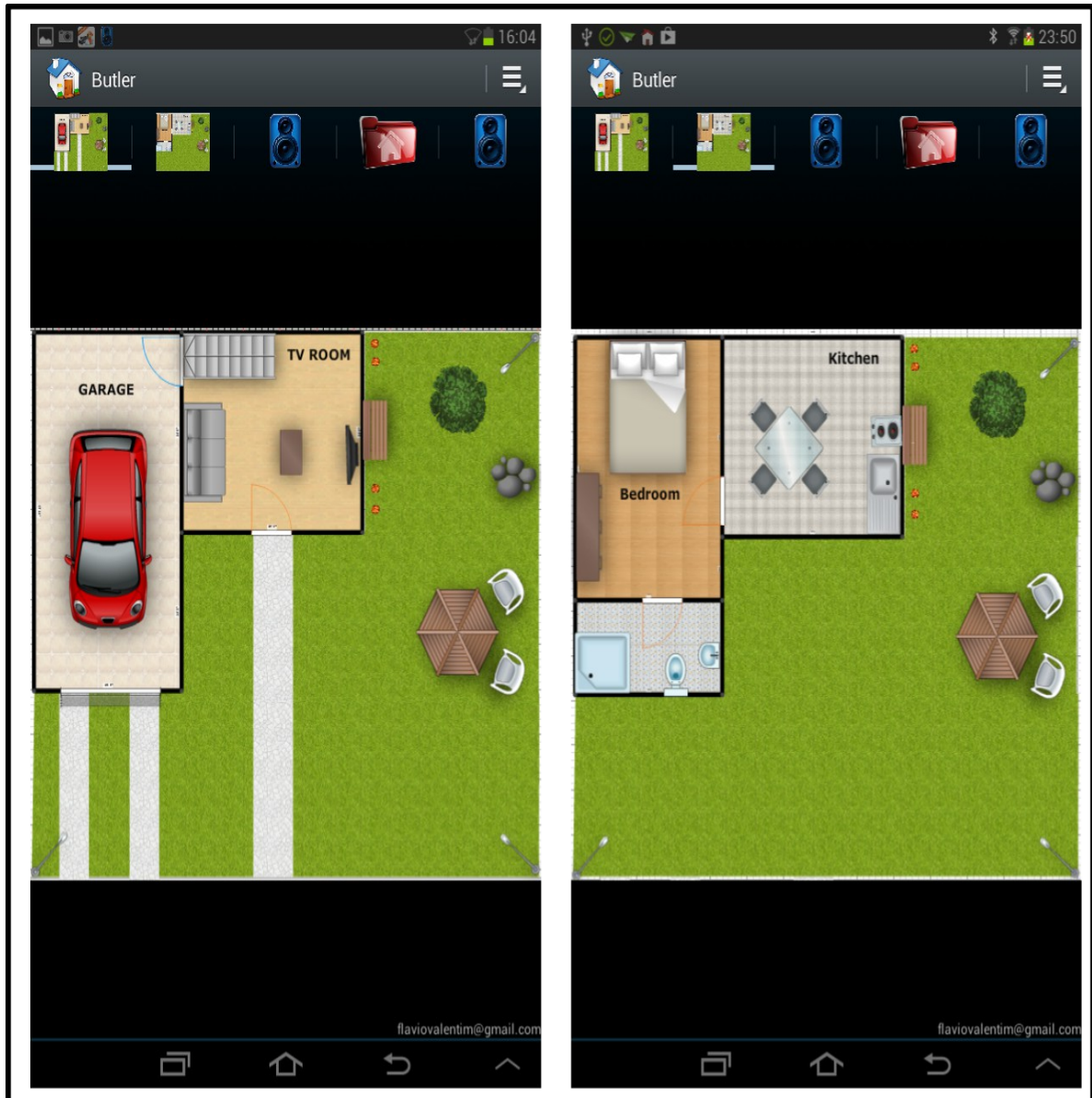
4.1.1 Interação com o Usuário

A plataforma Android fornece um amplo conjunto de ferramentas que tornam possível a criação de ambientes virtuais com os quais o usuário pode interagir sobretudo a partir de um simples toque na tela de seu *smartphone*.

Partindo deste princípio e no intuito de apresentar um sistema cuja utilização seja

intuitiva, decidiu-se utilizar a ilustração da planta baixa de uma residência, de forma que cada parte da casa possa ser controlada a partir de um toque em sua respectiva área na ilustração (Figura 4.2).

Figura 4.2: Planta baixa exibida ao usuário na IHM em plataforma Android.

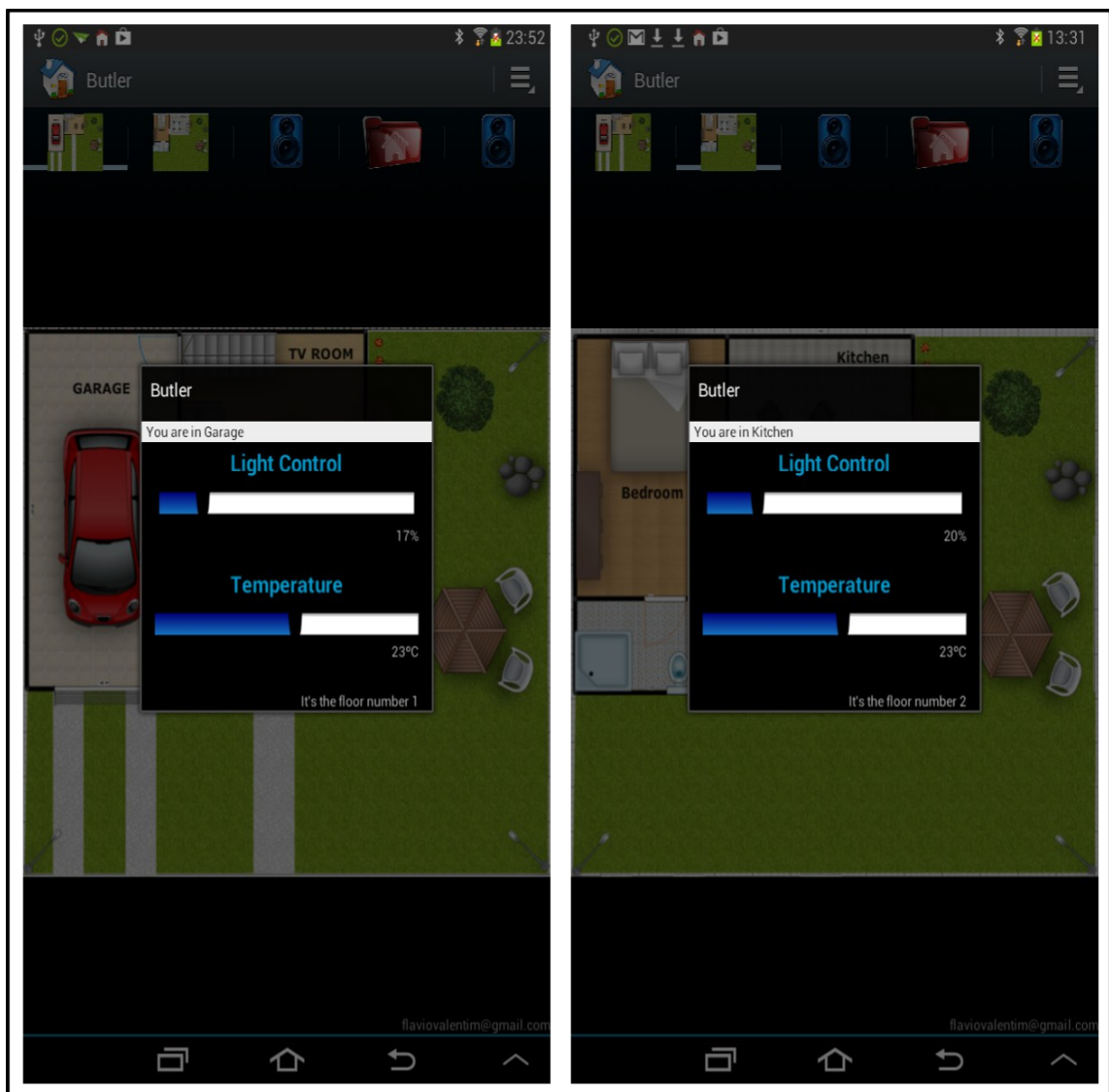


Fonte: Produção do próprio autor.

A tela principal apresenta a imagem de uma planta baixa referente a um modelo de residência que possui dois andares. As figuras tiveram as regiões referentes a cada cômodo mapeadas, de forma que ao tocar nas mesmas, serão exibidas ao

usuário as opções de comando disponíveis, que para o protótipo construído serão o controle da intensidade de luz e temperatura, restando portanto ao usuário apenas escolher o valor que lhe é de interesse para os referidos parâmetros, conforme ilustrado na Figura 4.3.

