FÁBIO LABEGALINI ZUCATO

Rede ZigBee Gerenciada por Sistema de Monitoramento Remoto Utilizando TCP/IP e GPRS

Dissertação apresentada à Escola de Engenharia de São Carlos da Universidade de São Paulo, como parte dos requisitos para a obtenção do título de Mestre em Engenharia Elétrica.
Área de concentração: Telecomunicações

Orientadora: Mônica de Lacerda Rocha

São Carlos
2009

FOLHA DE JULGAMENTO

Candidato(a): Engenheiro **FÁBIO LABEGALINI ZUCATO.**

Dissertação defendida e julgada em 18.12.2009 perante a Comissão Julgadora:

Mônica de Lacerda Rocha — Aprovado

Profª. Drª. **MÔNICA DE LACERDA ROCHA (Orientadora)**
(Escola de Engenharia de São Carlos/USP)

Aprovado

Prof. Associado **LUIZ GONÇALVES NETO**
(Escola de Engenharia de São Carlos/USP)

Daniel Moutinho Pataca — Aprovado

Dr. **DANIEL MOUTINHO PATACA**
(Centro de Pesquisa e Desenvolvimento em Telecomunicações/CPqD)

Prof. Titular **GERALDO ROBERTO MARTINS DA COSTA**
Coordenador do Programa de Pós-Graduação em Engenharia Elétrica e
Presidente da Comissão de Pós-Graduação

Agradecimentos

A todos na N3E que me deram todo o suporte em meus planos mirabolantes de virar mestre.

A minha orientadora, Profa. Dra. Mônica de Lacerda Rocha, pela prontidão de sua atenção em todos os momentos que precisei.

Aos amigos de São Carlos, pois só por eles suportei o fim das regalias de minha pequena cidade. Aprendi a tê-los em minha cozinha, ainda que não lhes deixasse mexer em minhas panelas. Eles terão minha gratidão e minha cumplicidade eternas.

Aos velhos amigos que cresceram comigo, presenciando minhas primeiras ressacas de carnaval e minhas primeiras desilusões com o futebol.

Aos novos amigos que vêm transformando minha vida em algo melhor e mais intenso, por mais irresponsável que pareça.

Aos meus primos que são meu tesouro e minha identidade. Tudo o que sou foi modificado de alguma forma pra se adaptar a eles.

Aos meus tios, meus heróis e meu respaldo. Pessoas idôneas e queridas. Por causa deles tornei-me leal como os cães e teimoso como as mulas.

A minha namorada, Cris, vida injetada em minha carne. Parte inseparável dos meus sonhos. Meu desassossego e minha calmaria.

A minha vó, a ternura de bengalas, Maria infinita. Algazarra forte nos lamentos fúteis, juventude eterna em seus cabelos brancos. Minha dama, minha passageira, meu xodó. Quantas restam tão Maria quanto ela?

Por fim, agradeço aos meus pais, Beto e Rita e ao meu irmão, Bruno, pela paciência em aturar minha rebeldia e atitudes infantis. Pelo apoio financeiro, moral e intelectual que me deram durante a vida toda. São minha maior alegria e meu sucesso mais relevante.

“A colheita é comum, mas o capinar é sozinho”

João Guimarães Rosa

Resumo

Este trabalho propõe a integração de técnicas de sensoriamento dinâmico, redes de dados sem fio e internet. A implementação dos nós da rede visa permitir o monitoramento de objetos que se movem tanto numa rede interna, limitada a uma edificação, quanto numa rede externa, através de coordenadas GPS's (*Global Position Systems*).

A rede sem fio, que utiliza o protocolo ZigBee, é composta por sensores, atuadores e lâmpadas e é dotada de mobilidade através de controles remotos[1]. A rede ZigBee é integrada, através de um *gateway*, a uma rede TCP/IP para permitir o monitoramento e a atuação remota sobre ela, via um servidor HTTP e/ou uma rede de dados celular (GPRS), que, quando fora do alcance dos nós da rede ZigBee interna, torna-se responsável pelo envio de coordenadas GPS na rede externa, garantindo a onipresença do monitoramento.

Além das adaptações na pilha TCP/IP e o desenvolvimento de um software que utiliza a rede GPRS para envio de coordenadas GPS, destacam-se, como contribuições originais desta tese: (i) a solução de problemas da pilha ZigBee original no tocante a endereçamento, que impossibilitava a mobilidade na rede - desta forma, uma nova técnica de endereçamento seqüencial foi implementada com sucesso; (ii) novo código que simulou o AES (*Advanced Encryption Standard*), tanto para encriptação quanto para desencriptação dos dados.

Testes para validação dos protótipos desenvolvidos são apresentados.

Palavras- chave : ZigBee, redes sem fio, servidores HTTP, TCP/IP, criptografia, GPRS, GPS, banco de dados.

[1] - O controle remoto movimenta-se irrestritamente pela rede interna, atualizando dinamicamente as relações de paternidade que regem a hierarquia da rede ZigBee

Abstract

This MSc Thesis proposes the integration of dynamic sensing techniques, wireless data network and Internet. The implementation of network nodes aims to allow monitoring of moving objects, either inside an internal network, limited to one area, or in an external network, through GPS's (Global Position Systems).

The wireless network, which uses the ZigBee protocol, is composed of sensors, actuators and lamps, and is endowed with mobility through remote controls. Thus, it is integrated, through a gateway, to a TCP / IP network to allow remote monitoring and acting on it via an HTTP server and / or a mobile data network (GPRS), responsible for sending the GPS coordinates on the external network, ensuring the ubiquity of monitoring.

Further to adaptations in the TCP/IP stack and the development of a software that uses GPRS protocol to send GPS coordinates, the main contributions of this work are: (i) proposal of a new addressing technique, based on a sequential numbering of nodes, instead of the standard one, thus solving problems related to mobility in the network; (ii) proposal of a new security code to emulate AES (Advanced Encryption Standard), either for data encryption or decryption.

Tests for validation of the developed prototypes will be presented.

Keywords: ZigBee, wireless network, HTTP servers, TCP/IP, cryptography, GPRS, GPS, database.

Índice

Índice de Figuras

Índice de Tabelas

GLOSSÁRIO

AES - Advanced Encryption Standard
AODV - Adhoc On Demand Distance Vector
API – Application Programming Interface
ARP - Address Resolution Protocol
ASK - Amplitude Shift Keying
BPSK/QPSK - Binary/Quardrature Phase Shift Keying
BSS/ESS - Basic/Extended Service Set
CCK - Complementary Code Keying
CDMA - Code Division Multiple Access
CGI - Common Gateway Interface
COFDM - Coded Orthogonal Frequency Division Multiplexing
CSMA – CA - Carrier Sense Multiple Access with Collision Avoidance
CVS - Concurrent Version System
DHCP - Dynamic Host Configuration Protocol
DSSS - Direct Sequence Spread Spectrum
EDGE - Enhanced Data for GSM Evolution
EEPROM - Electrically-Erasable Programmable Read-Only Memory
ERB - Estação Rádio-Base
FFD – Full Function Device
FHSS/DSSS - Frequency Hopping/Direct Sequence Spread Spectrum),
FTP – File Transfer Protocol
GFSK - Gaussian Frequency Shift Keying
GGSN – Gateway GPRS Support Node
GPRS - General Packet Radio Service
GPS - Global Position System
GSM - Global System for Mobile Communications
GTP – GPRS Tunneling Protocol
HCI – Host Controller Interface
HLR – Home Location Registers
HSDPA - High-Speed Downlink Packet Access
HTML - HyperText Markup Language
HTTP - Hypertext Transfer Protocol
IDE - Integrated Development Environment
IEEE - Institute of Electrical and Electronics Engineers
IMEI - International Mobile Equipment Identity
LAN - Local Area Network
LDO - Low-Dropout
LM – Link Manager
LMP – Link Manager Protocol
L2CAP – Logical Link Control and Adaptation Protocol
MAC – Media Acces Control
MAN - Metropolitan Area Network
MB-OFDM - Multiband OFDM
MEO – Medium Earth Orbit
M-QAM - M-ary Quadrature Amplitude Modulation
MT – Mobile Terminal

NAT - Network Address Translation
OBEX - Object Exchange
OFDM - Orthogonal Frequency Division Multiplexing
O-QPSK - Offset- Quadrature Phase-Shift Keying
OSI - Open Systems Interconnection
PAN – Personal Area Network
PCI – Placa de Circuito Impresso
PHP - Personal Home Page
PPP – Point-to-Point Protocol
RAM - Read Access Memory
RFCOMM - Radio Frequency Communication
RFD – Reduced Function Device
RTC - Real Time Clock
SDP – Service Discovery Protocol
SGSN – Serving GPRS Support Node
SIM - Subscriber Identity Module
SMD - Superficial Mounting Device
SMS - Short Message Service
SMTP - Simple Mail Transfer Protocol
SPI - Serial Peripheral Interface
SQL - Structured Query Language
SRAM - Static Random Access Memory
TE – Terminal Equipment
TCP/IP - Transmission Control Protocol / Internet Protocol
TCS - Telephony Control Specification
TDMA - Time Division Multiple Access
UMTS - Universal Mobile Telecommunication System
URL - Uniform Resource Locator
USART - Universal Synchronous Asynchronous Receiver Transmitter
USB - Universal Serial Bus
UWB - Ultra Wide Band
WCDMA - Wideband Code Division Multiple Access
WEP - Wired Equivalent Privacy
WiMax - Worldwide Interoperability for Microwave Access
WLAN – Wireless Local Area Network
WPA - Wi-Fi Protected Access
WPAN – Wireless Personal Area Network

CAPÍTULO 1

Introdução

As tecnologias que permitem o monitoramento remoto de pessoas e equipamentos, a qualquer momento e de/em qualquer lugar vêm se tornando um recurso cada vez mais comum e necessário. A difusão maciça da internet possibilitou o acesso a um banco de dados incomensurável com uma rapidez de busca e, sobretudo, uma agilidade na atualização das informações que banalizou o convívio *on line* e em tempo real. Além disso, monitorar a própria casa à distância representa mais do que uma grande comodidade, é sinônimo de segurança.

O sensoriamento, bem como as redes de dados sem fio e a internet, são recursos bem evidentes no ramo tecnológico atual, mas a integração de todos eles ainda requer pesquisa e desenvolvimento. O sensoriamento tradicional é, em geral, estático, o que significa que acompanhar um objeto fora dos domínios de um ambiente monitorado torna-se tarefa mais complexa. Dessa maneira, deixando de lado a discussão sobre ética de privacidade e priorizando um monitoramento 'onipresente', o foco desse trabalho está na implementação de nós que funcionem tanto em uma rede interna (limitada por uma edificação) quanto externa.

O objetivo central é tornar os protótipos desenvolvidos compactos e economicamente viáveis para comercialização. Para tanto, através do Programa RHAE – Inovação (Recursos Humanos em Áreas Estratégicas – Inovação) do Conselho Nacional de Pesquisa (CNPq), a empresa N3E Nova Empresa Equipamentos Eletrônicos, com sede na cidade de São Carlos, obteve apoio financeiro para a pesquisa (iniciação científica e a presente dissertação de mestrado, ambas ligadas à Escola de Engenharia de São Carlos, USP) e para o desenvolvimento de um conjunto de equipamentos eletrônicos destinados à automação residencial e comercial. É de grande relevância, portanto, a observação da pesquisa realizada em empresas incubadas, imbuídas em buscar soluções mercadológicas. A pesquisa em pequenas empresas, com o intuito de desenvolver tecnologias e produtos de forte impacto junto ao consumidor final, ao invés de limitar-se a ramos de pesquisas orientadas apenas à prospecção tecnológica, é uma prática que vem se tornando cada vez mais comum. A parceria empresa/universidade vem, gradualmente, alcançando sucesso considerável no propósito de materializar as idéias da inovação. Dessa maneira, ainda que o mestrado

seja individual, uma equipe auxilia no trabalho, principalmente no que tange às montagens de *hardwares*. Dentro da N3E, um mestrado com o tema ZigBee [1], foi defendido em 2007 por um dos engenheiros da equipe, Ferdinando Mosignore, e norteou o início do projeto. O mestrado atual é uma evolução da pesquisa anterior, muito em razão dos conhecimentos adquiridos ao longo do tempo, conseqüente da maior difusão desse padrão *wireless*, tão pouco evidente a época do primeiro trabalho.

A motivação do trabalho é desenvolver uma rede sem fio aplicando um protocolo ainda pouco explorado[2], conhecido como ZigBee, composta por sensores, atuadores e lâmpadas, além de dotá-la de mobilidade através de controles remotos. Esta rede é integrada, através de um *gateway*, a uma rede TCP/IP (*Transmission Control Protocol/Internet Protocol*) para permitir o monitoramento e a atuação remota sobre ela, via um servidor HTTP (*Hypertext Transfer Protocol*) e/ou uma rede de dados de celular (GPRS, *General Packet Radio Service*), responsável pelo envio de coordenadas GPS na rede externa, garantindo a onipresença do monitoramento. Mais especificamente, o projeto é direcionado à construção de protótipos para monitoramento remoto com visualização do *status* de uma rede interna e atuação na mesma via sites de Internet, extrapolando o monitoramento a um dispositivo que saia dessa rede interna por meio do envio de coordenadas GPS, armazenadas num banco de dados externo que viabilizará sua localização com a ajuda de um servidor de mapas.

De uma visão mais ampla, a arquitetura completa da rede pode ser vista na Figura 1. A rede é composta por 2 servidores. O servidor interno que está localizado dentro as casa ou de um estabelecimento comercial e onde estarão os demais nós ZigBee e um externo que armazenará as coordenadas GPS que serão enviadas por um nó misto, parte ZigBee, parte GPRS, que será chamado ZGPRS. A opção por um servidor externo deu-se por conveniência. A N3E já tinha uma hospedagem contratada junto ao Terra Empresas para alocar a página da empresa com todos os recursos para programação WEB. A hospedagem inclui servidores FTP, SMTP e HTTP, banco de dados, módulo PHP, etc... É preciso deixar claro que nada impede que esse servidor também seja alocado no mesmo ambiente ZigBee, mas não poderia ser o mesmo servidor da rede ZigBee, pois, como será explicado, o servidor interno tem limitações que impediriam a alocação de um banco de dados e de páginas complexas com linguagem PHP. Ambos

[2] - Apesar de já ter sido lançado há algum tempo, os desenvolvedores mostram relutância em substituir tecnologias que vêm dando certo há muitos anos pelo ZigBee. A grande desconfiança está, no entanto, na falta de uma padronização mais coesa de seus protocolos e de rotinas de trocas de mensagens.

servidores poderão ser visualizados de qualquer lugar do mundo por qualquer usuário que detenha um login e uma senha válidos.

Como mencionado, o nó ZGPRS tem um funcionamento diferenciado (misto). Ele pode ser, por exemplo, um controle de portão que, ao detectar que está fora da rede ZigBee, começa a procurar por satélites para que possa ser informado de sua coordenada, recebendo essa posição de tempos em tempos com um intervalo configurável e enviando, via rede de dados celular (GPRS), essa posição ao servidor externo que, por sua vez, atualiza a nova informação em seu banco de dados. Quando o nó ZGPRS reintegra-se a rede ZigBee, ele pára de mandar as coordenadas e passa a funcionar como um outro nó na rede, além de servir como uma interface entre a rede celular e a rede interna. Ou seja, seria possível enviar um SMS, por exemplo, para ser informado do status de um sensor ou ainda configurar esse nó para avisar, via SMS, um determinado celular quando um sensor disparar.

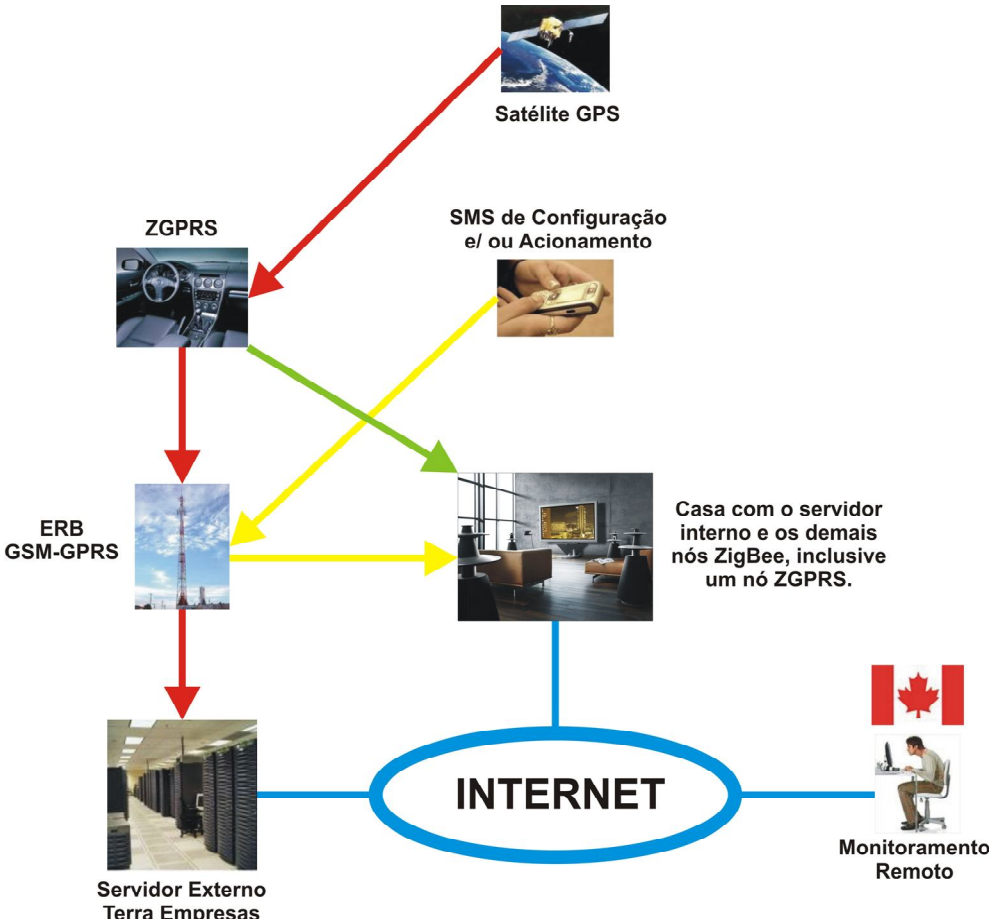


Figura 1 - Monitoramento Onipresente

O hardware envolve microcontroladores, RTC's (*Real Time Clock*), sensores, leitores biométricos, *modems* celulares e GPS. Os diversos softwares estão presentes nos micro-controladores e nas WebPages, escritos em várias linguagens (C, Java, Javascript, PHP, HTML, dentre outras). A forma de endereçamento dos nós da rede implementada é uma alternativa ao método de endereçamento tradicional de redes ZigBee. Um novo procedimento de atribuição de chaves de segurança também foi desenvolvido para suprir a ausência das rotinas de criptografia da pilha ZigBee adotada. E, ainda que a segurança proposta seja mais simplificada que a descrita pela especificação ZigBee, ela resolve um problema de vulnerabilidade presente na concepção original.

A dissertação está estruturada em 6 capítulos. O segundo capítulo apresenta uma revisão das tecnologias e principais conceitos envolvidos, ou seja, redes sem fio, ZigBee, TCP/IP, GPS e GPRS, no enfoque de como o projeto foi concebido para atender aos seus objetivos. O Capítulo 3 descreverá o desenvolvimento do hardware, enquanto o capítulo 4, o desenvolvimento do software. O capítulo 5 apresenta os testes de validação do protótipo e o capítulo 6 as conclusões do trabalho. Ao final, são apresentadas as referências bibliográficas e as publicações geradas.

CAPÍTULO 2

Tecnologias Envolvidas e Servidores de Monitoramento

O monitoramento e o gerenciamento remoto da rede ZigBee, com a proposta adicional do monitoramento de localização via satélite (GPS), utilizando a rede de dados celular e hospedagem de páginas de Internet, extrapola a abordagem dos conceitos de redes sem fio, envolvendo diversas linguagens de programação. Além do enfoque central desta dissertação, dado ao ZigBee por ser este uma peça fundamental na originalidade do trabalho, muitos estudos foram feitos com relação aos periféricos que integram e incrementam a rede e são também apresentados e discutidos ao longo deste capítulo.

2.1. Redes Sem Fio

O panorama das comunicações sem fio (ou *wireless*) é bastante vasto, abrangendo tanto redes domésticas quanto metropolitanas, como ilustrado na Figura 2.

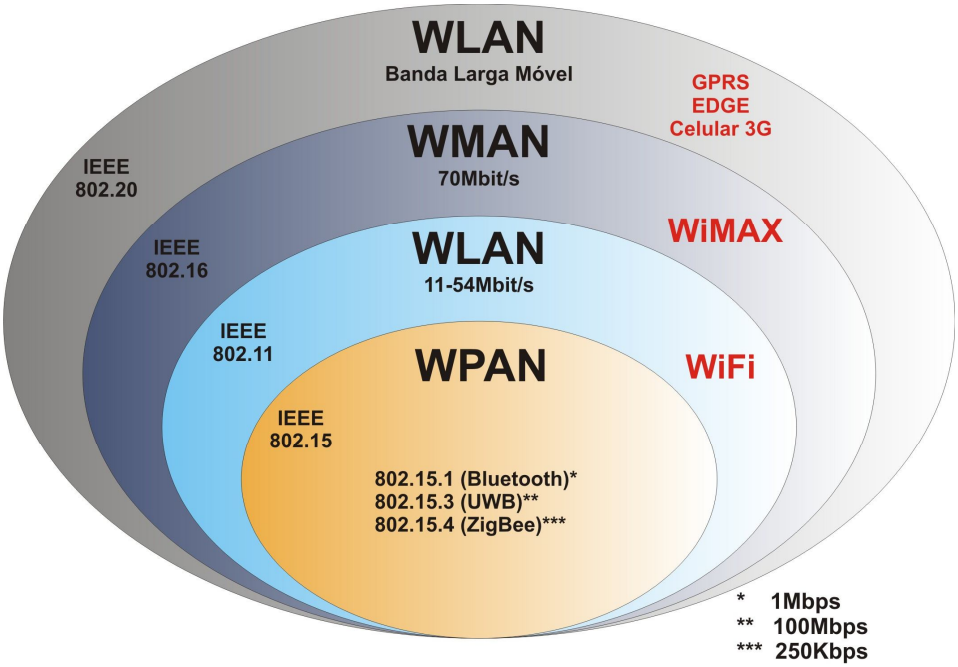


Figura 2 - Padrões Wireless. [2]

No âmbito metropolitano as redes de dados via celular, que já concretizam sua terceira geração[3] com técnicas de acesso, modulação e correção de erros bastante eficientes, conseguem transferências multimídia a taxas de transmissão da ordem de 2 Mb/s [3]. Além do celular, algumas outras tecnologias de redes sem fio, muitas vezes motivadas pela dificuldade da construção de estruturas cabeadas, começam a ganhar corpo. A transmissão fibra-rádio (*fiber-radio*), consistindo numa rede com núcleo (*backbone*) de fibra óptica e acesso via rádio, é uma opção que tem despertado bastante interesse [4]. Nestes sistemas, o *link* de rádio pode usar ondas micrométricas numa tecnologia já padronizada chamada WiMax (*Worldwide Interoperability for Microwave Access*) .

Com relação às redes locais (WLAN's, *Wireless Local Area Networks*) pode-se citar a conhecida rede WiFi, difundida em empresas, residências, cafés (*cybers*), hotéis e aeroportos. Como toda rede sem fio, o WiFi oferece a comodidade da mobilidade e da eliminação dos cabos de conexão. Definido pela norma IEEE 802.11 [6], o WiFi representa um importante avanço nas redes de computadores.

Por fim, as redes pessoais (WPAN's, *Wireless Personal Area Networks*) são as que abrigam as tecnologias Bluetooth e ZigBee. Essas duas soluções *wireless*, apesar de serem definidas pela mesma norma, possuem disparidades significativas, como resumido na Tabela 1 que mostra um comparativo dos vários padrões sem fio. Portanto, ZigBee e Bluetooth ocupam nichos do mercado distintos e não concorrentes.

As redes WPAN estão incluídas na norma IEEE 802.15 [8] e foram criadas, inicialmente, para interconectar dispositivos como *laptops*, fones de ouvido e celulares. Nessa concepção, os dispositivos deveriam estar dispostos num raio bem pequeno (máximo de 10 metros) [7], como indicado na Figura 3.

[3] - Já se encontram em operação tecnologias de quarta geração como o HSDPA (*High-Speed Downlink Packet Access*), que é capaz de transmitir a taxas de 10 Mb/s em uma banda de 5 MHz.

Padrão	Bluetooth	UWB	ZigBee	WiFi
Especificação IEEE	802.15.1	802.15.3	802.15.4	802.11a/b/g
Banda de freqüência	2,4 GHz	3.1-10,6 GHz	868/915 MHz; 2,4 GHz	2,4 GHz; 5 GHz
Taxa de transmissão	1 Mb/s	110 Mb/s – 480Mb/s	250 Kb/s	54 Mb/s
Alcance	10m	10 m	10 - 100 m	100 m
Potência nominal	0 - 10 dBm	-41,3 dBm/MHz	(-25) - 0 dBm	15 - 20 dBm
Número de canais	79	1-15	16 (2,4 GHz)	14 (2,4 GHz)
Largura de banda do canal	1 MHz	500 MHz – 7,5 GHz	0,3/0,6 MHz; 2 MHz	22 MHz
Modulação	GFSK	BPSK, QPSK	BPSK (+ ASK), O-QPSK	BPSK, QPSK COFDM, CCK, M-QAM
Espalhamento	FHSS	DS-UWB, MB-OFDM	DSSS	DSSS, CCK, OFDM
Mecanismo contra interferências	*Frequence Hopping* Adaptativo	*Frequence Hopping* Adaptativo	Seleção dinâmica de freqüência	Seleção dinâmica de freqüência
Célula básica	Piconet	Piconet	Star	BSS
Extensão da célula	Scatternet	*Peer-to-peer*	*Cluster tree, Mesh*	ESS
Número de nós	8	8	> 65000	2007
Encriptação	EQ stream cipher	AES	AES	WEP,AES

Tabela 1 - Comparação entre Zigbee e outras tecnologias de transmissão sem fio de dados [7]

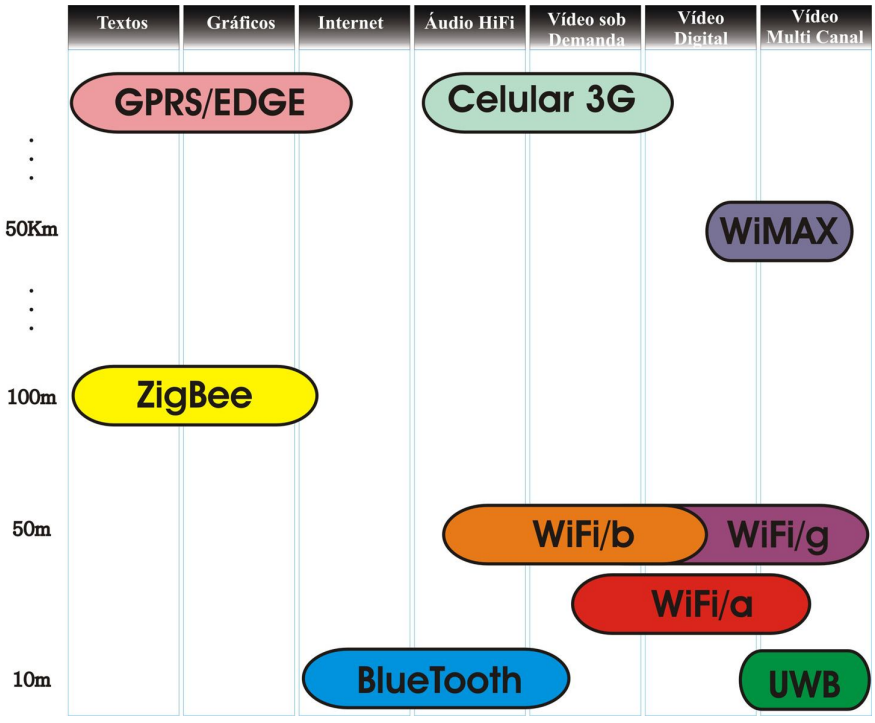


Figura 3 - Transmissão de dados e serviços [9].

2.1.1. Bluetooth

O Bluetooth foi a primeira tecnologia proposta no grupo WPAN. Bem difundido nos dias de hoje, o Bluetooth está presente em diversos aparelhos eletrônicos. Sua taxa de transmissão de 1 Mb/s, no entanto, não permite, por exemplo, transferências de vídeos em tempo real. Para esse propósito, foi criado o UWB (*Ultra Wide Band*). Com taxas da ordem de 480 Mb/s num raio de 3 m, o UWB consegue ser tão rápido quanto o USB 2.0 [7], seu diferencial é a utilização de um espectro de freqüência amplo (500 MHz – 7,5 GHz), com modulação OFDM (*Orthogonal Frequency-Division Multiplexing*) e com portadoras mudando muito rapidamente, deixando-o praticamente imune a qualquer interferência [10].

Retornando ao tópico principal desse item, o Bluetooth é uma tecnologia *ad-hoc* que suporta unicamente a topologia em estrela, com hierarquia mestre-escravo. Sua rede é chamada *piconet* e nela devem estar presentes, obrigatoriamente, um único mestre e no máximo sete escravos. Todo o sincronismo, gerenciamento de endereços e de transmissão e controle de janelas (*slots*) de tempo são feitos pelo mestre. Os escravos só podem encaminhar mensagens ao mestre, sendo proibidas as conexões entre escravos.

Com a permissão de apenas sete dispositivos na rede e com a ausência de protocolos de roteamento, o Bluetooth estaria muito limitado. Para tentar amenizar essa restrição, podem ser criadas *scatternets*, que são a união de duas *piconets* por meio de um nó ponte, podendo tanto ser representado por um escravo quanto por um mestre (Figura 4). Contudo, um nó só pode ser mestre numa única *piconet* e, se o mestre de uma *piconet* for usado como uma ponte para formação de uma *scatternet*, ele deverá, obrigatoriamente, ser um escravo na outra *piconet*. Entretanto, como os escravos são totalmente dependentes de seus mestres e como os nós de uma *piconet*, com exceção do nó ponte, não têm qualquer conhecimento do mestre da outra *piconet*, essa tentativa de roteamento para suprir algoritmos mais complexos é extremamente limitada.

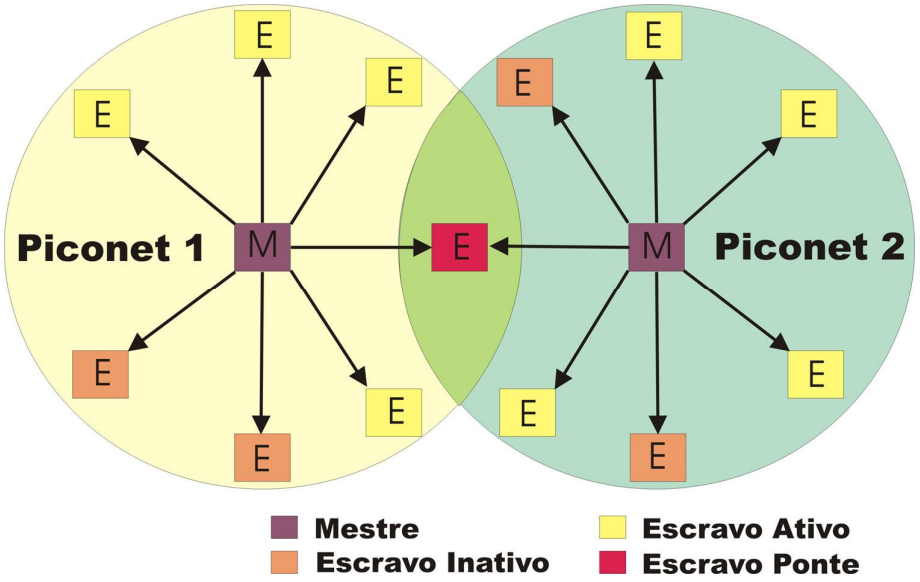


Figura 4 - Scatternet Bluetooth [12].

O Bluetooth, definido como uma arquitetura de camadas de protocolo, pode ser estruturado como indicado na Figura 5. Um grupo de protocolos de *middleware* permite que aplicações já existentes e novas aplicações operem sobre links Bluetooth. Os protocolos de padrões industriais incluem *Point-to-Point Protocol* (PPP), *Internet Protocol* (IP), *Trasmission Control Protocol* (TCP), *Wireless Application Protocol* (WAP), dentre outros. O *RFcomm*, por exemplo, é um protocolo criado pelo consórcio Bluetooth para imitar uma porta serial padrão, permitindo a conexão de periféricos como teclados e mouses.

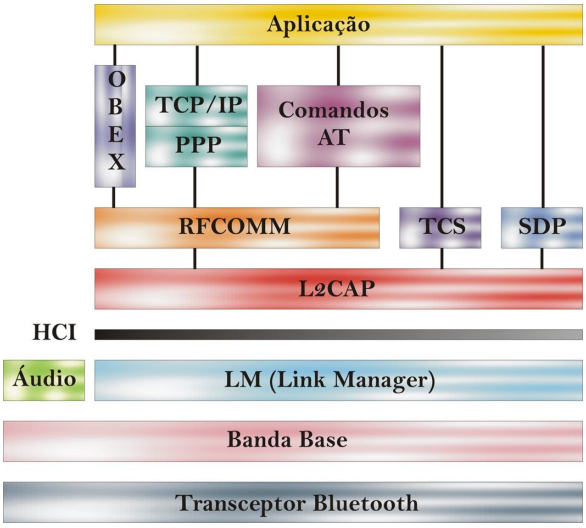


Figura 5 - Camadas Bluetooth [13] (as siglas são apresentadas no glossário).

A camada física do Bluetooth utiliza *frequency hoping* como técnica de codificação. São 79 canais comutados continuamente por um mecanismo de saltos de freqüência. Isso significa que, a cada *slot* de tempo, a freqüência do canal é trocada (salto em freqüência) segundo um padrão conhecido pelo transmissor e pelo receptor, sendo uma técnica bastante eficiente contra interferências e ruídos.

A camada banda-base define como os dispositivos Bluetooth localizam outros dispositivos e como se conectam aos mesmos. É nesta camada também onde se definem estratégias de detecção de erros, criptografia, transmissão e retransmissão de pacotes. Esta camada suporta dois tipos de enlaces: *Synchronous Connection-Oriented* (SCO) e *Assynchronous Connection-Less* (ACL).

Enlaces SCO são caracterizados pela atribuição periódica de um *slot* de tempo a um dispositivo e é utilizado, basicamente, na transmissão de voz, que requer transmissões de dados rápidas e consistentes. Um dispositivo que estabeleceu um link SCO possui, em essência, determinados *slots* de tempo reservados para seu uso. Seus pacotes são tratados como prioritários e serão processados antes de pacotes ACL.

Já um dispositivo que opera sobre um link ACL pode enviar pacotes de tamanho variável em intervalos irregulares. Entretanto, este tipo de link não possui reserva de *slots* de tempo para seus pacotes.

O protocolo de gerência de enlace, LMP (*Link Manager Protocol*), administra a alocação da taxa de transferência de dados, taxa de transferência de áudio, autenticação através de métodos de desafio-resposta, níveis de confiança entre dispositivos, criptografia de dados e controle do gasto de energia.

A camada L2CAP (*Logical Link Control and Adaptation Protocol*) serve de interface entre os protocolos de camadas superiores e os protocolos de transporte de camadas inferiores. Esta camada também é responsável pela fragmentação e remontagem de pacotes [14].

2.1.2. ZigBee

Nenhuma das tecnologias *wireless* anteriores ao Zigbee era dedicada a operar a taxas de bits típicas para dispositivos simples como sensores, lâmpadas e outros equipamentos para automação industrial e residencial.

Um problema na concepção do Bluetooth e do UWB é a ausência de rotinas de roteamento. Suas redes só suportam hierarquia mestre/escravo e topologia em estrela.

Assim, a pilha ZigBee foi construída com protocolos de roteamento de ordem superior, responsáveis pela expansão da rede, aceitando topologia, estrela, malha (*mesh*) e árvore.

O ZigBee, definido pela norma IEEE 802.15.4, surgiu para preencher uma lacuna deixada pelas redes sem fio. Observando as redes sem fio existentes, que incluem WiFi, WiMax, Bluetooth, Celular (GPRS – EDGE), percebe-se que nenhuma delas é dedicada a uma rede de equipamentos simples que não requisite grande complexidade e onde a economia de baterias e o baixo custo sejam essenciais, como ocorre numa rede de sensores, iluminação, refrigeração e afins.

Consumo (baterias), custo e complexidade reduzidos são as prioridades do padrão ZigBee. Esses princípios guiaram o seu desenvolvimento. Para efeito de comparação, enquanto um nó Bluetooth, alimentado por bateria, funciona por cerca de 1 semana, um nó ZigBee, alimentado pela mesma bateria, funciona por cerca de 1 ano. Isso é conseqüência do longo tempo em que os dispositivos mais simples ficam em modo *sleep*[4], acordando de tempos em tempos e perguntando por mensagens que lhe foram enviadas enquanto estavam dormindo.

É considerado um padrão "aberto" por deixar disponíveis suas especificações e documentos, entretanto, para poder usar o logo ZigBee é necessário ser parte da Aliança ZigBee e pagar uma taxa de $3500 por ano, além de submeter todos os seus produtos ZigBee a testes, igualmente pagos, para homologação [15]. Esse é, sem dúvida, um dos principais empecilhos na aceitação mercadológica do ZigBee.