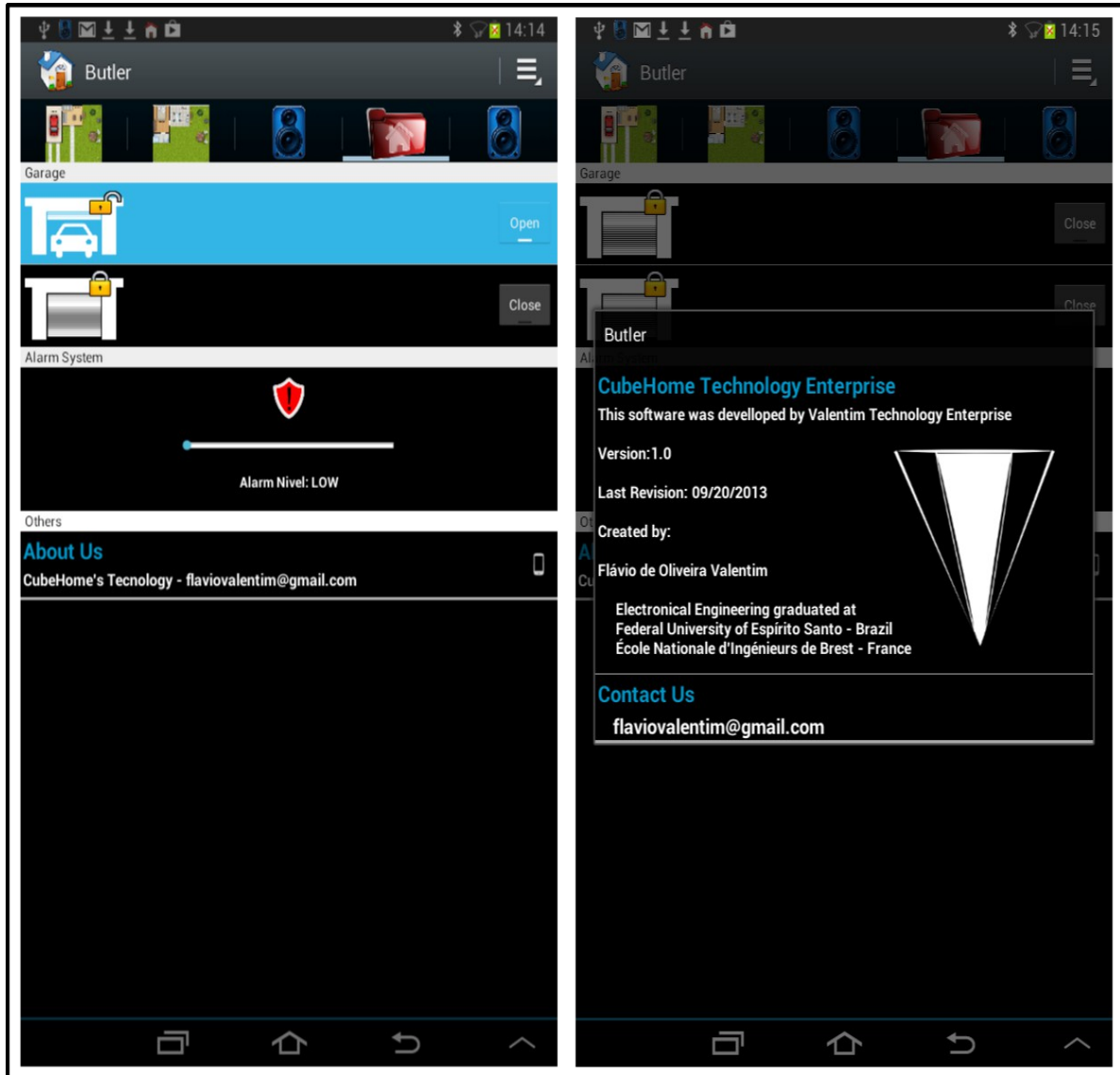
Figura 4.3: Opções de comando básicas disponibilizadas ao usuário.



Fonte: Produção do próprio autor.

A Figura 4.4 mostra o segundo grupo de parâmetros controláveis, que visa simular um sistema de segurança eletrônica e abertura e fechamento do portão da garagem. Ainda neste grupo, acrescentou-se as informações referentes ao desenvolvedor.

Figura 4.4: Comandos de controle de alarme e abertura de garagem.

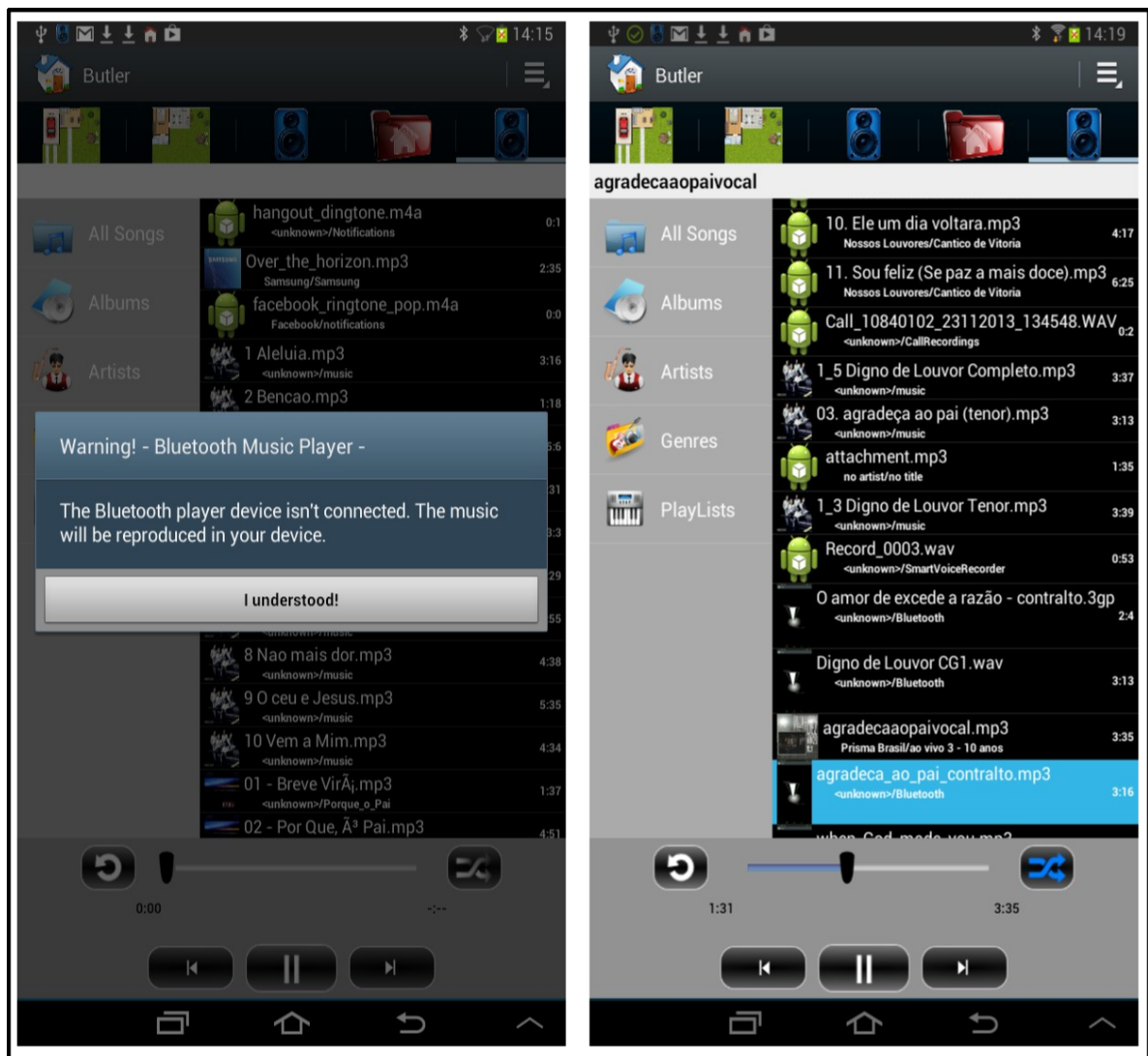


Fonte: Produção do próprio autor.

Por fim, será possível ao usuário executar os áudios disponíveis em seu aparelho *smartphone* no ambiente doméstico utilizando comunicação *Bluetooth* (Figura 4.5). Nesta função optou-se por uma solução que não exigisse ao usuário a inserção de um dispositivo de memória que armazenasse os áudios para execução, o que a tornaria incômoda fugindo ao princípio de integração dos sistemas. Dessa forma, entendeu-se que a melhor opção seria uma solução que permitisse a execução das músicas disponíveis em seu próprio aparelho *smartphone*, integrando os sistemas.

Naturalmente, o dispositivo transmissor deverá estar no raio de alcance do receptor, que neste caso é de até 10 metros.

Figura 4.5: Interface do tocador de áudio.



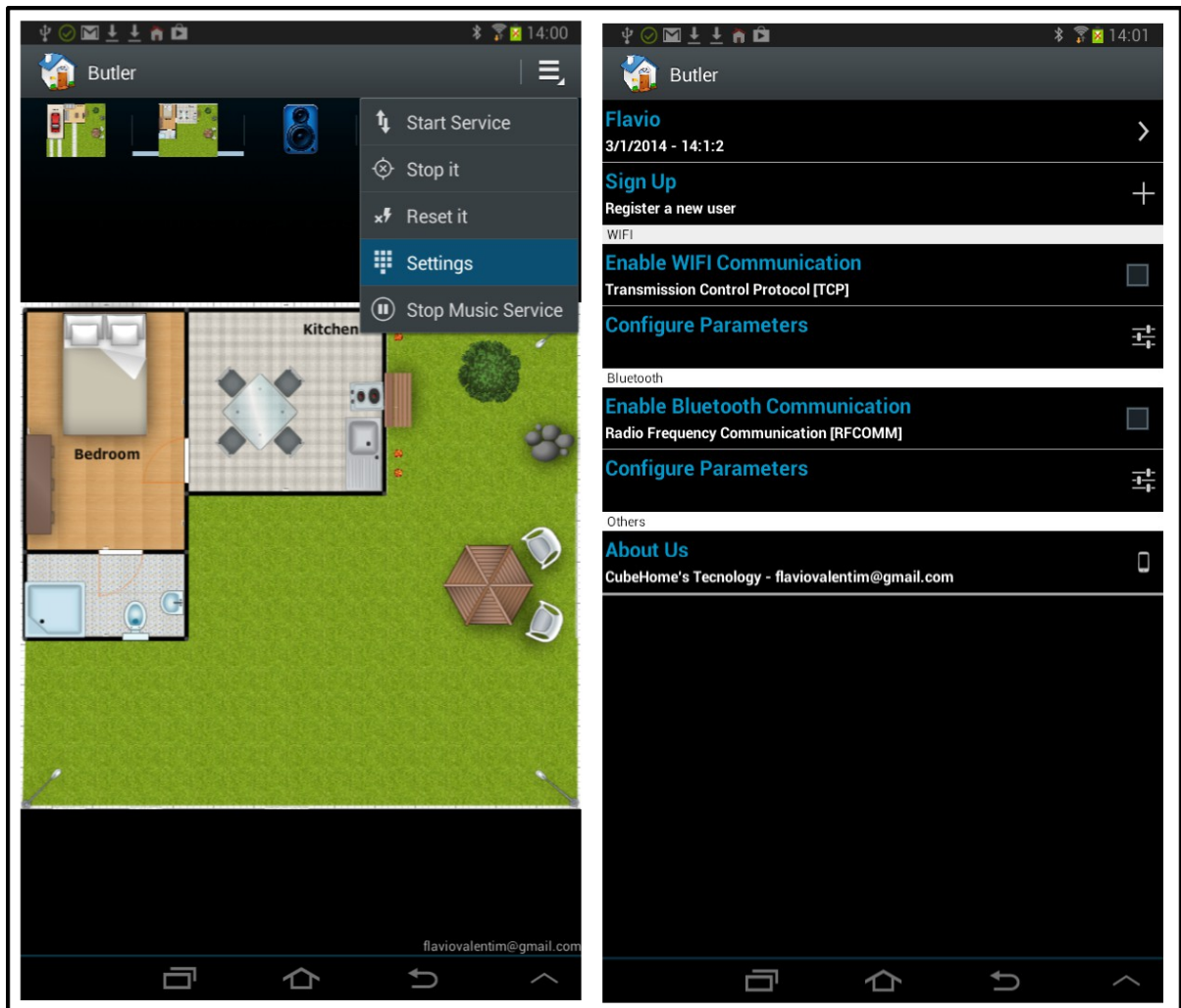
Fonte: Produção do próprio autor.

4.1.2 Configuração

Para a IHM em questão, utilizou-se um banco de dados que associa o usuário com as ações selecionadas, de forma que se tenha um registro que possa ser consultado para possíveis correções. Isto cria um ambiente próprio à implementação de algoritmos de predição de ações baseados nas escolhas passadas de cada

usuário. A Figura 4.6 mostra as opções de configurações gerais disponibilizadas ao usuário.

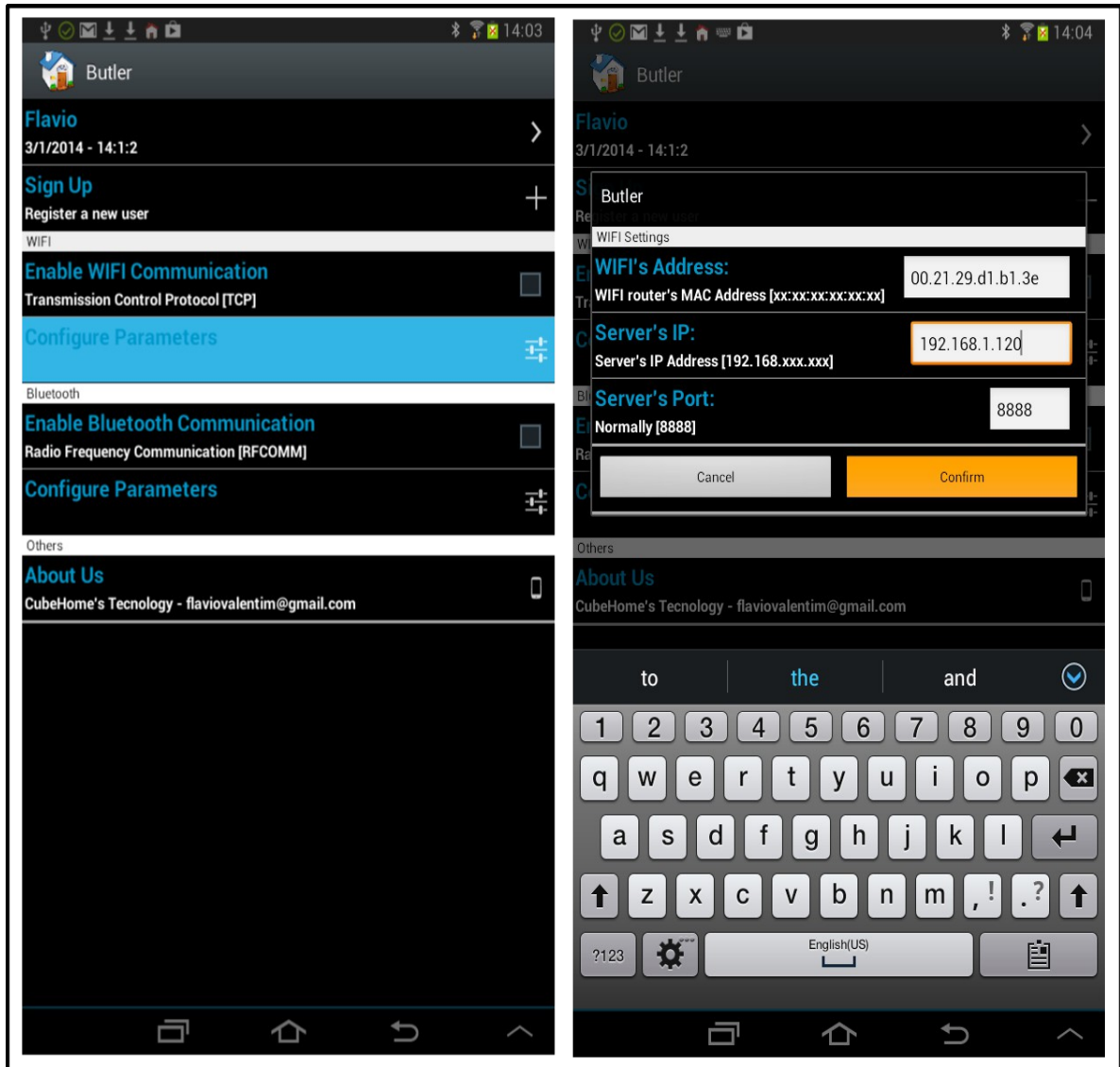
Figura 4.6: Opções de configuração gerais.



Fonte: Produção do próprio autor.