2.1.2.1. A pilha ZigBee

A estrutura ZigBee, dividida em camadas, tomando o modelo OSI por referência, pode ser vista na Figura 6. As camadas superiores são de responsabilidade do programador e determinam a singularidade do seu produto. O ZigBee é definido nas camadas intermediárias. Elas são ocupadas pela pilha de algum membro da *ZigBee Alliance*. Aí estão inclusos os algoritmos de roteamento, as regras para formação e integração da rede, dentre outros. A pilha escolhida para o projeto foi construída pela Microchip [16]. Por fim, a camada inferior é de responsabilidade do IEEE.

[4] - Os nós ZigBee, chamados RFD, para economia de bateria, estão em constante estado de *sleep*. Em termos de software, isso significa que dentro do *loop* principal, depois de verificar os estados de seus indicadores de atuação (*flags*) e constatar que todas as suas tarefas estão cumpridas, o dispositivo entra num modo de contenção de corrente, desligando o transceptor e periféricos adicionais. Quando retorna ao seu estado ativo, ele requisita sincronismo ao seu pai e recebe mensagens que lhe foram enviadas nesse meio tempo.

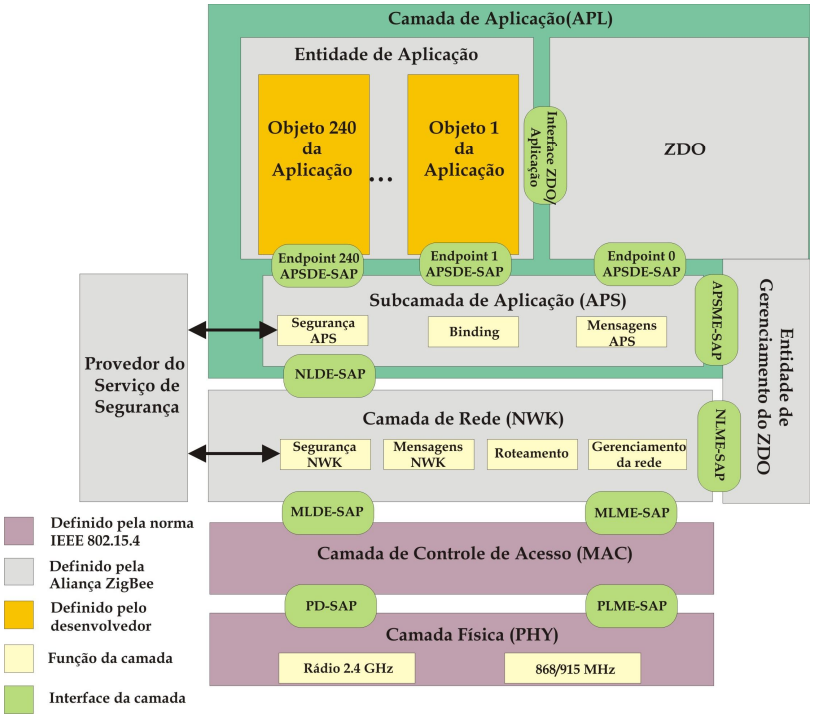


Figura 6 - Arquitetura da pilha ZigBee [17].

A camada física está ligada à ativação e desativação do transceptor e suas respectivas funções, como detecção do nível de energia dos canais, qualidade de cada *link* e seleção dos canais, além de transmitir e receber pacotes pelo meio físico. Utilizam-se três freqüências não licenciadas para sua operação, sendo a de 2,4 GHz para uso global, trabalhando em 250 kb/s. As demais freqüências são usadas para aplicações muito específicas na Europa e EUA e possuem taxas de bits inferiores. Todas elas utilizam o DSSS (*Direct Sequency Spread Spectrum*), mas diferem no tipo de modulação e na taxa de *chips*[5].

A camada MAC (*Media Access Control*) é responsável pelo acesso ao meio. Utiliza-se, como no WiFi, o CSMA – CA (*Carrier Sense Multiple Access/Colision Avoidance*), com a opção de usar ou não *beacons* que são sinais de sinalização, enviados pelo coordenador da rede. Isso é necessário em uma rede onde os nós entram em modo *sleep*. Quando acordam precisam ouvir o canal para serem sincronizados.

A escolha do *beacon* implica na utilização de CSMA-CA com *slot*. Depois que o nó é sincronizado, ele 'sabe', exatamente, o tempo em que pode iniciar a transmissão, sendo avisado, junto com o quadro de *beacon*, se já existe algum nó transmitindo e quanto tempo a transmissão irá durar. Caso se opte pelo modo *non-beacon*, utiliza-se o

[5] *Chip* é um tipo especial de pulso presente no DSSS; mega chips por segundo é a medida da velocidade com a qual os chips são gerados nos sinais DSSS.

CSMA-CA convencional, no entanto, os nós que acordam, por não estarem sincronizados, precisam saber quando poderão transmitir. Isso é feito através de um processo de *polling*[6] ao coordenador, que consiste numa consulta pela autorização de comunicação. Os nós requisitam ao coordenador um *slot* para transmitir algum dado ou para receber algum pacote que lhe foi enviado enquanto dormia, aguardando a disponibilidade e obtendo assim, um sincronismo temporário [18].

2.1.2.2. Quadros

Existem vários tipos de quadros (*frames)*. O quadro da Figura 7 é um modelo 'genérico'. O cabeçalho da camada física está relacionado ao sincronismo, tendo funcionalidades iguais a outros protocolos do gênero que podem ser encontrados em [12]. O que diz respeito à peculiaridade do ZigBee é o cabeçalho da camada MAC, como explicado a seguir [17].

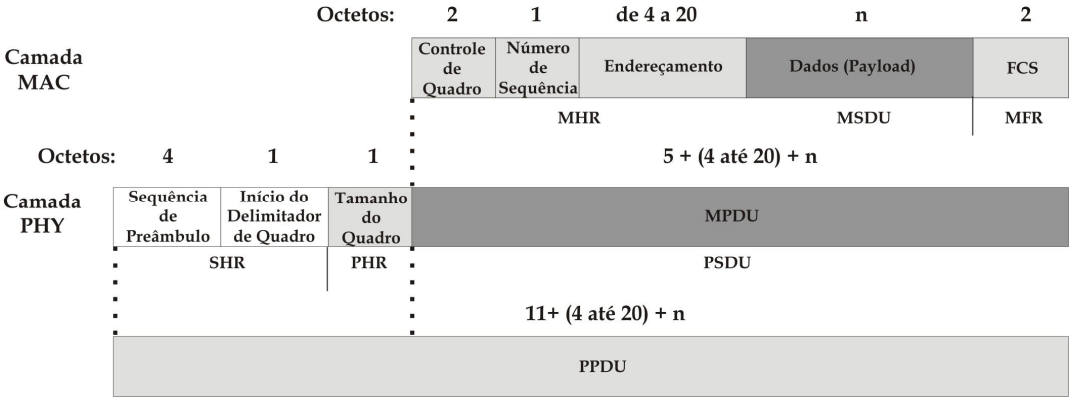


Figura 7 - Estrutura do quadro de dados ZigBee [17].

O Controle de Quadro (*Frame Control)* define diversas informações que serão usadas no nó destino, como: se o nó que originou o quadro faz parte da mesma rede, o tipo de nó de que se trata (FFD ou RFD, *Full Function Device* e *Reduced Function Device*, respectivamente), o tipo de quadro, i.e., quadro de dados, quadro de *beacon* ou quadro de comando (nesse caso é um quadro de comando). Essas são informações que farão o nó destino tomar as decisões corretas em seu software.

O Número de Seqüência (*Sequence Number)* é usado para o controle de dados faltantes, muito comum na transmissão por pacotes da internet.

---

[6] - O *Pooling* é um protocolo para evitar colisões na transmissão. O nó que entra em modo *sleep*, perde o sincronismo da rede. Para que possa transmitir e/ou receber mensagens ao acordar, é feita uma notificação de seu novo estado. Essa notificação é entendida pelo seu pai como uma reivindicação de um *slot* de transmissão. É o pai, portanto, que cuidará para que cada um de seus filhos transmita e receba mensagens num *slot* de tempo distinto, evitando colisões.

O campo de endereço tem um tamanho bem flexível, pois, dependendo do tipo de requisição, é preciso enviar tanto o endereço do nó na rede, chamado *short address*, quanto o seu endereço de placa, o *MAC address*, além do endereço da rede de que faz parte, o *PAN Address*.

O campo de comando é a requisição ou uma resposta feita: se o nó fonte está requisitando uma associação a uma determinada rede ou respondendo a uma requisição que o nó destino já tinha feito.

A carga do comando está relacionada à hierarquia de endereços, que será explicada mais adiante. Dentro de um mesmo nó, podem existir entidades menores, como dois botões que utilizam o mesmo rádio.

O FCS (*Frame Check Sequence*) refere-se aos caracteres *checksum* adicionados ao quadro para detecção e correção de erro [12].

2.1.2.3. Tipos de dispositivos e topologias

Existem dois tipos de dispositivos no ZigBee: FFD (Full Function Device), dotado de funcionalidades plenas, tipicamente alimentado pela rede de energia e o RFD (Reduced Function Device), dotado de funcionalidades específicas e limitadas, alimentado por bateria. Na compilação, certas habilidades são dadas a nós específicos. O FFD pode ser: (i) coordenador, um por rede, responsável pelo gerenciamento da rede; (ii) roteador, responsável pela expansão da rede.

RFDs são *End Devices*, que não podem servir de ponte entre dois outros nós, em outras palavras, não têm capacidade de roteamento.

Quanto à topologia, em conseqüência de sua capacidade de roteamento, existe grande flexibilidade implementada na pilha ZigBee, ao contrário do que acontece com o Bluetooth. Entre as opções, a topologia em estrela é a mais simples. Todos os nós periféricos estão ligados ao coordenador e o caminho das mensagens é único.

A topologia em árvore já tem habilidade de roteamento, mas os roteadores só podem estar ligados ao coordenador, sendo proibidos *links* com outros roteadores. A topologia em árvore já permite uma expansão melhor que a topologia em estrela, mas se um *link* for quebrado a transferência de dados é cessada. Esse problema é resolvido na topologia em malha. As topologias são descritas na Figura 8.

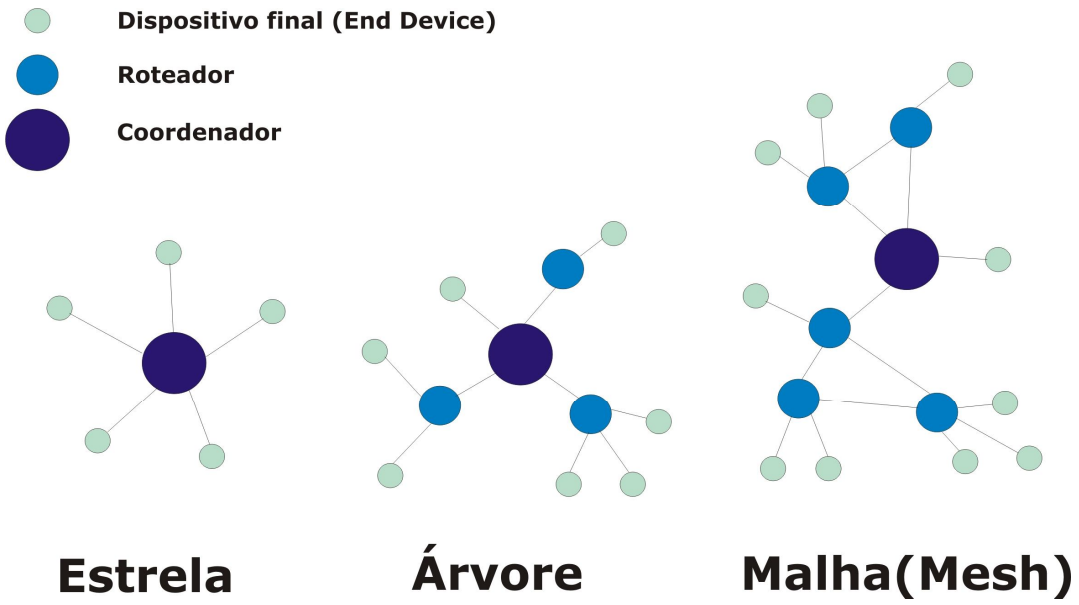


Figura 8 - Topologias ZigBee [19].

2.1.2.4. Hierarquia de endereços

Para relacionar os dados a diversas partições de um mesmo rádio, existe uma hierarquia de endereços, como é comum no serviço de correio (rua, cidade, bairro, etc...). O endereço mais específico é o atributo. Atributo é a funcionalidade: o interruptor tem a funcionalidade de acender ou apagar uma lâmpada. A lâmpada, por sua vez, tem a funcionalidade de estar acesa ou apagada. Atributos estão englobados em *clusters*. Na Figura 9, a lâmpada só tem *cluster* de entrada, mas o *switch* tem tanto *cluster* de entrada quanto de saída. O *cluster* de entrada nessa figura será usado para estabelecer seu modo de funcionamento (acendimento automático, por exemplo). *Clusters* estão englobados em *endpoints*. Eles descrevem uma unidade física, como um contato de interruptor ou uma lâmpada. Nesse caso só temos um *endpoint* por rádio, mas seria possível ter dois interruptores numa mesma caixa. Cada nó tem seu próprio rádio, que descreve o endereço mais genérico no ZigBee.

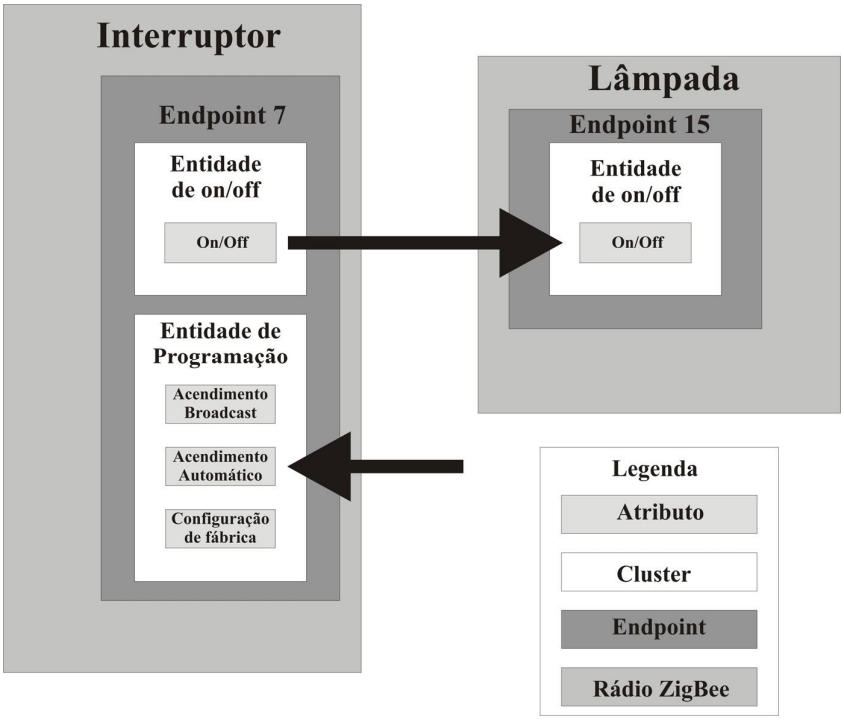


Figura 9 - Hierarquia de Endereços [18]**.**

2.1.2.5. Tipos de Mensagens

Na pilha ZigBee existem algumas variantes nos tipos de mensagens:

a) Broadcast (ponto-multiponto): Todos os nós da rede recebem a mesma mensagem. Este tipo de mensagem é usado em operações como a inicialização da rede pelo coordenador e a descoberta da melhor rota até o nó destino.

b) Unicast (ponto-a-ponto)*:* apenas o nó destino recebe essas mensagens, os demais nós, ao perceberem que não são o destino daquela mensagem podem roteá-la se conhecerem o destinatário. As mensagens *unicast* podem ser:

b.1) Mensagens diretas : Os cabeçalhos devem conter todas as informações de endereços, tanto da fonte quanto do destino. Também devem ter no cabeçalho a rota a seguir, obedecendo a uma tabela de roteamento, além de um campo que é atualizado a cada nó intermediário para atualizar o número de roteadores por que a mensagem passou até chegar ao destino.

b.2) Mensagens indiretas: nesse tipo de mensagem o nó fonte manda a mensagem ao coordenador, que armazena uma tabela com as informações de endereço desse nó associadas às informações de endereço do destino, salvas na mesma tabela. Assim, os nós fonte e destino não têm qualquer informação um sobre o outro. Para *end devices*, nós extremamente simples, esse tipo de mensagem é melhor do que a mensagem direta,

pois eles podem economizar energia e memória quando evitam cabeçalhos muito longos. Esse tipo de mensagem é conseguido através da operação de *binding*, explicada na próxima seção.

2.1.2.6. *Binding*

O esquema do *binding* pode ser entendido a partir dos nós indicados na Figura 10. Num mesmo nó existem dois interruptores. No outro nó, estão dispostas quatro lâmpadas. Um interruptor manda uma diretiva ao coordenador, requisitando um *binding*. Isso pode ser feito ao se pressionar um botão, por exemplo. Essa diretiva chega ao coordenador, que começa a construir uma tabela com as informações passadas pelo primeiro nó, aguardando uma segunda requisição de *binding* que deve vir da lâmpada. Assim, o *cluster* de saída do interruptor fica amarrado ao *cluster* de entrada da lâmpada. Nesse caso, um interruptor está amarrado a três lâmpadas. Isso significa que o acionamento do *switch* 1 atuará em três lâmpadas.

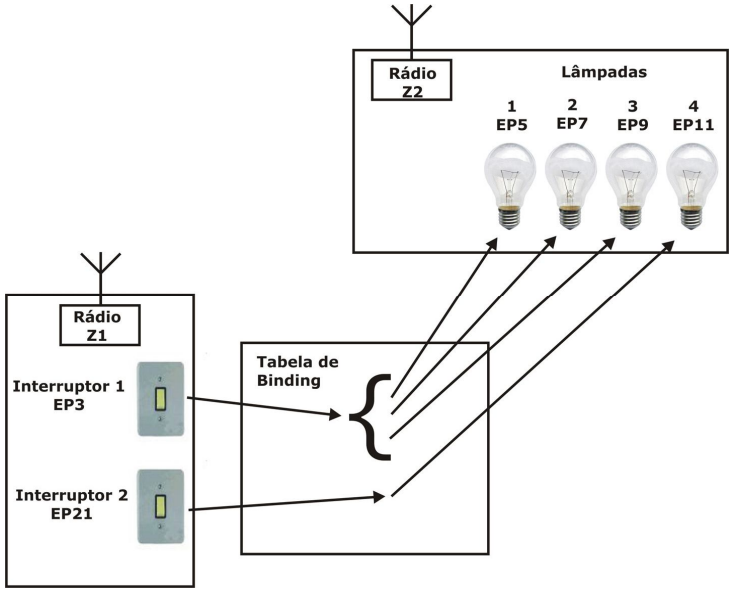


Figura 10 – *Binding* [17**].**

2.1.2.7. Roteamento

Todos os nós têm uma tabela de vizinhança com os endereços de seus nós vizinhos e sua relação “familiar” com os mesmos: pai ou irmãos. A escolha dos pais é feita por uma rotina para a descoberta da melhor rota, sendo a melhor rota aquela com o menor número de saltos (*hops*) até o coordenador, o chamado AODV (*Adhoc On Demand Distance Vector*). E isso é similar no roteamento. Se o nó fonte não tem

qualquer informação de roteamento até o nó destino, ele envia uma mensagem em *broadcast* com o endereço do nó destino. Todos os nós ao alcance do nó fonte recebem essa mensagem. Caso tenham a informação do caminho até o destino em sua tabela de roteamento, respondem ao nó fonte com o custo até o mesmo (número de *hops*), e o nó fonte pode escolher aquela rota com o menor custo, caso não a tenham, dão um encaminhamento (*forward*) com uma mensagem em *broadcast* incrementando o campo de número de saltos. Também há a capacidade de atualizar a tabela de roteamento. Se a mensagem é marcada com *ack* (*acknowledgment*) e ela chega por uma rota diferente daquela salva em sua tabela com um número de hops menor do que o contido na mesma, pode ser feita uma atualização (*update*).

2.2. Servidores de Monitoramento

São diversas as aplicações baseadas no padrão TCP/IP, tais como servidores HTTP, softwares de supervisão e programas p2p (*peer to peer*) [20]. O modelo TCP/IP é muito bem difundido e padronizado de forma que diversas pilhas comerciais podem ser encontradas à disposição de qualquer usuário desenvolvedor de aplicação. A pilha escolhida para o servidor da rede interna foi desenvolvida pela Microchip (disponível para download em www.microchip.com).

Como as regras, os protocolos, a negociação e a conexão (i.e., o perfil de transferência) já são padronizados, não há novidade presente nessa pilha. O monitoramento proposto no trabalho é, estritamente, embasado em aplicações da Internet (Web). Um usuário deverá digitar em seu 'navegador' (*browser*) de preferência: o endereço da Web Page que alocará as informações da rede interna, no caso do ZigBee, e as posições de satélite, no caso do deslocamento externo, com os dados do GPS sendo enviados para uma página Web, via GPRS.

A intenção é desenvolver uma aplicação de um servidor HTTP hospedeiro de uma página de monitoramento e atuação, conciliado a um servidor FTP para transferência de arquivos relevantes que possam viabilizar a visualização indiscriminada em qualquer parte do mundo. Nesse servidor da rede interna está uma inovação significativa, explicada a seguir.

2.2.1. Servidor Interno

A pilha é escrita em C e toma por base o modelo de camadas TCP/IP, ainda que com algumas ressalvas para que o software funcione melhor, como o acesso a camadas inferiores. Camadas superiores acessam uma ou mais camadas que não estão diretamente abaixo de si. Cada camada é implementada por um arquivo fonte separado (arquivo.c). O modelo de camadas da pilha da Microchip é mostrado na Figura 11.

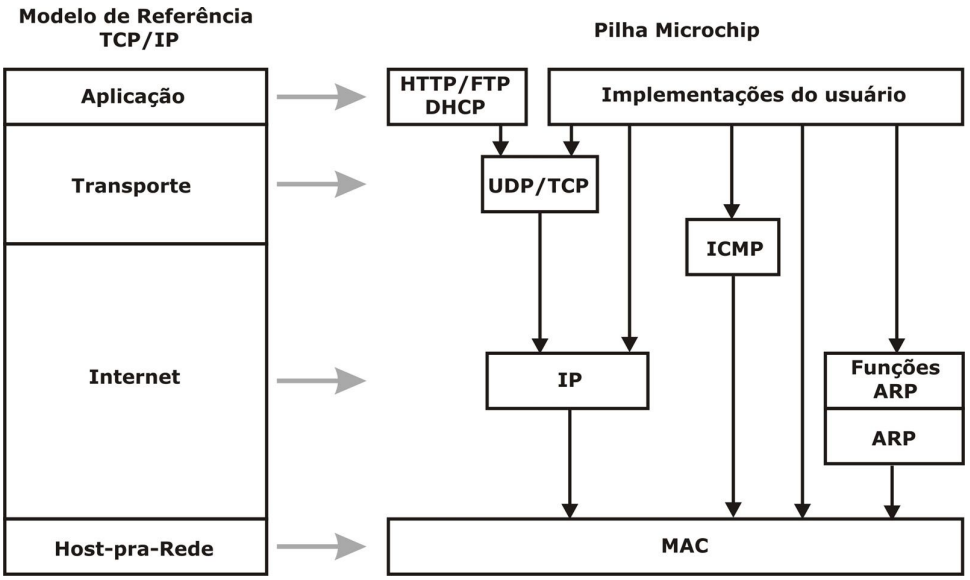


Figura 11 - Modelo de Referência TCP_IP x Pilha TCP_IP da Microchip [21**].**

A grande inovação é alocar um servidor HTTP dentro de um *chip*, dispensando o uso de um computador dedicado a essa funcionalidade. Com isso, será possível construir um único nó, dotado de funcionalidades ZigBee (por conveniência, um coordenador, já que é quem gerencia todo o tráfego de mensagens) e TCP/IP funcionando como um servidor HTTP. Desta forma, o servidor constitui-se num *gateway* entre a rede mundial de computadores e a rede ZigBee. Isso tudo pode ser realizado num espaço físico reduzido e a um custo inferior ao de disponibilizar um computador exclusivo para funcionar como servidor.

Notar que a hospedagem da página será limitada à memória do módulo, inviabilizando páginas complexas com linguagens SQL e PHP, além de figuras e afins muito ‘pesados’, mas hospedando páginas funcionais, tendo a dimensão física de seu hardware compensando as limitações gráficas de suas páginas.

2.2.1.1. As camadas [12]

A. Media Access Control (MAC)

Essa camada se vale de um controlador de interface de rede (NIC), que implementa tanto a camada física (PHY) quanto a camada de acesso (MAC). A pilha utiliza uma memória RAM estática (SRAM), presente nos NIC's para servir como *buffer* na transmissão, até que uma camada superior leia os quadros e os processe ou, alternativamente, os retransmita, caso uma confirmação não chegue dentro de um tempo pré-determinado. O usuário pode definir o tamanho do *buffer* e da lista de quadros armazenados usando *defines.*

A camada SLIP (*Serial Lines Internet Protocols*) é uma opção simplificada da camada MAC, usando, ao invés de cabos Ethernet, cabos seriais em conexões ponto a ponto, onde um computador funciona como um cliente. Interrupções do tipo USART são usadas para detectar tráfego de dados.

B. *Address Resolution Protocol (ARP)*

Diz respeito ao protocolo ARP tradicional. O ARP é uma forma simples de descobrir o endereço MAC de um determinado hospedeiro (*host*) a partir do seu endereço IP. Sem ele, diversos serviços, incluindo o DHCP (*Dynamic Host Configuration Protocol*), não funcionariam.

A camada ARP da pilha é implementada pelos módulos ARP, que criam as diretivas responsáveis pelas funcionalidades do protocolo, e ARPTask, que utiliza as diretivas e responde pelos serviços intrínsecos ao ARP.

C. Internet Protocol (IP)

É o protocolo responsável pelo endereçamento de todos os *host*'s da rede.

Na pilha, as camadas superiores usam as diretivas da "camada IP", requisitando os pacotes IP, interpretando-os e tomando a ação correspondente. A especificação IP determina que o *host* transmissor gere um único identificador para cada pacote transmitido e isso permite que o *host* remoto descarte pacotes duplicados.

D. Internet Control Message Protocol (ICMP)

É o protocolo usado para gerar relatórios de erros aos *hosts* transmissores de pacotes IP, servindo para correção e adaptação. Assim como a "camada IP", a camada

ICMP é passiva, sendo a ação tomada pelas camadas superiores através da chamada das diretivas ICMP.

Quando habilitado, o ICMP pode responder a pacotes "*ping*", permitindo a *hosts* remotos identificarem a presença de seu *host* alvo.

E. Transmission Control Protocol (TCP)

É o protocolo fundamental na transmissão. É ele o responsável pelo estabelecimento e término da conexão, controle de fluxo e de erro e a transmissão, propriamente dita, dos dados.

A pilha disponibiliza *sockets* TCP que sustentarão a conexão dos usuários ao servidor HTTP da aplicação. A limitação do número de *sockets* para conexões simultâneas estará ligada à memória disponível.

Ao contrário das implementações TCP/IP habituais nos servidores, todos os *sockets* na pilha compartilham um único *buffer* de transmissão devido às limitações de RAM do hardware, criando um problema evidente quando um único *socket* reservar toda a disponibilidade do *buffer* para si. Sendo assim, as aplicações desenvolvidas sobre essa camada devem ser capazes de gerenciar as conexões simultâneas.

F. User Datagram Protocol (UDP)

O UDP, como o TCP, está relacionado à transmissão, no entanto é dedicado a tráfego de dados que exigem tempo real, como áudios e vídeos, onde a confirmação de pacotes é menos importante que o fluxo dos mesmos. Um pacote perdido num vídeo não tem razão de ser retransmitido, pois sua relevância já se perdeu com o transcorrer da cena. Como na proposta deste trabalho será usado apenas o servidor HTTP, o UDP não tem relevância significativa.

G. Dynamic Host Configuration Protocol (DHCP)

É o protocolo utilizado para adquirir um endereço IP e uma máscara de rede dinamicamente (de maneira automática) a partir de um servidor DHCP. Portanto depois de ser conectado a um *gateway* (um roteador, por exemplo) que funcione como um servidor DHCP, o hardware, contendo a pilha, será capaz de requisitar e adotar o endereço fornecido pelo mesmo.

Também é possível optar por um endereço estático. A escolha deve ser disponibilizada na aplicação da maneira mais conveniente.

Os servidores HTTP e FTP são parte da aplicação a ser desenvolvida. A pilha descrita oferece alguns exemplos desses servidores e eles devem ser alterados de acordo com a necessidade da aplicação. Todas as diretivas da aplicação, criadas nos diferentes arquivos de HTTP, FTP e DHCP, são utilizadas pela camada *stack task*, que realiza as ações necessárias.

2.2.2. Servidor Externo

O servidor externo será dedicado às informações de coordenadas GPS enviadas pela rede de dados GPRS. Notar que o servidor é externo por questão de conveniência, pois a N3E já possui um domínio registrado (www.n3e.com.br) que facilita a alocação de suas páginas de monitoramento. Assim, seria possível alocá-las num servidor interno, desde que se tome o cuidado de reservar uma porta diferente do servidor responsável pela rede ZigBee. A recíproca, porém, não é verdadeira. Não poderíamos colocar a página de monitoramento da rede ZigBee num servidor externo, uma vez que temos a dependência do hardware para armazená-la. Além disso, o hardware ZigBee/TCP_IP é o referencial que tomamos para discernir um servidor interno do externo.

O servidor das páginas de coordenadas, por sua vez, necessita dos requisitos básicos de um servidor convencional: banco de dados, módulo PHP, servidores FTP, HTTP e SMTP. Todos os aplicativos podem ser encontrados para *download* na rede WEB e instalados numa máquina que tenha os recursos necessários para operá-los. Para um monitoramento sem “janelas”, teríamos que disponibilizar uma máquina confiável que ficasse operando continuamente. Em função desta característica e também pela facilidade do domínio já registrado, optou-se pelo servidor externo em detrimento do interno.

O provedor de serviços da N3E é o Terra Empresas (Figura 12), que disponibiliza um controle de acesso UNIX funcionando como um painel de controle remoto. Dentro desse painel de controle é possível atribuir permissões, gerenciar arquivos e o banco de dados, configurado através de uma ferramenta bastante amigável chamada PHPMyAdmin [22]. Existem diversas opções de banco de dados (BD): Oracle, Access, DB2, Firebird, SQL Server, Postgree, dentre outros. Alguns deles são gratuitos como o mais conhecido BD, o “MySql”, justamente o que é usado pelo Terra Empresas e que será descrito um pouco mais à frente.

Figura 12 - Controle de acesso UNIX do provedor Terra Empresas

A hospedagem do Terra Empresas inclui, como todo provedor de domínios, servidores SMTP (*Single Mail Transfer Protocol*) e FTP. O servidor FTP possibilitará a transferência dos códigos fonte das páginas aos diretórios do servidor HTTP. A Figura 13 ilustra o painel de controle UNIX do "Provedor Terra Empresas".

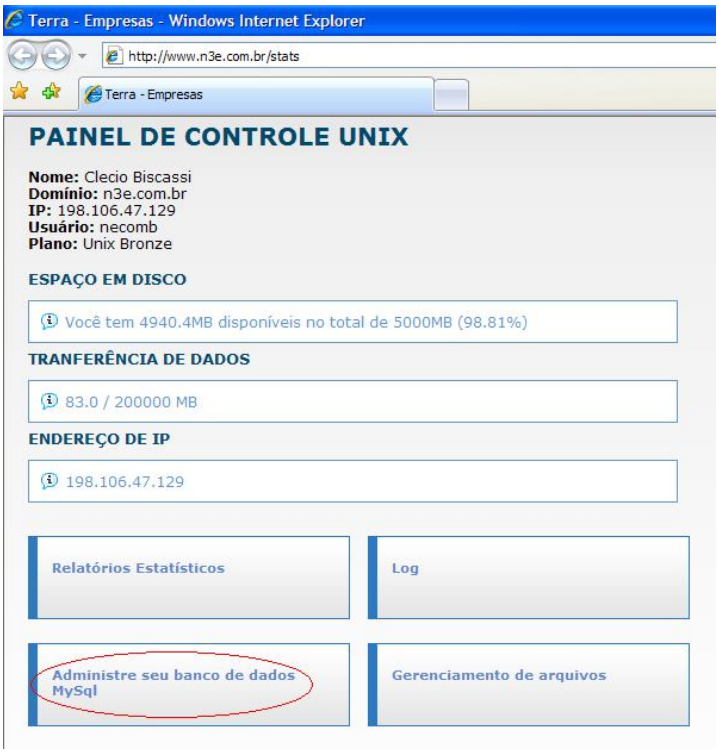


Figura 13 - Painel de controle UNIX do provedor Terra Empresas

2.2.2.1. PHP [23]

Originalmente chamado de *Personal Home Page Tools* e depois, por razões de facilidade de distribuição, rebatizado de PHP, é uma linguagem destinada a sítios da Web (*WEB sites*) dinâmicos, interagindo com os usuários através de variáveis passadas pela URL (*Uniform Resource Locator*) ou por formulários. Ao contrário do Javascript, o PHP é executado diretamente no servidor, sendo enviado para o cliente apenas HTML puro. Desta maneira, permite a interação com bancos de dados e aplicações existentes no servidor com a vantagem de não expor o código fonte para o cliente. De fato, a maior parte do que o PHP realiza é invisível ao usuário final, tendo pouquíssima influência sobre qualquer coisa relacionada à aparência de uma página Web. Alguém visualizando uma página PHP não será capaz de dizer que não foi escrita em HTML porque o resultado final do PHP é HTML.

PHP tem como uma das características mais importantes o suporte a um grande número de bancos de dados, como Interbase, MySQL, Oracle, Sybase, PostgreSQL e vários outros. Além disso, o que pode ser feito por algum programa CGI também pode ser feito com PHP, como coletar dados de um formulário, gerar páginas dinamicamente ou enviar e receber *cookies*.

2.2.2.2. MySQL [24]

É um sistema de gerenciamento de banco de dados que utiliza a linguagem SQL (*Structured Query Language* - Linguagem de Consulta Estruturada). O agrupamento de informações inter-relacionadas, dispostas em linhas e colunas, constituindo uma planilha estruturada, é um banco de dados. Um banco de dados abrange desde informações simples, por exemplo, uma lista de festa de aniversário, até tabelas gigantescas de clientes corporativos.

Para adicionar, acessar e processar dados armazenados em um banco de dados de um computador, é necessário um sistema de gerenciamento de bancos de dados como o Servidor MySQL que auxilia a máquina onde está hospedado a manipular o banco que detém.

Originalmente desenvolvido pela empresa sueca TCX, para trabalhar, de maneira rápida, com grandes escalas de dados e sem a necessidade de hardwares de custos inviáveis, o MySql é definido como um servidor de banco de dados capaz de realizar tarefas simultâneas a usuários distintos, valendo-se da linguagem SQL. O SQL foi desenvolvido pelo Departamento de Pesquisas da IBM, estabelecendo-se como a

linguagem padrão de Banco de Dados Relacional. Trata-se de uma linguagem simples, de fácil interpretação, bastante intuitiva nas operações de gravação e leitura das tabelas. Suas principais características são:

- Suporta diferentes plataformas, tais como Win32, Linux, FreeBSD e Unix.
- Suporta as API's (*Application Programming Interfaces*) de linguagens como PHP, Perl, C, C++, Java e Pynton.
- Suporta múltiplos processadores.
- Possui um sofisticado sistema de senhas criptografadas flexível e seguro.
- Código fonte escrito em C e C++ e testado com uma variedade de diferentes compiladores.
- O cliente pode conectar o MySQL através de conexões TCP/IP

2.2.2.3. PHPMyAdmin [22]

É um projeto de código aberto em PHP para administrar a base de dados MySQL através de uma interface web.

O PHPMyAdmin é um programa de distribuição livre em PHP, criado por uma comunidade sem fins lucrativos. É uma ferramenta bastante completa que permite acessar todas as funções típicas da base de dados MySQL através de uma interface WEB muito intuitiva.

Funciona como um software para a manipulação do banco de dados. É uma forma versátil e poderosa de se acessar bancos de dados remotos, disponibilizando todas as funções que teriam que ser digitadas em linhas, em algum *prompt* de comandos em botões e formulários.

A aplicação em si não é mais do que um conjunto de arquivos escritos em PHP. A ferramenta permite a criação de tabelas, a inserção de dados nas tabelas existentes, a navegação pelos registros das mesmas, sua edição e remoção, a execução de sentenças SQL e a realização de um *backup* da base de dados. A Figura 14 ilustra uma página de phpMyAdm para a administração do banco de dados remoto. Nessa página estão os campos da tabela XT65, criada para armazenamento das coordenadas GPS. Essa tabela está salva no banco de dados *necomb*, comum a todas as outras tabelas usadas na construção do site da N3E.

A tabela XT65 (correspondente ao modelo usado para o módulo GPRS da Siemens) é dotada dos campos:

- ID: responsável pela contagem dos dados, automaticamente incrementado.