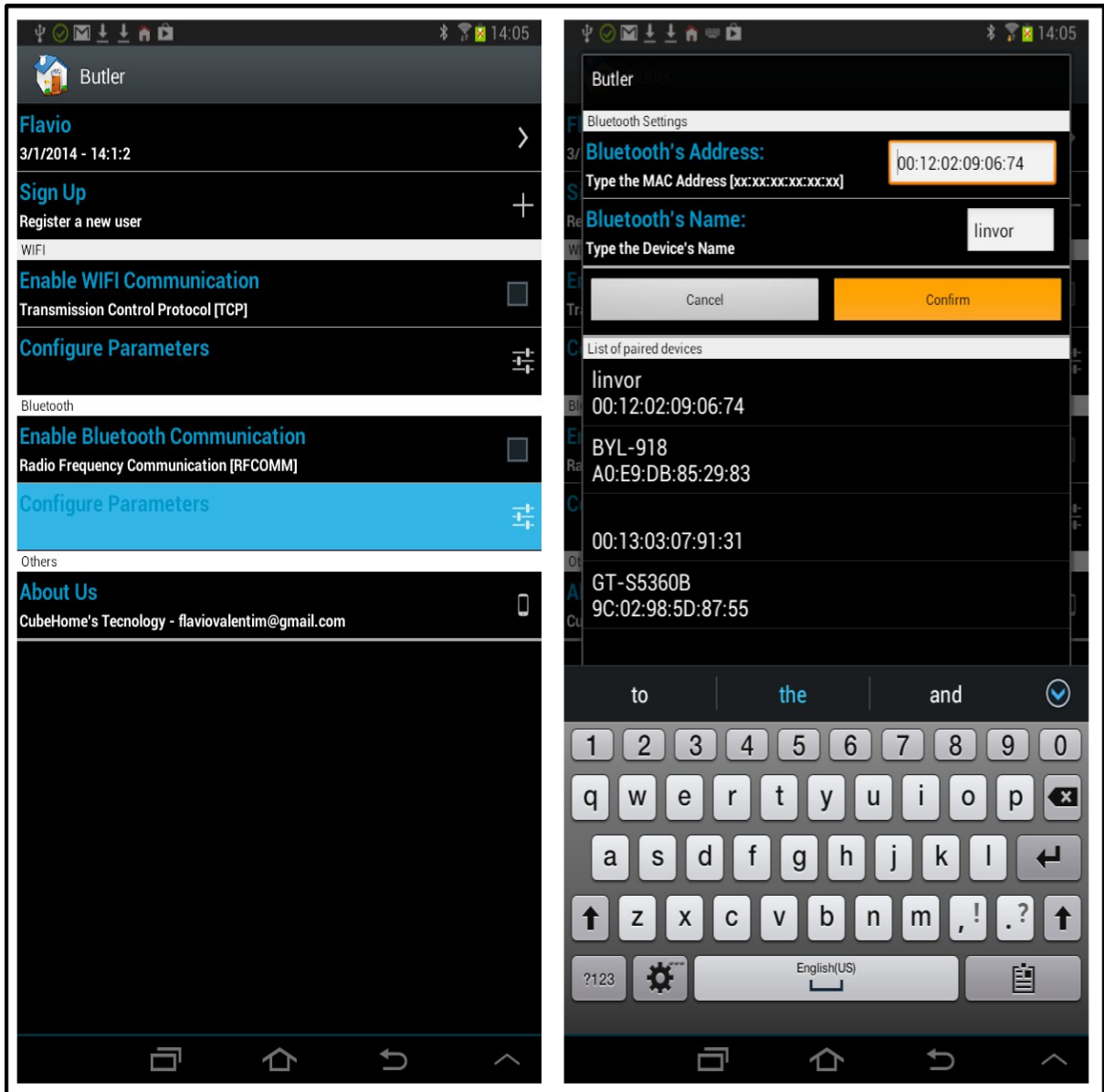
Para que seja possível estabelecer uma conexão do tipo TCP, é preciso informar ao sistema alguns dados mínimos, conforme ilustrado na Figura 4.7: endereço de IP; endereço MAC e porta.

Figura 4.7: Configuração de conexão TCP.



Fonte: Produção do próprio autor.

De maneira análoga à conexão TCP, para a utilização do protocolo *Bluetooth*, se faz necessário informar os seguintes dados (Figura 4.8): endereço MAC e nome do dispositivo receptor.

Figura 4.8: Configuração de conexão *Bluetooth*.

Fonte: Produção do próprio autor.

4.1.3 Codificação das Mensagens

No que se refere à composição das mensagens que codificam os comandos acionados pelo usuário, cada pacote será preenchido conforme descrito na seção 3.1.3.

Para o protótipo desenvolvido neste projeto, fixou-se o valor dos endereços de

cada sensor conforme descrito na Tabela 4.1. No entanto, buscou-se preparar uma estrutura que permita a melhoria deste projeto no sentido de tornar possível uma configuração dinâmica com a possibilidade de excluir ou adicionar novos sensores de maneira simplificada, de forma que um usuário sem conhecimento técnico aprofundado possa fazê-lo.

Tabela 4.1: Endereço dos sensores.

<i>Região da Casa</i>	<i>Ação</i>	<i>Endereço Octadecimal</i>
Garagem	Luz	5
	Temperatura	15
Sala de TV	Luz	4
	Temperatura	14
Quarto	Luz	3
	Temperatura	13
Cozinha	Luz	2
	Temperatura	12
Outros	Portão 1	1
	Portão 2	11

Fonte: Produção do próprio autor.

Tabela 4.2: Comandos disponíveis e seus respectivos códigos.

<i>Comando</i>	<i>Pseudônimo (Alias)</i>	<i>Código Hexadecimal</i>
Liga/Desliga	mONOFF	A1
Controle PWM	mPWM	A2
Portão de Garagem	mGARAGE	A3
Simulação de Presença	mSIMULATION	A4
Senha de Autenticação	mPASSWORD	F1
Endereço do Sensor	mNODEADDRESS	F2
Modo de Operação	mMODEOPERATION	F3

Fonte: Produção do próprio autor.

Dentre os comandos descritos na Tabela 4.2, os três últimos estão disponibilizados em uma primeira versão apenas via acesso serial diretamente nos sensores. Fato este que será mais profundamente abordado na seção 4.3.

4.2 Central de Controle - CC

Para a implementação da CC foram utilizados os seguintes módulos:

- Arduino Mega 2560;
- Arduino *Ethernet*;
- *Bluetooth*;
- NRF24L01;

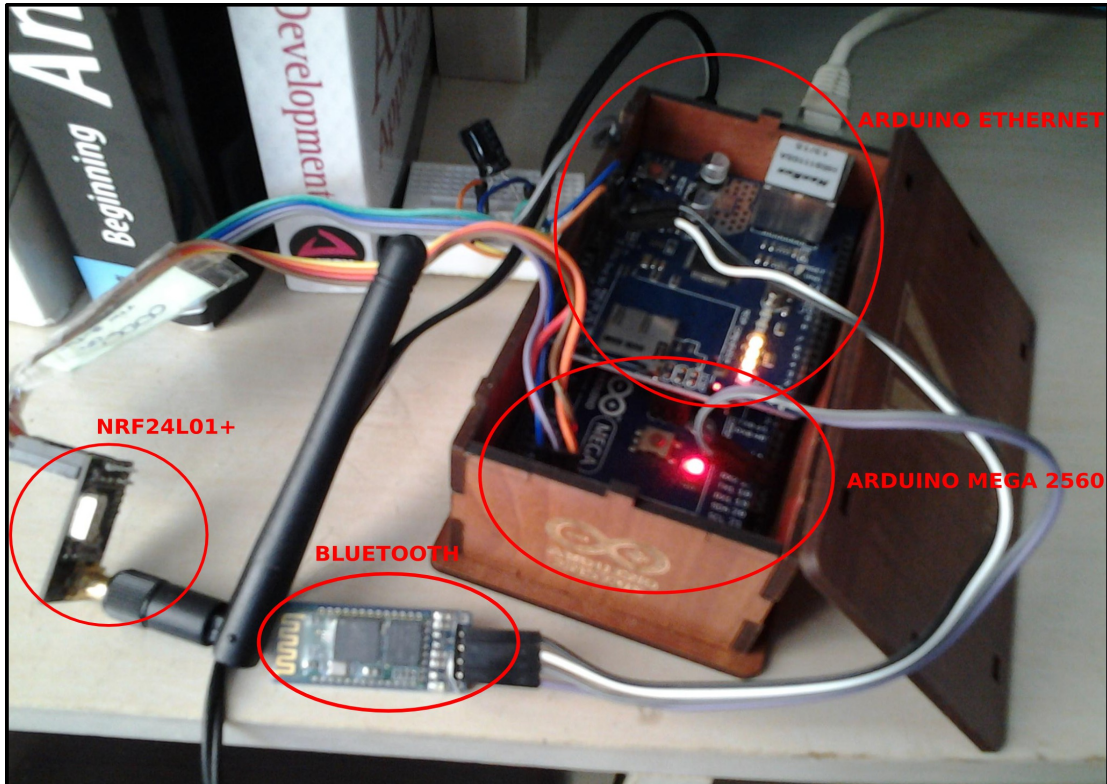
A escolha do módulo Arduino Mega 2560 se deu devido a seus 256KB de memória *flash*, visto que a maior parte da lógica implementada no sistema será concentrada neste módulo. Além disso, a disponibilidade de várias portas de comunicação serial facilita o processo de desenvolvimento e tratamento de erros, uma vez que se pode utilizar uma delas exclusivamente para depuração e uma segunda porta apenas para comunicação com o módulo *Bluetooth*.