- IMEI (*International Mobile Equipment Identity*): número único para todo módulo XT65, funciona como um MAC *Address*. Ele serve para identificar a tabela onde são gravadas as informações de cada módulo, ou seja, se existirem dois módulos, eles terão IMEI's diferentes e serão, conseqüentemente, salvos em tabelas diferentes, desde que previamente cadastrados[7]. Ao enviar o IMEI, ele é comparado aos IMEI's cadastrados nessa outra tabela e os dados são salvos na tabela correspondente a esse número.
- Data : Informada pela consulta do módulo ao satélite GPS.
- Hora: Também informada pelo GPS, baseada no fuso horário do meridiano de Greenwich.
- ST: é o *status* do módulo. Isso é explicado com mais detalhes adiante, mas corresponde à veracidade dos dados. Para informar a altitude em que se encontra, por exemplo, o módulo precisa estar enxergando, pelo menos, 4 satélites. Já para informar a coordenada, são necessários apenas 3, já que se trata de uma geometria planar. O módulo é programado em software para só enviar os dados se obtiverem *status* 2 (bidimensional) ou 3(tridimensional).
- Latitude: é a coordenada que dá informação da distância que o ponto está do equador, em outras palavras, é a coordenada medida no sentido norte-sul, ditada pelas linhas imaginárias paralelas ao equador.
- Longitude: é a coordenada leste-oeste que informa a distância do ponto ao meridiano de Greenwich.
- Altitude: é uma posição de terceira dimensão que pode ser estimada pela triangulação de, no mínimo, 4 satélites GPS.
- Servidor de Data : é usada apenas como redundância para o horário do GPS. Ele poderia ser usado para o caso de o módulo não achar o satélite e, conseqüentemente, não conseguir enviar informações realistas, no entanto, com a restrição do envio apenas das coordenadas com o *status* 2 ou 3, esse dado não tem relevância.

[7] O cadastro é feito em outra tabela chamada *Login*, que contém os campos IMEI, Senha e o Login (que é igual ao nome da tabela onde são gravados os dados), propriamente dito. Ao fazer o *login*, corretamente, o usuário, acessará a tabela correspondente.

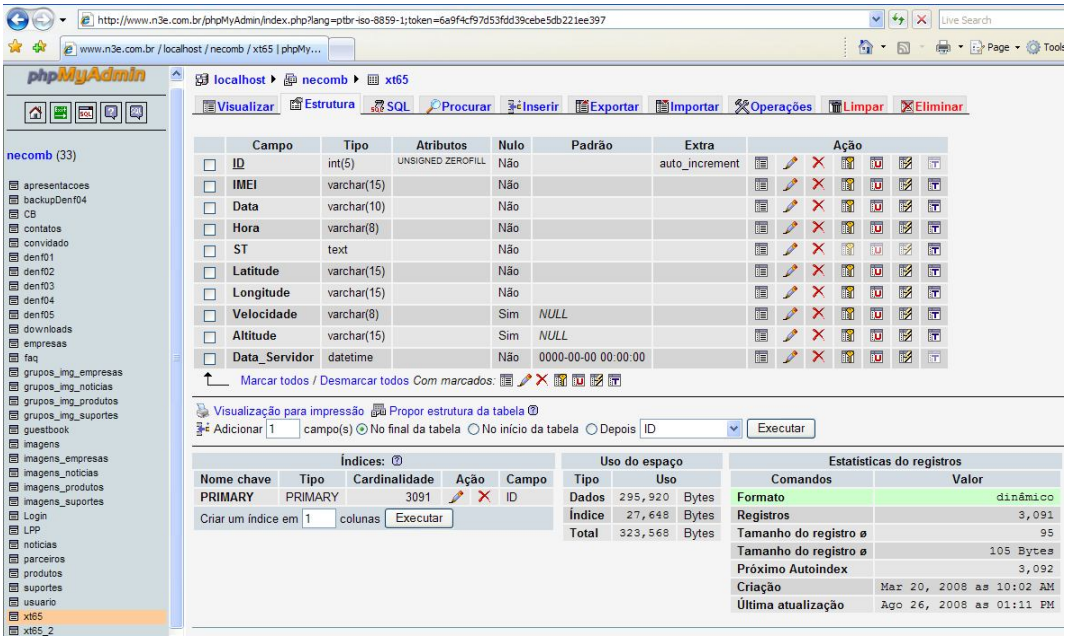


Figura 14 - Página do phpMyAdmim para administração do banco de dados remoto.

2.3. GPS

É uma ferramenta muito importante nos sistemas de navegação, já sendo amplamente utilizada em embarcações aéreas e marítimas [26]. Mais recentemente, os automóveis passaram a acrescentá-lo em seus computadores de bordo, guiando-se por mapas das ruas e descobrindo os melhores trajetos, além de estar presente em equipamentos profissionais no ramo de topografia, agrimensura, etc... As grandes metrópoles, com seus problemas urbanos, motivaram o uso civil dos equipamentos GPS. Hoje, muitos modelos de telefones celulares já são equipados com receptores GPS.

Criado pelo Departamento de Defesa dos EUA, o sistema é composto por uma constelação de 27 satélites (24 em operação e 3 extras, caso haja falha nos outros). São os chamados satélites de órbita média - MEO (*Medium-Earth Orbit*), alimentados por luz solar e pesando cerca de 1,5 tonelada. Circundam o globo terrestre a aproximadamente 19.300 quilômetros de altura, como ilustrado na Figura 15, descrevendo duas rotações completas a cada dia. As órbitas são dispostas de modo que, a qualquer hora do dia, em qualquer lugar na Terra, haja pelo menos quatro satélites "visíveis" no céu [26].

Figura 15 - Órbita dos satélites GPS [26**]**

O GPS foi criado durante a Guerra Fria para que os submarinos americanos pudessem determinar com exatidão a posição dos mísseis soviéticos. O uso civil foi permitido depois de um Boeing coreano ter sido derrubado por mísseis russos, em 1983, ao sobrevoar o espaço aéreo da União Soviética, por falta de um equipamento de navegação. Em 1995 o sistema já estava totalmente operacional para os militares. Os civis, embora tivessem acesso ao mesmo, possuíam uma localização distorcida para evitar que equipamentos "inimigos" fizessem uso do GPS para atacar o próprio país que o criou. Essa distorção acabou em 2000 por intervenção do então presidente norte-americano, Bill Clinton, e em 2004, o presidente George W. Bush confirmou que o sistema não teria custo para os usuários.

Hoje, mais de 95% das unidades de GPS produzidas destinam-se a uso civil. O *Commerce Department* dos EUA afirma que o volume de vendas anual do GPS ultrapassa 20 bilhões de dólares e o crescimento é notável.

Além do GPS, o Glonass russo também é um sistema efetivo de posicionamento por satélite. Existem mais dois sistemas em implantação: o Galileo, europeu, e o Compass, chinês **[**27**]**.

2.3.1. Funcionamento do GPS

A função de um receptor GPS é localizar três ou mais satélites, determinar a distância do ponto alvo até cada um dos satélites e utilizar essa informação para deduzir sua própria posição. Essa operação é baseada em um princípio matemático simples chamado trilateração **[**28**]**.

2.3.1.1. Trilateração 2-D

Para ilustrar esse conceito, será usado um exemplo regional, no estado de São Paulo. Supondo que alguém esteja perdido em uma cidade do Brasil, desconhecendo quaisquer evidências de clima e vegetação, isto é, essa pessoa não possui qualquer noção de sua localização dentro do estado em que está perdido, embora possua um mapa do estado.

Ao avistar um morador da cidade, pergunta: "Onde eu estou?". A resposta indica que está a 12 km da cidade de Ibaté. Isso ajuda, mas não resolve, porque tudo o que se pode saber é que a pessoa está sobre uma circunferência imaginária de 12 km de raio, cujo centro localiza-se em Ibaté.

Um outro morador informa que a pessoa está a 40 km da cidade de Descalvado. Assim, a pessoa perdida sabe que está também sobre outra circunferência imaginária, com 40 km de raio, com centro em Descalvado. Portanto, ela pode deduzir que está na intersecção das duas circunferências e já pode limitar bastante sua localização.

Se um terceiro morador lhe disser que ela está a 25 km da cidade de Analândia, então ela já pode se localizar. A terceira circunferência imaginária marcará um único ponto de intersecção entre as 3 circunferências (marcado com um "X"). E é aí que conclui que está em algum ponto no centro de São Carlos, como ilustrado na Figura 16.

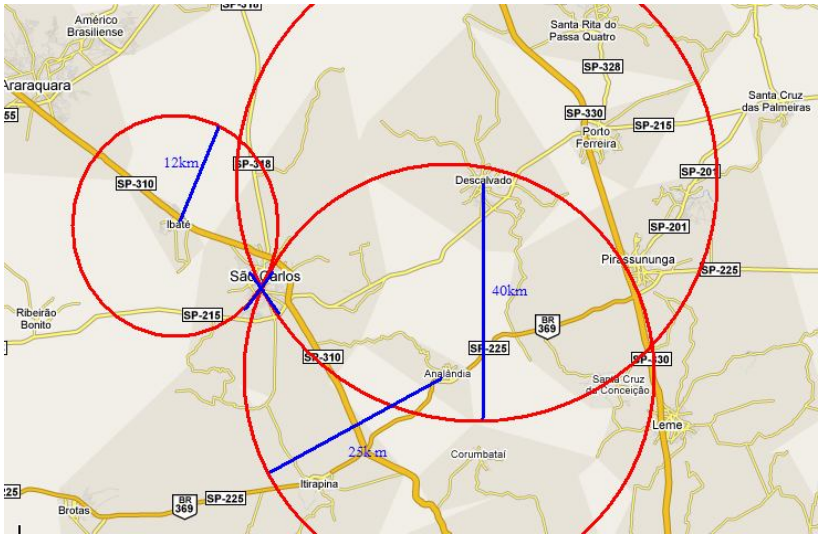


Figura 16 - Trilateração 2D.

2.3.1.2. Trilateração 3-D

A trilateração tridimensional, apesar de mais complexa, não é muito diferente da bidimensional, e, ao invés de uma série de círculos, tem-se uma série de esferas.

Aqui os pontos de referências são os satélites e eles serão análogos aos moradores do exemplo da trilateração bidimensional. Como já foi dito, estes satélites dão duas voltas na terra por dia, nem sempre todos eles estão visíveis em todos os pontos da Terra. Analogamente à trilateração bidimensional, é preciso que pelo menos três satélites estejam visíveis. Para tanto, como os satélites não estão parados sobre a superfície da terra, suas rotas são tais que pelo menos quatro estarão sempre visíveis de qualquer ponto da Terra.

Se alguém souber que está a 16 km do satélite A no céu, deduzirá que pode estar em qualquer lugar da superfície de uma esfera imaginária com um raio 16 km. Se souber também que está a 24 km do satélite B, poderá sobrepor as duas esferas, que se cruzam em um círculo perfeito. Se souber a distância até um terceiro satélite, obterá uma terceira esfera, que cruza com esse círculo em dois pontos (Figura 17). A própria Terra pode agir como uma quarta esfera - apenas um dos dois pontos possíveis estará na superfície do planeta em si, de forma que se pode eliminar o ponto no espaço. Apesar disso, os receptores geralmente olham para quatro ou mais satélites para melhorar a precisão e fornecer informações exatas sobre altitude [28].

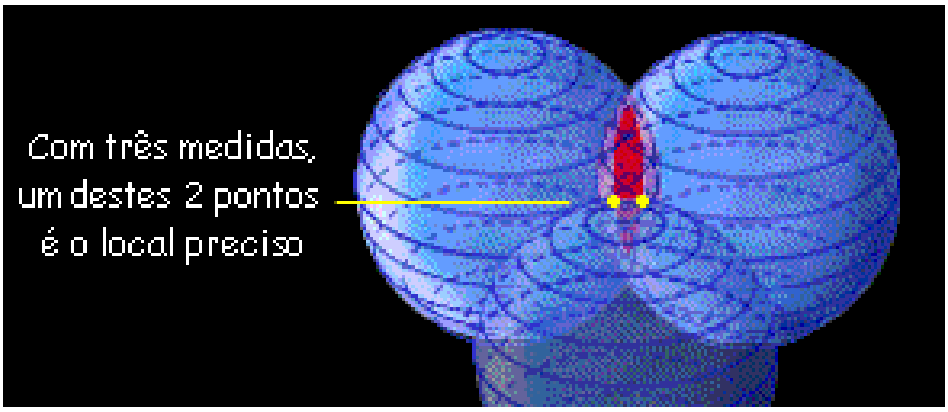


Figura 17 - Trilateração 3D [26].

2.4. GPRS (General Packet Radio Service)

O GPRS (*General Packet Radio Service*), uma vertente do GSM, é um serviço baseado em comutação por pacotes para comunicações sem fio. As mensagens são divididas em pacotes, transmitidas até as estações móveis GSM e transformadas em quadros compatíveis com as redes IP.

No GPRS os usuários ficam conectados constantemente, só pagando pelos dados transmitidos e não pelo tempo de conexão. Teoricamente, pode atingir taxas de transmissão de 171,2 kb/s, porém essa taxa só é possível no GPRS usando todos os oito *slots* de tempo do *frame* GSM simultaneamente. Isso corresponderia a quase três vezes a velocidade dos modems discados comuns de 56 kb/s.

2.4.1. Comutação por pacotes *[12]*

Na comutação por pacotes (Figura 18) as mensagens são transformadas em partes menores, os “pacotes”. Cada um desses pacotes tem um número seqüencial que auxilia na remontagem da mensagem original no lado do *host* receptor. Os pacotes não precisam chegar na ordem certa nem seguir a mesma rota dos outros pacotes da mensagem. De fato, ao verificar o número seqüencial de cada pedaço e constatar algum pacote faltante, existem protocolos que permitem que o *host* receptor requisite que o *host* emissor envie novamente aquele determinado pacote.

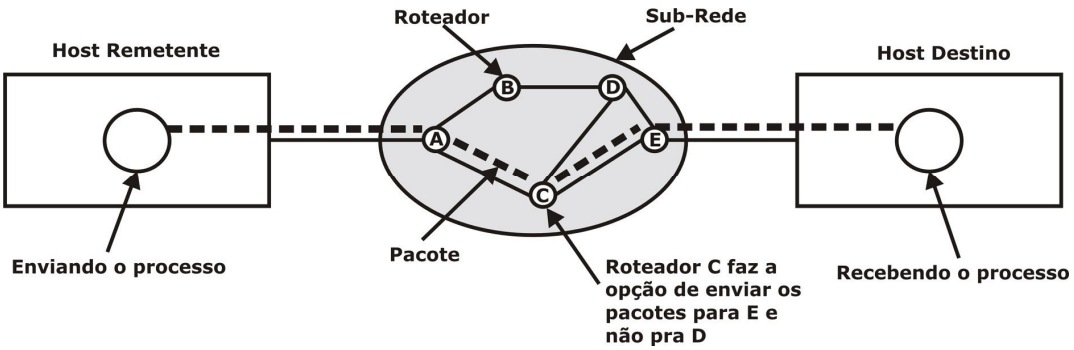


Figura 18 - Comutação por pacotes [12**].**

Assim, os pacotes não precisam seguir a mesma rota, eles são roteados individualmente, existindo um algoritmo de roteamento que prioriza as rotas de menor custo. As decisões de roteamento são tomadas em caráter local. Quando um pacote chega ao roteador A, cabe ao roteador A decidir se esse pacote deve ser enviado na linha para B ou na linha para C.

Vantagens:

- Comutação por pacotes utiliza recursos mais eficientemente;
- Tempo de iniciar e terminar ligações é muito pequeno;
- É mais flexível (não se preocupa muito com o que foi enviado, desde que seja possível colocar em formato de pacote);
- Emissor e receptor podem transmitir em taxas diferentes;
- Tipos diferentes de computadores podem se comunicar em rede de comutação por pacotes;
- Redes de comutação por pacotes não recusam uma conexão; no máximo, atrasam a ligação até que o pacote possa ser transmitido;
- Comutação por pacotes gera tráfego em rajadas (*burst*), que é mais usado nas redes de computadores;

Desvantagens:

- Pouca garantia nos atrasos;
- Algoritmos são mais complexos;
- Demasiados pacotes poderão conduzir a um congestionamento da rede comutada por pacotes: pacotes que não são guardados ou entregues podem ser descartados;
- Pacotes podem chegar em tempos diferentes e numa ordem diferente da que foram enviados.

2.4.2. Evolução GSM [30]

O GPRS (2,5G) é considerado uma evolução do GSM, como ilustrado na Figura 19, sendo um serviço de dados que não abrange as chamadas convencionais de voz. Um celular 2,5G é um aparelho capaz de realizar chamadas de voz, SMS, enfim, todos os recursos dos equipamentos 2G e também tem a capacidade de acessar uma rede de dados, como a Internet, através do GPRS, mas opera sobre a mesma rede GSM, com as mesmas bandas de transmissão, os mesmos *slots* de tempo e assim por diante.

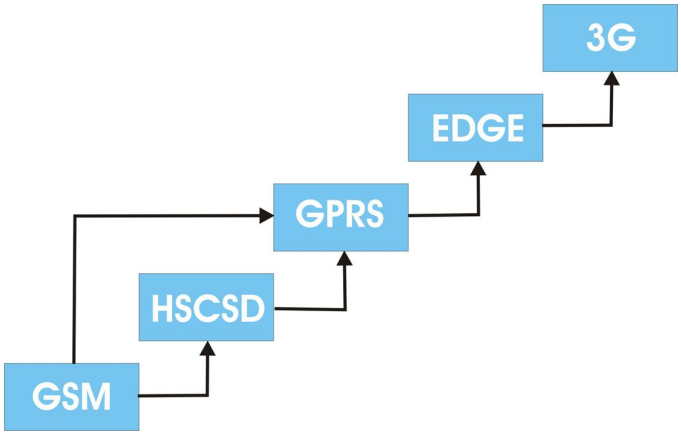


Figura 19 - Evolução GSM [30].

As inovações tecnológicas, novas formas de codificação e modulação, permitiram um salto gigante na rede de dados da telefonia celular. A chegada da terceira geração do celular (3G) viabilizou o "*always on*", ou seja, a capacidade de uma pessoa ficar o tempo todo *on line*. Antes, porém, de surgir a terceira geração com taxas de transmissão que não discriminam qualquer tipo de serviço de dados, alguns passos tiveram que ser dados para amenizar a passagem do GSM puro (2G) ao 3G e por isso, as fases intermediárias são chamadas 2,5G [31]. Na seqüência evolutiva mostrada na Figura 19 as etapas são:

- GSM (*Global System for Móbile*): É a primeira geração digital de celular, conhecida como 2G. Baseado na comutação for circuitos, atinge velocidade máxima de 9,6 kb/s.
- HSCSD (*High Speed Circuit Switched Data*): É a primeira tentativa de dotar o GSM de uma taxa de transmissão maior, através da atribuição de um número maior dos *slots* de tempo a um mesmo usuário ou aplicação, permitindo velocidades de até 64 kb/s. Esse tipo de serviço seria direcionado a placas PCI para notebooks, mas não vingou.

- GPRS: É a primeira tecnologia de comutação por pacotes na rede GSM, facilitando a integração da rede de dados celular com as redes baseadas no IP. Trata-se da geração 2,5G, sendo o primeiro dos passos para amenizar a falta de estrutura para uma terceira geração. Em sua concepção o GPRS atingiria bandas teóricas de 170 kb/s, mas, na realidade, atinge as velocidade dos modems discados convencionais de 56 kb/s.
- EDGE (*Enhanced Data for GSM Evolution*): É a segunda tentativa de suprir o 3G. Baseado em técnicas de modulação mais eficientes e dedicando até 8 *slots* de tempo a uma mesma aplicação, o EDGE atinge taxas de 384 kb/s, chegando ao limite da capacidade do TDMA, ficando muito aquém do que se prometeu para a terceira geração de celular.
- 3G: embora o comitê responsável pelo GSM dissesse sempre que as gerações 2,5 seriam intermediárias ao 3G, eles estavam apenas adiando problema da incapacidade das redes GSM para abrigar uma terceira geração, com quadros, técnicas de codificação e modulação e grande parte da infra-estrutura muito diferentes de tudo que existe no GSM. Essa terceira geração é chamada WCDMA.

2.4.3. Arquitetura do Sistema GPRS [30]

As partes do sistema GPRS que realizam a comutação dos pacotes são chamadas de SGSN (*Serving GPRS Support Node)*. Constituindo o centro da rede, este nó permite o roteamento para as demais partes e é a interface entre a rede de comutação por pacotes (rede de dados) e a rede de comutação por circuitos.

A Figura 20 ilustra a configuração de uma rede sem fio com GPRS, cujas partes são descritas a seguir:

- SGSN (*Serving GPRS Support Node)*: é o nó que envia e recebe pacotes de dados das estações móveis. Ele envia requisições aos HLR's (*Home Location Registers*) para obter perfis de dados dos assinantes GPRS e detecta novas estações GPRS numa determinada região.
- GGSN (*Gateway GPRS Support Node*): gerencia o roteamento entre a rede GPRS e outras redes de dados (Internet, por exemplo). Também é responsável por controlar a alocação de endereços IP e por traduzir os

formatos de pacotes de endereços externos para o formato de endereçamento GPRS e vice-versa.

- GTP (*GPRS Tunnelling Protocol*): protocolo que encapsula pacotes IP para que eles possam trafegar na rede GPRS.
- BSS (*Base Station System*): é uma estação preparada para reconhecer e enviar os dados dos assinantes até o SGSN.
- TE (*Terminal Eqquipment*): é o terminal onde o usuário final trabalha. Caracterizado pelo computador, o sistema recebe endereçamento IP para conectividade em uma rede local ou Internet.
- MT (*Mobile Terminal*): é como um modem que fornece a conexão do TE na rede GPRS utilizando uma ligação com o SGSN. É estabelecido um túnel entre o TE e o SGSN.

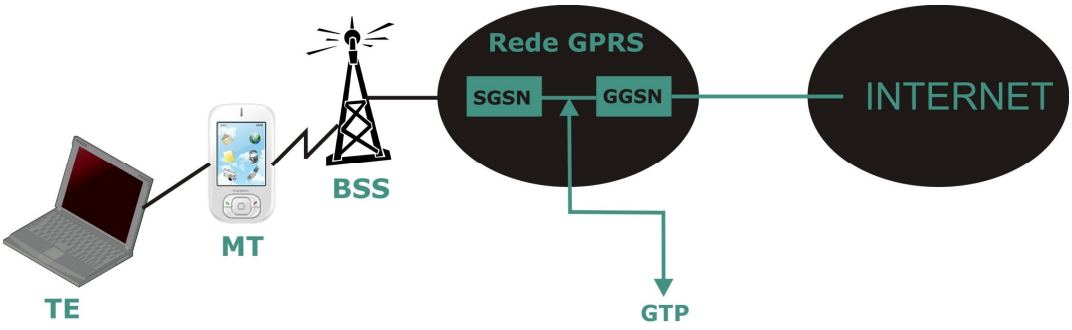


Figura 20 - Arquitetura GPRS [30].

2.4.4. Comandos AT [32][33]

Para que se estabeleça uma comunicação com o modem, passando-lhe as instruções, são usados os chamados “Comandos AT”. O modem os interpretará, retornando a informação requisitada ou a confirmação de correto recebimento do comando ou de erro na sintaxe do mesmo ("OK" ou "ERROR").

Os modems seguem o padrão Hayes, uma fabricante de modems da década de 80. Naquela época não havia um padrão de *driver* dos modems e, portanto, quando um novo modelo de modem chegava às lojas, a sintaxe de comandos também mudava completamente. A Hayes foi a primeira a padronizar uma série de comandos que passaram a ser adotados pelos modelos mais novos, permitindo o uso do mesmo *driver* de modelos anteriores até que uma nova versão fosse adaptada. Notar que existem comandos específicos de cada modem, variando de fabricante para fabricante, e para

sabê-los é necessário consultar os manuais dos mesmos, mas existe uma série de comandos que são comuns a todos os modems e esses seguem o chamado padrão Hayes.

Os comandos AT permitem controlar várias funções do modem e podem ser usados quando digitados na linha de comando de um programa como HyperTerminal.

Exemplos de comando

- AT: é o principal comando que deve ser enviado ao modem, quase sempre antes de outro comando. Serve para chamar a ATenção do modem, informando-o que o que vem a seguir é uma seqüência de comandos que ele deve interpretar. Os manuais dos modems normalmente trazem uma lista dos comandos possíveis para este determinado modelo. Alguns comandos se tornaram padrão entre todos os modems, como:
- ATD: para tirar o modem "do gancho" e mandá-lo discar o número que vier a seguir. Ex: "ATD99999999".
- ATZ: Carrega a configuração previamente salva na memória do modem.
- ATA: Atende a chamada. Quando alguém liga e o modem detecta a chamada, digitando-se "ATA" seguido de <ENTER> o modem atenderá a chamada e iniciará o procedimento de conexão com o outro modem.

CAPÍTULO 3

Hardware

Esse trabalho propõe a construção de oito tipos de nós que são discriminados em software para que a rede de monitoramento possa dar a informação correta sobre cada um, bem como direcionar as ações do usuário. Por exemplo, a ação de acionamento do *dimmer* por parte do usuário terá que ser encaminhada a um *dimmer* e não a um controle remoto.

Todos os nós ZigBee foram construídos com um componente chamado PIXIE, desenvolvido pela empresa Flexipanel [34], já completamente preparado para receber uma pilha ZigBee, dotado de antena, do transceptor CC2420 e do microcontrolador PIC18F4620, mostrado na Figura 21.

De maneira geral, sem contar os kits de desenvolvimento, usados para familiarização com as pilhas ZigBee e/ou TCP/IP, todos os hardwares foram esquematizados na empresa, incluindo a parte de *layout* e roteamento de PCI's, utilizando a ferramenta ORCAD. A maioria das placas foi impressa, ainda que de maneira bem artesanal, dentro da N3E. Além disso, a soldagem, inclusive de componentes SMD, também foi feita na empresa.

Este capítulo apresenta uma descrição completa do hardware.

Figura 21 – Módulo PIXIE.

3.1. Coordenador/TCP_IP

É o gerenciador da rede, o *gateway* entre as duas redes (Ethernet-ZigBee). Nele estão as tabelas de *binding* e as páginas HTML para visualização remota. Todas as

mensagens trocadas na rede definirão um novo *status* a ser salvo nesse nó e atualizado na página Web hospedada por ele.

O coordenador incorpora dois componentes. Um direcionado ao ZigBee, utilizando o PIXIE e o outro relacionado à rede ETHERNET, que utiliza o PIC18F97J60. A interface entre essas redes é feita pela comunicação serial (Figura 22). Quando um módulo precisa se comunicar com o outro, para atualizar a entrada de novos nós ou para realizar a requisição vinda do site, por exemplo, uma string é colocada no RX de um, ativando a interrupção da USART do outro. Os campos da *string* são tratados e a ação é realizada.

O sinal AC é transformado na fonte em um sinal DC de 3V3, que alimenta os dois módulos.

A parte do hardware do ZigBee é bem simples, sendo a complexidade toda concentrada no software. Os únicos periféricos são os leds de *status* e o botão de exclusão. Os leds servem para indicar a entrada de novos filhos, *status* de formação da rede e advertências como tabela de *binding* cheia. O botão exclusão serve para apagar todos os registros da tabela de *binding* e da tabela de vizinhança. Apertando esse botão a rede deverá ser totalmente reconfigurada.

O hardware da parte ETHERNET é um tanto mais complexo. O PIC18F97J60 suporta duas SPI's (*Serial Peripheral Interface*) e duas USART's (*Universal Synchronous Asynchronous Receiver Transmitter*). A segunda USART foi destinada em software para as configurações iniciais, entre elas: IP inicial, gateway, máscara de rede e as configurações necessárias para estabelecer uma rede ETHERNET. A SPI1 está atrelada à memória. Na memória estão salvas as páginas WEB do servidor, os nós ZigBee que já integraram a rede, as informações da última atualização, senhas do servidor HTTP e FTP e outras informações que não podem se perder. É utilizada a memória da Microchip 25LC256, com capacidade de 256 kbits. Na SPI2 está ligado o RTC (Real Time Clock) da Dallas Semiconductor, DS1305, utilizado para vincular as atualizações da rede ZigBee com o momento exato em que aconteceram, portanto, ao se acender uma lâmpada, além de se ver a alteração do *status* do nó, também será possível ver a mudança no horário de sua última atualização. O horário fica salvo na EEPROM, junto com as demais informações do nó. A configuração do RTC é feita, assim como as configurações da rede ETHERNET, pela USART 2.

A parte ETHERNET possui três botões que servem para restaurar configurações originais, apagar as informações de senhas e nós ZigBee na EEPROM e discernir as

configurações da rede ETHERNET e do RTC. Os leds servem para indicar o tráfego ETHERNET e o sensor de temperatura é um adendo para incrementar as informações do ambiente. Esse sensor pode ser substituído ou auxiliado por outros sensores da rede ZigBee. Ele é ligado a uma porta analógica do PIC18F97J60, que faz uma quantificação digital do valor. O conversor AD do PIC é de 10 bits. O RTC é alimentado em 3V3, portanto, temos 2^{10} valores para representar 3300 mV. O sensor utilizado é o TC1047 da Microchip. Sua temperatura varia linearmente com o nível de tensão que coloca na entrada analógica do PIC. Assim, pode-se, facilmente, fazer a conversão de tensão em temperatura.

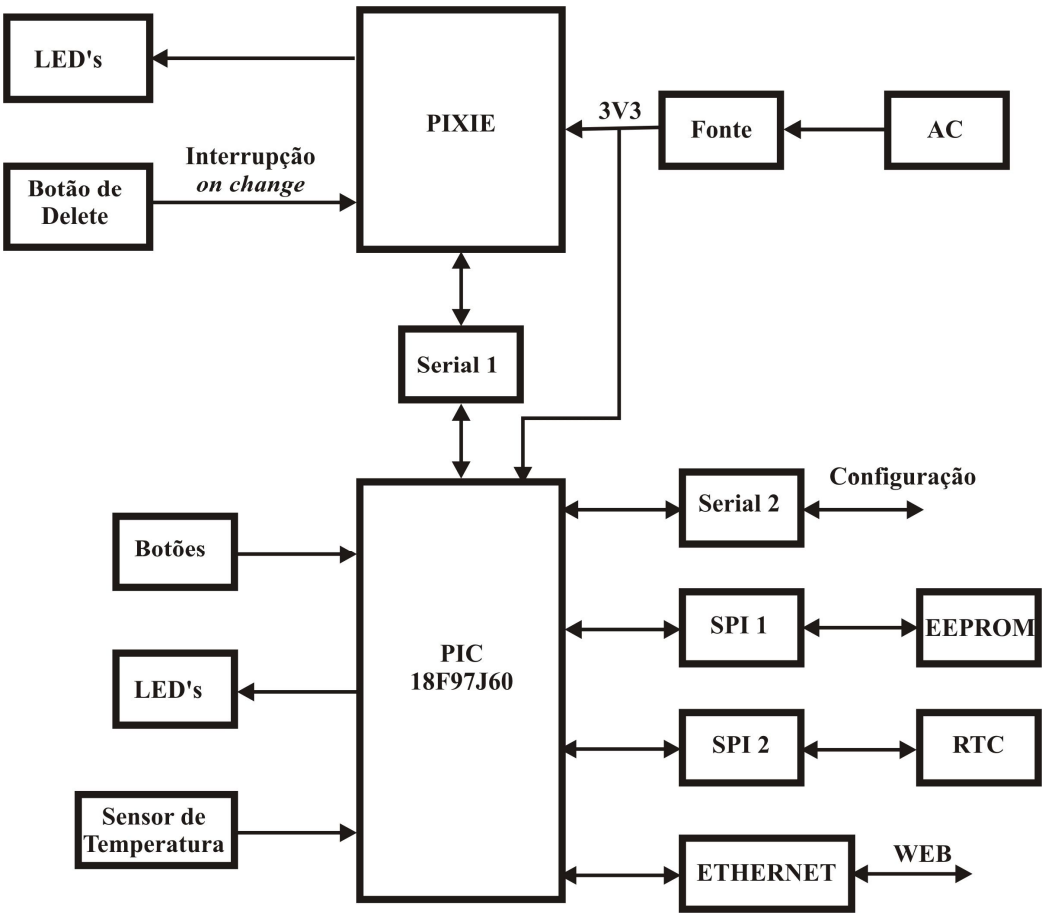


Figura 22 - Diagrama de blocos do Coordenador/TCP_IP

3.2. Dimmer

A Figura 23 apresenta um diagrama de blocos do *dimmer*. Ele é o responsável pela parte de iluminação da rede. Em seu hardware, o sinal AC é aplicado na fonte que possui um circuito para transformá-lo num sinal de saída DC de 3V3 (tensão nominal do PIXIE).

O circuito detector de zero é alimentado com a tensão AC e, como o próprio nome diz, ele é capaz de detectar a passagem da senóide AC pelo valor zero. Sua saída é uma onda quadrada com valor máximo de 3V3, obtido quando a senóide passa pelo zero na crescente (negativo para o positivo). Quando a senóide está na descendente (positivo para o negativo), a onda quadrada apresenta seu valor mínimo de 0V. Essa onda quadrada alimenta o comparador do microcontrolador do PIXIE. O funcionamento do comparador é simples: uma das portas referentes a essa função é alimentada com uma tensão de referência de 1,6 V e a outra alimentada com a onda quadrada já referida. Quando a onda quadrada passa de 3V3 para 0 V e vice-versa, ela, obviamente, passa também pelo valor de 1,6 V, igualando a tensão nas duas portas e ativando a interrupção do comparador. A mudança de 3V3 para 0 V e vice versa é exatamente o momento em que a senoide AC passa pelo 0 V.

A intensidade da carga dependerá do momento de acionamento do triac. O triac também é alimentado com a onda AC. Dependo do instante em que o pulso em seu *gate* é dado, a potência transferida à carga é alterada. O pulso serve para que o triac conduza. Se esse pulso é dado bem no início do ciclo da senóide, grande parte da potência original é repassada à carga e a intensidade luminosa de uma lâmpada que estiver representando a carga se aproxima da intensidade máxima. Por outro lado, se o pulso só for dado no final do ciclo, a porcentagem da potência original repassada à carga é bem menor e a intensidade luminosa é pequena. Portanto, o princípio do efeito *dimmer*, i.e. da variação da potência na carga, está em alterar o momento desse pulso. Esse momento é controlado pelo tempo de interrupção do *timer*, configurado pelo microcontrolador. Após receber a interrupção do comparador, a interrupção do timer é habilitada. O *dimmer* terá uma variável em software com o valor atual da carga e esse valor servirá de parâmetro para configurar o tempo de interrupção do *timer*. Assim, a toda interrupção do comparador, um pulso, controlado pela interrupção do *timer* e, conseqüentemente, pelo nível de potência atual da carga é dado no *gate*.

Existem dois leds de *status* que servem para indicar o sucesso ou a impossibilidade na integração da rede, a entrada de um novo filho, o desenrolar e a confirmação do *binding*.

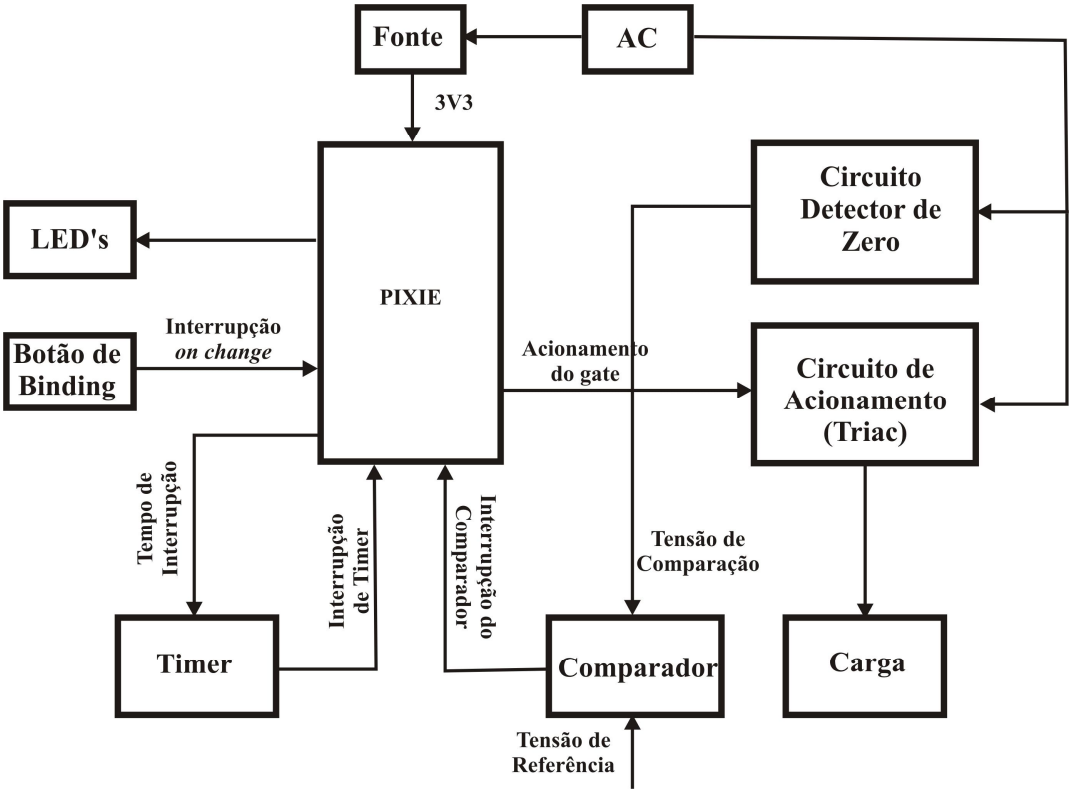


Figura 23 - Diagrama de blocos do Dimmer

3.3. Controle Remoto

O controle remoto visto na Figura 24 foi criado para se deslocar livremente, fazendo *binding* com qualquer nó em qualquer cômodo dentro da rede. Ele pode, por exemplo, criar cenários de iluminação com os dimmers dentro de um ambiente, possibilitando ao usuário acionar a configuração que melhor se adapte a ocasião: criar cenários bem iluminados para estudo ou a meia luz para um jantar. Possui leds de sinalização para confirmar acionamentos e, portanto, se o usuário estiver em seu quarto e lembrar que deixou a luz da sala acesa ele pode, através do controle, apagá-la e receber a confirmação do desligamento em um de seus leds.

O controle remoto é um RFD e, portanto, é alimentado por baterias. O LDO (l*ow dropout*) regula a tensão total das pilhas para a tensão nominal de 3V3 do PIXIE. Como

dito, é um nó acionador que pode se deslocar livremente pela rede, atualizando sua paternidade, quando verifica que está fora do alcance de seu antigo pai.

O hardware é, praticamente, formado por leds e botões. Existem 12 botões e, logicamente, não seria possível atribuir uma porta com interrupção "*on change*" (PORTB) a cada botão. Construiu-se então uma matriz de botões. São sete portas do PIXIE, 4 do PORTA e 3 do PORTB(4 x 3 = 12). Cada botão está associado a duas portas e, obviamente, nenhum deles está associado às mesmas duas portas. Além disso, todos eles estão associados a uma porta do PORTB. Assim, o pressionamento pode ser detectado pela interrupção sendo, em seguida, feita a leitura da PORTA para saber qual botão foi acionado.

Também não existiriam portas suficientes para os leds (6 no total). Para suportá-los foi utilizado o s*hift register* da Philips 74HC/HCT164. É um *shift register* de 8 bits, o que significa que pode controlar 8 saídas, sendo cada bit referente a uma de suas saídas. Duas portas do PIXIE lhe são necessárias: uma para controlar o *clock*, que possui uma disposição temporal de níveis alto e baixo, determinada em software para a leitura dos dados que vêm da outra porta. Assim, se tivermos na porta de dados uma palavra 11000011, por exemplo, estaremos acionando os leds ligados às saídas 1, 2, 7 e 8.

O teclado de membrana, fabricado pela empresa Visuart, foi uma opção escolhida para minimizar o tamanho da placa impressa. A interface com a placa impressa é feita por um conector de 14 vias, onde estão as saídas dos *shift registers*, as portas que serão usadas na matriz de botões impressa no teclado de membrana e o GND comum do hardware.

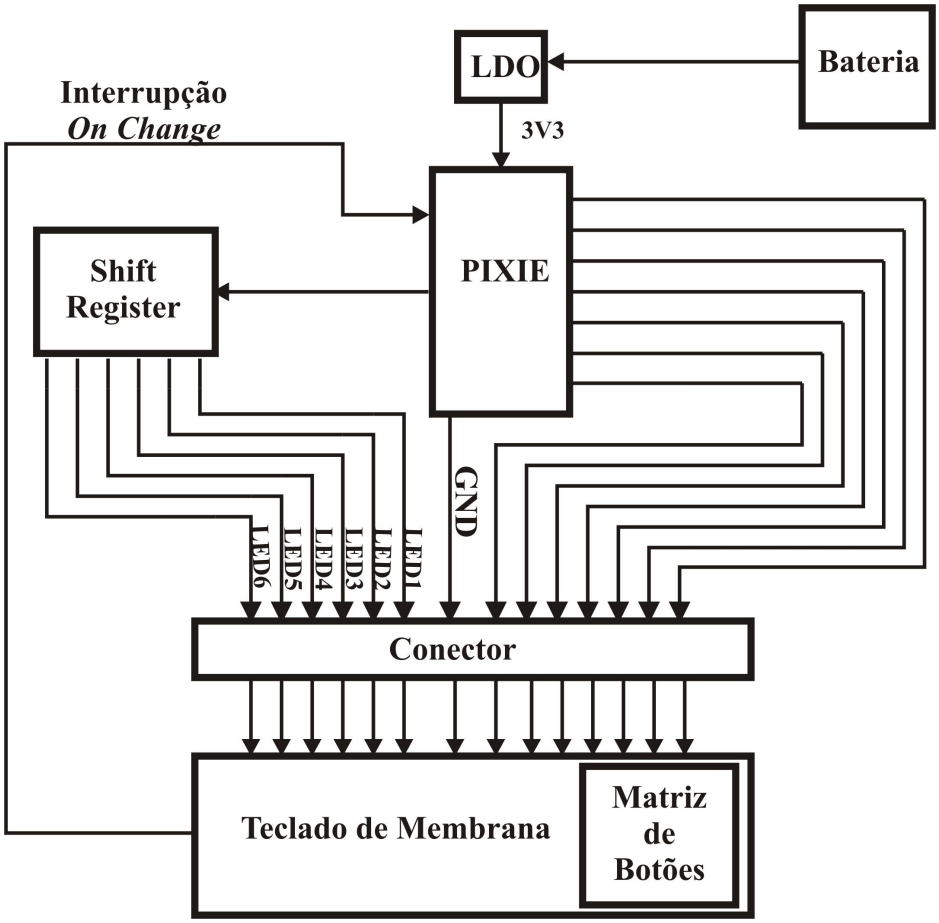


Figura 24 - Digrama de blocos do Controle Remoto

3.4. Biometria

A biometria é um nó adicional que pode ser usado em ambientes de acesso: da porta de entrada de uma casa até a escotilha de um cofre. Ela deve estar amarrada a outro nó, tipicamente, o atuador, que faz o papel de uma fechadura, mas nada impede que a biometria esteja amarrada a um dimmer, apesar de não ser a opção mais inteligente a de usar a leitura biométrica para fazer o simples acionamento de uma lâmpada

Por ser um RFD, a alimentação é obtida de baterias, totalizando 5 V. Os 5 V servirão para alimentar o LDO que o transformará na tensão nominal 3V3 do PIXIE e para alimentar o processador biométrico, cujos pinos são ligados ao conector descrito na Figura 25.

A biometria possui três botões. Um deles destinado ao *binding*, um destinado ao procedimento de cadastramento de impressões digitais e o último ao acionamento da carga que foi associada a esse nó por meio de um *binding*.

Toda a comunicação entre o processador biométrico e o PIXIE é feita por intermédio da comunicação serial. Nos procedimentos de cadastramento e acionamento, a impressão digital que repousa sobre o leitor é decifrada por seus sensores e os dados são enviados ao processador biométrico que, no cadastramento, salva a digital em sua memória, informando o *status* de sucesso ao PIXIE pela serial e, no acionamento, vasculha sua memória em busca de uma digital que case com a digital que foi obtida no leitor, informando ao PIXIE se a digital já foi ou não cadastrada. Assim, o PIXIE pode tomar a decisão de acionar ou não a carga por meio de uma mensagem na rede.

O processador biométrico e a interface de configuração utilizados são fabricados pela empresa Duodigit. Possui uma memória *flash*, onde salva as digitais e possui respostas-padrão para indicar o *status* de leitura que é utilizado no software.

Existem 2 leds de *status* que servem para indicar o sucesso ou a impossibilidade na integração da rede, o desenrolar e a confirmação do *binding*, do cadastramento de digitais e do acionamento.

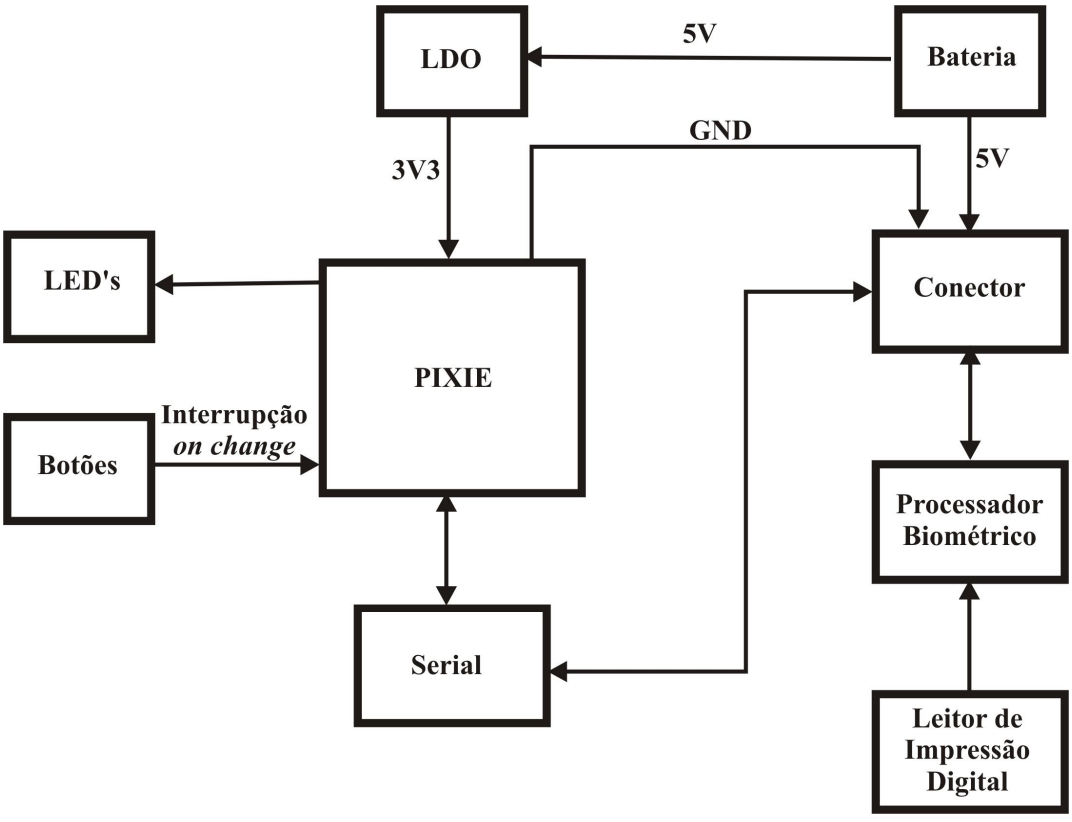


Figura 25 - Diagrama de blocos da Biometria

3.5. Atuador

É a fechadura dos ambientes de acesso que pode ser aberta por um acionador: um controle remoto, um botão de um interruptor ou um nó biometria.

A alimentação do atuador (Figura 26) é transferida a uma fonte reguladora que transforma a entrada AC em uma saída DC de 12 V. Essa servirá para alimentar o relê e o LDO. O LDO transforma os 12 V na tensão nominal 3V3 do PIXIE.

A única interrupção de Hardware é a do botão de *binding*.

Ao receber uma mensagem de acionamento da rede ZigBee, o PIXIE muda o *status* da porta ligada no circuito de acionamento. O nível alto da porta faz com que seja conduzida uma diferença de potencial no relê. O relê é acionado e transfere a tensão para sua saída que está ligada ao conector que, por sua vez, está ligado à fechadura. Ao receber a tensão de 12 V a fechadura abre.

Existem 2 leds de *status* que servem para indicar o sucesso ou a impossibilidade na integração da rede, a entrada de um novo filho, o desenrolar e a confirmação do *binding*.

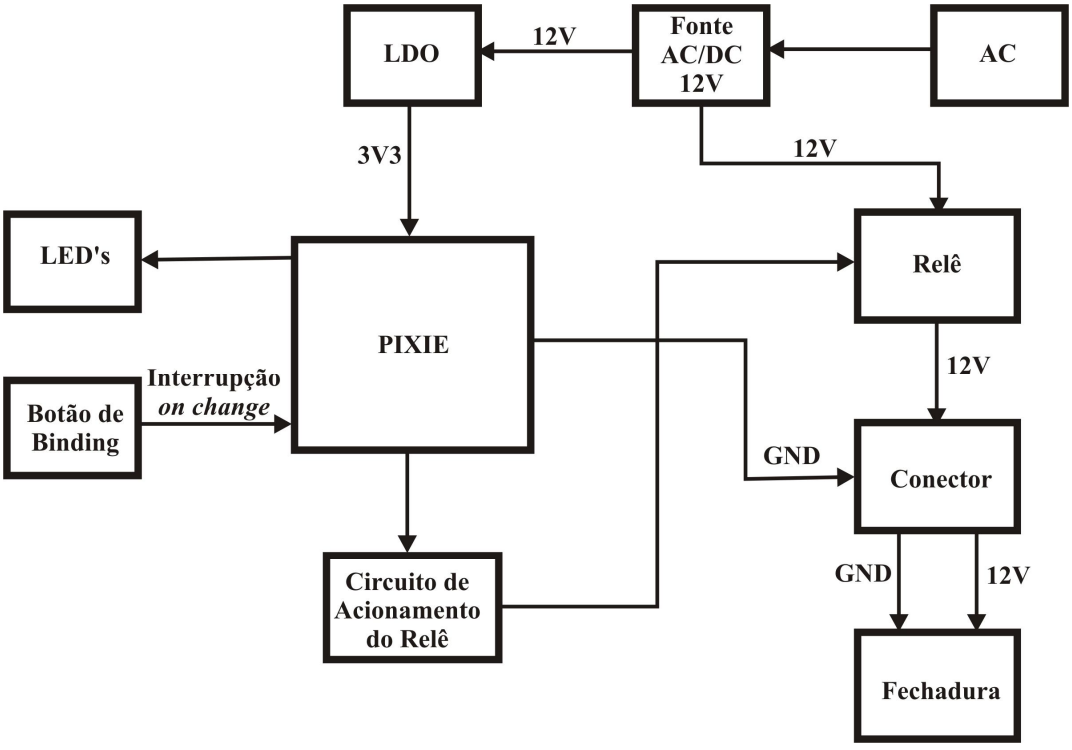


Figura 26 - Diagrama de blocos do Atuador

3.6. Sensor

É o nó responsável pela informação dinâmica da rede. Aqui está exemplificado um sensor de temperatura, mas a rede pode ser equipada com os mais variados sensores (incêndio, presença, umidade, etc...) e, através das leituras que eles fornecem, pode-se construir uma rede bastante informativa, monitorada através de páginas na web.

Por ser um RFD, a alimentação do sensor (Figura 27) é obtida de baterias, totalizando 5 V. Os 5 V servirão para alimentar o LDO que transformará os 5 V na tensão nominal 3V3 do PIXIE e que também servirá para alimentar o sensor.

O sensor possui apenas 1 botão ligado a uma das portas do POTRB e que, portanto, gera uma interrupção *on change* quando acionado. Esse botão serve para fazer a configuração do *set point* da temperatura, ou seja, para estabelecer o intervalo de normalidade de funcionamento, colocando um patamar inferior e um patamar superior da temperatura. Se a temperatura lida pela SPI estiver acima do patamar superior ou abaixo do patamar inferior, uma mensagem direta é enviada ao coordenador com um *status* de alarme e o coordenador poderá informar isso ao site. Depois de acionar o botão, um menu é apresentado na serial e a configuração pode ser feita com um software de comunicação, como o HyperTerminal.

Existem 2 leds de *status*. Um para indicar o sucesso ou a impossibilidade na integração da rede e o outro para indicar o sensor acionado.

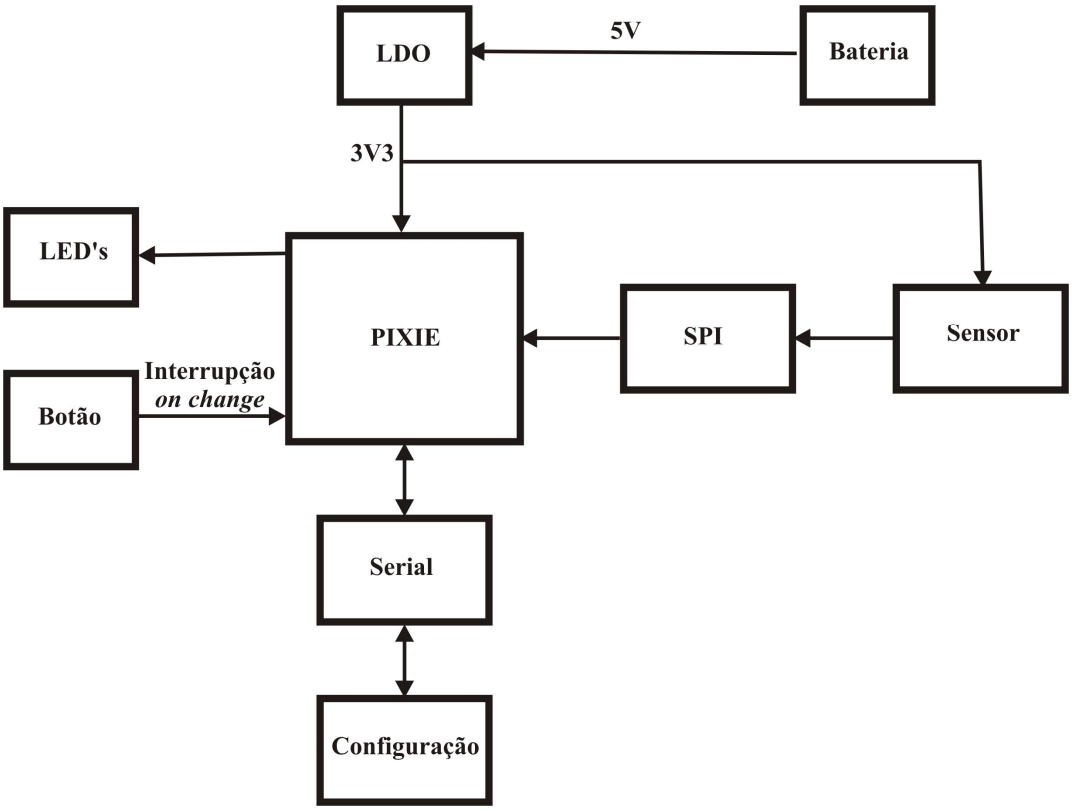


Figura 27 - Diagrama de blocos do Sensor

3.7. Interruptor

É um nó acionador como o controle remoto com a diferença de não se movimentar constantemente na rede, apesar de ser possível mudá-lo constantemente de lugar. Ou seja, ele é, normalmente, um nó fixo, usado para acionar os dimmers dentro de um cômodo, mas nada impede que seja trocado de cômodo para acionar outros dimmers.

A Figura 28 apresenta o diagrama de blocos do interruptor. O sinal AC é aplicado na fonte que possui um circuito para transformá-la num sinal de saída DC de 3V3 (tensão nominal do PIXIE).

É o circuito mais simples entre todos os nós.

A placa foi inserida num gabinete de plástico, característico das construções civis com um grupo de 3 interruptores com 2 estados cada um. Os estados servem para determinar a interrupção *on-change*, que norteiam seu funcionamento. Temos, portanto, 3 botões. Dois deles servem como botões de *dimmer* de uma carga qualquer, havendo uma lógica de inversão simples para aumentar e diminuir a intensidade da carga. O

outro botão serve para a realização do *binding* e deve atuar em conjunto com os outros dois para distinguir que botão será amarrado com uma determinada carga. Em suma, o *binding* será realizado em duas etapas, o pressionamento do botão de modo *binding* e o envio da mensagem de *binding*, propriamente dito, pelo pressionamento de um dos outros dois botões.

Existe apenas 1 led de *status*, que serve para indicar o sucesso ou a impossibilidade na integração da rede, a entrada de um novo filho, o desenrolar e a confirmação do binding e a confirmação (Ack) de um acionamento.

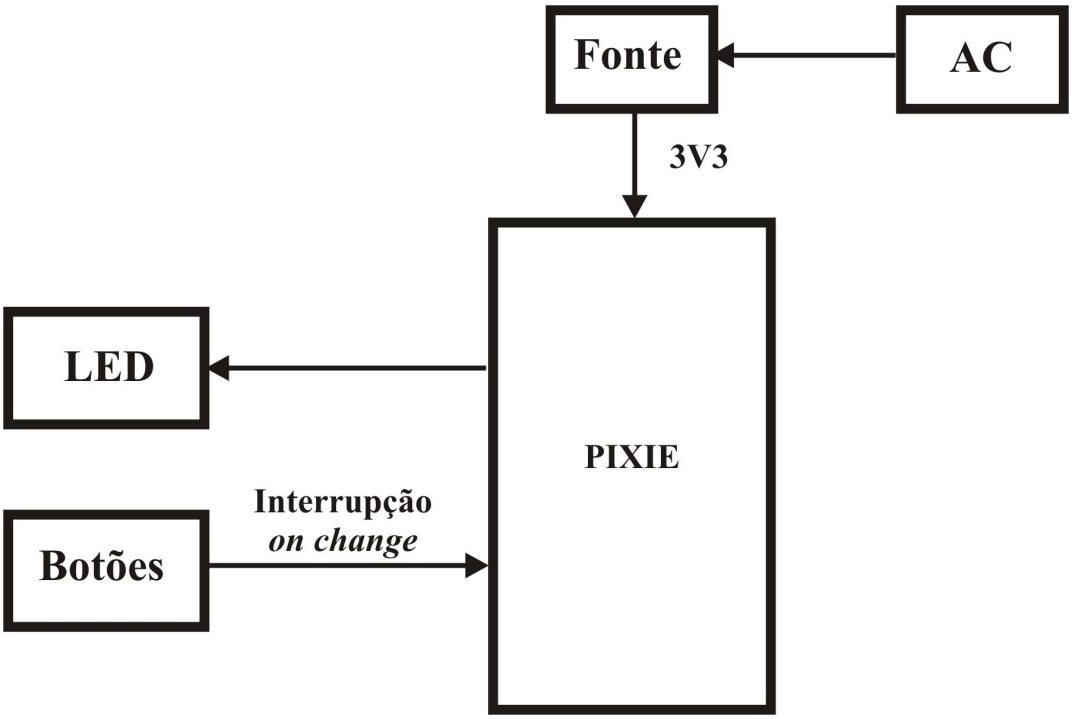


Figura 28 - Diagrama de blocos do Interruptor

3.8. ZigBee/GPRS [35]

A bateria de 3V7 adotada possui três terminais: além do GND e do VCC, um terceiro terminal é usado para controle de temperatura, nele é soldado um termistor (NTC) cujo sinal de saída é usado para controle em software do XT65 (Figura 29). O NTC varia a sua resistência de acordo com a temperatura, caso atinja um patamar crítico, a aplicação Java é destruída. Os mesmos 3V7 alimentam a parte do ZigBee, chegando a um LDO que o converte em 3V3, tensão nominal do PIXIE.

Figura 29 – Módulo XT65 para implementação do controle via GPRS.

A parte do ZigBee é extremamente simples. A idéia é que esse módulo funcione com uma chave. Ao sair de casa, o módulo estará desvencilhado da rede ZigBee e será dedicado ao monitoramento das coordenadas GPS, enviadas, via rede GPRS, a um servidor externo. Ao chegar em casa, ele será reintegrado à rede ZigBee e possuirá um botão com o *binding*, previamente feito, com o atuador presente na porta da casa. Portanto, possuirá 2 botões, um para ser amarrado ao atuador ou a qualquer outro nó que se queira e um botão de acionamento. Possui também dois leds de *status* para indicar o desenrolar da integração na rede, o *binding* e o ack do nó a que está amarrado.

Na parte do XT65 existem vários periféricos necessários à rede GSM e ao próprio hardware do modem. Como indicado na Figura 30, o XT65 não deve ser apenas alimentado com a bateria, ele deve ser iniciado e isso é feito com o circuito de ignição que tem uma onda pré-definida para acionar o módulo.

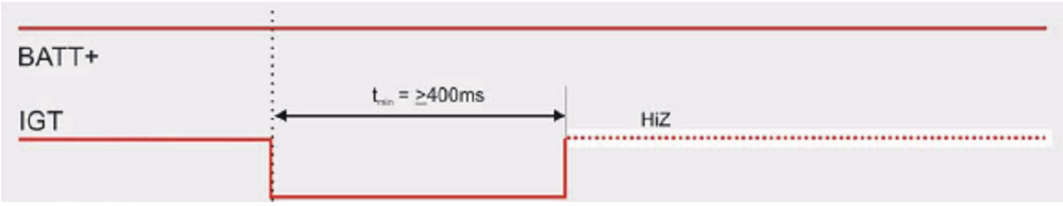


Figura 30 - Onda de representação do acionamento do XT65 [35]

A chave de liga-desliga, além de alimentar o circuito de ignição, é ligada a um pino de I/O do XT65 para indicar ao software quando a aplicação deve ser encerrada.

O hardware, quando em funcionamento, está conectado tanto à ERB GSM quanto aos satélites GPS e, dessa maneira, além da antena ZigBee, deve possuir mais duas antenas: a antena GPS usada é produzida pela Trimble e a GSM foi confeccionada

em circuito impresso, utilizando a técnica de corrosão com percloreto de ferro que será citada no tópico de layout de PCI's. As medidas foram baseadas na antena de um produto já fixado no mercado, o LPP (l*ocalizador portátil pessoal*) da empresa Redecamp [37] e sua qualidade foi avaliada mediante um comando AT [36], específico desse módulo Siemens, que retorna a potência do sinal recebido da ERB GSM. O parâmetro retornado serviu para validar a antena produzida, pois ela conseguiu potências equivalentes a antenas GSM comerciais produzidas pela Amphenol.

O circuito de carregamento, assim com o de ignição, é sugerido pela Siemens. Usando portas do XT65, pode-se detectar, em software, o *status* do carregamento, havendo uma lógica entre a leitura dessa porta e da porta da chave de liga-desliga para direcionar o desligamento do hardware.

O botão de emergência também está ligado a uma porta de I/O do XT65 e seu acionamento funciona para diferenciar as coordenadas enviadas ao servidor, bem como o intervalo entre os envios. Seu pressionamento será entendido pelo servidor externo como uma coordenada de pânico e serão mostradas de maneira diferente das demais, além de mandar, automaticamente, um SMS aos celulares pré-cadastrados (máximo de 3). Existem 3 leds: um para indicar o *status* do carregamento, um para indicar o *status* da rede GSM e outro para o *status* do GPS.

A interface entre os dois módulos é feita pela serial. A atuação na rede ZigBee através da rede GPRS pode ser feita por envio de SMS's. Assim, estando o módulo integrado na rede ZigBee, basta um celular enviar uma mensagem com uma *string* no formato correto com os campos de login e senha e os outros possíveis, para que esse nó reencaminhe a ação a outro nó, cujo endereço também é passado na mesma *string*. O diagrama de blocos do nó ZigBee/GPRS é mostrado na Figura 31.

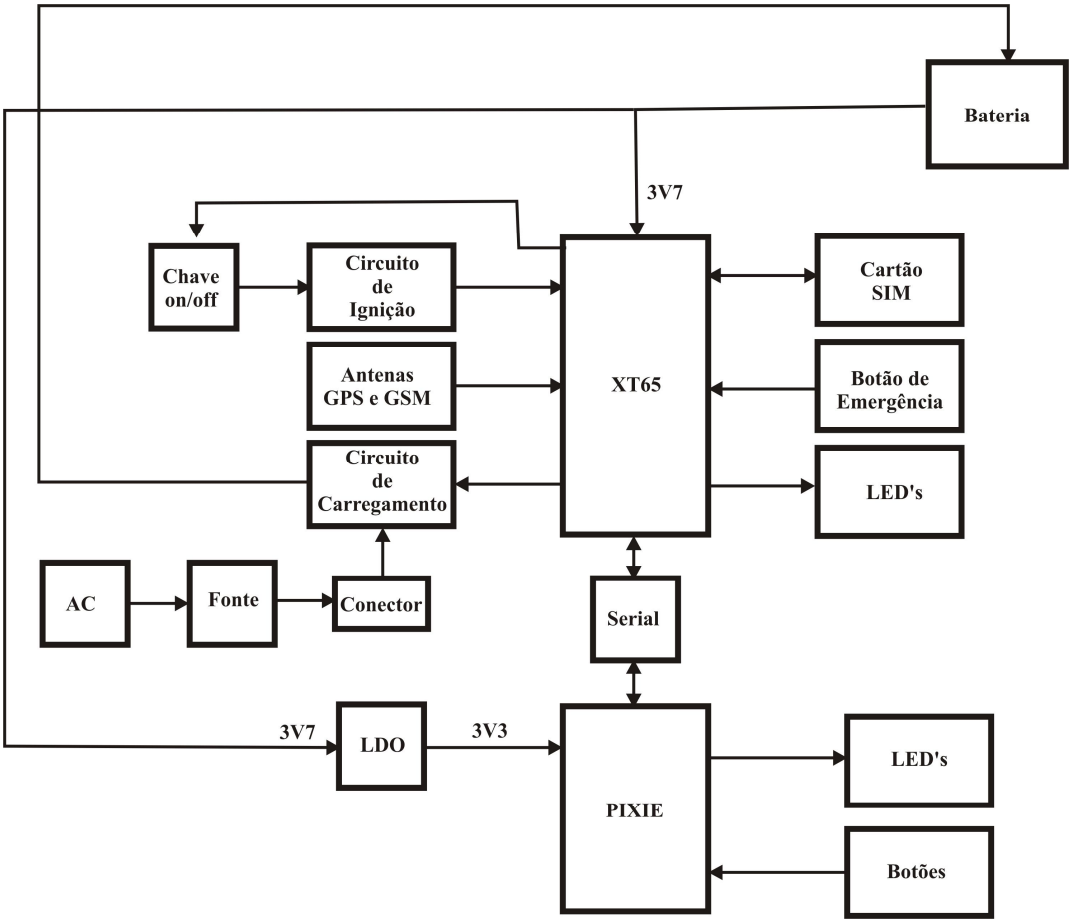


Figura 31 - Diagrama de blocos do ZigBee/GPRS

3.9. Layout das PCI's

Todos os layout's das placas foram feitos utilizando-se o software Orcad. O circuito elétrico pode ser construído com a ferramenta Capture CIS. Com ela realizam-se as conexões entre os componentes. A maioria dos componentes já está inclusa nas bibliotecas inerente ao programa, mas também é possível criar seus próprios componentes e incluí-los em sua própria biblioteca. A Figura 32 mostra um exemplo de circuito elétrico construído com o Capture CIS.

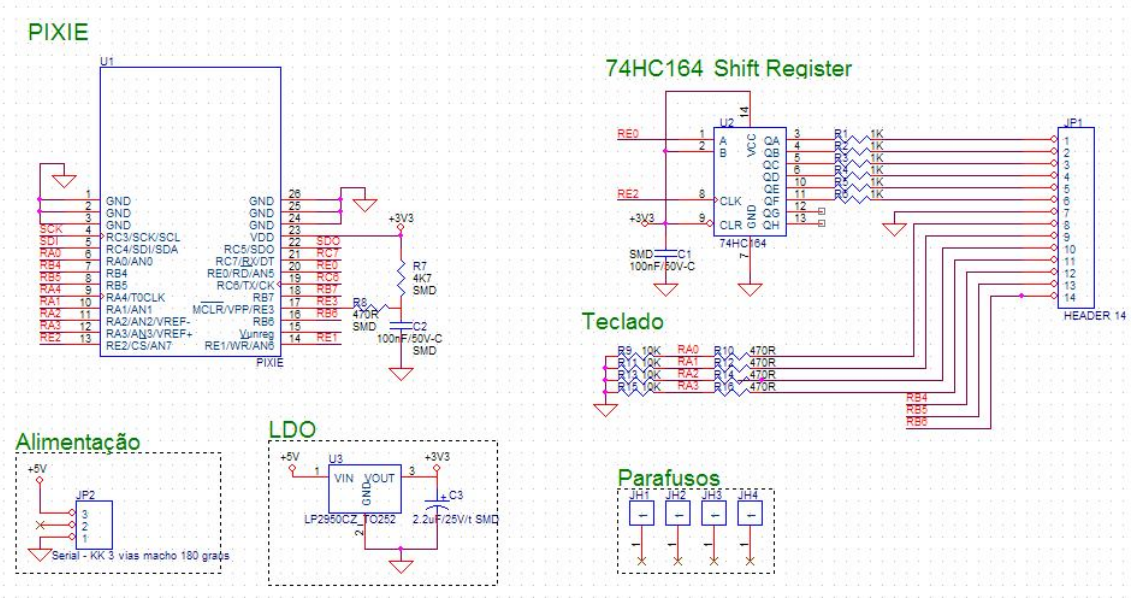


Figura 32 - Esquema elétrico do Controle Remoto

Cada um desses componentes tem seu próprio *footprint*, ou seja, o desenho de suas dimensões reais para ser impresso em uma PCI. A maioria desses *footprint's* também está inclusa nas bibliotecas do Orcad, mas alguns componentes requisitarão o desenho de um footprint novo que será adicionado a uma biblioteca particular.

Uma vez que grande parte dos componentes é SMD, é necessário ficar atento aos seus encapsulamentos. Ainda que não se tenha o *footprint* específico do PIC18F97J60, por exemplo, sabe-se que ele possui um encapsulamento TQFP de 100 pinos e o *footprint* desse componente é padronizado e está pronto na biblioteca. No entanto, outros componentes, a citar o PIXIE, não possuem footprint que os possa representar, já que se trata de uma disposição entre dimensão e posição dos pinos, totalmente peculiar. Portanto, o layout nada mais é que a representação das ilhas de soldas onde serão posicionados os pinos dos componentes e das trilhas que ligarão um ponto a outro, além da representação espacial real de cada componente, o que impede a sobreposição dos mesmos e permite o dimensionamento da placa como um todo e seu espaçamento de trilhas.

Depois de pronto o esquema elétrico no Capture e depois de ser associado a cada componente seu próprio *footprint*, cria-se o roteamento da placa. Essa ferramenta irá colocar os *footprint's* numa planilha CAD com as respectivas ligações que terão que ser transformadas nas trilhas, podendo ser dispostas em várias camadas. As placas feitas na N3E, por ser usada uma técnica bem artesanal, têm, no máximo, 2 camadas. O desenho

das trilhas e a disposição dos componentes são feitos com outra ferramenta do Orcad, chamada Layout Plus. A Figura 33 mostra um exemplo de *layout* feito com o Layout Plus.

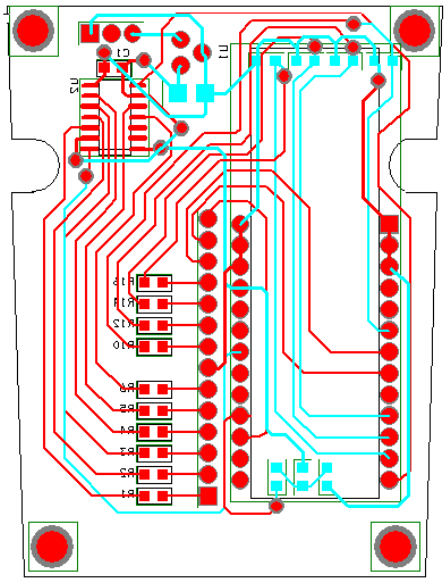


Figura 33 - Layout do Controle Remoto

Depois de pronto o layout é preciso corroer a placa. Existem vários tutoriais na internet que explicam como fazê-lo.

O primeiro passo é imprimir o layout mostrado na Figura 33, preferencialmente numa impressora a laser, pois o *tonner* é passado à placa com mais facilidade que a tinta de uma impressora jato de tinta, melhorando a nitidez das trilhas e, preferencialmente também, num papel fotográfico. Empiricamente, o papel com que se obtiveram os melhores resultados foi o papel gloss. Também se deve tomar cuidado para imprimir com o espelhamento correto. Em princípio, se a placa tiver 2 camadas, é preciso separar as camadas. Em outras palavras, é preciso imprimir, separadamente, a camada *bottom* (camada inferior) e a camada t*op* (camada superior). Para a camada top, pode-se intuir que, depois de impresso, o desenho ficará invertido ao pousarmos a face da folha na face da placa e, portanto, o desenho deve ser impresso espelhado. Já a camada *bottom*, como ela deve estar espelhada em relação à camada *Top*, deve-se imprimir sem espelhamento e quando tiver a face disposta sobre a placa, o espelhamento será intrínseco.

A placa utilizada é a de fibra de vidro que retorna um resultado melhor que as comuns de fenolite, apesar de essa última ser mais fácil de lidar, principalmente, com

relação à furação. Deve-se limpar muito bem a placa com uma esponja de aço para ficar completamente livre de gordura. Depois de imprimir uma camada e colocá-la sobre a placa, pressiona-se o desenho com um ferro de passar bem quente, fazendo o papel de catalisador na transfusão da tinta do papel para a placa. Aplica-se esse processo dos dois lados. Eventualmente alguma trilha pode não sair bem feita na placa, para corrigi-la basta utilizar uma caneta de retroprojetor.

Depois das trilhas e ilhas de solda terem sido impressas no cobre da placa, é preciso fazer a corrosão, ou seja, retirar toda a parte metálica da placa que não fará parte do circuito, ficando metalizada apenas a área sob a tinta do layout.

Caso necessário, os contornos dos componentes podem também ser impressos na placa, de modo a facilitar a montagem dos mesmos. Para isso, basta repetir os passos descritos anteriormente, utilizando a camada de silk do layout.

Antes de chegar ao layout final do equipamento, um protótipo mais simples é feito para poder validar as funcionalidades de software. Por meio das Figura 34 a Figura 38 pode-se ter uma idéia da miniaturização necessária dos componentes e da PCI para que o protótipo fique com aspecto de produto final, permitindo sua comercialização.