Os módulos *Bluetooth* e Arduino *Ethernet* consistem em periféricos que tornam possível o controle do sistema via protocolos *Bluetooth* e TCP respectivamente. Este visa tornar possível ao usuário comandar o sistema através da mesma rede de acesso à internet. Em um primeiro momento, o controle será feito no âmbito local, visto a necessidade de se tratar alguns aspectos de segurança essenciais antes de disponibilizar acesso remoto.

O NRF24L01 é, sob a ótica de Redes de Computadores, a 'Porta de Entrada Padrão' (do inglês - *Gateway Default*) entre a CC e a RSSF, visto que todas as mensagens trocadas por ambas as partes deverão passar pelo mesmo.

A Figura 4.9 mostra o protótipo da CC indicando cada circuito que a compõe:

Figura 4.9: Protótipo da central de controle testada.



Fonte: Produção do próprio autor.

4.2.1 Formatação e Encaminhamento das Mensagens

As mensagens transmitidas através do dispositivo NRF24L01 possuem uma codificação própria e por esse motivo, se faz necessário decodificar a mensagem recebida na CC para adequá-la ao protocolo utilizado na RSSF. Esse mecanismo tem a vantagem de criar uma estrutura que facilita a implementação de melhorias de segurança no sistema. Isto porque, torna possível adicionar uma camada extra de filtros de bloqueio e autenticação diretamente na CC, sem que seja necessário implementar grandes alterações no projeto.

O algoritmo desenvolvido para a CC é responsável por verificar se um pacote proveniente da IHM está completo através da conferência dos valores contidos nos campos inicial e final do mesmo. Em seguida ele rearranja os dados recebidos em uma estrutura que possa ser encaminhada à RSSF.

4.3 Rede de Sensores Sem Fio - RSSF

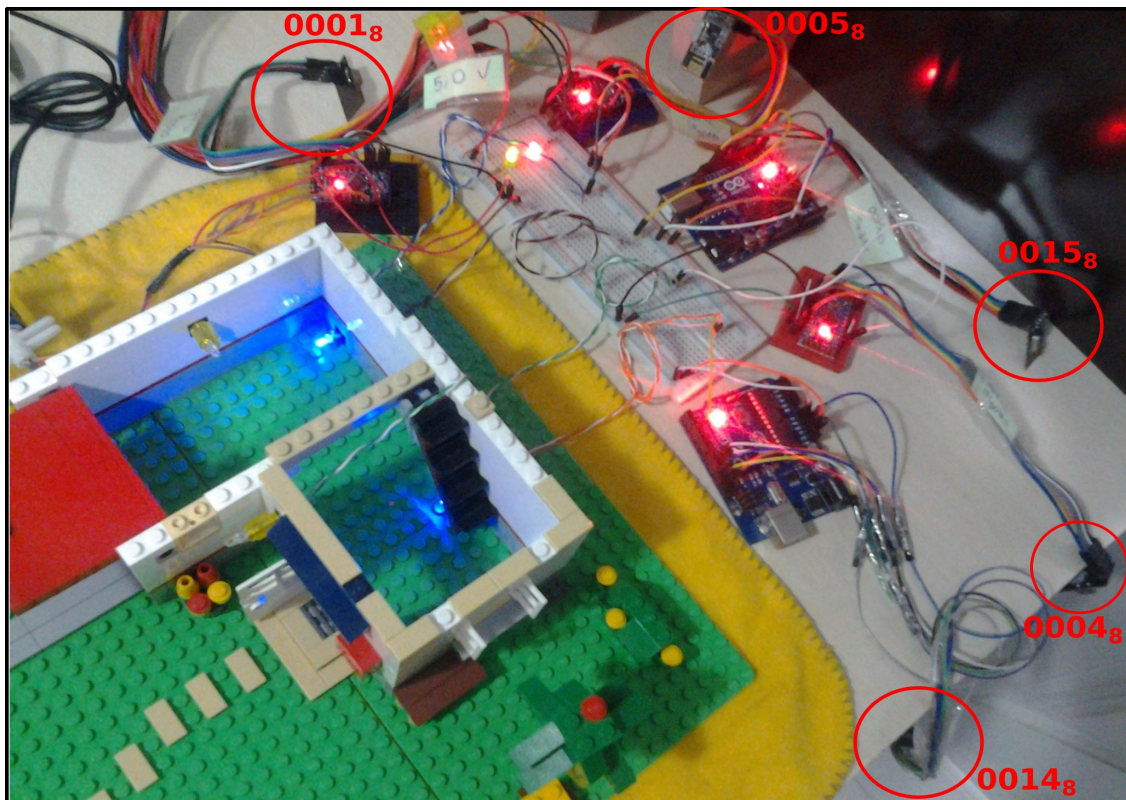
Na implementação de cada um dos nós que compõem a RSSF optou-se pela utilização dos seguintes módulos:

- Arduino Nano;
- NRF24L01+;

O módulo Arduino Nano utiliza um microcontrolador ATmega328 em uma placa de dimensões bem reduzidas (ver seção 2.2.3) e 32KB de memória flash, suficiente à execução do programa desenvolvido. Enquanto que, o NRF24L01+ é o responsável pela troca de mensagens sem fio. Esse conjunto mínimo compõe o *hardware* de cada nó.

A Figura 4.10 mostra a RSSF implementadas no protótipo com seus respectivos endereços no formato octadecimal.

Figura 4.10: Nós da rede formada e seus respectivos endereços octadecimais.

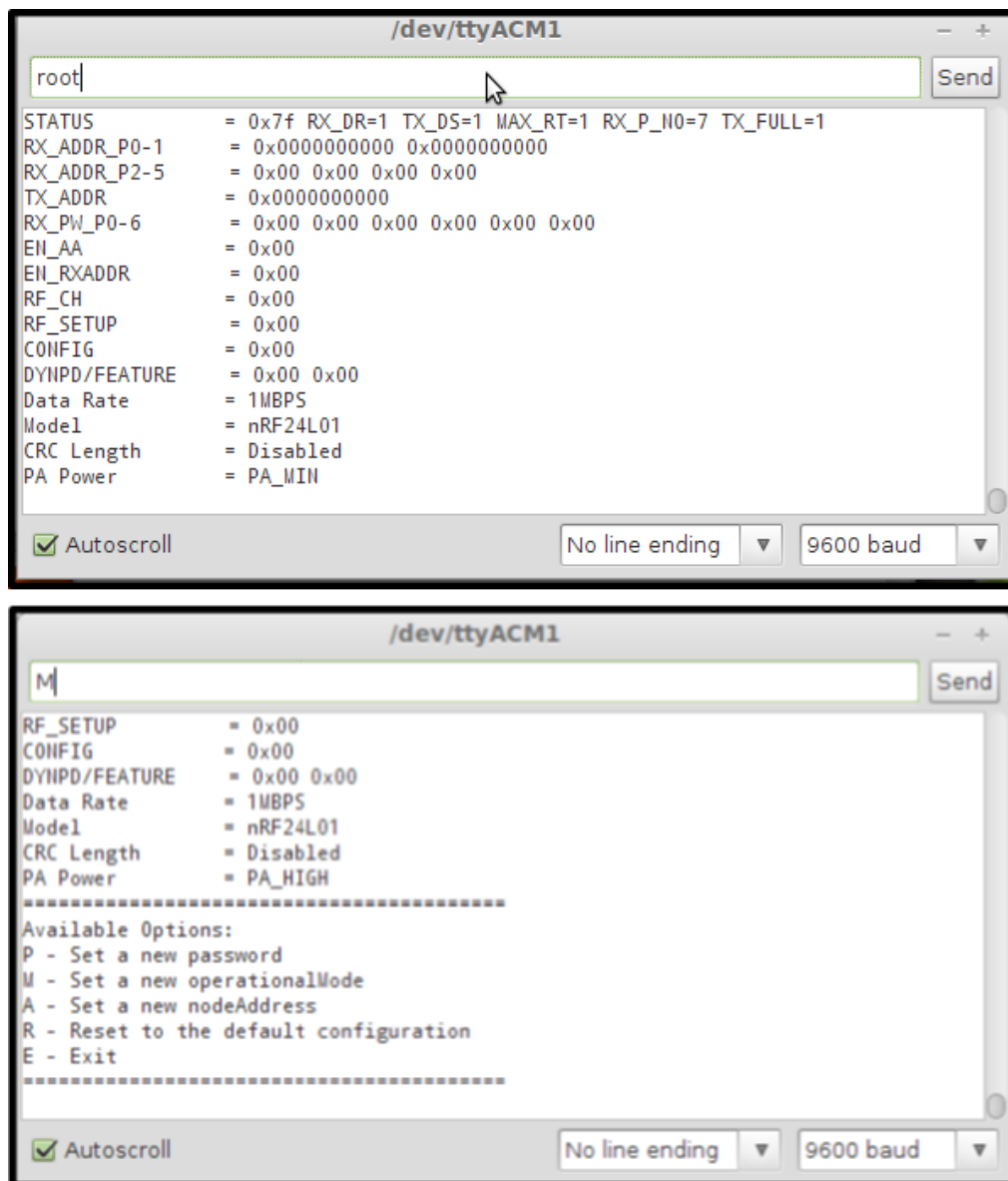


Fonte: Produção do próprio autor.