Figura 34 - Coordenadores

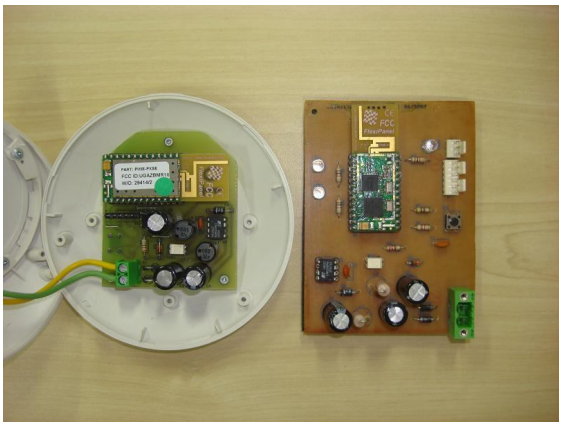


Figura 35 - PCI's dos Coordenadores

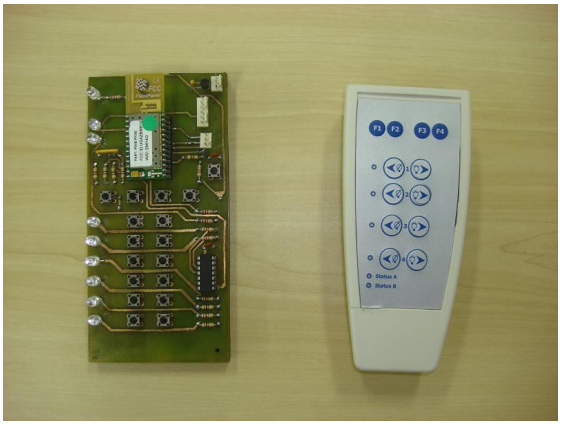


Figura 36 - Controles Remotos

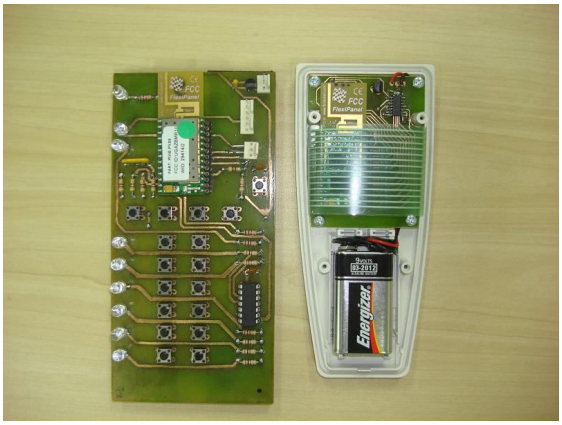


Figura 37 - PCI's dos Controles Remotos

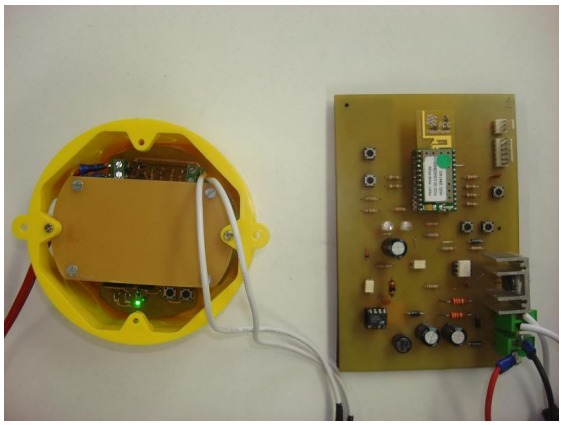


Figura 38 – *Dimmers*

Durante o projeto, foram confeccionadas cerca de 15 PCIs, sendo que a maior parte foi feita em dupla face. Em alguns casos, devido à necessidade de miniaturização dos componentes, especialmente no controle remoto, foram utilizados vários componentes SMD, o que aumentou a complexidade da confecção da PCI.

3.9.1. Coordenador/TCP_IP

O coordenador, como já dito anteriormente, é o nó principal da rede. Inicialmente, projetou-se um nó que possuísse apenas a funcionalidade de Coordenador, para uma rede mais simples, formada, por exemplo, por um coordenador, um controle remoto e um *dimmer*. Este coordenador pode ser visto nas Figura 39 a Figura 44.

Figura 39 – Coordenador: fator dimensional comparativo

Figura 40 – Coordenador: Vista Frontal

Figura 41 – Coordenador: Vista Lateral

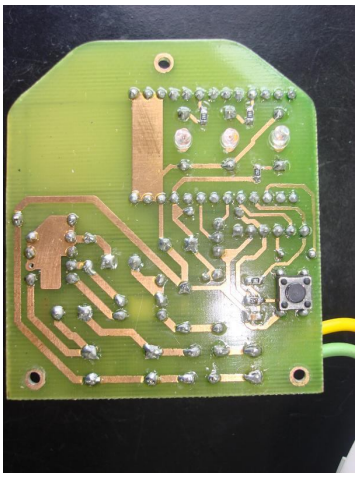


Figura 42 – Coordenador: *Bottom* da PCI

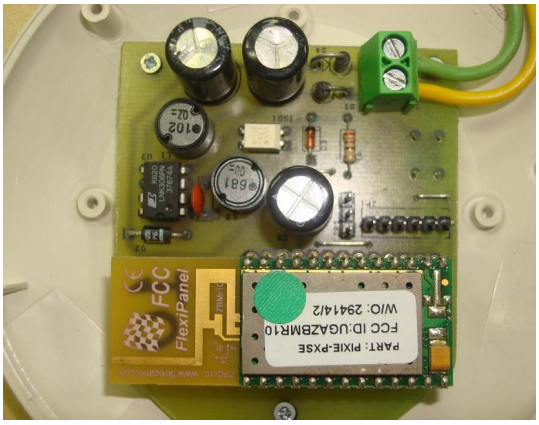


Figura 43 – Coordenador: *Top* da PCI

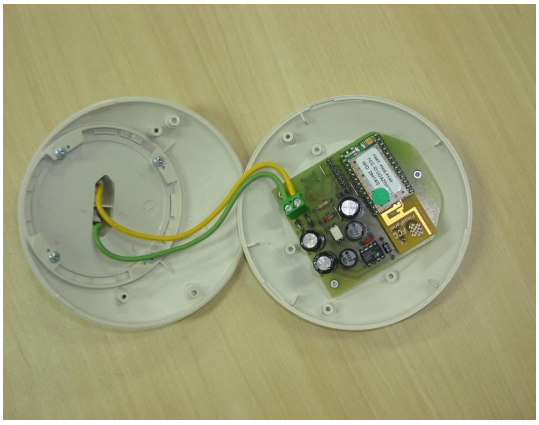


Figura 44 - Coordenador Aberto

Além desse tipo de nó, foi também desenvolvido o Coordenador/TCP_IP (Figura 54), para o contato da rede ZigBee com o "mundo externo" através da rede TCP/IP, permitindo que este nó aloque páginas de internet para serem acessadas de qualquer lugar do planeta. Na Figura 45 e na Figura 46, porém, pode-se visualizar o coordenador/TCP_IP que foi montado utilizando kits de desenvolvimento da Microchip (Figura 55), sendo eles o PICDEM Net, do lado esquerdo, para desenvolvimento do software TCP/IP e o PICDEM Z, do lado direito, para o desenvolvimento do software ZigBee.

Além destes dois kits, foi montado um RTC (*Real-Time Clock*) para se poder atualizar, em tempo real, os horários de mudança de *status* dos nós da rede ZigBee na página de monitoramento.

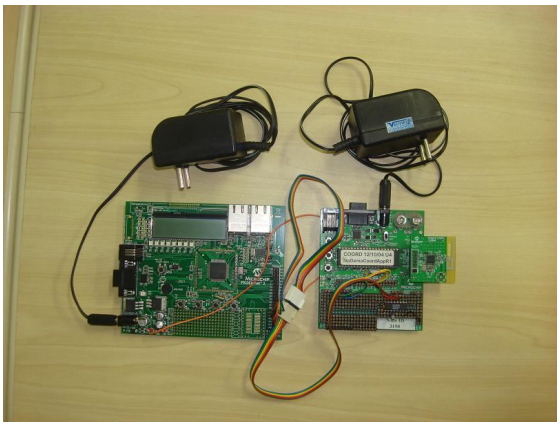


Figura 45 - Coordenador TCP/IP

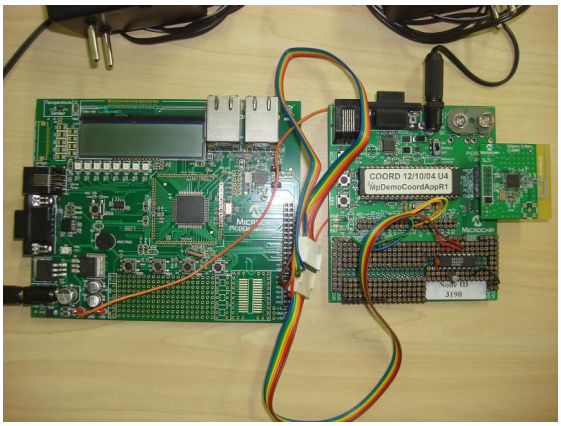


Figura 46 - Kits de desenvolvimento

3.9.2. *Dimmer*

Seguindo a idéia de embutir o hardware ZigBee, o *dimmer* foi criado de tal forma que pudesse ser encaixado dentro de uma caixa octogonal comum de teto utilizado em residências. O gabinete octogonal e a PCI do *dimmer*, podem ser bem visualizados na Figura 47 a Figura 49.

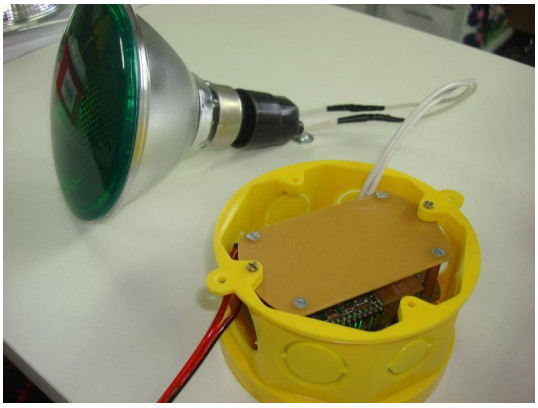


Figura 47 – *Dimmer*

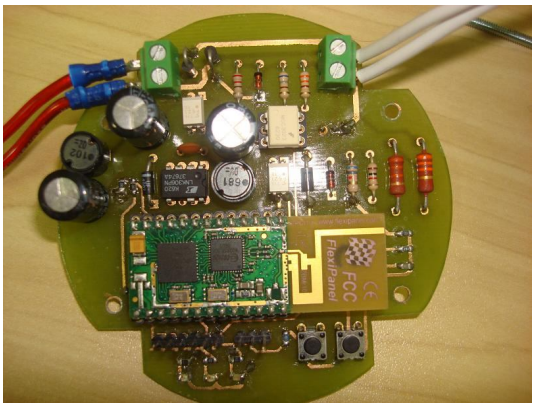


Figura 48 – *Dimmer*: *Top* da PCI

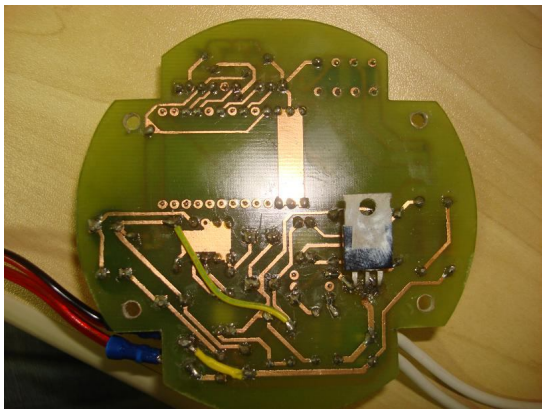


Figura 49 – *Dimmer*: *Bottom* da PCI

3.9.3. Controle Remoto

É um dispositivo do tipo RFD (*Reduction Function Device*) e, portanto, alimentado com pilhas ou bateria. É dotado de uma série de botões para que possa fazer um controle bem amplo por todos os cômodos da casa. Optou-se, por questões de estética e espaço, por utilizar um teclado de membrana. A PCI, contendo apenas componente SMD, ficou bem reduzida, sendo dimensionada para caber num gabinete produzido pela OKW, que já possui um espaço reservado para a bateria (Figura 50 a Figura 53).

Figura 50 - Controle Remoto: efeito comparativo

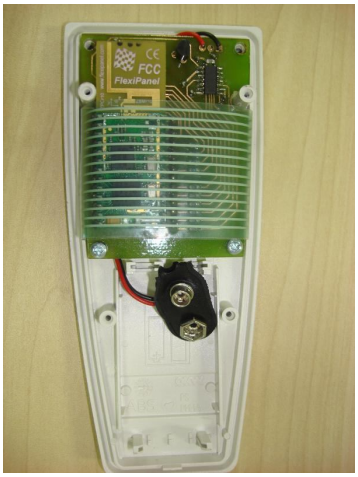


Figura 51 - Controle Remoto aberto

Figura 52 - Controle Remoto: *Top* da PCI

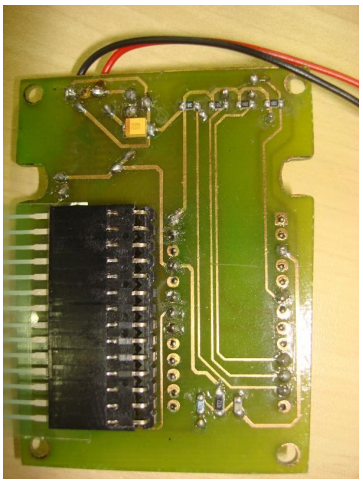


Figura 53 - Controle Remoto: *Bottom* da PCI

3.9.4. Biometria

A biometria é dotada de um leitor digital e botões para interface e mensagens ZigBee. Esse nó também está apenas em fase de protótipo, mas com a funcionalidade já testada e aprovada (Figura 54).

A idéia de agregar a Biometria na rede ZigBee surgiu em paralelo a outro produto, já comercializado pela N3E, para controle de acesso de arquivos eletrônicos. Assim, já tendo o conhecimento do hardware do leitor biométrico usado no controle de acesso, optou-se por agregar essa propriedade também ao ZigBee.

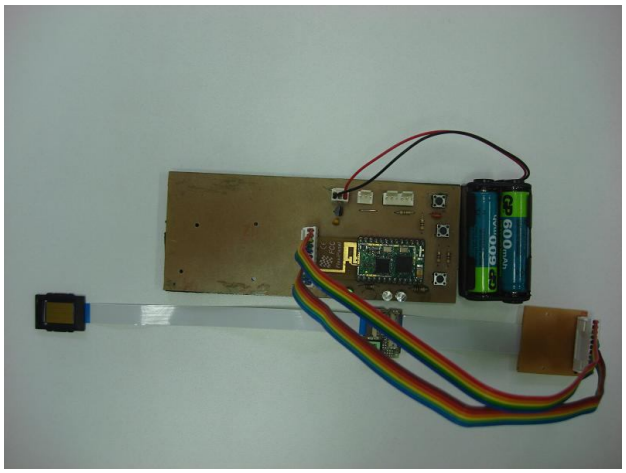


Figura 54 – Biometria

3.9.5. Atuador

Visto na Figura 55, é o par direto da biometria. Serve com um delimitador de ambientes, funcionando como um cofre ou uma fechadura. A abertura é controlada por relés, possuindo botões para a rotina de *bindings.*

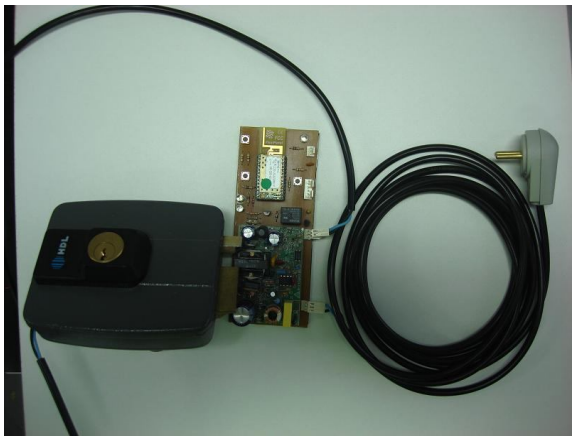


Figura 55 – Atuador

3.9.6. Sensor

É o principal alvo de monitoramento da rede. Independentemente do que irá controlar é ele que determinará os *status* mais visados do usuário. Sensores de presença para alertar um possível arrombamento, sensores de temperatura para detectar incêndios e as mais variadas propriedades que se queira controlar. Ele manda mensagens

diretamente ao coordenador para que esse possa dispor a informação de acionamento no site.

Ainda não foi definido um hardware final. Por ora, foi utilizado o próprio kit da Microchip (PICDEM Z), já mostrado anteriormente, para funcionar como Sensor ZigBee (no kit existe um sensor de temperatura que se comunica via SPI). Falta portanto, definir outros tipos de sensores para se ter uma rede bem dinâmica com vários parâmetros de ambiente.

3.9.7. Interruptor

Durante os testes e a utilização, analisando criticamente, não só com a visão de desenvolvedor, mas também assumindo o lugar do usuário do sistema, surgiu a necessidade de se criar um "controle remoto fixo", ou seja, botões que pudessem ser utilizados para acionar cargas que ficassem dentro do próprio cômodo ou em locais adjacentes, posicionados onde o usuário tivesse fácil acesso.

Foi então criado um controle remoto mais simples, que pudesse ser embutido na parede, fazendo o papel de um interruptor. Esse controle remoto pode ser visto na Figura 56 e Figura 57.

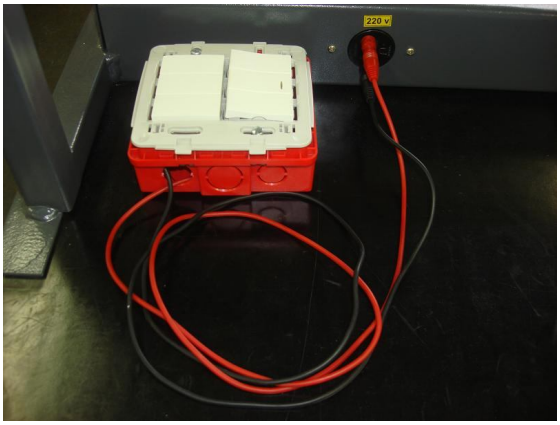


Figura 56 - Interruptor Fechado

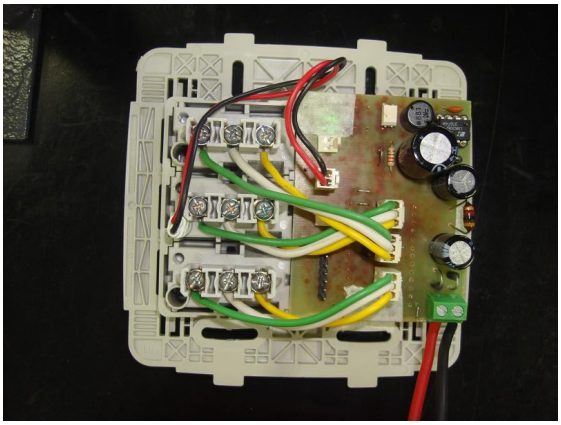


Figura 57 - Interruptor Aberto

3.9.8. ZigBee/GPRS

Esse nó, assim como a biometria, foi idealizado junto com um projeto paralelo feito com o Departamento de Enfermagem da UFSCar. A proposta da Universidade Federal era uma parceria em que a N3E desenvolveria um dispositivo para monitoramento de idosos com mal de Alzeihmer que fosse discreto para não constranger os usuários.

O hardware foi baseado num dispositivo que já era utilizado pela universidade, o LPP da Redecamp que, além de fornecer os localizadores, também disponibilizava um portal para o monitoramento das coordenadas.

Teve-se, assim, que se desenvolver o dispositivo e também disponibilizar a visualização das coordenadas a um custo mais razoável, comparado ao que era oferecido pela Redecamp.

A princípio, foi desenvolvido um protótipo que é o que está sendo utilizado para os testes com a rede ZigBee, como pode ser visualizado na Figura 58 e Figura 59. Os nós comunicam-se pela serial.

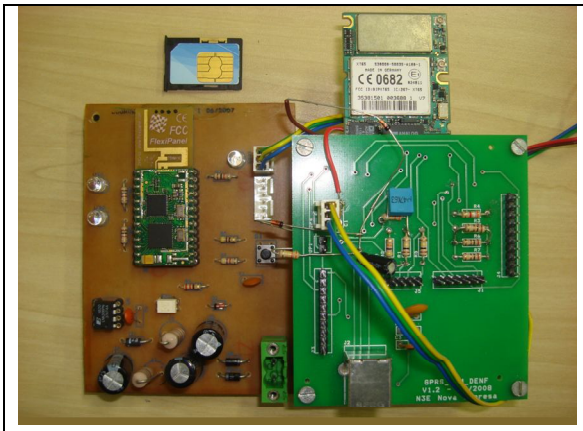


Figura 58 - ZigBee/GPRS

Figura 59 - Antenas GSM (superior) e GPS (Inferior)

Mais tarde, depois de validar o protótipo, foi feita a PCI final (Figura 60 e Figura 61) para ser colocada num gabinete discreto, como bolsas e cintos e que será a base para a confecção da PCI final para o dispositivo ZigBee/GPRS.

Tanto a placa protótipo quanto a placa final, foram confeccionadas pela empresa Micropress [38].

Figura 60 - Placa final + bateria do localizador da UFSCar

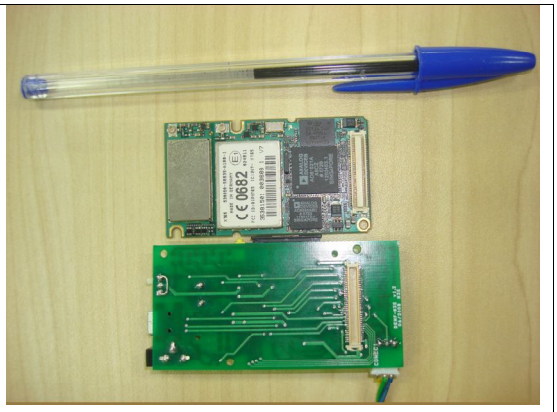


Figura 61 - Parte traseira da placa final + XT65

CAPÍTULO 4

Software

Neste capítulo são descritos os vários softwares que compreendem o núcleo desta tese de mestrado. Eles tiveram que ser escritos em diversas linguagens para implementar a pilha ZigBee, as páginas de Internet, o módulo GPRS, a conexão TCP/IP, etc.

A parte inicial do trabalho foi apresentada em 2007, na *IEEE-SBMO International Microwave and Optoelecttronics Conference* (IMOC), [P1]. O artigo, publicado nos anais da conferência, aborda a concepção básica da rede, com fotos dos primeiros protótipos e sem integração com nenhuma outra tecnologia, mas devido ao seu caráter inovador foi escolhido para uma apresentação oral na conferência.

As páginas de Internet, relacionadas ao servidor interno, já estão no formato final, apesar de ainda faltarem algumas implementações, principalmente a parte relacionada à possibilidade de configurações pelo site, como o *setpoint* de sensores.

Com relação ao servidor externo, as páginas foram criadas para atender ao plano proposto para o servidor de monitoramento do departamento de enfermagem da UFSCar para os idosos com Mal de Alzheimer e, tendo obtido boa aceitação para esse fim, também não terão mudanças drásticas, inclusive, foi a partir desse projeto que surgiu a idéia de integrar ZigBee, GPRS e GPS. A proposta era a de que os idosos fossem monitorados o tempo todo, afinal, pessoas com Mal de Alzheimer, têm perdas muito bruscas de memória que não permitem lacunas no monitoramento. O tráfego de dados pela rede GPRS é, no entanto, tarifado e o envio de coordenadas durante 24 horas por dia, 7 dias por semana pode representar um custo proibitivo. Diante dessa situação, surgiu a solução de usar o ZigBee, em detrimento do GPS, enquanto os idosos estivessem em zonas de segurança, representadas pelo próprio lar, por casas de repousos, etc... O ZigBee poderia ser usado, dentro desse ambiente, para monitorar os sinais vitais dos idosos e, a partir do momento em que deixassem as zonas de segurança, as coordenadas GPS voltariam a ser enviadas. Apesar dessa proposta não ter sido implantada, ela impulsionou a idéia desse monitoramento onipresente que pareceu relevante para o trabalho apresentado aqui.

Portanto, com as páginas de monitoramento já prontas, os maiores desafios estão nos softwares dos micro-controladores: o PIC18F4620, da pilha ZigBee; o PIC18F97J60, da pilha TCP/IP e o ARM7, do módulo GPRS.

4.1. Páginas Web

Como já dito, a rede possui 2 servidores com características bem distintas. Enquanto o servidor interno tem grande limitação de memória, uma vez que todo processamento é feito por um único microcontrolador, podendo hospedar apenas páginas bem simples, escritas unicamente em HTML e Javascript, o servidor externo é dotado de todos os recursos disponíveis para programação para Web, com disponibilidade de módulos PHP e MySql, não havendo limitações gráficas para construção das páginas armazenadas. Nessa seção serão mostradas as páginas presentes em ambos os servidores, bem como o processo de transferência do código escrito para essas páginas, mediante seus servidores FTP.

4.1.1. Servidor Interno

É o responsável pelas atualizações da rede ZigBee. Apresenta certa limitação de memória, mas para as pretensões do projeto, ele é mais que suficiente, com dimensão física de seu hardware compensando a limitação gráfica de suas páginas.

4.1.1.1. Servidor FTP (*File Transfer Protocol*)

O FTP é o protocolo para transferência de arquivos. Ele será usado exatamente para que se possa transferir os códigos fonte e figuras das páginas Web, sendo gravadas na memória do hardware. No entanto, essas informações devem ser salvas num formato consistente para que o micro-controlador, depois de buscar as informações da memória, possa interpretá-las e executá-las com as propriedades corretas. A rigor, o que será feito é transformar um arquivo numa imagem com um formato padrão chamado MPFS (*Microchip File System*). Essa imagem pode ser gerada com a ajuda de um programa (mpfs.exe), que também está disponível para *download*, executado através do *prompt* de comandos. Para a aplicação desse trabalho, em particular, a imagem será transformada num arquivo binário (arquivo.bin). Dependendo da extensão do arquivo, um byte correspondente será gerado no arquivo.bin. Um arquivo HTML terá cada um dos seus caracteres transformados no byte correspondente segundo o padrão ASCII. Já arquivos

.gif, referentes a imagens, terão os pixels transformados em bytes segundo uma biblioteca de cores pré-definida no software.

A transferência da imagem transformada (*upload*), ou seja o arquivo.bin, é feita pelo *prompt* de comandos. É implementada uma autenticação com *login* e senha e o servidor é completamente compatível com a imagem MPFS, sendo a transferência realizada com um simples comando "*put*".

Apenas uma conexão por vez é permitida, sendo usados 2 *sockets* durante a operação e desabilitando-se os *sockets* durante a mesma para qualquer outra funcionalidade, portanto, se uma conexão estiver estabelecida com um cliente HTTP, ela será interrompida até o processo de transferência terminar. Também é estabelecido um limite de tempo (*time-out*) para que uma conexão "órfã" ou uma conexão problemática, que tenha ocorrido durante o *upload*, não ocupe o servidor FTP indefinidamente.

4.1.1.2. Servidor HTTP [12]

Depois de feito o carregamento ascendente (*upload*) das páginas WEB pelo FTP e adquirido um endereço IP, estática ou dinamicamente, o servidor HTTP já está apto a receber conexões. Lembrar que o endereço recebido numa rede interna é um IP privado (ex: 192.168.1.1) e não é o que será visível aos usuários de fora da rede. Os usuários da rede mundial de computadores só "vêem" o IP real que pertence ao *modem* da rede interna. Para direcionar a requisição HTTP de um usuário externo ao servidor, é necessário criar uma regra NAT (*Network Address Translation*) no roteador, amarrada a uma porta (a porta tradicional de servidores HTTP é a 80). Sendo assim, também é aconselhável que um usuário que deseje fazer o monitoramento remoto adquira um serviço de IP real fixo junto à sua operadora, pois os endereços atribuídos dinamicamente via DHCP mudam constantemente, não sendo possível prevê-los. Além disso, é necessário atribuir um endereço privado, também fixo, pois a regra NAT será baseada nesse endereço.

Ao requisitar o IP real, que será interpretado e encaminhado ao IP interno pela regra NAT criada, o usuário irá visualizar uma página inicial definida no código fonte. Essa página, que já foi transferida por *upload* através de FTP, será buscada pelo micro-controlador em sua EEPROM (*Electrically-Erasable Programmable Read-Only Memory*) e transmitida pelo *socket* TCP até o *browser*, permitindo a visualização pelo usuário.

O *browser* interpreta que se trata da linguagem HTML, portanto, as seqüências de bits (*strings*) especiais que correspondem a comandos HTML, são corretamente representadas na tela. Notar que a simplicidade do servidor impede a instalação e o uso de recursos avançados de programação para a web como o PHP e o SQL, assim toda a parte interativa da página, como chamada de funções, tem que se valer, exclusivamente, do Javascript.

Um recurso conveniente de um sítio (*site*) de monitoramento é a visualização dinâmica do *status* de sua rede. Por exemplo: o monitoramento da pressão instantânea de um duto de gás - essa informação tem que ser atualizada instantaneamente. Para tanto, o servidor suporta páginas dinâmicas, capazes de atualizar o *status* constantemente. Isso é feito chamando arquivos CGI (*Common Gateway Interface*) dentro das páginas HTML por funções Javascript. Dentro desses arquivos são colocados caracteres especiais que, depois de transferidos via TCP, são interpretados pelo software do micro-controlador, dando-lhe a informação de que o *status* precisa ser atualizado. Portanto, o procedimento se faz da seguinte maneira: no evento de carregamento (*onload*) de uma página HTML é feita a leitura do arquivo CGI, contendo um caractere especial e um código para identificação do comando (Ex: %22). Essa *string* é interpretada pelo micro-controlador e o comando é realizado. O resultado é passado a uma *string* interna que é transmitida via TCP e salva em uma pseudo variável Javascript, reconhecida pelo HTML mediante um ID. Assim, ao darmos o comando:

- Loading...

o *browser* mostra a *string* salva na tela.

Um sistema de monitoramento que permite atuação remota necessita de segurança. Dessa forma, o servidor contará com uma senha de autenticação *default* que poderá ser mudada pelo próprio *site*. A mudança de senha, sua autenticação e os comandos de atuação são dados por eventos de *onclik* nos botões correspondentes. Os comandos são passados por método POST com os caracteres especiais que identificam esse método, junto com o código do comando e atributos adicionais que o eventual comando possa ter. No software, cada campo é interpretado e a decisão correta é tomada.

A. Página Login

É a página inicial, a página que será exibida (Figura 62) quando o IP real em conjunto com a porta específica, for digitado no *browser*.

Figura 62 - Página Login do servidor HTTP interno

Como mostrado na Figura 62, a página tem dois campos de preenchimento. Para que a página de monitoramento possa ser visualizada, é necessário que a senha esteja coincida com a que está salva na memória do hardware. O *login* serve, unicamente, para diferenciar as conexões. Como já foi dito, é possível que uma única conexão ocupe toda a memória do *socket* e isso deve ser corrigido na aplicação. Propôs-se, então, limitar o número de *logins* simultâneos em apenas dois[8], sendo um com visualização dinâmica (atualização constante do *status*) e o outro com visualização estática. O que for digitado no campo "Login" será passado pela URL e o software poderá distinguir quem está visualizando a página dinâmica e quem está visualizando a página estática. Caso um terceiro usuário tente se conectar, ele será advertido com a informação de que o servidor já está repleto.

O *link* "trocar senha" encaminha o usuário a uma página de alteração da senha atual.

Ao apertar o botão "Login", o comando passado pelo método *POST* carrega, como atributo, 2 campos: o campo Senha, que é comparado à senha atual, possibilitando uma advertência ao usuário se a senha estiver errada; e o campo Login, usado para a autenticação do usuário, permitindo uma notificação ao mesmo se a visualização tiver que ser estática, em razão de um outro usuário já conectado, ou ainda uma notificação

[8] - O servidor só ficaria comprometido com mais visualizações dinâmicas, pois o processamento não é rápido o suficiente para dar uma atualização satisfatória a múltiplos usuários. Visualizações estáticas requerem apenas uma única abertura de *socket* TCP, no entanto, se for permitido um número irrestrito de usuários e se, por ventura, vários deles resolverem atuar (requisitar um acionamento de *dimmer*, por exemplo) ao mesmo tempo, essa ação pode derrubar o servidor.

de servidor repleto se 2 usuários já estiverem conectados ao servidor (são permitidos apenas 2 usuários por vez, sendo uma visualização dinâmica e uma estática).

B. Página Senha

Na página senha (ilustrada na Figura 63), três campos têm de ser preenchidos: a senha antiga, a nova senha e a confirmação da nova senha. Os três campos são passados como atributos do comando para alterar senha. Se eles forem coerentes, há uma notificação do sucesso, caso contrário uma mensagem de alerta é exibida.

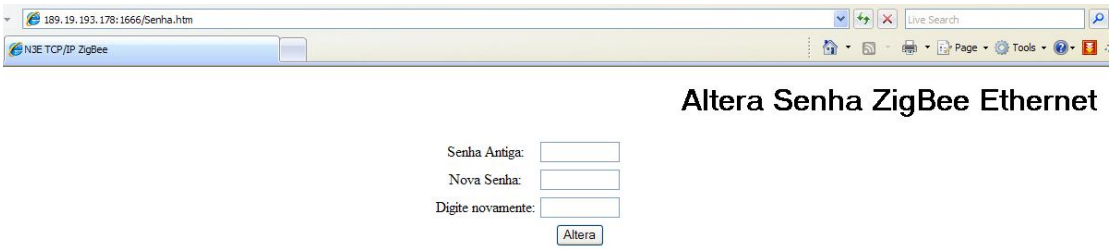


Figura 63 - Página para alteração de senha do servidor HTTP interno

C. Páginas Index e Estático

Essas são as páginas de monitoramento. A diferença entre elas já foi explicada. A página Index.htm terá uma visualização dinâmica e a atualização do *status* dos nós será feita automaticamente enquanto o usuário não fechar a página. A página Estático.htm terá visualização estática e uma eventual mudança no *status* da rede só poderá ser percebida se for dado um comando de atualização (*refresh*) na página. Também existe um limite de tempo (*time out*): o visualizador da página Estática é obrigado a atualizar a sua página em menos de 2 minutos, caso contrário o seu tempo expira e se quiser voltar a visualizar a rede, será obrigado a conectar-se novamente. O *time out* da página dinâmica é curto. Uma vez que a atualização do *status* é quase instantânea, o *time out* só irá ocorrer se o usuário tiver fechado a página. Caso isso ocorra e exista um segundo usuário com visualização estática conectado, a visualização dinâmica lhe será passada automaticamente na próxima vez em que esse atualizar sua página e, conseqüentemente, o *login* para a página estática ficará vago para um usuário futuro.

Nessas páginas existirá uma tabela com as informações dos nós com os campos:

- ID : será usado para identificar a que nó o comando será enviado.
- Endereço: é o endereço do nó na rede ZigBee.

- Tipo: é o tipo de nó de que se trata (*Dimmer*, Coordenador, Sensor, etc.)
- Nome: nome usado para detalhar o nó ZigBee (Ex: luz_corredor)
- *Status*: é o *status* atual que terá sentido apenas para nós com a propriedade de alternar sua condição, como o *dimmer* e o sensor.
- Ultimo Dado: data e hora da última atualização (qualquer alteração como o *status*, a entrada na rede, etc...). Essa informação será pega do RTC (*Real Time Clock*) implementado em hardware.

Haverá os campos de texto "índice" e "nome" e os botões "ON", "OFF" e "DIMMER".

O *link* "SEGURANÇA" encaminha o usuário para a página Crip.htm, que será mostrada mais adiante. Todas as configurações de chave criptográfica e aceitação de novos nós na rede serão feitas por essa página.

O campo "índice" servirá para encaminhar o comando ao nó correto. Esse campo deve ser preenchido com o ID da tabela, correspondente ao nó que se deseja acionar.

O campo "nome" servirá para detalhar os nós da rede. Colocando-se o ID da tabela, preenchendo o nome e clicando no botão "Renomear" altera-se o campo "Nome" da tabela.

Os botões "ON", "OFF" e "DIMMER", servirão para o acionamento, sendo que, para o botão DIMMER, é necessário escolher um nível de operação do *dimmer*, que é um dispositivo utilizado para variar a intensidade de uma lâmpada. Ele consiste de graduadores que, através da diminuição ou aumento da tensão e, portanto, um aumento da potência média de uma lâmpada, controlam a intensidade da luz produzida pela mesma. Assim, ele só irá funcionar para os ID's que corresponderem a nós do tipo "Dimmer". A Figura 64 mostra a página Index.htm.

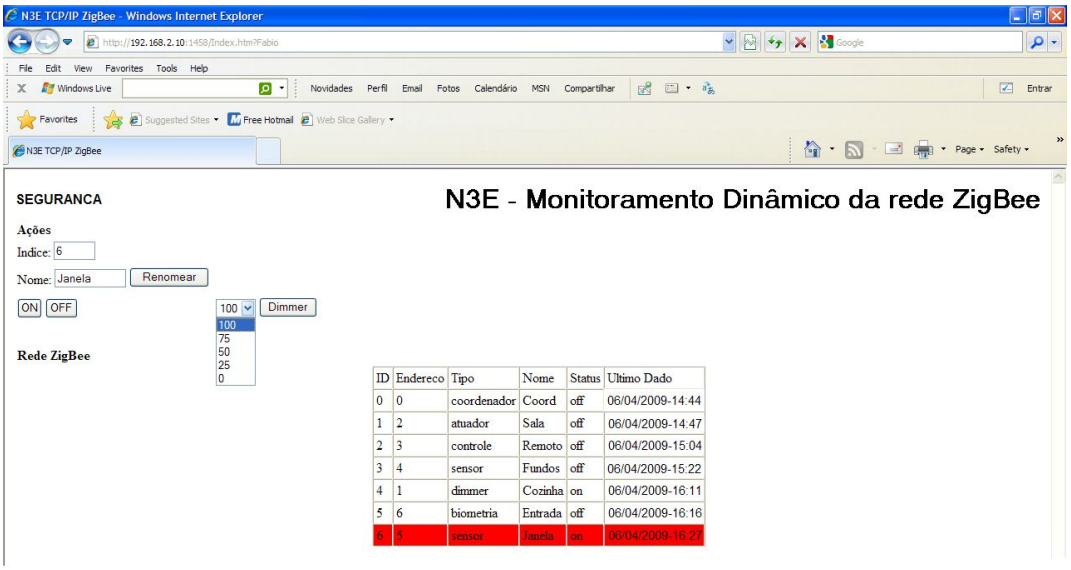


Figura 64 - Página Index.htm

4.1.2. Servidor Externo

É o responsável em receber e armazenar as coordenas GPS enviadas pelo nó ZGPRS. Possui algumas ferramentas facilitadoras para transferência de arquivos e criação de tabelas no banco de dados como o PHPMyAdmim.

4.1.2.1. FTP Externo

A hospedagem do site da N3E inclui o servidor FTP, imprescindível para a transferência dos códigos fontes de suas páginas. Como qualquer servidor FTP, é protegido por senha.

Para acessar um servidor FTP pode-se optar por linhas de comando num prompt ou, de uma maneira mais agradável, pelo próprio navegador, especificando o protocolo FTP que será utilizado: ftp://www.n3e.com.br o que resultará na página indicada na Figura 65.

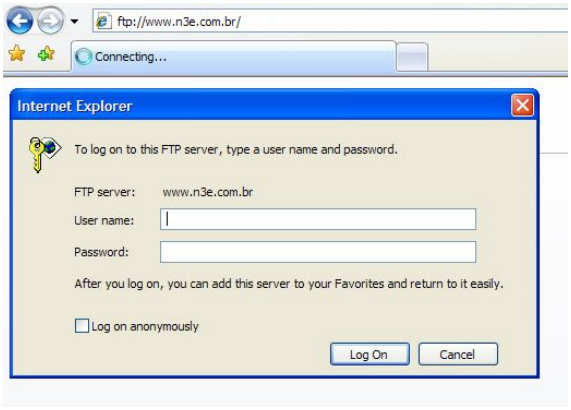


Figura 65 - Autenticação do usuário para o acesso ao servidor FTP externo

Existe a opção de visualizar as pastas de maneira bem similar à interface do Windows Explorer, facilitando bastante o trânsito de arquivos, bastando arrastá-los para a pasta correta, como ilustrado na Figura 66.

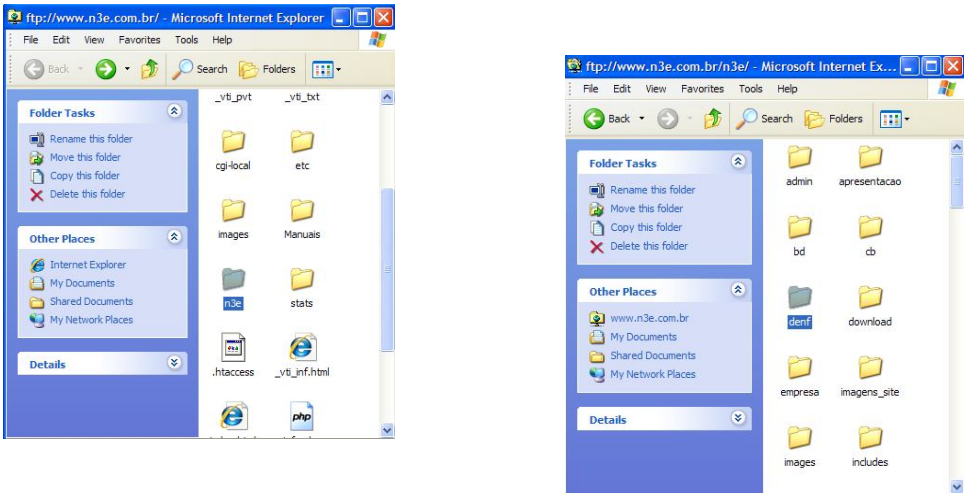


Figura 66 - Diretórios dos códigos fontes das páginas PHP/HTML

Os arquivos (.php) são salvos em diretórios e subpastas (Figura 67) e constituem o caminho da URL que será acessada no Browser, ou seja, para acessar as coordenadas referentes aos testes deste projeto, por exemplo, deve-se digitar no navegador: www.n3e.com.br/n3e/denf/login_XT.php.

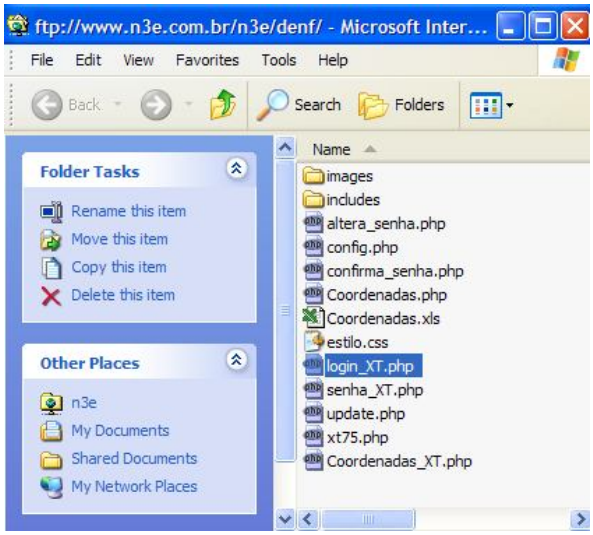


Figura 67 - URL completa para o acesso à página inicial de login do servidor HTTP externo

4.1.2.2. HTTP Externo

O servidor HTTP externo é o responsável pela interface com o usuário. Através de páginas PHP em conexão com o banco de dados, um usuário, desde que tenha a senha de acesso, pode visualizar, de qualquer local do mundo, por um equipamento com conexão à Internet, a localização do módulo ZGPRS que, como já foi salientado, estará ativo a partir do momento que se desvencilhar da rede interna ZigBee.

Todo módulo GPRS tem seu próprio IMEI, que, ao sair da linha de produção, será previamente cadastrado com um *Login* fixo e uma senha temporária *default*. O envio de uma coordenada nada mais é que o estabelecimento de uma conexão, como ocorre num *browser* qualquer. O módulo terá que ser configurado para conectar-se a um endereço específico, que corresponderá a uma página no servidor HTTP, responsável por tratar os dados e salvá-lo no banco de dados. A página a ser configurada é:

www.n3e.com.br/n3e/denf/xt75.php [9]

Todos os dados são passados via URL, inclusive o IMEI do módulo que norteará o destino da gravação.

Essa página irá quebrar a String que é enviada pela URL nos seus campos correspondentes, fazendo os ajustes necessários para correção de fuso horário, padrões de data, etc...

[9] Os testes foram desenvolvidos tanto com módulos xt65 quanto com módulos xt75, por isso a nomenclatura da página é xt75.php. A única diferença entre os dois é que o xt75 tem disponível uma tecnologia de transmissão de dados mais rápida e eficiente que o GPRS puro. É o chamado EDGE que é uma forma aperfeiçoada do GPRS

A URL passada tem o seguinte padrão:

http://www.n3e.com.br/n3e/denf/xt75.php?imei=353815010036881&coord=2008/06/12,13:04:40,21.9920996,S,047.8828516,W,678,003.99,3"

Como pode ser visto, os valores que são passados às variáveis PHP na página referida são o “imei” e o “coord”.

O “imei” é intuitivo e corresponde ao IMEI do módulo, já o “coord” passa o restante das informações, sendo cada campo separado por vírgulas, usadas como caractere delimitador para compor os campos da tabela. O “coord” é dividido da seguinte maneira:

Data	Hora	Latitude	Hemisfério	Longitude	Hemisfério	Altitude	Velocidade	Status

- Data: padrão AAAA/MM/DD
- Hora: HH:MM:SS
- Latitude: formato graus decimais, onde cada grau é dividido em frações decimais.
- Hemisfério: Norte (N) ou Sul(S).
- Longitude: formato graus decimais, onde cada grau é dividido em frações decimais.
- Hemisfério: Leste (E) ou Oeste (W).
- Altitude: com relação ao nível do mar, medida em metros (m).
- Velocidade: medida em km/h.
- *Status*: é referente ao número de satélites que o módulo está enxergando. A coordenada só será transmitida se o s*tatus* for 2 (2 dimensões) ou 3 (3 dimensões).

4.1.2.3. Páginas PHP

A. login_XT.php

Para visualizar as coordenadas o usuário terá primeiro que conectar-se, não sendo possível o acesso direto à página onde a tabela se encontra, pois é feita uma verificação de conexão antes que a página seja mostrada. Portanto, o usuário terá que acessar (Figura 68): www.n3e.com.br/n3e/denf/login_XT.php

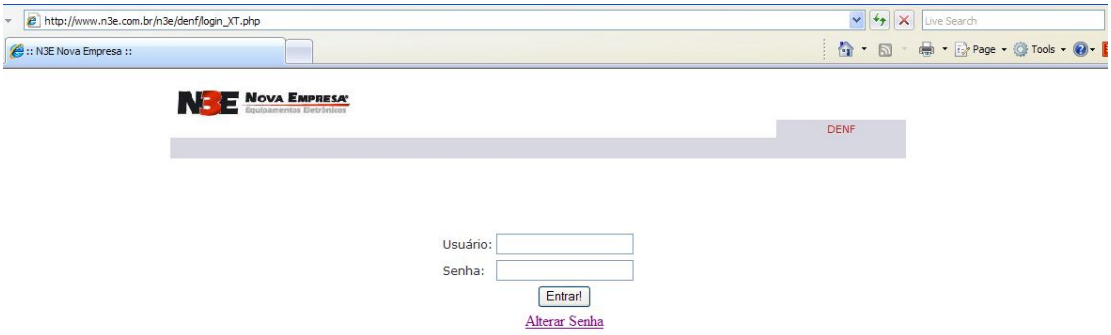


Figura 68- Página de Login do servidor HTTP externo

Nessa página, o usuário terá que fornecer o *Login*, que é fixo e amarrado ao IMEI do equipamento e uma senha *default* que pode ser alterada pelo usuário clicando em "Alterar Senha". Ao preencher os campos e dar um "*submit*" no botão "Entrar", os dados são passados por métodos POST a uma página "secreta" (senha.php), que consulta o banco de dados e compara os dados preenchidos com os dados contidos na tabela *Login* do MySQL. Se os dados forem coerentes, o usuário é encaminhado à página com a tabela de coordenadas, caso contrário recebe um alerta de erro e é reencaminhado à página de *Login*.

B. altera_senha.php

Como as coordenadas só dirão respeito ao usuário detentor do módulo GPRS, ele tem a opção de alterar a senha *default* por outra que só ele irá saber. Assim, quem alterá-la terá que informá-la às outras pessoas interessadas na localização do módulo. Como é de praxe em qualquer cadastro de mudança de senha, o usuário deve fornecer o seu *Login*, a senha antiga, a nova senha e a confirmação da nova senha na página vista na Figura 69.

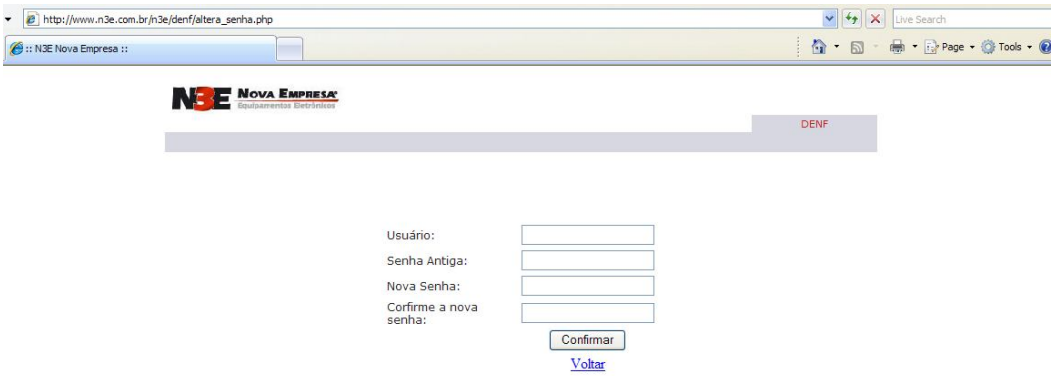


Figura 69 - Página para alteração de senha do servidor HTTP externo

Ao preencher os dados, eles são passados a uma página secreta (confirma_senha.php), que irá vasculhar a tabela *Login* do banco de dados e comparar os dados de Login e Senha com os que já estavam cadastrados, em seguida, será feita a verificação de igualdade dos campos de nova senha e de sua confirmação. Caso os dados confiram, o usuário é encaminhado à página de *Login*, onde já irá conectar-se com a nova senha. Se a senha antiga ou o usuário estiver errado, o usuário é advertido com uma alerta de erro de senha, caso a senha esteja correta, mas os campos de nova senha e confirmação não forem iguais, o usuário também é alertado sobre esse erro.

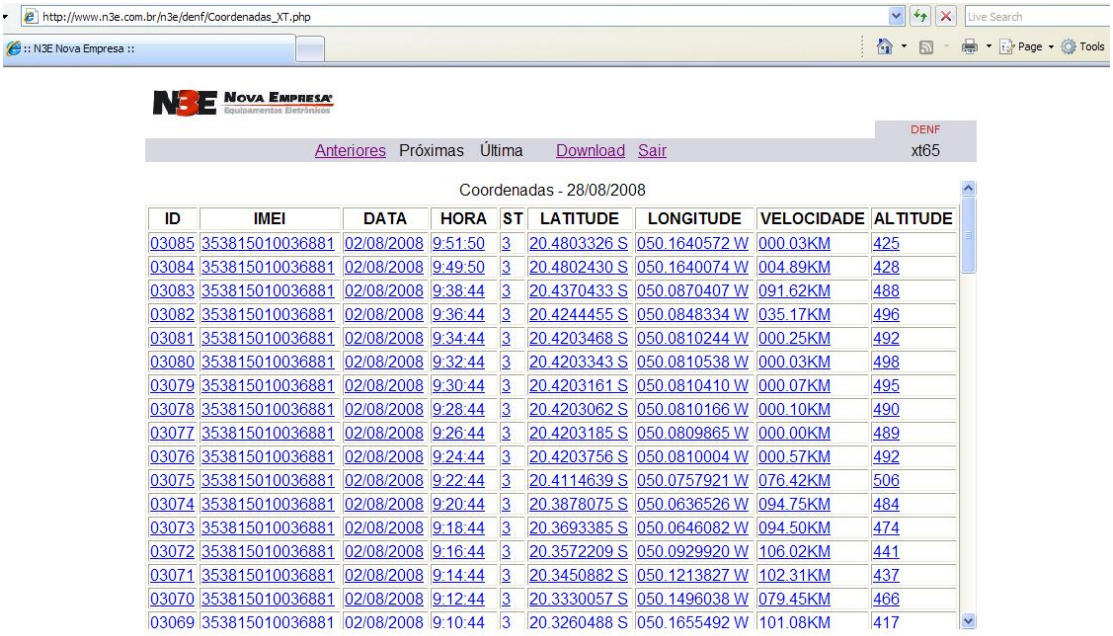
Ao clicar no *link* "Voltar", o usuário é encaminhado à página de *Login*.

C. Coordenadas_XT.php

Por fim, a tabela com as coordenadas. Os dados que foram passados pelo módulo à página xt65.php foram salvos na tabela correspondente ao seu IMEI no banco de dados. Ao conectar-se, o usuário fornece seu *login*, cujo nome é idêntico à tabela de coordenadas correspondente do módulo. Assim, ao abrir a página "Coordenadas_XT.php", uma leitura é feita na tabela xt65 (para esse caso) e seus dados são mostrados numa tabela HTML (tag <table>). São mostradas 100 linhas por vez, com os campos (colunas) já mencionados anteriormente, ordenados pelo ID, ou seja, dados mais recentes são mostrados primeiro, com a possibilidade de visualização dos dados mais antigos através do *link* "Anteriores" e de avanço para os dados recentes, através do *link* "Próximas". O *link* "Última" aponta para a página com os dados mais recentes e o link "Sair" aponta para a página de *Login*.

Muitos softwares de topografia utilizam informações de GPS para triangular regiões e estabelecer zonas limites e muitos deles podem ser munidos diretamente com uma tabela em planilha do tipo Excel em sua entrada (um exemplo de planilha é visto na

Figura 70). Diante disso e também para um histórico *backup*, com a possibilidade de limpar o banco de dados, decidiu-se disponibilizar as informações da tabela em um *link* para download de uma planilha do Excel. Isso é conseguido clicando no link "Download".



ID	IMEI	DATA	HORA	ST	LATITUDE	LONGITUDE	VELOCIDADE	ALTITUDE
03085	353815010036881	02/08/2008	9:51:50	3	20.4803326 S	050.1640572 W	000.03KM	425
03084	353815010036881	02/08/2008	9:49:50	3	20.4802430 S	050.1640074 W	004.89KM	428
03083	353815010036881	02/08/2008	9:38:44	3	20.4370433 S	050.0870407 W	091.62KM	488
03082	353815010036881	02/08/2008	9:36:44	3	20.4244455 S	050.0848334 W	035.17KM	496
03081	353815010036881	02/08/2008	9:34:44	3	20.4203468 S	050.0810244 W	000.25KM	492
03080	353815010036881	02/08/2008	9:32:44	3	20.4203343 S	050.0810538 W	000.03KM	498
03079	353815010036881	02/08/2008	9:30:44	3	20.4203161 S	050.0810410 W	000.07KM	495
03078	353815010036881	02/08/2008	9:28:44	3	20.4203062 S	050.0810166 W	000.10KM	490
03077	353815010036881	02/08/2008	9:26:44	3	20.4203185 S	050.0809865 W	000.00KM	489
03076	353815010036881	02/08/2008	9:24:44	3	20.4203756 S	050.0810004 W	000.57KM	492
03075	353815010036881	02/08/2008	9:22:44	3	20.4114639 S	050.0757921 W	076.42KM	506
03074	353815010036881	02/08/2008	9:20:44	3	20.3878075 S	050.0636526 W	094.75KM	484
03073	353815010036881	02/08/2008	9:18:44	3	20.3693385 S	050.0646082 W	094.50KM	474
03072	353815010036881	02/08/2008	9:16:44	3	20.3572209 S	050.0929920 W	106.02KM	441
03071	353815010036881	02/08/2008	9:14:44	3	20.3450882 S	050.1213827 W	102.31KM	437
03070	353815010036881	02/08/2008	9:12:44	3	20.3330057 S	050.1496038 W	079.45KM	466
03069	353815010036881	02/08/2008	9:10:44	3	20.3260488 S	050.1655492 W	101.08KM	417

Figura 70 - Página de visualização das coordenadas GPS

Entretanto, as informações dadas dessa maneira, apenas com números, não têm muito significado para o usuário. Sem um software de topografia ou um servidor de mapas, as coordenadas são quase inúteis. Existem alguns servidores de mapas que podem ser instalados junto com o servidor HTTP, funcionando como um módulo do mesmo, como é feito com o MySQL. Esses servidores possuem pacotes bem interessantes de mapas com nome de ruas, estabelecimentos comerciais e outros recursos. No entanto, a maioria deles é paga. Outro obstáculo na implementação é que o servidor HTTP é externo, seria preciso instalar o servidor de mapas na máquina do Terra Empresas o que seria inviável. Uma solução mais simples é usar um servidor de mapas bastante conhecido, acessível a qualquer usuário: o Google Maps **[**25**] (**Figura 71**)**.

Ao fazer uma busca no Google Maps, colocando uma coordenada no campo de busca, percebe-se que a página aberta segue sempre um padrão, em outras palavras, a coordenada digitada é passada via URL e o servidor usa essa informação para abrir o

mapa da região correspondente. Assim, cada linha da tabela, oferece um link para o Google Maps, passando como parâmetro, na URL, sua coordenada:

http://maps.google.com.br/maps?f=q&hl=pt-BR&geocode=&q=047.8830394 W+21.9926896 S

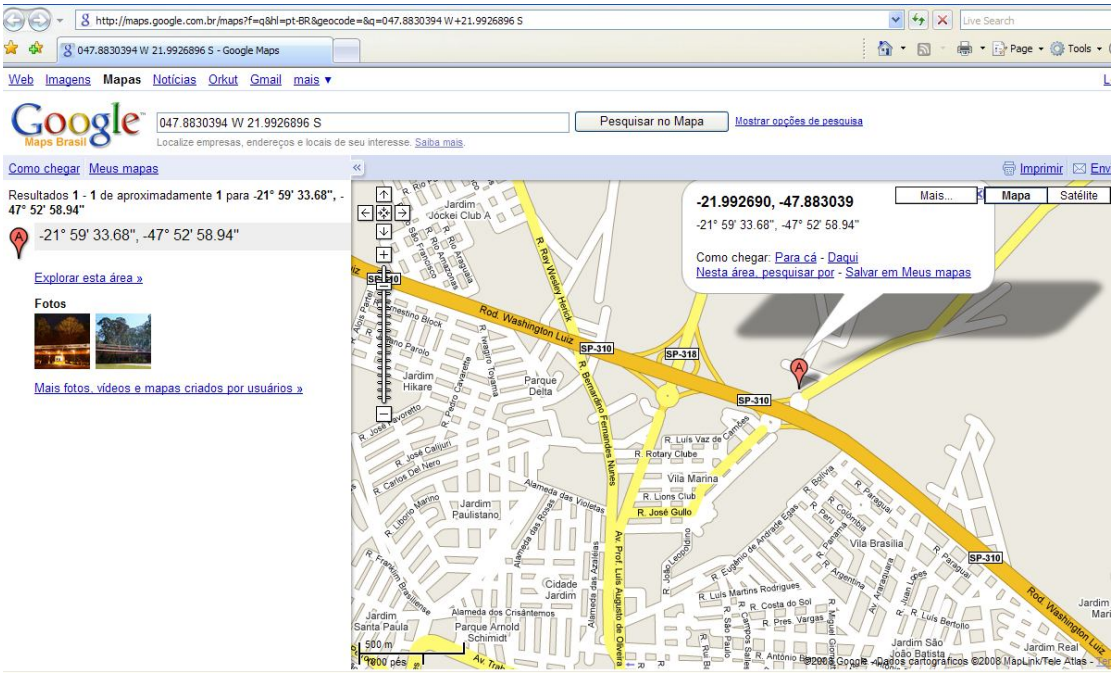


Figura 71 - Servidor de mapas Google Maps

Além do recurso do mapa que um servidor comum forneceria, o Google Maps ainda oferece o recurso do satélite, complementando a localização do módulo, tornando-se uma solução barata e ao mesmo tempo poderosa.

4.2. ZigBee

O consenso oficial da pilha é de que os desenvolvedores atuem apenas na camada de aplicação e que as camadas inferiores sejam padronizadas pela norma IEEE 802.15.4. As camadas intermediárias, apesar de aceitarem variações na maneira de se escrever o software, uma vez que são distribuídas por empresas diferentes (Microchip, Freescale, dentre outras), deveriam desempenhar as mesmas funções. Isso seria o ideal para que produtos desenvolvidos por diferentes empresas, ainda que tivessem funcionalidades distintas, pudessem integrar a mesma rede, tendo o mesmo tamanho de cabeçalho, a mesma rotina de entrada da rede, o mesmo padrão de roteamento, etc. No entanto, para suportar os requisitos propostos, diversas mudanças tiveram que ser feitas.

Percebeu-se que alguns *bugs* da pilha oficial não permitiriam mobilidade na rede. Diante disso, alterações significativas foram feitas em camadas inferiores, transformando a pilha desenvolvida em algo bastante peculiar, sobretudo na forma de integração da rede, atribuição de endereços e quadros de mensagens.

Outro requisito que se fez necessário é a parte de chave criptográfica. Apesar de a criptografia ser prevista pela documentação ZigBee, ela não estava implementada na versão da pilha da Microchip utilizada nesse trabalho. A forma de atribuição da mesma para que haja uma limitação na integração de uma rede particular, sendo segura contra invasões de nós externos é mais uma contribuição deste trabalho.

Toda a parte de software é armazenada num servidor dentro da N3E, havendo um imprescindível controle de versões, viabilizado por um CVS (*Concurrent Version System*) que identifica, dinamicamente, os arquivos alterados e as diferenças entre o arquivo atual e o arquivo da última versão salva, permite baixar arquivos de versões antigas e salvar novas versões com os devidos comentários das atualizações feitas. Esse tipo de ferramenta é essencial para softwares em processo de criação. O software CVS utilizado é grátis e foi desenvolvido pela empresa Tortoise [39].

4.2.1. *Endpoints*, *Clusters* e atributos da pilha N3E

Para integrar todos os nós na rede e distinguir as mensagens características de cada um, tanto no envio quanto na recepção, criaram-se diferentes *endpoints* e *clusters* e, dentro desses, os possíveis atributos. O conjunto dá a assinatura de cada tipo de mensagem e o nó que o recebe pode, através dessas duas informações, saber o número de campos que foram colocados no *payload* e, conseqüentemente, usá-los de maneira correta. Em cada nó devem ser discriminados os *clusters* de entrada e saída que podem abrigar.

Para a nova forma de endereçamento e para a distribuição da chave criptográfica, foram criadas novas diretivas na camada NWK e MAC, no entanto, a grande maioria das mensagens, sobretudo as mensagens relacionada ao endereçamento, estão ao nível da camada APS, apesar disso, variáveis intrínsecas das camadas inferiores são alteradas na camada de aplicação e, portanto, as funções dessas camadas também são alteradas. Resumindo, a maior parte das mensagens é criada na camada de aplicação, mas suas conseqüências não estão restritas apenas a essa camada, como é

recomendado pela norma ZigBee, elas influenciam as camadas inferiores mediante a alteração de suas variáveis.

4.2.1.1. *Endpoints*

Cada nó pode ter até 4 tipos de *enpoints* e, não necessariamente, ter apenas um de cada tipo.

a) *Endpoints* Acionadores

São os *endpoints* inseridos nas mensagens de acionamento, direcionadas a uma carga, característicos de um interruptor e também em mensagens de configuração

Eles serão, dominantemente, os *endpoints* remetentes das mensagens, mas também poderão ser *endpoints* destinos nas mensagens de ACK do acionamento.

Esse tipo de *endpoint* estará presente nos nós:

- **Biometria**: a leitura correta de uma digital irá enviar uma mensagem ao nó a que está amarrada. Essa é uma típica mensagem de acionamento, servindo para abrir uma fechadura e até mesmo acender uma lâmpada.
- **Controle Remoto**: é o nó que mais possui *endpoints* acionadores. Cada um de seus botões possui um *endpoint* diferente para que possa ser distinguido no momento do *binding*. Portanto, possui quatro endpoints acionadores (4 grupos de botões) em funcionamento normal, além do modo de funcionamento adicional de separar o grupo de botões por cômodos. Nesse modo, pode-se ter até quatro cômodos e, dentro de cada cômodo, usar os quatro grupos de botões para acionar as cargas presentes nesse cômodo. Temos portanto 4 *endpoints* em funcionamento normal e 4 (cômodos) x 4 (grupos de botões) em funcionamento do modo cômodo, totalizando 20 *endpoints*.
- **Interruptor**: dois *endpoints* acionadores. Cada botão de acionamento representa um *endpoint* diferente.
- **Coordenador:** o coordenador é o responsável por encaminhar os acionamentos vindos dos sites, portanto, também deverá ter um e*ndpoint* acionador. Além de enviar mensagens de configuração, cujos parâmetros serão determinados pelo site.
- **GPRS/ZigBee**: dentro da idéia desse nó funcionar como uma chave, ele deve ser amarrado a uma fechadura (atuador) e, portanto, possuirá um

e*ndpoint* acionador. Além disso, podem existir acionamentos e configurações feitos por SMS, justificando ainda mais a existência desse tipo de *endpoint*.

- **Interruptor**: O interruptor é o exemplo mais claro de um acionador. Possui dois botões independentes e, conseqüentemente, dois *endpoints* acionadores.
- **Sensor (temperatura)**: Apesar de não ser um acionamento óbvio, o sensor lê, constantemente, os valores da temperatura ambiente e, dependendo desse valor e do *setpoint* pré-configurado, pode ser gerado o acionamento de um alarme sonoro e/ou visual na página do site.

b) *Endpoints* Receptores

São os alvos dos *endpoints* acionadores e também poderão receber mensagens de configuração vindas do site ou de mensagens SMS. Todos os nós terão esse *endpoint* em razão de todos estarem propensos a alguma configuração, mas os nós que têm esse *endpoint* em decorrência do acionamento são:

- **Atuador**: Possui o relé como "carga" e, dessa maneira, possui a propriedade de ser acionado por controles, biometria, site e SMS.
- **Coordenador**: O site pode ser considerado a "carga" do coordenador. Uma vez que as atualizações têm que ser expostas na página de internet, o coordenador é, obrigatoriamente, um *endpoint* destino de um acionamento. Isso fica mais evidente se pensarmos num *status* de alarme de um sensor de temperatura que gera um acionamento num campo específico da tabela do site.
- ***Dimmer***: É o exemplo mais claro de um e*ndpoint* receptor. A lâmpada a ser controlada pelo *dimmer* sempre estará amarrada a um e*ndpoint* acionador.

  GPRS: A "carga" desse nó é a rede celular. O nó pode receber a requisição, por parte da rede ZigBee, para enviar um SMS a um determinado celular em resposta, por exemplo, a um alarme de um sensor.

c) *Endpoint* de Endereçamento

É o e*ndpoint* remetente e destino que será enviado em todas as mensagens relacionadas a forma de endereçamento criada. Todos os nós o possuem, afinal, todos os nós participam da formação da rede.

d) *Endpoint* ZDO (*ZigBee Device Object*)

O ZDO é uma entidade prevista na pilha ZigBee, presente na camada APS, responsável por toda mensagem de dados, ou seja, que use diretivas intrínsecas da camada superior APS e que não fazem parte do escopo do desenvolvedor. Em outras palavras, o ZDO cuida das mensagens de dados que não cabem aos desenvolvedores. A gama dos clusters dentro desse *endpoint* é ou, pelo menos, deveria ser, fixa. O ZDO é o responsável por enviar, por exemplo, a requisição e a resposta de uma tentativa de *binding*, requisição e resposta de um endereço de rede, baseado num endereço MAC, enfim, várias mensagens descritas na documentação da pilha. A rigor, a única mensagem do âmbito ZDO que está sendo usada no software é a de requisição e resposta de *binding*.

4.2.1.2. *Clusters*

Dentro da hierarquia de endereços, os clusters estão imediatamente abaixo dos *endpoints*, isso quer dizer que eles são mais específicos que os *endpoints*. Numa análise prática, se fôssemos escrever uma carta a um determinado nó, o endereço de rede seria a cidade, o *endpoint* o bairro e o *cluster* seria a rua. Os *clusters* podem ser divididos em *cluster* de saída e *cluster* de entrada, dependendo da direção em que são encaminhadas as mensagens e, apesar de a pilha fazer distinção entre eles, atribuindo-lhes nomes diferentes, na implementação feita, eles terão o mesmo nome, posto que é possível diferenciar com clareza em software quando um nó está enviando e quando está recebendo uma mensagem.

Só para dar um exemplo, dentro do *endpoint* ZDO, temos 2 *clusters* relacionados ao *binding*. Um *cluster* de entrada que é a requisição vinda de outro nó, chamado, *Binding*_Req e um de saída que é a resposta a essa requisição, chamado *Binding*_Rsp, no entanto, ambos tratam de uma mensagem de *binding* e poderiam chamar-se apenas *binding*. Assim, fica relatado que nas implementações realizadas, não será feita essa distinção.

- ***On_Off_Cluster***: utilizado no acionamento do *dimmer* e do atuador e no ACK a esse acionamento. Será, portanto, *cluster* de saída durante o envio do acionamento por parte de um controle ou de interruptor e de entrada durante a recepção do ACK. No *dimmer* e no atuador ocorrerá o contrário: será cluster de entrada durante a recepção de acionamento e de saída durante o envio do ACK.
- **Biometria_*Cluster***: relacionado ao acionamento enviado pela nó Biometria, estará amarrado a nós que possuem uma carga a ser acionada como o *dimmer* e o atuador. Segue a mesma regra do *On_Off_Cluster* para a determinação de sua direção.
- **Temperatura_*Cluster***: relacionado ao envio do valor da temperatura.
- **Cena_*Cluster*** : Esse *cluster* está restrito ao par controle remoto/*dimmer*. É usado em mensagens para estabelecer um cenário específico de iluminação, criando ambientes propícios para cada ocasião.
- **SMS_*Cluster***: Relacionado às mensagens que saem do nó ZGPRS. Esse cluster é usado no *endpoint* acionador tanto para o acionamento, propriamente dito, quanto para as mensagens de configuração, sendo assim, também poderá ser recebido num *endpoint* receptor.
- **Ethernet_*Cluster***: Vale o mesmo que foi dito para o SMS_*Cluster*, sendo que a mensagem de acionamento ou configuração partirá do coordenador.
- **Acrescenta_Filho_*Cluster***: Usado por um nó pai na verificação da validade de um endereço atribuído a um filho junto ao coordenador e, portanto, na resposta do coordenador e também no *broadcast* enviado na rede após a confirmação da validade do endereço que será atribuído ao próximo nó que entrar na rede.
- **Muda_Filho_*Cluster***: Usado para que um pai avise seu filho que o endereço que lhe havia atribuído é inválido e, conseqüentemente, também é usado no ACK do filho.
- ***Coord_Come_Back_Cluster***: Se um determinado nó tenta enviar uma mensagem ao coordenador e esse não responde, ele avisa a toda a rede que nenhum nó poderá se integrar enquanto o coordenador não voltar e essa volta é detectada pela mensagem enviada pelo coordenador com esse *cluster*.

4.2.1.3. Atributos

Dentro de cada *cluster* existem atributos distintos que darão o endereço completo da mensagem. Usando a mesma analogia anterior, o atributo, na carta escrita ao nó, funcionaria como o número da casa. Com essa informação o nó saberá com exatidão, os campos que devem ser pegos no *payload* e poderá tomar a iniciativa coerente à recepção de uma determinada mensagem.

a) *On_Off_Cluster*

a.1) Atributo *On_Off*: único atributo do cluster, é usado no envio de mensagens de acionamento pelo controle remoto ou pelo interruptor e na recepção do ACK das mesmas, conseqüentemente, é usado por seus alvos, na recepção das mensagens de acionamento e envio do ACK.

b) Biometria_*Cluster*

b.1) Atributo *Send_Bio*: mesma situação do Atributo *On_Off* com a diferença que as mensagens partem da Biometria.

c) Temperatura_*Cluster*

c.1) Atributo Send_Temp: mesma situação dos atributos anteriores com a diferença que a mensagem parte do sensor de temperatura.

d) Cena_*Cluster*

d.1) Atributo *Send*_Cena: é o atributo usado na mensagem direta que é enviada pelo controle remoto a todos os *dimmers* integrantes de um determinado cenário e também é usado no envio e recepção do ACK dessa mesma mensagem.

d.2) Atributo *Send*_Cena_Coord: Como a mensagem enviada pelo Controle Remoto é direta aos *dimmers*, o Coordenador não teria conhecimento do acionamento e, portanto, não poderia fazer a atualização no site. Para que não haja esse problema, após receber a mensagem e enviar o ACK ao controle com o atributo *Send*_Cena, o *Dimmer* enviará uma mensagem direta ao coordenador com o atributo *Send*_Cena_Coord e também receberá um ACK.

e) SMS_*Cluster*

e.1) Atributo *Send*_SMS: usado para os acionamentos SMS's, por parte do nó GPRS, e seus ACK's.

e.2) Atributo *Send*_SMS_Conf: usado para as configurações que podem ser feitas via SMS e em seus ACK's.

e.3) Atributo *Send*_SMS_Coord: mesma idéia do Atributo *Send*_Cena_Coord.

f) Ethernet_Cluster

f.1) Atributo *Send*_Ethernet: usado para os acionamentos vindos do site, que serão enviados pelo coordenador ao endereço especificado na página, e também usado no ACK a essas mensagens.

f.2) Atributo *Send*_Ethernet_Conf: usado para as configurações requisitadas pelo site e nos ACK's relativos a essas mensagens de configuração.

f.3) Atributo *Send*_Ethernet_NewKey: usado para que o coordenador informe a nova chave da rede digitada na página Crip.htm (mostrada mais adiante) e também usado na recepção dessa mensagem nos demais nós.

f.4) Atributo *Send*_Ethernet_NotKey: usado para que o coordenador informe aos demais nós que a segurança da rede foi desabilitada e também usado na recepção dessa mensagem nos demais nós.

f.5) Atributo *Send*_Ethernet_Key: usado pelo coordenador na mensagem direta que envia ao pai de um nó requisitante, a chave de desbloqueio digitada no site (veja a parte de segurança descrita adiante) e usada na recepção dessa mensagem por parte dos pais.

g) Acrescenta_Filho_*Cluster*

g.1) Atributo *Send*_Filho_*Permit*: Usado pelo pai para perguntar ao coordenador se o endereço que atribuiu a seu filho está correto, ou seja, se aquele endereço é exatamente o endereço que foi determinado pela rede para ser atribuído ao próximo nó que viesse a integrar a rede. O coordenador responde com o mesmo atributo, indicando sucesso ou endereço inválido e, dependendo, dessa resposta, o pai toma as atitudes necessárias.

g.2) Atributo *Send*_Filho: Usado no envio pelo pai de um *broadcast* na rede, para que todos os nós atualizem o endereço que será atribuído ao próximo nó que integrar a rede.

h) Muda_Filho_*Cluster*

g.1) Atributo *Send*_Filho_*Change*: Se o coordenador não aprovar o endereço que foi atribuído a um filho, ele responde ao pai com o *status* de erro, e o pai usa esse atributo para informar ao filho que o endereço tem que ser mudado.

g.2) Atributo *Send*_Filho_Ack: Usado pelo filho, para responder a mensagem que foi enviada para a mudança de endereço.

g.3) Atributo *Send*_Filho_Deserd: Usado pelo pai para informar o seu filho que ele será deserdado e para o restante da rede que nenhum outro nó poderá integrar-se à rede, pois, possivelmente, o Coordenador está fora da rede ZigBee. Essa atitude é tomada depois que o pai tentou enviar várias mensagens com o Atributo *Send*_Filho_*Permit* e o Coordenador não respondeu.

i) Coord_*Come_Back_Cluster*

i.1) *Send_Come_Back*: Usado pelo coordenador sempre que ele é reiniciado. Ele envia uma mensagem de broadcast que será recebida por toda rede. Se a rede estiver bloqueada para a entrada de novos nós em razão de uma mensagem de atributo *Send_Filho*_Deserd, ela voltará a ser capacitada à entrada de novos nós.