4.3.1 Configuração do Modo de Funcionamento

O programa base implementado para cada nó da rede permite a alteração dos parâmetros básicos de funcionamento através de uma interface de comunicação serial acessível a partir da palavra-chave **'root'**. A Figura 4.11 mostra o acesso às configurações com um exemplo de alteração do parâmetro Modo de Operação (ver Tabela 4.3).

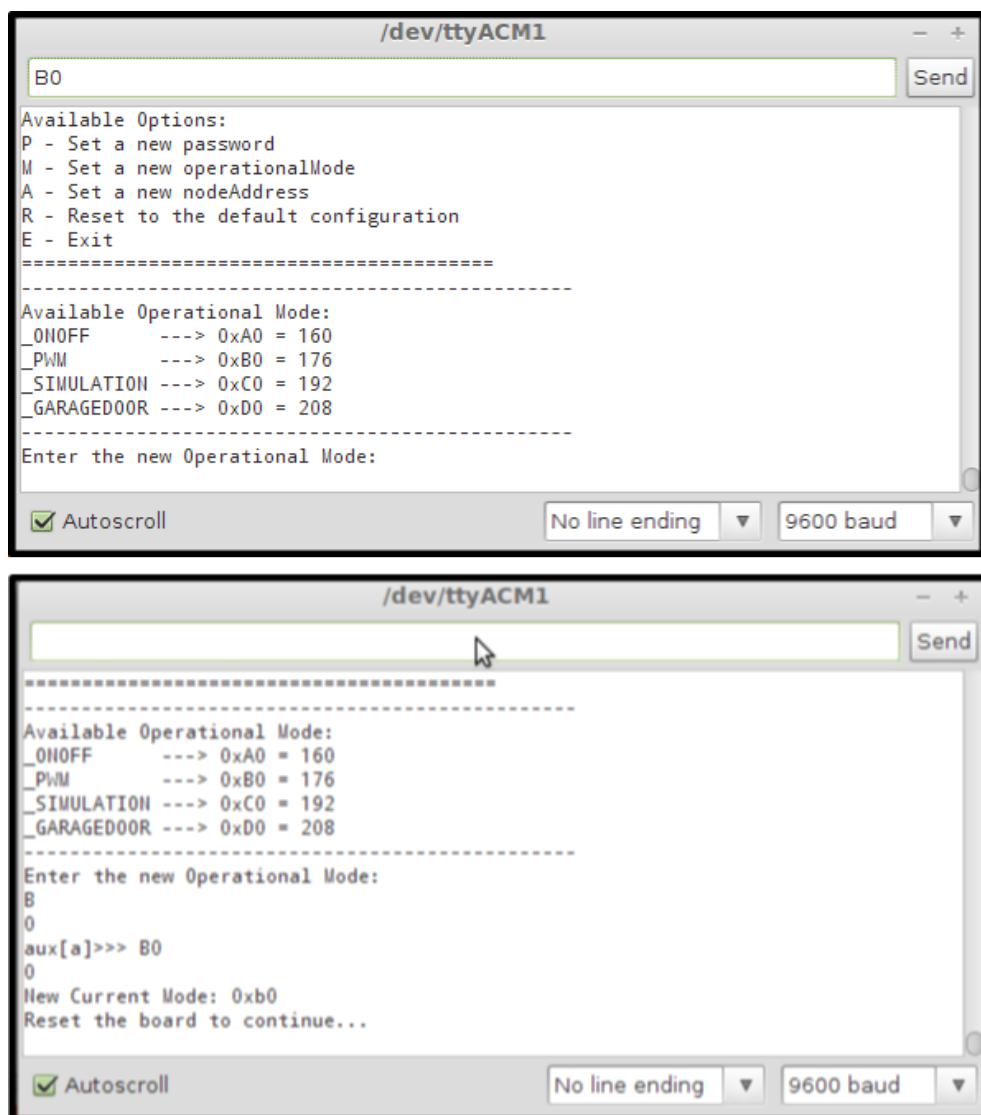
Figura 4.11: Acessando a interface de configuração via comunicação serial.



Fonte: Produção do próprio autor.

Como a configuração de cada nó é feita durante a inicialização do microcontrolador, para que uma alteração em sua configuração tome efeito, se faz necessário reinicializar o mesmo. Assim, na Figura 4.12, é indicado ao usuário a necessidade de se reinicializar o módulo após a alteração da configuração do nó.

Figura 4.12: Modificando o modo de operação do atuador através de interface serial.



Fonte: Produção do próprio autor.

A Tabela 4.4 mostra os códigos referentes aos modos de operação pré-programados em cada um dos nós da rede.

Tabela 4.3: Parâmetros configuráveis de cada nó da rede.

Parâmetro	Pseudônimo	Observação
Senha de Autenticação	password	Até 10 caracteres
Modo de Operação	operationalMode	Ver Tabela 4.4
Endereço de Rede	nodeAddress	Ver Seção 3.3.1
Restaurar Padrões	reset	

Fonte: Produção do próprio autor.

Tabela 4.4: Modo de funcionamento dos módulos microcontrolados.

Modo de Operação	Pseudônimo (Alias)	Código Hexadecimal
Liga/Desliga	ONOFF	A0
Controle PWM	PWM	B0
Simulação de Presença	SIMULATION	C0
Portão de Garagem	GARAGEDOOR	D0

Fonte: Produção do próprio autor.

4.3.2 Decodificação e Execução de Comandos

Ao receber uma mensagem, o microcontrolador autentica a mesma através da verificação do campo de *'password'* e executa uma ação, caso o comando solicitado seja válido. Esta ação consiste em ligar/desligar uma saída digital ou controlar uma saída PWM (Ver seção 3.3.2).

Os comandos implementados são os mesmos descritos na Tabela 4.2 e estão mapeados em funções que executam finalmente a ação solicitada.

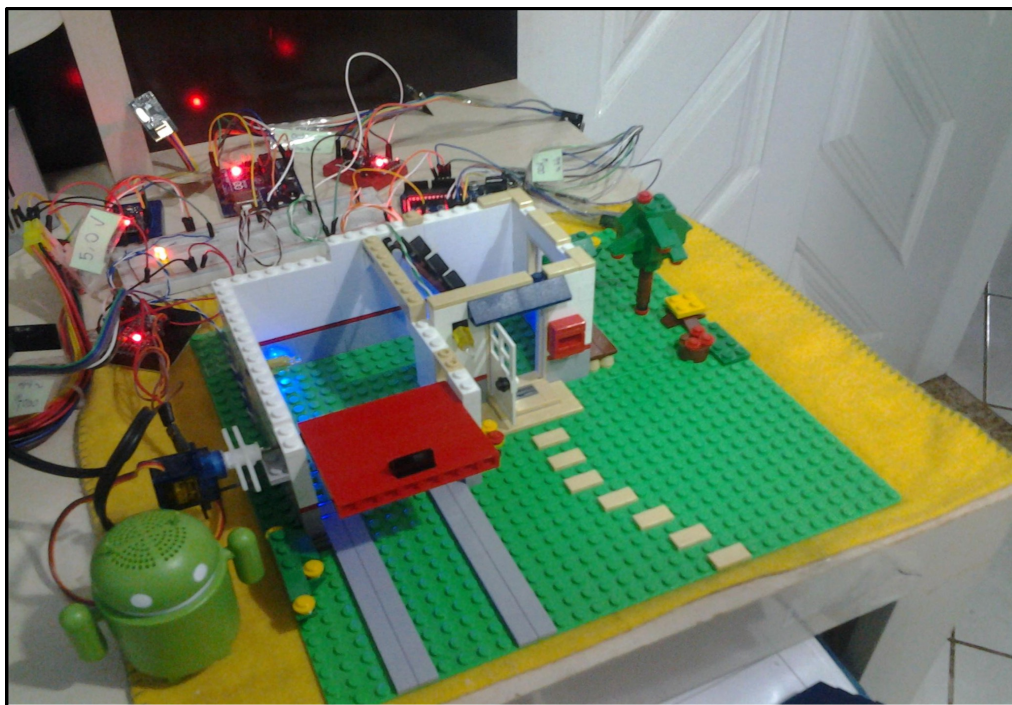
A vista exterior do modelo de ambiente residencial sob o qual os comandos foram executados pode ser vista na Figura 4.13, enquanto a Figura 4.14 permite a visualização de seu interior.

Figura 4.13: Vista exterior do modelo de ambiente residencial utilizado.



Fonte: Produção do próprio autor.

Figura 4.14: Vista interior do modelo de ambiente residencial utilizado.



Fonte: Produção do próprio autor.

4.4 Interface com Atuadores Analógicos

Para que todo o sistema digital descrito possa interagir com o mundo analógico é preciso que haja a conversão dos comandos gerados em sinais analógicos capazes de controlar grandezas de interesse, como por exemplo a potência dissipada em um circuito de iluminação.

Esta seção tratará portanto, de apresentar soluções para a aplicação do sistema proposto em um ambiente real, permitindo o controle de potência, temperatura e ações de ligamento e desligamento remoto.

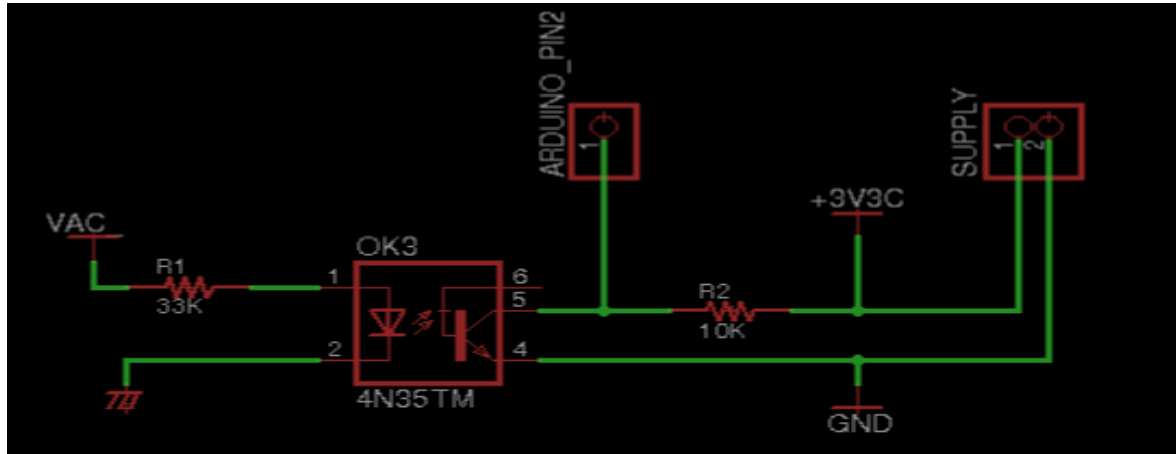
4.4.1 Controle de Potência (*Dimmer*)

Uma vez que os sistemas de iluminação são alimentados eletricamente com tensão alternada, o seu controle consiste na regulação da potência dissipada, que pode ser feito através da manipulação da Tensão RMS (*Root Mean Square*), que matematicamente é a média quadrática da tensão enquanto que do ponto de vista elétrico é definido como o equivalente em tensão contínua que dissiparia a mesma potência em uma mesma carga.

Esse tipo de controle pode ser feito com o uso de um Tiristor que consiste em um dispositivo semiconductor que permite o controle da passagem de corrente através de um sinal de controle aplicado a um de seus terminais: o *gate*.

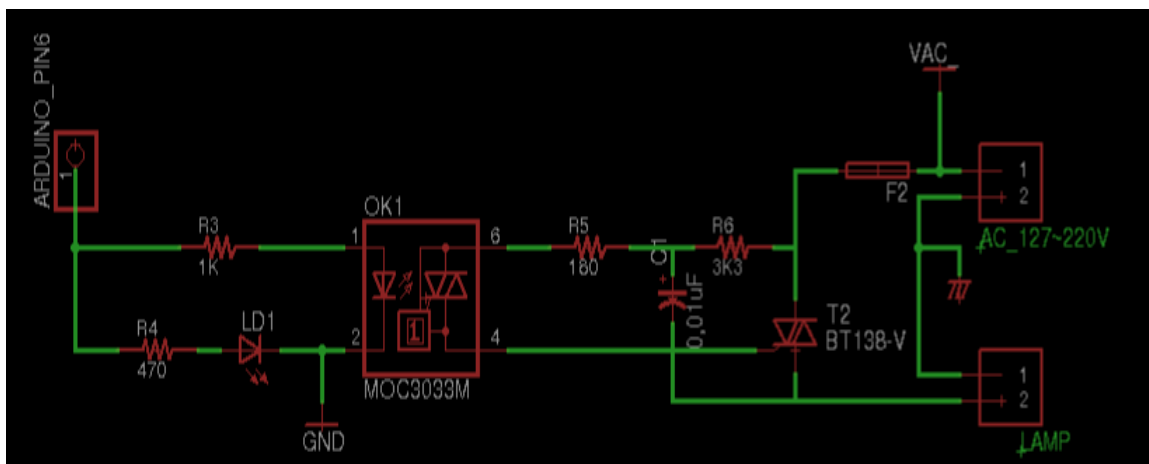
A Figura 4.15 mostra o esquemático de um circuito que permite a detecção da passagem por zero de um sinal senoidal. Essa detecção será configurada no microcontrolador para gerar um interrupção e dessa forma tornar possível a geração de um sinal sincronizado para acionar o tiristor do circuito de controle de potência cujo esquema se encontra disponível na Figura 4.16. Em ambos os circuitos é utilizado um acoplador ótico para permitir interação de um sinal senoidal de tensão doméstica com um sistema digital.

Figura 4.15: Esquemático do circuito de detecção de cruzamento de zero.



Fonte: Produção do próprio autor.

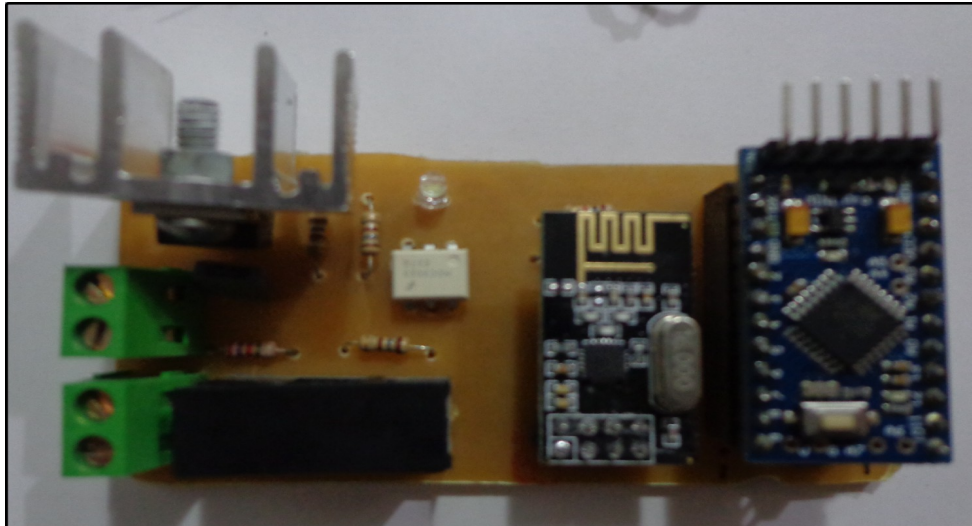
Figura 4.16: Circuito de controle de potência através de um tiristor.



Fonte: Produção do próprio autor.

O circuito final implementado (Figura 4.17) pode ser usado igualmente para aplicações de funcionamento do tipo chave, ou seja, com os estados ligado e desligado. Neste caso basta aplicar um sinal de entrada digital.

Figura 4.17: Circuito de acionamento e controle de potência implementado.



Fonte: Produção do próprio autor.

4.4.2 Controle de Temperatura

A utilização de aparelhos de ar condicionado tem aumentado grandemente nos últimos anos sobretudo em países como o Brasil, devido às suas características climáticas. Com o aumento do consumo, cresce também as opções de marcas e modelos disponibilizadas no mercado, dentre os quais destacam-se: Samsung, Consul, Electrolux, Gree, Midea e muitos outros.

Em geral, esse tipo de aparelho pode ser comandado através de um controle remoto que envia sinais de infravermelho codificados em frequência e observando esse fato, a solução apresentada consiste em decodificar esses comandos, criando assim um circuito clone que poderá ser integrado a um dos nós da rede e, portanto, ser controlado através da IHM apresentada na seção 4.1.

Para o circuito de captura utilizou-se o dispositivo TL1838, que consiste em um receptor infravermelho projetado para a recepção de sinais na faixa de 38kHz encapsulados em um mesmo circuito com três pinos:

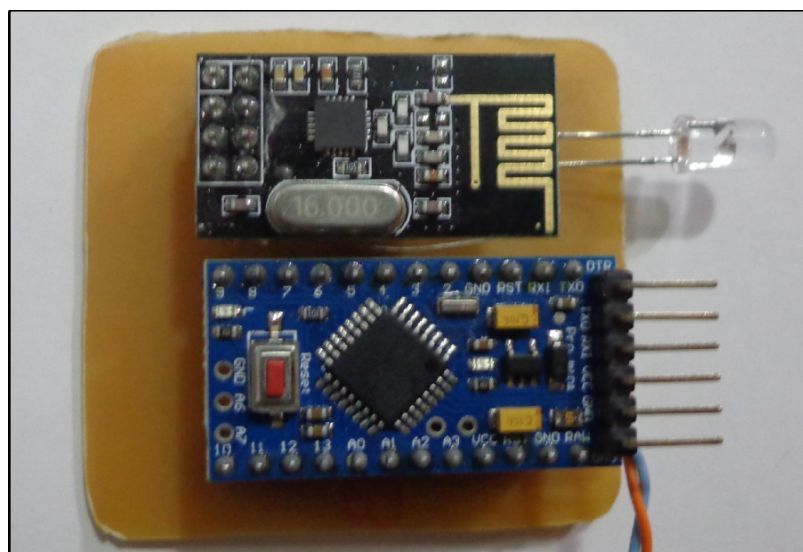
- Vcc e Gnd: referentes à alimentação do circuito que, conforme documentação técnica, deve estar entre 2.7 e 5.5 Volts.

- Saída: sinal de entrada filtrado, onde o valor de saída 'Vcc' corresponde à ausência de sinal infravermelho enquanto que 'Gnd' corresponde à presença desse sinal.

O processo de decodificação dos comandos de um controle infravermelho consiste em capturar os sinais emitidos pelo mesmo através de um receptor, interpretá-los na forma de zeros e uns (código binário) para então buscar compreender a lógica utilizada. Para isso, pode-se utilizar um microcontrolador cujo programa de controle avalie a variação de um sinal de entrada infravermelho. Existem diversas bibliotecas de código aberto disponibilizadas para a plataforma Arduino que auxiliam esse processo, tendo sido utilizada nesta aplicação a biblioteca desenvolvida por Ken Shirriff e aprimorada por Chris Young (YOUNG, 2014).

O circuito de emissão (Figura 4.18) consiste em um *led* infravermelho em série com uma resistência para limitar o valor da corrente no circuito, ligado a uma saída digital de um microcontrolador, neste caso uma placa Arduino Nano, responsável por gerar os pulsos na frequência de 38kHz e na sequência referente ao comando decodificado anteriormente.

Figura 4.18: Circuito de controle de dispositivos através de emissor infravermelho.



Fonte: Produção do próprio autor.

5 CONCLUSÃO

Sistemas de automação residencial sem fio prometem mudar definitivamente a rotina das pessoas, visto que aumentam consideravelmente a sensação de conforto e comodidade ao permitirem uma maior interação do mesmo com o seu usuário. Controlar uma casa a partir de um aparelho *smartphone* já possível e, devido aos avanços tecnológicos e desenvolvimento de tecnologias de comunicação cada vez mais baratas e de baixo consumo, tem se tornado cada vez mais acessível.

Os objetivos de desenvolvimento deste projeto foram alcançados através de um sistema completo de automação, que integra uma rede e atuadores sem fio controlada a partir de uma IHM. Esta, pode ser acessada através de qualquer dispositivo Android, como um simples aplicativo, não havendo necessidade de grande conhecimento técnico para operá-lo, o que aumenta enormemente o potencial de aceitação no mercado.

A solução proposta tem como característica o fato de poder ser aplicada a diversos tipos de sistemas como por exemplo: em sistemas de segurança predial, monitoramento e controle de ambientes fechados, entre outros. Uma outra vantagem consiste em sua característica de baixo custo:

- Nó de rádio da RSSF e atuadores: entre 15 e 30 dólares.
- CC: Aproximadamente 40 dólares.
- IHM: Por se tratar de um *software*, seu valor não pode ser estimado.

Apesar de os resultados obtidos com o protótipo implementado terem sido satisfatórios, algumas melhorias são ainda desejáveis para que se possa disponibilizar uma primeira versão ao mercado:

- **IHM:** Visando uma maior aceitação do mercado de tecnologia, realizar a portabilidade do aplicativo desenvolvido para os sistema IOS, da fabricante Apple, e Windows, da Microsoft.
- **RSSF:** Para aumentar a confiabilidade do sistema, é necessário criar um mecanismo que cuide do reenvio de mensagens, quando não houver

confirmação do recebimento das mesmas. Além disso, um algoritmo que diagnostique as condições da rede, verificando os nós ativos e tratando problemas de conexão dessa natureza.

- **Circuitos finais:** As placas podem ser redesenhadas no intuito de terem seus tamanhos reduzidos.

Com este trabalho conclui-se que o uso das tecnologias disponíveis nos aparelhos *smartphones*, aliado a sistemas que utilizem microcontroladores de baixo custo diminuem consideravelmente os custos de um sistema de automação residencial, sendo portanto um caminho perfeitamente viável ao desenvolvimento dos mesmos.

REFERÊNCIAS BIBLIOGRÁFICAS

ARDUINO. Arduino Mega 2560. Disponível em: <<http://arduino.cc/en/Main/arduinoBoardMega2560>>. Acesso em: 21 dez. 2013.

ARDUINO. Arduino Nano. Disponível em: <<http://arduino.cc/en/Main/arduinoBoardNano>>. Acesso em: 21 dez. 2013.

ARDUINO. Arduino Shield Ethernet. Disponível em: <<http://arduino.cc/en/Main/ArduinoEthernetShield>>. Acesso em: 21 dez. 2013.

BARAKA K., GHOBRI M., MALEK S., KANJ R., KAYSSI A.. Low cost Arduino/Android-based Energy-Efficient Home Automation System with Smart Task Scheduling. 2013 Fifth International Conference on Computational Intelligence, Communication Systems and Networks, pp. 296-301, 2013.

DEITEL P. , DEITEL H.. Java Como Programar. 8th Ed., Pearson, 2010.

HSIAO R. S., LIN D. B., H. P. LIN, CHUNG C. H., CHENG S. C.. Integrating Zigbee Lighting Control Into Existing Building Automation Systems. IET International Conference on Information Science and Control Engineering (ICISCE 2012),, Shenzhen-HongKong, 2012.

KEER D.. Android dominates 81 percent of world smartphone market, nov. 2013.. Disponível em: <http://news.cnet.com/8301-1035_3-57612057-94/android-dominates-81-percent-of-world-smartphone-market/>. Acesso em: 18 mar. 2014.

MANIACBUG. RF24Network for Wireless Sensor Networking, mar. 2010. Disponível em: <<http://maniacbug.wordpress.com/2012/03/30/rf24network/>>. Acesso em: 17 mar. 2014.

MYSEHOME. C-Bus Home Control. Disponível em: <<http://www.mysehome.com/>>. Acesso em: 8 Julho 2014.

MUHAMMAD A. MAZIDI, SARMA NAIMI, SEPHER NAIMI. The AVR Microcontroller and Embedded Systems. 1st ed., Prentice Hall, 2011.

NORDIC. NRF 24L01+ - Ultra low power 2.4GHz RF Transceiver IC. Disponível em: <www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01P>. Acesso em: 4 nov. 2013.

SHANI, Su J., NIE L., QU S.. Design of Multi-Point Wireless Temperature Measuring System. Modelling, Identification and Control, pp. 422-425, June 2012.

TANENBAUM, ANDREW W. S.. Computer Networks. 4th ed., Prentice Hall, 2003.

WISER. Wiser Home Control. Disponível em: <<http://www.wiserhomeautomation.com/index.htm>>. Acesso em: 8 Julho 2014.

YOUNG C.. An Arduino library for encoding and decoding infrared remote signals, jun. 2014.. Disponível em: <<https://github.com/cyborg5/IRLib>>. Acesso em: 13 maio 2014.

Z-WAVE ALLIANCE. Z-Wave Home Control. Disponível em: <<http://www.z-wave.com/>>.

Acesso em: 8 Julho 2014.