4.2.2. Endereçamento

A forma de atribuição de endereços original é baseada na profundidade que o nó está na rede e de sua natureza (RFD, FFD). A profundidade é medida pelo número de *hops* que o nó está do coordenador, ou seja, ao receber um pedido de integração de outro nó, o nó receptor, se tiver a capacidade de ser pai (FFD), atribui ao novo filho um endereço, baseado na profundidade que ele próprio se encontra na rede. Cada nó FFD tem um limite na sua tabela de vizinhança. Caso essa tabela esteja cheia, ele é capaz de rotear a requisição a um de seus vizinhos (pai ou filhos) e esses podem aceitar a requisição do nó órfão. Entretanto, essa forma de endereçamento apresenta uma falha significativa. A maioria das mensagens da rede é enviada de forma indireta (por meio de *bindings*). Ao enviar uma requisição de *binding*, o nó remetente envia o endereço baseado em sua profundidade (o endereço que recebeu de seu pai), e é esse endereço que é amarrado ao endereço de outro nó na tabela de *binding* do coordenador. Se o nó remetente se deslocasse na rede, de forma a sair do alcance de seu pai original, ele teria que enviar uma nova requisição de entrada na rede, com o *status* de órfão. Quando outro nó, com capacidade de adotá-lo, recebesse essa requisição, atribuiria ao nó órfão um endereço diferente do original, baseando-se agora numa outra profundidade. Dessa maneira o *binding* que foi feito na posição anterior seria inválido, pois ao vasculhar sua tabela, o coordenador não encontraria nenhuma informação do novo endereço.

Uma nova forma de endereçamento foi criada para suportar a mobilidade do controle remoto.

4.2.2.1. Endereçamento proposto

Os endereços não serão mais baseados na profundidade. Eles serão seqüenciais, ou seja, os nós serão numerados seqüencialmente, de acordo com a ordem que integraram a rede e, portanto, todos os nós da rede que têm a capacidade de ser pai (qualquer FFD) deverão saber qual o endereço será atribuído ao próximo filho que, eventualmente, lhes requisitar a integração. Isso será possível pelo envio de um *broadcast*, por parte do pai, assim que a integração se confirmar.

Descrevendo em detalhes desde o início[10]:

O filho integra a rede como um novo nó, fornecendo ao pai o último endereço que lhe foi atribuído e o valor da última PAN que integrou. Se for a primeira integração, ambos os valores serão 0xFFFF. O pai compara o valor da PAN fornecida pelo filho com o valor de sua própria PAN, se eles forem coincidentes, significa que o nó já havia se integrado anteriormente só que, ao tentar comunicar-se com seu antigo pai, não obteve sucesso e procurou por outro pai em potencial. O novo pai irá lhe atribuir o mesmo endereço informado pelo filho. Nessa ocasião, a integração está finalizada, não é preciso informar a rede a respeito dessa integração, pois esse nó, obrigatoriamente, já foi anunciado alguma vez no passado. Caso o número da PAN seja diferente, trata-se, realmente, de uma nova integração e essa deverá ser conhecida pelos demais nós. A forma de endereçamento mais simples, quando a requisição é feita diretamente ao coordenador, está mostrada na Figura 72[11].

[10] - A descrição relata o que foi modificado do endereçamento habitual da pilha ZigBee, deixando de lado, por ora, os procedimentos de segurança que precedem a integração dos nós. Essa parte é descrita pouco mais adiante.

[11] - Os horários presentes na figura servem apenas para se determinar a seqüência das mensagens, não descrevendo, de maneira nenhuma, a latência das mesmas.

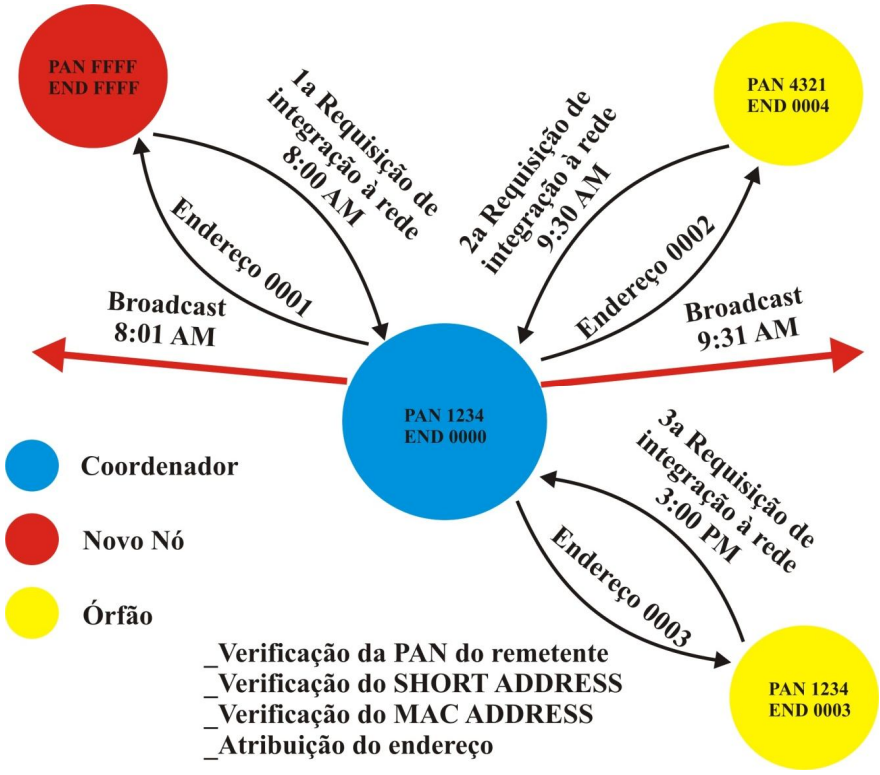


Figura 72 - Endereçamento 1

Depois que o pai atribui o endereço a seu filho, ele pergunta ao coordenador se aquele endereço é válido, enviando uma mensagem direta ao coordenador com o EP_FILHO, o Acrescenta_Filho_*Cluster* e o Atributo *Send*_Filho_*Permit*. Nesse momento, o pai desabilita sua capacidade de receber novas integrações. Essa mensagem exige um ACK e se o tempo de espera for ultrapassado, o pai tentará reenviá-la por duas vezes e poderá adotar dois procedimentos diferentes, dependo do recebimento ou não da resposta do coordenador:

Caso 1: Coordenador não responde o ACK depois de todas as tentativas

Se ele não responde, inicia-se o processo para deserdar o antigo filho. O pai elimina o filho de sua tabela de vizinhança e envia uma mensagem de *broadcast*[12], com o EP_FILHO, o Muda_Filho_*Cluster* e o atributo *Send*_Filho_*Deserd*, contendo, no *payload* da mensagem, o endereço do filho que será deserdado.

Todos nós integrados na rede, portanto, recebem essa mensagem e nessa recepção fazem uma verificação do endereço enviado no *payload* (endereço_antigo) e do nó remetente, que é pego do cabeçalho do *frame* da camada NWK. Se o endereço do

[12] O *broadcast* justifica-se, pois como não houve uma confirmação da validade do endereço, pode haver outro nó com o mesmo endereço já integrado na rede e o procedimento seria inválido

payload for igual ao seu próprio endereço e se o nó remetente for o seu pai, esse nó muda seu endereço de rede e o endereço da sua PAN para 0xFFFF, elimina as informações de seu antigo pai da tabela de vizinhança e configura os bits que confirmam sua condição de novo nó. O nó deserdado irá, novamente, tentar integrar a mesma rede. Essa integração, no entanto, não será possível, pois os demais nós que forem roteadores, ao receberem esse broadcast e verificarem que não são o filho a ser deserdado, bloquearão sua capacidade de integrar novos nós. Essa capacidade só será restabelecida quando o coordenador se reintegrar a rede. Todo o procedimento é ilustrado na Figura 73.

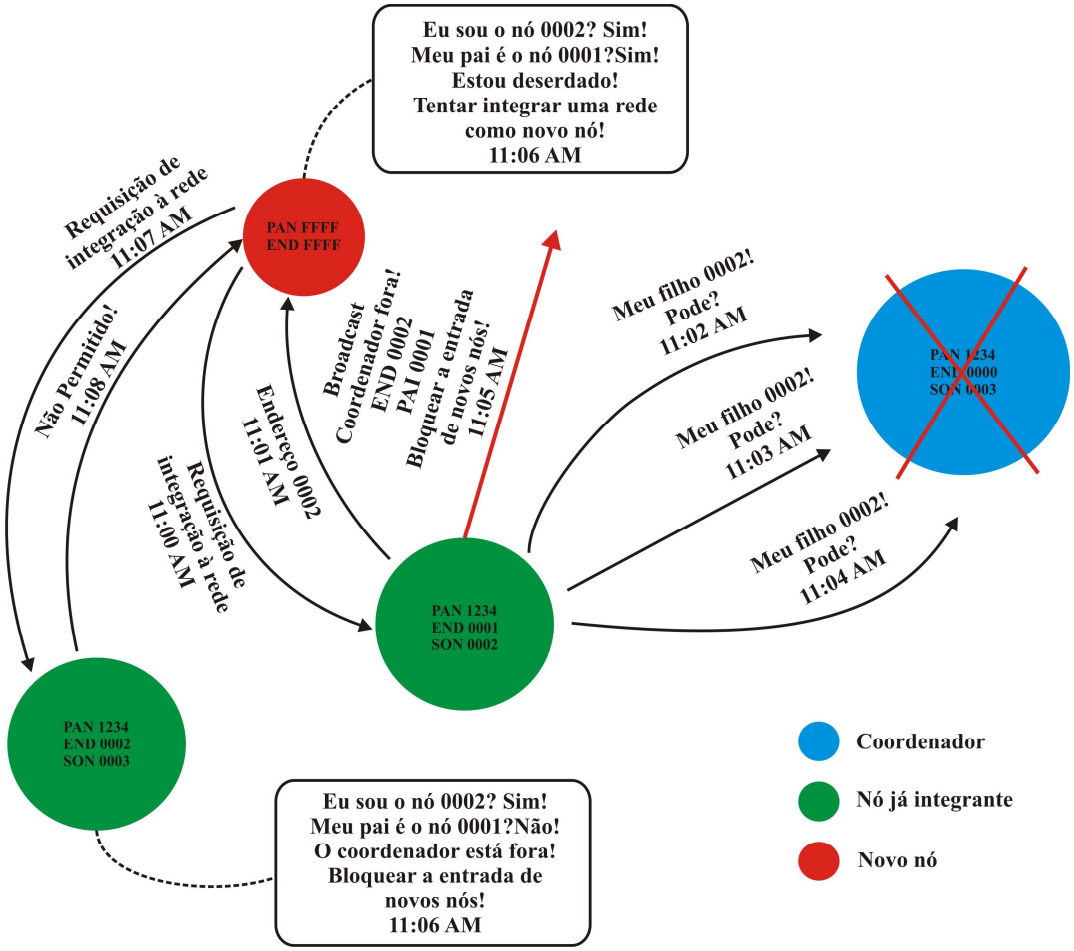


Figura 73 - Endereçamento 2

Ao se reintegrar ele envia uma mensagem de broadcast com o EP_FILHO, o Coord_*Come_Back_Cluster* e o Atributo *Send_Come_Back*. Os roteadores, ao receberem essa mensagem, têm sua capacidade de integração restabelecida.

Caso 2: O coordenador recebe a mensagem enviada pela função Filho *Permit*

O coordenador faz uma verificação do valor do endereço atribuído ao filho pelo roteador remetente e o compara com o valor do endereço seqüencial da rede.

Caso 2.1: O valor enviado pelo roteador é o valor correto

O coordenador responde o ACK com o *status* de sucesso. O roteador recebe o ACK com o EP_FILHO, o *cluster* Acres_Filho e o atributo *Send*_Filho_*Permit* e o *status* de sucesso. Esse caso de endereçamento está ilustrado na Figura 74.

Nesse momento cancela as tentativas de reenvio da mensagem, restabelece sua capacidade de integração de novos nós, incrementa o endereço seqüencial de atribuição de novos nós, salva-o em sua EEPROM e envia a mensagem de broadcast, contendo esse novo endereço seqüencial e o tipo de nó que o integrou[13], que confirmará a integração ao restante da rede. Essa mensagem terá o EP_FILHO, o Acrescenta_Filho_Cluster e o Atributo Send_Filho.

Ao receber essa mensagem os nós atuam de maneira diferente:

- **Coordenador**: Imprime na serial o endereço e a natureza do nó ingressante para a atualização no site. Atualiza seu endereço seqüencial e o salva em sua EEPROM.
- **Nós já integrantes FFD's**: Atualizam o endereço seqüencial e o salvam na EEPROM.
- **Nó Integrado**: Se for um FFD, atualiza o endereço seqüencial e o salva na EEPROM. Confirma que o *broadcast* está relacionado ao seu próprio ingresso e só então salva o valor de PAN e o seu próprio endereço na EEPROM:

[13] O campo do tipo de nó serve para o coordenador atualizar as informações no site. Ele discrimina os oito tipos de nós propostos no projeto.

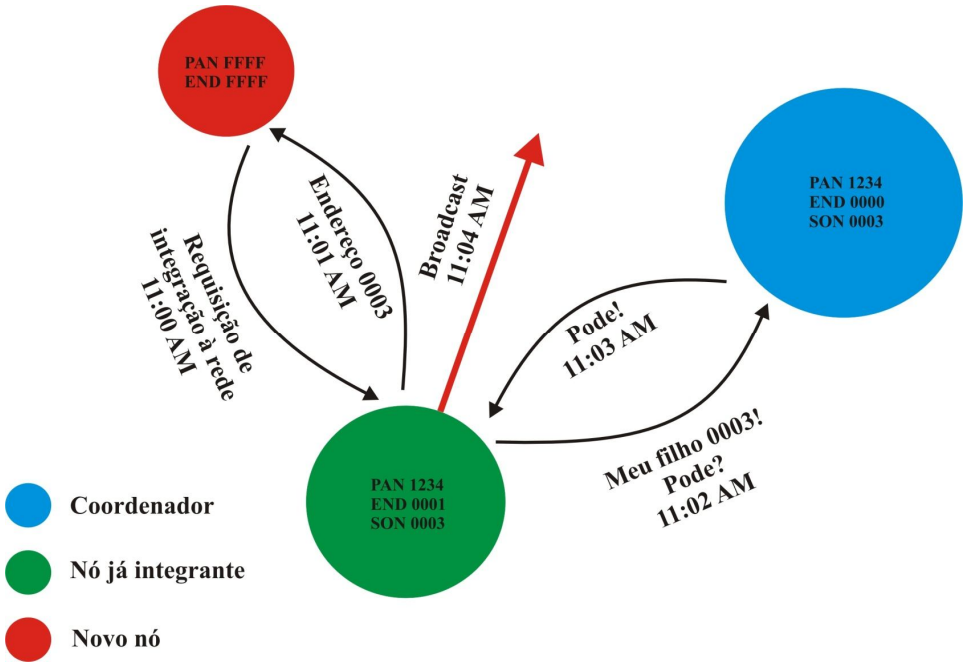


Figura 74 - Endereçamento 3

Caso 2.2: O valor de "próximo_filho" informado pelo *dimmer* está errado

O coordenador envia o ACK com um *status* de erro junto com o valor correto do endereço seqüencial ao roteador remetente. O roteador recebe o *status* de erro e o valor do endereço corrigido. Com o endereço errado e com o endereço corrigido irá trocar as informações de sua tabela de vizinhança, ou seja, irá procurar o endereço velho na sua tabela e mudá-lo para o valor corrigido. Então irá enviar uma mensagem de *broadcast* que servirá para que o filho troque seu endereço. Essa mensagem irá conter, em seu *payload*, o endereço antigo e o endereço corrigido. A mensagem fará uso do EP_FILHO, do Muda_Filho_*Cluster* e do *Send*_Filho_*Change*. Essa mensagem, apesar de ser um *broadcast* (que se justifica pelo fato do endereço estar errado, portanto o *broadcast* visa evitar conflito com o endereço de algum outro nó já integrado), exigirá uma confirmação de recebimento e o nó tentará reenviar três vezes essa mensagem, caso não receba a confirmação do filho, ele desistirá da integração e apagará também a nova entrada de sua tabela de vizinhança.

Ao receber esse *broadcast*, os nós irão comparar o endereço enviado com o seu próprio endereço e o endereço remetente com o endereço de seu pai. O nó que confirmar esses valores irá adotar o endereço corrigido enviado por seu pai e o salvará numa variável volátil, sem salvá-lo na EEPROM, pois isso só acontecerá quando receber o *broadcast* final e, por fim, enviará a confirmação do recebimento ao seu pai

em mensagem direta contendo o *status* de sucesso com o EP_FILHO, o Muda_Filho_*Cluster* e o Atributo *Send*_Ack_Filho. Depois que o pai receber essa confirmação de sucesso ele irá enviar a mesma mensagem de *broadcast* já referida acima para que os demais nós da rede atualizem o endereço seqüencial, fazendo os mesmos processos de verificação e gravação na EEPROM também já referidos. Esse último caso está representado na Figura 75.

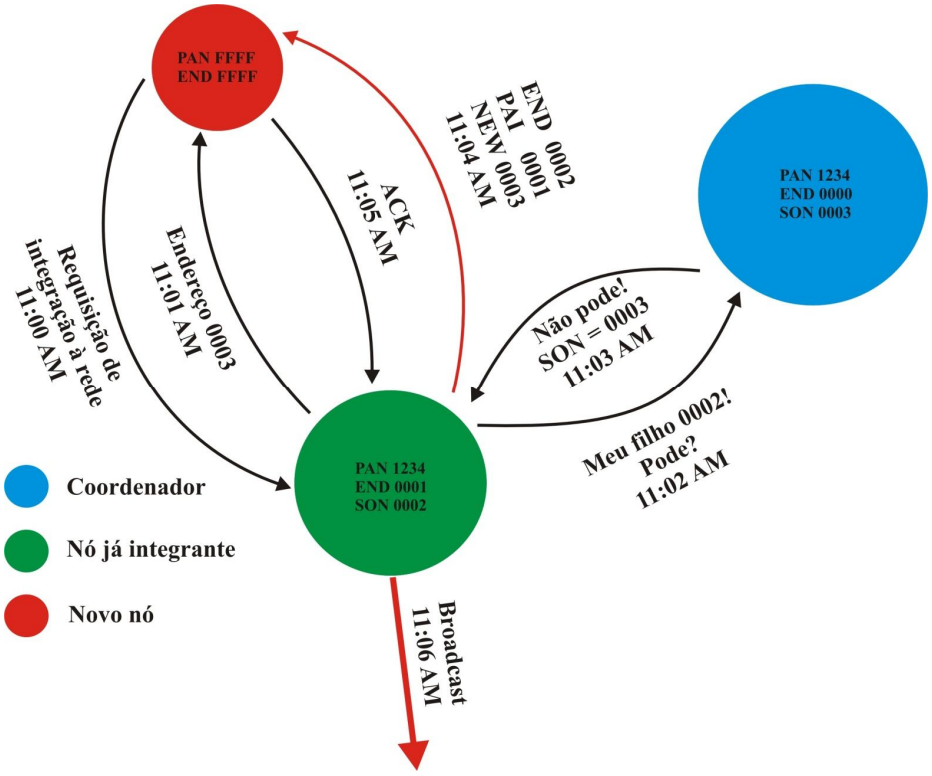


Figura 75 - Endereçamento 4

4.2.3. Segurança

A criptografia determinada pela especificação ZigBee é o AES (*Advanced Encryption Standard*) com chave de 128 bits, com reutilização dessa chave para todas as camadas, mas de forma independente entre elas, ou seja, se os dados da camada MAC requerem segurança, estes serão criptografados sem a necessidade de os dados da camada NWK o serem.

Existem 2 tipos de chaves para transmissão de mensagens:

Link Key: usada em mensagens *unicast* (de um nó para outro). Chave compartilhada somente entre os dois nós.

Network Key: usada em mensagens *broadcast* (todos os nós da rede a recebem). Chave conhecida por todos os nós da rede.

A especificação dá ainda a opção de se estabelecer um centro de segurança que é um nó dedicado ao estabelecimento e à distribuição dessas chaves.

Os detalhes de toda a parte de segurança, distribuição de chaves e a criptografia dentro de cada camada podem ser encontrados em [17].

A pilha da Microchip, distribuída gratuitamente, não possui implementadas as rotinas de segurança, sendo assim, um novo procedimento de segurança teve que ser desenvolvido para esse trabalho e, apesar de a segurança da especificação ZigBee, ser muito bem estruturada, ela apresenta um breve momento de vulnerabilidade: quando o um novo nó entra em uma rede segura, ele precisa adquirir a chave atual. A única maneira de recebê-la, porém, é através de um nó já integrante que a envia sem criptografia, ficando a mercê de pessoas mal intencionadas. O método de criptografia proposto para esse trabalho, ainda que mais simplificado, resolve essa vulnerabilidade.

4.2.3.1. Método de segurança proposto

O ponto de partida para estabelecer a segurança da pilha apresentada nesse trabalho foi de que a criptografia deveria ser a mesma usada no ZigBee (AES, com chave de 128 bits) e que era preciso resolver o breve momento de vulnerabilidade que a segurança ZigBee tem no momento de entrada na rede.

A configuração da segurança é toda feita pelo servidor HTTP interno. A página será descrita adiante.

Propôs-se, então, a se escrever um código em C que simulasse o AES (*Advanced Encryption Standard*) tanto para encriptação quanto para desencriptação dos dados.

4.2.3.1.1. AES [12] [41]

O processo de criptografia é, de maneira bem simplificada, descrito na Figura 76.

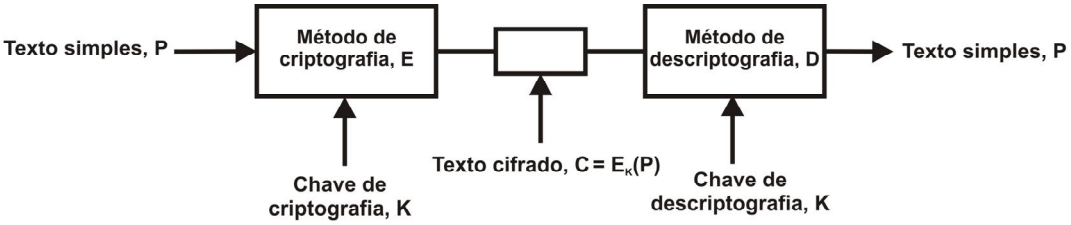


Figura 76 – Criptografia [12]

Da Figura 76 deduzimos que:

$$D_k(E_k(P)) = P,$$

onde D é a função matemática de descriptografia, E é a função matemática de criptografia, K é a chave criptográfica e P é o texto puro.

A criptografia é conhecida há milhares de anos. O imperador Júlio César já usava um algoritmo muito simples de substituição de letras do alfabeto para encriptar suas cartas. Em 1883 um militar flamengo, chamado Auguste Kerckhoff enunciou:

"*Todos os algoritmos devem ser públicos; apenas as chaves são secretas*"

que ficou conhecido como *Princípio de Kerckhoff.*

Desde Júlio César, porém, os algoritmos tornaram-se muito mais complexos. Se o seu algoritmo, naquela época, fosse público, suas cartas não enganariam ninguém, pois se tratava apenas de um deslocamento linear das posições das letras dentro do alfabeto. Hoje os algoritmos são tão complexos que, mesmo que uma pessoa tenha acesso a vários trechos de texto cifrado e conheça toda a matemática do algoritmo, sem a chave ela não será capaz de decifrar uma mensagem por completo.

O AES é considerado um algoritmo de chave simétrica: usam a mesma chave para criptografia e descriptografia. Ele surgiu de um concurso de criptografia, proposto pelo NIST (*National Institute of Standards and Technology),* o órgão do departamento de comércio dos Estados Unidos. Entre os requisitos feitos por esse departamento estavam:

_O algoritmo teria de ser uma cifra de bloco simétrica

_Todo projeto teria de ser público

_Deveriam ser admitidos tamanhos de chaves de 128,192 e 256 bits

_Teriam que ser possíveis implementações de software e hardware

_O algoritmo teria que ser público ou licenciado em termos não – discriminatórios

O algoritmo vencedor, anunciado em 2001 é chamado Rijndael uma junção dos nomes de seus criadores, dois jovens criptógrafos belgas chamados Joan Daemen e Vincent Rijmen.

O AES tem tamanho de bloco fixo de 128 bits e tamanho da chave variável de 128, 192 e 256 bits.

De modo geral, o algoritmo do AES pode ser mostrado na Figura 77. Nas entradas da encriptação e desencriptação tem-se um bloco de 128 bits, esse bloco é copiado para um vetor de estado e modificado a cada estágio de encriptação ou

desencriptação. Depois do estágio final, o vetor de estado é copiado para uma matriz de saída. Os estágios detalhados do algoritmo podem ser encontrados em [41].

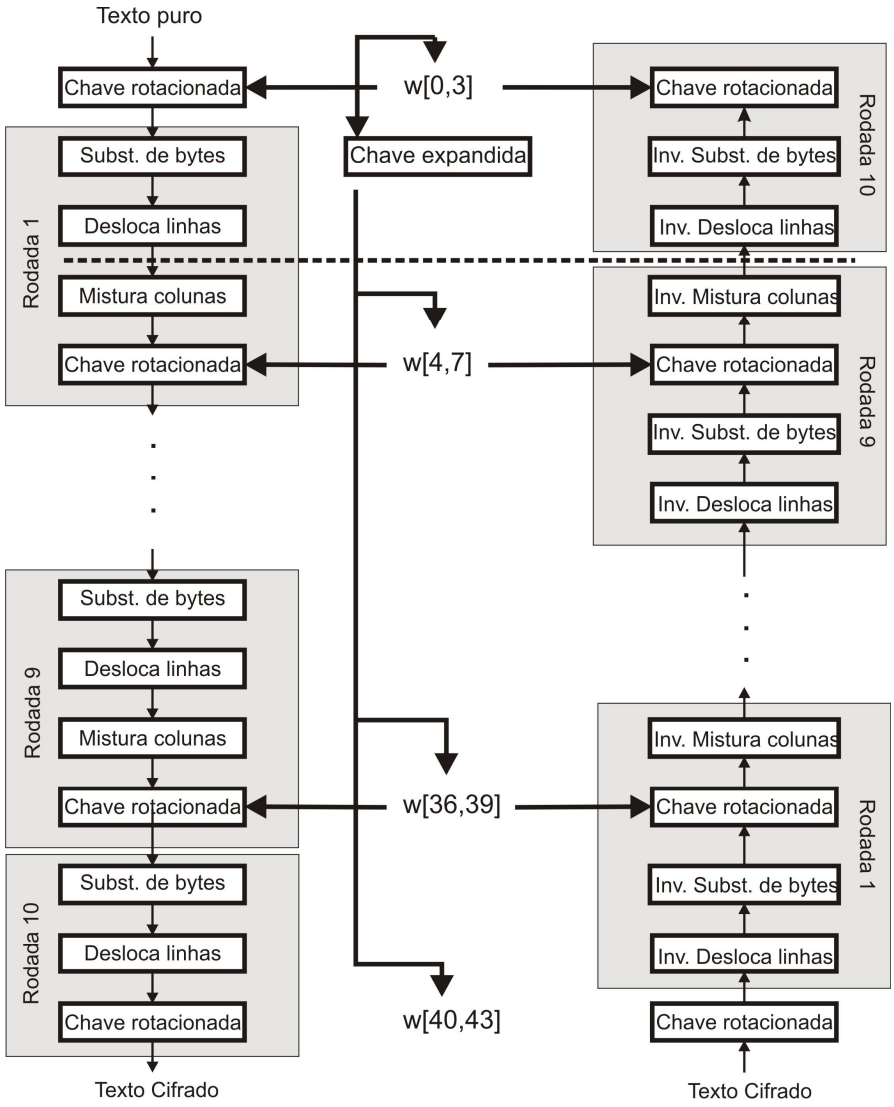


Figura 77 - Encriptação e Desencriptação AES [41]

4.2.3.1.2. Descrição

Na nova pilha desse trabalho, com a segurança implementada, o coordenador, ao formar a rede, cria uma chave randômica, baseada no valor do *timer* do microcontrolador. Ela funciona como uma chave *default* e todos os nós que entrarem na rede receberão essa mesma chave até que se opte por alterá-la.

Ao entrar na rede, a primeira requisição de um novo nó é a diretiva *Beacon_Request*. Essa diretiva é recebida pelo pai que responde com parâmetros imprescindíveis para o sincronismo de todas as mensagens. Na resposta a essa mensagem o pai já informa ao novo nó se a rede está ou não usando uma chave de segurança. Assim, o software procederá de maneiras distintas, dependendo do uso ou não da segurança:

Caso 1 (sem segurança): independente de a rede estar ou não usando segurança é preciso que todos os nós tenham uma chave mestra comum, para que, quando a segurança for habilitada e uma nova chave for escolhida, essa chave possa ser criptografada com a chave mestra pelo coordenador (que estará conversando com o servidor HTTP que recebe a nova chave) e descriptografada por todos os demais nós da rede. Por essa razão, o coordenador, ao formar a rede, gera a chave randômica *default* referida acima. Essa chave é enviada, no estado puro (sem qualquer criptografia) na resposta ao *Beacon_Request*, juntamente com a informação de que a rede não está usando segurança. O nó recebe a resposta, salva a chave numa variável, marca num cabeçalho que a rede que irá ingressar não está usando segurança e prossegue com as diretivas habituais de entrada da rede ZigBee. Depois que é aceito e recebeu todas as informações adicionais, necessárias à pilha proposta por esse trabalho para a nova forma de endereçamento, os valores são salvos na EEPROM.

Caso 2 (com segurança): se a rede está usando segurança, o método de entrada na rede fica consideravelmente diferente. Ao receber o *Beacon_Request* o pai marca o bit que dirá ao possível filho que a rede está usando segurança, mas não envia, em hipótese alguma, a chave da rede, pois como o filho ainda não possui nenhuma chave, a única maneira de o filho recebê-la, seria enviando-a descriptografada e isso era um dos fatores que se pretendia corrigir da especificação ZigBee. O filho, ao receber a resposta ao *Beacon_Request* com a informação que a rede está usando segurança, envia uma diretiva nova chamada *Key_Request*, contendo o seu endereço MAC. Se o futuro pai desse nó não for o coordenador, ao receber essa diretiva, ele enviará uma mensagem direta ao coordenador contendo as informações relevantes sobre o nó requisitante. O coordenador irá salvar o endereço MAC do requisitante, o endereço de rede do nó que lhe enviou a mensagem direta, e colocará essas informações num vetor ordenado por

ordem de requisições[14] e informará ao servidor HTTP que um novo nó, de determinado endereço MAC[15], está requisitando a entrada na rede.

Todo nó, com exceção do coordenador, deverá sair com uma chave de desbloqueio única, da mesma maneira que saem com o endereço MAC único. Essa chave de desbloqueio é que será digitada na página do servidor para liberar sua entrada.

Depois que essa chave é digitada, vinculada a um determinado índice, o servidor passa ao coordenador que, baseado no índice, fará a verificação das informações que tem e irá proceder de 2 formas dependendo de sua relação com o nó requisitante:

A) Se o pai do nó, cuja requisição tem índice X, é ele mesmo: o coordenador mandará uma diretiva nova de resposta chamada *Key_Response*, contendo o endereço MAC de seu possível filho, e mais 32 bytes[16] criptografados pela chave de desbloqueio:
_No primeiro bloco de 16 bytes, são criptografados, com a chave de desbloqueio, o valor do índice que o nó ocupava no vetor de nós requisitantes e mais quatro bytes gerados aleatoriamente e que servirão de verificação da chave que será enviada. Esses quatro bytes também são salvos num vetor, de tal maneira, que o coordenador sabe que os quatro bytes randômicos gerados para o índice X foram Y,K,W e Z.
_No segundo bloco de 16 bytes, irá enviar a chave da rede criptografada com a chave de desbloqueio.

B) Se o pai do nó, cuja requisição tem índice X, não é ele: o coordenador enviará uma mensagem direta ao pai desse nó, com o *cluster* Ethernet_CLUSTER e o atributo Send_Ethernet_Key, contendo o número MAC do nó requisitante e a chave de desbloqueio criptografada com a chave atual da rede, afinal, o pai já é um integrante da rede e, portanto, já possui essa chave. Ao receber essa mensagem o pai compara o endereço MAC enviado no *payload* dessa mensagem e verifica se aquele endereço consta em seu vetor de nós requisitantes. Em caso afirmativo, ele descriptografa a chave de desbloqueio de seu filho, com a chave da rede e faz o mesmo procedimento descrito acima no caso A para o envio da diretiva *Key_Response*.

---

[14] - Oito é o número máximo de requisições. Depois disso, novos nós só poderão entrar, conforme as liberações forem feitas no site.
[15] - O endereço MAC, além de discriminar o nó no site, também serve para o pai impedir que um mesmo nó faça mais de uma requisição de entrada.
[16] - Como foi dito, o AES só aceita blocos de 128 bits (16 bytes). No *Key_Response* , seriam precisos 21 bytes, como só é possível mandar blocos múltiplos de 16, é enviado um bloco com 32 bytes.

O nó requisitante recebe o Key_Response e descriptografa os 32 bytes do *payload* com sua chave de desbloqueio, colocando os 16 bytes da chave da rede numa variável, que será salva na EEPROM após a confirmação de que já está integrado na rede, e colocando os 5 bytes de verificação em uma outra variável. Feito isso, ele segue os procedimentos de entrada habitais do ZigBee com o adendo de que na diretiva de associação, será enviado no *payload*, os cinco bytes de verificação (índice + 4 bytes randômicos gerados pelo pai), criptografado com a chave da rede que lhe foi recém enviada. O pai, ao receber a diretiva de associação, irá descriptografar o *payload*, com a chave da rede e fazer a verificação se os 4 bytes randômicos que ele gerou para o índice X, correspondem aos 4 bytes enviados, de maneira criptografada, pelo nó requisitante. Em caso afirmativo, significa que seu filho tem a chave correta e já pode fazer parte da rede.

A partir daí, é a aplicação que decidirá quais mensagens serão criptografadas, marcando um determinado bit antes da transmissão das mensagens. Na proposta das mensagens criptografadas desse trabalho todo o *payload* da mensagem será criptografado, independentemente da camada da qual esse *payload* faz parte, ao contrário do que a especificação propõe de se ter uma segurança dedicada a cada camada da pilha. Dessa maneira, em aplicações que utilizem o ZigBee para tranferir informações sigilosas, pode-se optar por marcar o bit de criptografia nas mensagens e qualquer pessoa mal intencionada que esteja analisando o canal, mas que não tenha a chave da rede, não será capaz de extrair os dados corretos.

4.2.3.1.3. Página Crip.htm

A página Crip.htm será responsável por toda a interface do usuário com a segurança de sua rede. Ela é acessada clicando no link “SEGURANÇA”, presente na página Index.htm ou Estatico.htm.

Ao ser aberta, o evento html de *onload* chama a função javascript para abrir a página dinâmica Key.cgi. Essa página tem um caractere de escape especial que será interpretado pelo servidor HTTP. Ao recebê-lo, o servidor saberá que deve requisitar ao coordenador o valor de sua chave de rede atual que será, imediatamente, impressa no campo “Chave” e, assim, o usuário poderá saber o valor atual e modificá-lo se preferir. A chave impressa, e os demais componentes da página podem ser vistos na Figura 78. Se o administrador da rede optar por não usar a segurança, basta escolher a opção “Segurança Desabilitada” e apertar o botão “Confirma”, o valor que está no campo de

texto será ignorado e o servidor irá alimentar (enviar uma mensagem pela serial) o coordenador com esse *status.* Ao receber a mensagem pela serial, o coordenador irá enviar uma mensagem de *broadcast* com o *cluster* Ethernet_CLUSTER e com o atributo Send_Ethernet_NotKey. Todos os nós da rede irão receber essa mensagem, de maneira descriptografada, uma vez que a segurança foi desabilitada, irão atualizar o cabeçalho que armazena os dados da rede com a informação que a rede está sem segurança e irão salvar esse cabeçalho na EEPROM. Se o administrador, por outro lado, resolver habilitar a segurança de uma rede desprotegida ou mesmo mudar a chave de uma rede já protegida, o coordenador receberá pela serial o novo *status* e a nova chave (presente no campo do texto), e mandará uma mensagem de *broadcast* com o *cluster Ethernet_CLUSTER* e com o atributo *Send_Ethernet_NewKey*, tendo em seu *payload* a nova chave criptografada pela antiga chave da rede[17]. Ao receber essa mensagem, os nós a descriptografam com a antiga chave e salvam a nova chave na EEPROM.

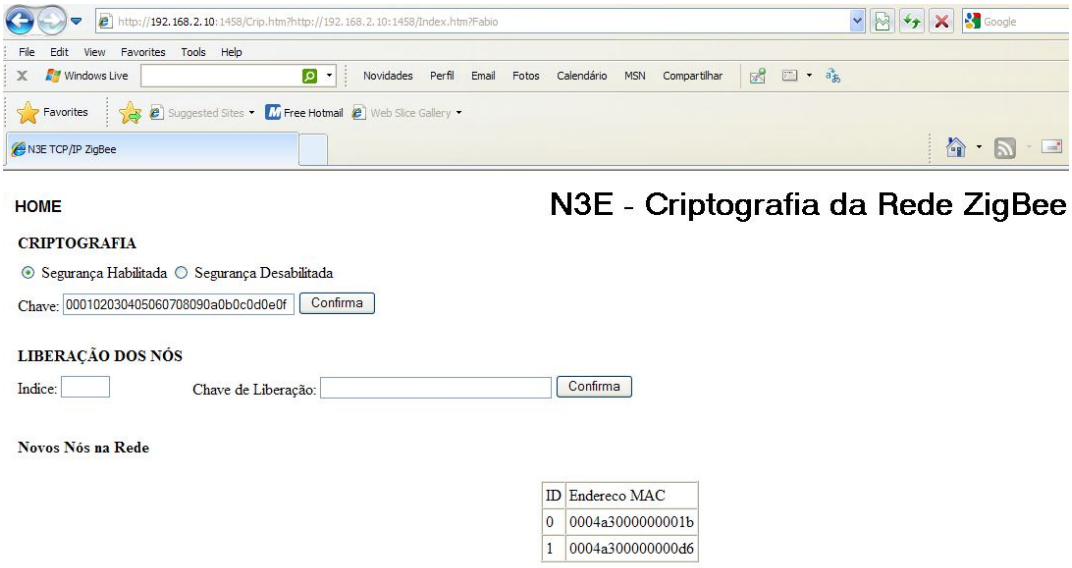


Figura 78 - Página Crip.htm

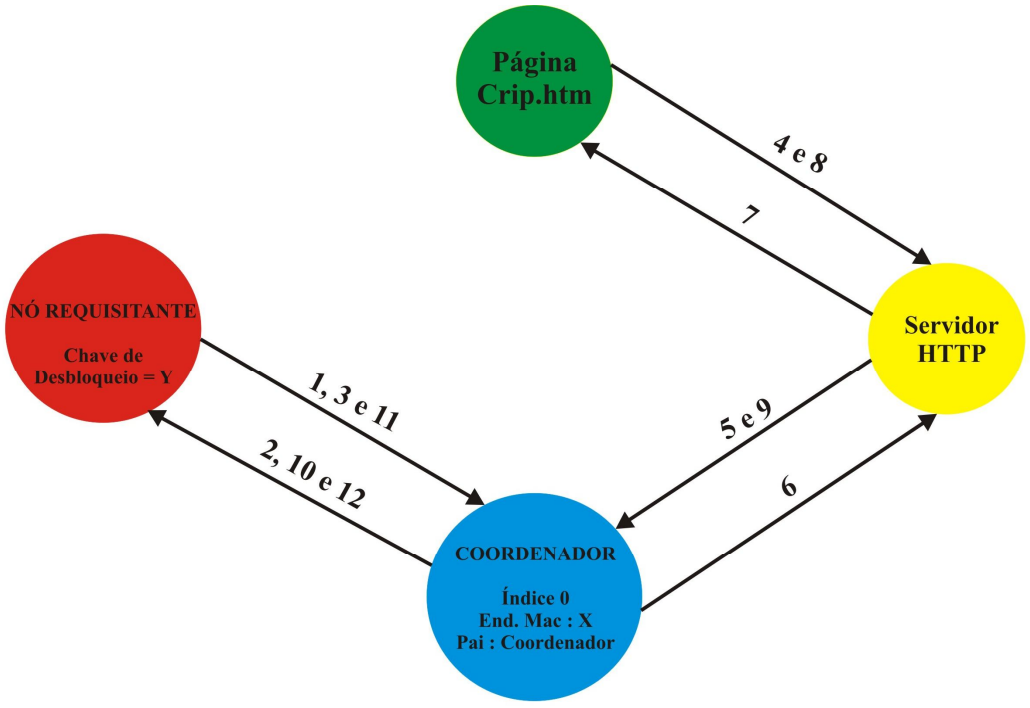
Na tabela presente na página Crip.htm da Figura 78, pode-se ver a tabela com os novos nós requisitantes da rede. Nela constam o índice do nó requisitante e seu endereço MAC. Essa tabela também é obtida por uma requisição do servidor ao

[17] - Lembrando que, mesmo que a rede nunca tenha sido protegida, existe uma chave *default* que é gerada, randomicamente, pelo coordenador na formação da rede e, nesse caso, é essa chave que será usada pra criptografar a chave nova.

coordenador, após o evento *onload* do HTML, abrindo a página Crip.cgi que, por sua vez, também possui um caractere de escape para orientar o servidor. Ao receber essa requisição pela serial, o coordenador vasculha o cabeçalho com os índices ocupados por nós requisitantes e informa ao servidor o endereço MAC dos mesmos, possibilitando a impressão da tabela. Para liberar um nó, como já foi explicado, o administrador deve saber a chave de desbloqueio correspondente àquele número MAC. De posse dela, basta preencher o campo índice, com algum dos valores presentes na tabela e digitá-la no campo "Chave de Liberação" e apertar o botão confirma. O coordenador fará a verificação já explicada, a respeito do pai daquele nó e enviará a mensagem apropriada (*Key_Response* ou a mensagem direta ao pai). Feito isso, ele apaga as informações referentes àquele índice e a tabela já é atualizada no site.

Tanto o campo do índice quanto o campo das chaves possuem função Javascript para verificação do número de dígitos e de dígitos válidos. Como só 8 requisições são permitidas, apenas dígitos de 0 a 7 são permitidos no campo "Índice" e como as chaves são formadas por números hexadecimais, apenas dígitos de 0 a 9 e letras de A a F são permitidos. Além disso, a chave deve conter, obrigatoriamente, 32 dígitos (16 bytes), se qualquer um desses critérios não for atendido um alerta Javascript será exibido com o erro.

As rotinas para a integração de nó em uma rede com a segurança habilitada são mostradas na Figura 79. Para esse caso em especial, a requisição é feita diretamente ao coordenador. Para o caso em que a requisição é feita a um nó diferente, considere o adicional de uma mensagem direta do nó ao coordenador, informando o endereço MAC de seu possível filho e de uma mensagem direta do coordenador a esse nó, informando-lhe a chave de desbloqueio do nó requisitante.



1 - Beacon_Request.
2 - *Beacon_Response*, com confirmação de segurança habilitada.
3 - Key_Request. Endereço MAC = X;
4 - Envio de caracteres de escape relativos à chave atual da rede e a tabela com os nós requisitantes da integração na rede.
5 - Solicita a lista de nós requisitantes salvos em em um vetor dentro do coordenador.
6 - Vasculha o cabeçalho dos nós requisitantes para responder à socilitação do servidor e envia o índice e o endereço MAC do nó.
7 - Envio dos dados de segurança ` fornecidos pelo coordenador.
8 - Chave de desbloqueio Y digitada no site.
9 - Envio da chave e do índice do nó a ser desbloqueado.
10 - *Key _Response:* Chave da rede e mais cinco bytes (índice + quatro bytes randômicos) de verificação criptografados com a chave de desbloqueio do nó X.
11 - Chave da rede e bytes de verificação descriptografados. Envio da diretiva *Association_Request,* contendo os bytes de verificação criptografados com a chave da rede.
12 - Descriptografia dos bytes de verificação e o envio do *Association_Response*, permitindo a integração do nó na rede

Figura 79 - Rotina de aceitação de um nó em uma rede com segurança habilitada

4.3. TCP/IP

Não existe nada inovador na pilha TCP/IP. Ela é parecida com tantas outras que podem ser encontradas para download na Web. A pilha é toda escrita em linguagem C e as camadas presentes no modelo OSI da internet, bem como seus principais protocolos, estão separados em arquivos “.c” e, ao contrário do que diz o modelo de referência TCP/IP, as camadas podem acessar, diretamente, qualquer outra camada, não sendo necessário que a camada acessada esteja imediatamente acima da camada que a acessa.

Apesar de não apresentar nenhuma inovação no método de conexão e protocolos, algumas mudanças tiveram que ser feitas para atender o propósito do trabalho.

4.3.1. Interrupções

O software habilita a prioridade de interrupções, sendo a interrupção de alta prioridade destinada ao *clock* em que se baseiam todas as rotinas de conexão, tempo de abertura de *socket*, tempo de expiração de protocolos, enfim, é esse *clock* que dá o embasamento temporal para todas as funcionalidades da pilha TCP/IP. A interrupção de baixa prioridade é destinada às requisições do projeto, relacionadas às interrupções seriais. As duas seriais disponíveis no PIC estão com a interrupção habilitada, sendo a interrupção serial 1, destinada à comunicação entre o microcontrolador da pilha TCP/IP e o microcontrolador da pilha ZigBee e a interrupção serial 2 destinada à parte de configuração.

A configuração é destinada a dois propósitos:

1) acertar os parâmetros referentes ao funcionamento da rede TCP, habilitando ou desabilitando o protocolo DHCP, inserindo o IP fixo e *gateway* padrão para redes que o exijam, alterando o número de hardware do nó e outras funções já padronizadas para menus de configuração em conexões de internet.

2) acertar a data e a hora do RTC que será imprescindível para as atualizações da rede ZigBee na página de Internet.

Toda a configuração tem que ser feita mediante um software Telnet/Serial como o HyperTerminal. Cada opção de configuração tem um botão dedicado, cujo pressionamento gera um Menu de opções na tela do HyperTerminal.

A comunicação entre ZigBee e TCP/IP adota uma padronização de *string*, ou seja, possui um caractere especial de início e outro de fim que permitem validar a transferência de informações.

4.3.2. Inicialização

Além das inicializações básicas da pilha, como os parâmetros usados em cada camada, a inicialização do hardware e *timer*, existe a inicialização implementada para atender os requisitos do site.

O hardware usa uma EEPROM externa que está dividida entre o armazenamento das páginas de internet que, como já foi dito, são transformadas num agrupamento binário pelo programa MPFS.exe da Microchip e transferidas ao módulo por FTP, e os dados que são necessárias para o site do projeto.

A inicialização implementada inclui:

1) Leitura da senha do site que servirá para autenticar o Login: Por default o módulo vem configurado com uma senha que pode ser alterada pelo usuário, como já foi mostrado em uma das páginas desenvolvidas para o site. Existe uma limitação no tamanho da *string* que pode compor a senha e, portanto, esse parâmetro tem uma localização específica num endereço da EEPROM. Depois de lida a memória, os bytes gravados são copiados a uma variável global que servirá para comparação com a senha digitada em cada *login*.

2) Leitura dos dados repassados pela rede ZigBee: Endereços dos nós já integrantes, última atualização, *status* e todos os dados presentes na tabela, também já exposta em uma das páginas desenvolvidas para o site. Esses dados são passados, via serial, pelo coordenador ZigBee e salvos na memória. Para copiar esses dados, foi reservado um espaço na memória de programa do PIC para a criação de um vetor de 50 posições (portanto, atualmente, pode-se armazenar 50 nós) e cada posição tem todos os campos possíveis. Explicando melhor, foi criada uma Struct "ZigBee", contendo todos os campos que podem ser designados a um nó e um vetor de 50 posições do tipo ZigBee, num espaço dedicado da memória de programa.

4.3.3. *Loop* Principal

O *loop* principal realiza, basicamente, três varreduras que norteiam todo o funcionamento do site:

4.3.3.1. Servidor HTTP

A cada *loop*, o software chama a função que atualiza o servidor HTTP. Essa varredura é responsável pelo dinamismo das páginas.

A primeira atividade dessa função é verificar o *status* da página que está sendo atualizada dinamicamente. Se o usuário que visualizava a página dinâmica se desconectou, a ocupação do *socket* fica ociosa e pode ser detectada e a ocupação dinâmica pode ser repassada a outro usuário que estiver visualizando a página estática. Esse repasse acontecerá na próxima vez que a página estática tentar ser atualizada pelo usuário.

Ao se digitar um endereço no *Browser* e pressionar a tecla "*Enter*", o servidor entende essa ação como um método HTTP GET. Detectado esse método, o servidor lê o nome da página requisitada e, com base nele, busca na EEPROM os bits que lhe são referentes, transformando-os no padrão entendido pelo *Browser* e colocando-os no *socket* da conexão. Assim, o usuário pode visualizar a página. Deve-se ressaltar que existem caracteres especiais padrões passados pela URL que permitem que, além do nome da página a ser aberta, o usuário passe parâmetros que irão discernir a maneira como essa página será aberta. Por exemplo, na página de *login*, quando o usuário digita a senha e aperta o confirma, ele passa como parâmetro, identificado por um caractere especial, a *string* que digitou no campo senha. Antes de abrir a página de visualização, o servidor compara a *string* digitada com a senha salva e só se elas forem coincidentes é que a página de visualização será aberta, caso contrário, uma página de erro será mostrada. Assim como o exemplo da senha, qualquer ação vinda do site, tipicamente atrelada ao pressionamento de um botão, envia parâmetros ao servidor que são interpretados e geram uma reação. As ações são diferenciadas por um padrão de campos na string enviada. O botão de acionamento de um *dimmer* tem um determinado código na *string*, que é enviada após o pressionamento, que o diferencia do pressionamento do botão de *login*.

Conforme vai colocando os bits da página requisitada no *socket*, o servidor os interpreta. Existem caracteres de escape que permitem a visualização dinâmica. Ao detectar o caracter "!", por exemplo, o servidor sabe que deverá ler o vetor ZigBee que foi criado e colocá-lo na página de visualização, intercalando *tags* HTML para que seja impressa uma tabela na posição correta da página.

4.3.3.2. Servidor FTP

É a varredura que permite o carregamento (*upload)* das páginas do site.

Quando um usuário digita o IP do nó em um *prompt* de comando, ele ocupa um *socket* FTP que é identificado pelo nó.

Depois de conectado, existe uma seqüência padrão do procedimento de validação do usuário e atuação no servidor: requisição de usuário e senha, que são comparados com os que estão salvos na EEPROM, e comando de transferência de arquivo que, depois de detectado, libera o *socket* para que os dados sejam transferidos.

Todo o procedimento é respaldado por um monitoramento do *timer*. Existe um tempo máximo para a realização de cada procedimento que interrompe a conexão caso seja extrapolado.

4.3.3.3. Gravação dos dados ZigBee

Quando o ZigBee se comunica com o TCP/IP ele gera a interrupção da serial 1, a *string* é passada a uma variável global e um marcador (*flag)* é acionado para identificar que a comunicação ocorreu. No *loop* principal esse marcador é constantemente avaliado. Quando ele é identificado como VERDADEIRO (TRUE), o software sabe que uma comunicação ocorreu e tratará de decifrar a string passada. Cada campo da *string* tem significado e, embora os campos sejam passados todos numa *string*, deverão ser transformados nos tipos específicos determinados pela Struct ZigBee, declarada na memória de programa. Portanto, dentro dessa rotina, os caracteres são contados para separar o campo e várias funções são chamadas para transformá-los num tipo específico. Além disso, a leitura do RTC é realizada e uma string, contendo data e hora, é associada aos demais campos que foram passados pelo coordenador ZigBee. Em seguida, caso as informações correspondam a dados de um novo nó, essa nova entrada é salva na EEPROM num endereço livre, caso correspondam a dados de um nó já integrante, as informações são editadas no endereço que o nó já mantinha na mesma EEPROM.

4.4. GPRS [40]

A programação JAVA é toda baseada na interface, classes e métodos (espécie de bibliotecas de “funções”), já desenvolvidos pela Siemens, especialmente dedicados ao módulo XT65, aliada a novos métodos e classes criados para utilizar os recursos do hardware.

O módulo XT65 é, basicamente, um modem e, portanto, todo o seu funcionamento é controlado via comandos AT. Existe um método específico da biblioteca da Siemens que envia esses comandos e esse método é crucial para o desenrolar de todo o software. Através dos comandos AT pode-se pedir informação do *status* de portas (nível alto ou baixo), permitindo o controle da lógica de leds, por exemplo.

A IDE escolhida para escrever os códigos foi o Eclipse, desenvolvido pela SUN. Os códigos escritos são referenciados à biblioteca Siemens. Ela possui ferramentas de desenvolvimento próprias, vindas com o módulo e que devem ser instaladas no computador e agregadas à IDE.

O código escrito (extensão .java) é compilado, gerando arquivos (.jar e .jad). Esses dois arquivos são salvos na memória do módulo e são decifrados pela máquina virtual do XT65 para realizar as funções previstas no código. A transferência dos arquivos é intermediada por um software bem simples de comunicação que transforma o módulo numa COM comum, como uma espécie de *pendrive* que oferece ao usuário a interface de sua memória interna, sendo possível arrastar os arquivos pra dentro dela, como é feito em qualquer pasta no Windows Explorer.

O código, de maneira simplificada, é estruturado da seguinte maneira:

Ele implementa a interface da Siemens com os métodos de início, parada e destruição da aplicação. Ao rodar o arquivo .jar, a máquina virtual é encaminhada ao método *start* (é o loop proncipal do software), iniciando, a partir daí, as rotinas de verificação de portas, envio de coordenadas, etc..., sendo que alguns fatores como sobre-tensão e escassez de memória, podem induzir o módulo a encerrar a aplicação.

Dentro do método *start* da aplicação são iniciados *timers*, cujas interrupções estão associadas às verificações e ao acionamento de portas, ou seja, a cada interrupção desses *timers* é possível fazer o acendimento de leds ou a verificação da posição de uma chave, por exemplo, tomando-se a decisão necessária conforme a leitura. O *status* de uma porta é uma resposta a um comando AT. Nessa parte também é feito todo o

procedimento de configuração que é baseado na leitura de um arquivo TXT salvo em sua memória.

As demais funcionalidades são baseadas em métodos *listener*, que funcionam como uma interrupção. O envio de coordenadas, por exemplo, tem o intervalo que é configurado no módulo por comandos AT, ao alcançar o tempo do intervalo, o módulo lança um evento que é escutado pelo método *Listener* e, discriminado o tipo de evento de que se trata, toma a iniciativa de enviar a coordenada via rede de dados celular. O recebimento de um SMS de programação também segue o mesmo padrão.

A configuração das propriedades do módulo como: tempo de envio de coordenadas; telefones celulares cadastrados para receber avisos de anomalias de comportamento, como bateria fraca, ou acionamento de emergência; configuração da APN da rede celular para o uso do GPRS, enfim, tudo isso é feito via SMS. As mensagens enviadas seguem um padrão bem específico, contendo campos *Login*, Senha, código do comando e parâmetros. Dependendo do tipo de comando, o número de parâmetros e seus caracteres permitidos variam. Tudo é verificado pelo módulo que replica a mensagem de configuração, alertando o remetente do sucesso ou do fracasso da configuração.

A comunicação serial também possui um pacote, contendo os métodos usados na configuração, leitura e escrita. A configuração é feita junto com a inicialização já comentada. A leitura é feita via verificação constante dentro de um *loop* na parte principal do programa e a escrita se dá conforme for recebido uma interrupção (*listener)* de SMS para um possível acionamento. É através da serial que GPRS e ZigBee comunicam-se.

Uma parte muito importante no JAVA é o sincronismo entre os métodos. É preciso estabelecer um *script* da realização das tarefas, adotando, inclusive, prioridades em suas execuções. Do contrário, erros sérios de programação podem ocorrer, ocasionando problemas no funcionamento do hardware. Esses erros geram a destruição da aplicação, chamando o método da interface Siemens previsto para isso.

A parte ZigBee desse nó é um software de RFD. Os RFD's, como já comentado, ficam, constantemente, entrando em modo *sleep* e quando acordam, enviam uma mensagem de *pooling* ao seu pai, requisitando sincronismo e, em seguida, recebendo e/ou enviando mensagens na rede. Quando esse nó manda um *pooling* e não recebe uma resposta é porque, possivelmente, já que se trata do nó responsável pelo monitoramento no mundo externo, saiu do alcance da rede. Depois de algumas tentativas frustradas de

receber a resposta ao *pooling*, o ZigBee manda pela serial a informação que o nó se desmembrou da rede e, a partir desse momento, o XT65 deverá colher informações de coordenadas GPS. A parte do software do ZigBee passa a demorar mais no modo *sleep* e marca a condição de que é um órfão, portanto, sempre que retorna do *sleep* tenta integrar uma rede como órfão. Quando, enfim, encontra uma rede, volta a informar o software Java que o modem pode parar de enviar coordenadas ao servidor externo. Dentro da rede, esse nó também pode enviar avisos de alarme vindos de sensores ZigBee, via SMS, aos celulares pré cadastrados e, logicamente, receber SMS's de acionamento e configuração de qualquer celular e reencaminhá-los aos nós da rede.

CAPÍTULO 5

Testes e Validação

Os testes abordam vários cenários possíveis na disposição da rede, abrangendo desde a nova forma de endereçamento, com a avaliação do comportamento do roteamento das mensagens decorrente da mobilidade do controle remoto, até a visualização das páginas de monitoramento na internet.

5.1. Endereçamento e mobilidade

Como já mencionado, uma nova forma de endereçamento teve que ser desenvolvida para suportar a mobilidade dos nós dentro da rede sem que ele perca suas entradas de *binding* na tabela do coordenador.

A nova forma de endereçamento seqüencial foi implementada na camada de aplicação, mas para que se adaptassem aos demais protocolos, sobretudo o procedimento de roteamento, foram necessárias algumas mudanças na camada de rede com a inclusão de *bytes* no cabeçalho e a forma de busca na tabela de vizinhança, que é a ferramenta crucial para todo o encaminhamento, recebimento e roteamento das mensagens. Sendo assim, simular situações de trocas de pais, forçando mensagens roteadas era essencial para a validação das mudanças feitas na pilha.

Dentro desse propósito, os endereços seqüenciais corresponderam às expectativas, 8 nós foram colocados na rede, incluindo 2 controles remotos deslocando-se constantemente. Quando o controle tentava um acionamento e estava fora do alcance do pai atual, ele entrava com uma requisição de órfão, sendo adotado por outro pai com endereço idêntico, mantendo as entradas de *binding*. No cabeçalho das mensagens enviadas (acionamento e *binding*) é passada a informação do pai atual. Se um antigo pai receber essa mensagem para roteá-la, por exemplo, e constatar que o nó não é mais seu filho, ele atualiza sua tabela de vizinhança, excluindo o antigo filho.

5.2. Binding

Existem botões dedicados ao *binding* em cada nó. Para concluí-lo deve-se realizar o seguinte procedimento:

Sempre são amarrados 2 nós por vez. Pressiona-se o botão de *binding* num primeiro nó. O coordenador recebe essa primeira diretiva e começa a preparar a entrada da tabela com o endereço e o *endpoint* correspondente. Caso uma segunda diretiva de *binding* de um outro nó chegue ao coordenador em menos de 5 segundos, ele completa o casamento dos dados na tabela, caso contrário ele descarta as entradas criadas. Portanto, é necessário pressionar o botão do outro nó (ou pelo menos um outro *endpoint*) que se queira amarrar em menos de 5 segundos. Há uma sinalização de leds para indicar o sucesso ou o fracasso da tentativa de *binding*.

O procedimento de *binding* foi completado com sucesso, inclusive dos nós móveis, trocando de pai, como já tinha sido salientado. Há um limite de memória por parte do coordenador para as entradas de *binding*, mas nada que comprometa a aplicação, sendo possível acrescentar uma memória sobressalente se necessário. Foram feitos *bindings* de um botão com várias cargas e de uma carga com vários botões. Tudo procedeu como esperado: um mesmo botão acionando várias cargas e uma carga sendo acionada por vários botões.

5.3. Acionamento, cenas, hierarquia de endereços e acknowlegement

Cada botão de *binding* está associado a um determinado *endpoint*, portanto, se tivermos mais de um botão de acionamento num mesmo nó, será preciso ter um número de botões de *binding* correspondente ou criar um procedimento para diferenciar um *endpoint* do outro, essa é uma grande facilidade da hierarquia de endereços. No controle remoto existem vários grupos de botões, cada um deles possuindo um *endpoint* diferente o que permite que tenham entradas de *bindings* independentes, apesar de terem o mesmo endereço (*short address*). Para fazer o *binding* foi criado um procedimento de modo *binding*. Um botão é pressionado para entrar no modo *binding* e para mandar a requisição do *binding*, deve-se pressionar o grupo de acionamento que se deseja amarrar com uma determinada carga. Para voltar ao modo de acionamento, deve-se pressionar

novamente o botão de modo *binding*. Esse é um exemplo feito para o controle remoto, mas também vale para outro nó qualquer que venha a ter mais de *endpoint* para ser amarrado.

Outra rotina no controle é a criação de cenas. É possível configurar cenários para uma ocasião especial ou para uma determinada hora do dia. O usuário pode querer um cenário a meia luz para conseguir um ambiente intimista ou configurar um caminho de luz até o seu quarto, enfim, uma série de possibilidades. Para isso deve-se adotar o seguinte procedimento:

Deve-se, primeiramente, preparar todas as cargas, passíveis da variação de intensidade luminosa (*dimmer*) até a intensidade que se deseja e colocá-la no modo de requisição de cena. Feito isso, o controle irá mandar um *broadcast* de requisição de cena. Todas as cargas que estiverem nesse modo responderão com seu endereço, o *endpoint* e a intensidade atual. O controle irá salvar essas informações numa tabela. É possível configurar até 10 cargas numa mesma cena e até 4 cenas diferentes (o controle possui 4 grupos de botões). Para o controle enviar a requisição de cena, é preciso pressionar um botão para entrar nesse módulo e em seguida pressionar um dos quatro grupos de botões para discriminar a cena, ou seja, cada grupo de botões pode fazer referência a um cenário diferente. Depois que o cenário foi salvo, o acionamento segue o mesmo padrão. É necessário pressionar um botão para entrar num modo de acionamento de cenas e em seguida pressionar o grupo de botões referente ao cenário desejado. Ao contrário dos acionamentos comuns, as cenas são obtidas por mensagens diretas, ou seja, elas são encaminhadas diretamente ao endereço salvo na tabela de cenas, não tendo, necessariamente, que passar pelo coordenador, como é feito com as mensagens indiretas que são encaminhas segundo as entradas da tabela de *binding*.

Uma rotina interessante que foi implementada foi a resposta ao acionamento do controle. Se o usuário decidir acender a luz da garagem, estando deitado em sua cama, é necessário que obtenha uma confirmação do acendimento. Cada grupo de botões (cada *endpoint*) possui um led correspondente. Ao se requisitar um acionamento de maneira indireta (*binding*) o *acknowledgement* será enviado apenas ao botão remetente. Isso é diferente da maneira que era implementada na pilha original que mandava o *ack* a todos os nós que estivessem amarrados a determinada carga. Isso sobrecarregava a rede e o buffer do coordenador não era capaz de encaminhar todas as mensagens se houvessem muitos *bindings*. Portanto, foi criada uma rotina para corrigir esse problema.

Todos os procedimentos de cenas, acionamento e ack's foram realizados com sucesso.

5.4. Biometria e Atuador

São dois nós casados. Eles se complementam. Para funcionar, deve ser feito, entre os dois, o *binding*, que segue os mesmo procedimentos já comentados no controle remoto e no *dimmer*, havendo um botão dedicado a isso tanto no nó Biometria quanto no nó Atuador, havendo também uma sinalização do *status* da tentativa.

Os seguintes testes foram feitos. O coordenador foi colocado no centro da sala. O atuador foi fixado num canto a cinco metros do coordenador e o nó biometria foi colocado em outro canto. O *binding* foi realizado com sucesso entre a biometria e o atuador. Em seguida, foi realizado um procedimento de cadastro de uma impressão digital, sendo salva no nó biometria. Depois de registrado e de feito o *binding*, ao colocar a digital no leitor e confirmar o cadastro, uma mensagem é enviada ao atuador, que abre sua fechadura eletromecânica.

Em seguida, o *binding* foi desfeito e o mesmo teste foi repetido. A impressão digital foi lida corretamente, mas a mensagem não foi enviada e a fechadura não abriu. Esse teste foi feito repetidas vezes, sempre obtendo sucesso.

5.5. Internet

As páginas mostradas anteriormente na explicação dos servidores interno e externo e o que foi dito a respeito de seu funcionamento já documentam parte dos testes de validação.

No servidor interno, mostraram-se extremamente satisfatórios os itens:

_atualização dinâmica de acionamento e entrada dos nós, com a mudança dos campos de *status* e do horário da atualização pego do RTC do servidor interno;

_validação dos usuários, com repasse automático da página dinâmica a um usuário de visualização estática após o tempo ocioso do *socket*;

_limitação do número de usuários logados para impedir a queda do servidor;

_atualização dos nós requisitantes de integração na rede com segurança habilitada, bem como a chave atual usada na rede, a mudança da chave atual e a liberação dos nós com a chave de desbloqueio.

Com relação ao servidor externo, ele já estava sendo usado para o projeto em parceria com o Departamento de Enfermagem da Universidade Federal de São Carlos e tem o respaldo de mais de 15 mil coordenadas enviadas, salvas no banco de dados da hospedagem do Terra Empresas e com a localização auxiliada pelo Google Maps. A precisão estipulada nos mapas do Google, tomando por base referências conhecidas, como a própria edificação da N3E e outros pontos estratégicos da cidade de São Carlos, é de 30 metros, um valor bastante aceitável para os propósitos do projeto.

CAPÍTULO 6

Conclusão

Diante da tendência crescente de monitoramento ininterrupto somada à queda das barreiras de conectividade, a utilização da pilha TCP para integrar uma rede doméstica com a rede mundial de computadores é extremamente relevante. Nesse caso, tem-se a opção de, além de inspecionar, atuar no funcionamento da rede através dos mais variados terminais com conexão à Internet. O desafio aumenta diante da possibilidade de fusão com uma rede de dados ainda pouco explorada, mas que foi criada exatamente para esse contexto de aplicação em redes de baixas taxas de transmissão e que primam por baixo consumo, como é uma rede de sensores e iluminação.

Os maiores obstáculos, conseqüentes da pouca difusão do ZigBee, foram sentidos no desenrolar do projeto. A escassez de bibliografias, fóruns de discussão e, principalmente, de especialistas na área, acarretaram dificuldades que não seriam percebidas em um padrão que já estivesse em evidência como o Bluetooth, por exemplo. Entretanto, não renderia oportunidades como as que vêm surgindo para a N3E de trabalhar com empresas interessadas em utilizar o ZigBee em seus projetos, tornando-se uma referência no assunto[18]. As falhas no protocolo, como o de endereçamento, tiveram que ser descobertas empiricamente e solucionados em âmbito local, resultando em alterações nas camadas inferiores que acabaram descaracterizando a norma que foi proposta. As rotinas de segurança, que não são distribuídas gratuitamente pela Microchip, tiveram que ser inteiramente desenvolvida para esse trabalho, originando um protocolo de integração da rede e de criptografia dos dados sensivelmente diferente do previsto pela especificação ZigBee. Todas essas alterações em camadas inferiores que não são da alçada dos desenvolvedores, transformaram a pilha atual em um código bastante peculiar em muitos aspectos, desvirtuando, de forma significativa, de toda concepção ZigBee e, embora tenha-se usado o nome ZigBee para definir o trabalho, posto que tudo foi amparado em suas diretivas e protocolos, a pilha atual é passível de uma abordagem totalmente nova.

[18] Os novos projetos que vêm surgindo são protegidos por um compromisso de sigilo e não podem ser detalhados aqui, mas tem havido bastante interesse nos conhecimentos de ZigBee da empresa.

Os testes que foram realizados mostraram-se bastante satisfatórios, confirmando a autonomia das baterias, a mobilidade do controle remoto, a atribuição dos endereços, o roteamento de mensagens, a criptografia dos dados e a validação das chaves de segurança. A transposição dos dados de acionamento e a integração dos nós para a página do site também está validada. Testes com o incremento da capacidade da memória externa para hospedar as páginas de internet, deixando-as mais elaboradas com a inclusão de configuração dos patamares de sensores e seus valores em tempo real (temperatura, umidade, etc...), deixando a rede mais funcional e dinâmica, ainda deverão ser feitos.

Além das adaptações na pilha TCP/IP e o desenvolvimento de um software que utiliza a rede GPRS para envio de coordenadas GPS, destacam-se, como contribuições originais desta tese: (i) a solução de problemas da pilha oficial no tocante a endereçamento, que impossibilitava a mobilidade na rede - desta forma, uma nova técnica de endereçamento seqüencial foi implementada com sucesso, ou seja, os nós serão numerados seqüencialmente, de acordo com a ordem que integraram a rede; (ii) novo código de segurança que simulou o AES (*Advanced Encryption Standard*), tanto para encriptação quanto para desencriptação dos dados.

Em suma, apesar de muito já ter sido feito e estar funcionando com um alto nível de satisfação, ainda há trabalho a se fazer. Componentes que requisitam importação já estão sendo comprados e estão chegando pouco a pouco, viabilizando o desenho de seus *footprints*. Em breve, já será possível rotear uma nova placa para o coordenador/TCP_IP e para o ZigBee/GPRS. Também estão sendo pesquisados novos tipos de sensores para dotar a rede ZigBee das mais variadas informações.

Para trabalhos futuros, a caracterização da rede deverá incluir testes de confiabilidade, raio de alcance (distância dos nós), tempos de saída e entrada de um nó da rede e limite de carga (sensores).

Referências

[1] Monsignore, Ferdinando, “Sensoriamento de ambiente utilizando o padrão ZigBee”. Dissertação (Mestrado), Escola de Engenharia de São Carlos – EESC (2007)

[2] White Paper. “Entenda o Wi-Fi e o WiMAX como Soluções de Acesso Metropolitano”, Intel Solutions (2004)

[3] http://www.mundowireless.com.br

[4] Yuanqiu Luo, Ting Wang, Steve WeinsteinIntegrating, “Optical and Wireless Services in the Access Network”, NEC Laboratories America, Inc. Technical Report (2005)

[6] IEEE 802.11 Working Group, IEEE Std 802.11. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications of IEEE 802.11. (1999).

[7] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen, “A Comparative Study of Wireless Protocols: Bluetooth, UWB, ZigBee, and Wi-Fi”, The 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON), Taipei, Taiwan (2007)

[8] http://ieee802.org/15/index.html

[9] http://www.meshnetics.co

[10] http://www.teleco.com.br

[11] http://wirelessbrasil.org

[12] Tanenbaum, Andrew S, “Computer networks”, 4th ed., Upper Saddle River, NJ : Prentice Hall PTR (2003)

[13] http://www.techtree.com

[14] Miller, Michael; tradução : Altair Dias Caldas de Moraes, Cláudio Belleza Dias “Descobrindo bluetooth”, Rio de Janeiro : Campus(2004)

[15] http://www.zigbee.org

[16] http://www.microchip.com

[17] ZigBee Alliance, “ZigBee Specification”, (2004)

[18] Microchip, “AN965 - Microchip Stack for the ZigBee™ Protocol”, (2004)

[19] Jing Sun, Zhongxiao Wang , Hong Wang, Xiaofen Zhang, “Research on Routing Protocols Based on ZigBee Network”, Shenyang Normal University, Shenyang, China, and JiLin Normal University, Siping, China (2008)

[20] Sanmin Liu, Hai Jin, Xiaofei Liao, Hong Yao, Deze Zeng, “TCPBridge: A Software Approach to Establish Direct Communications for NAT Hosts”, Huazhong University of Science and Technology, Wuhan,China, (2008)

[21] Microchip, “AN833 – The Microchip TCP_IP Stack”, (2002)

[22] http://www.phpmyadmin.net

[23] Niederauer, Juliano, “Desenvolvendo websites com PHP”, Novatec, São Paulo (2004)

[24] Cox Junior, Fred, “Programando para WEB com PHP/MySQL”, http://www.apostilando.com.br, 2ª ed. (2001)

[25] http://maps.google.com.br

[26] http://www.popa.com.br

[27] http://pt.wikipedia.org/wiki/Sistema_de_Posicionamento_Global

[28] http://informatica.hsw.uol.com.br/receptores-gps.htm

[29] Theodore S. Rappaport, “Wireless Communications: Principle and Pactice”, New York, N.Y.; Upper Saddle River, N.J. : Institute of Electrical and Electronics Engineers: Prentice Hall PTR(1996)

[30] White Paper, “GPRS – General Packet Radio Service”, Usha Communications Technology (2000)

[31] Sanjiv Nanda, Krishna Balachandran, and Sarath Kumar, “Adaptation Techniques in Wireless Packet Data Services”, IEEE Communications Magazine (2000)

[32] http://en.wikipedia.org/wiki/Hayes_command_set

[33] Siemens Cellular Engine, “XT65 – AT Commands Set”, Version 01.001, Siemens Documents(2007)

[34] www.flexipanel.com

[35] Siemens, “XT65/XT75 Hardware Interface Description”, Siemens Cellular Engine, Alemanha (Janeiro de 2007)

[36] Siemens, “XT65 AT Command Set”, Siemens Cellular Engine, Alemanha (Janeiro de 2007)

[37] www.redecamp.com.br

[38] www.micropress.com.br

[39] www.tortoisecvs.org

[40] Siemens, “Java User’s Guide”, Siemens Cellular Engine, Alemanha (Dezembro de 2006)

[41] Stallings, William, “Cryptography and Network Security Principles and Practices”, Upper Saddle River, NJ : Pearson/Prentice Hall (2005)

Publicações

[P1] Fábio L. Zucato, Clecio A. Biscassi, Ferdinando Monsignore, Francis Fidélix, e Mônica L. Rocha, “ZigBee for Building Control Wireless Sensor Networks” IMOC (2007).

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo