UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

JÔNATAS ROMANI RECH

Desenvolvimento de um Gerente de Rede WirelessHART

Trabalho de Diplomação apresentado como requisito parcial para a obtenção do grau de Engenheiro de Computação

Prof. Dr. João Cesar Netto
Orientador

Msc. Ivan Muller
Co-orientador

Porto Alegre, julho de 2012

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Rech, Jônatas Romani

Desenvolvimento de um Gerente de Rede WirelessHART / Jônatas Romani Rech. – Porto Alegre: Instituto de Informática da UFRGS, 2012.

55 f.: il.

Trabalho de Diplomação – Universidade Federal do Rio Grande do Sul. Curso de Graduação em Engenharia de Computação, Porto Alegre, BR–RS, 2012. Orientador: João Cesar Netto; Co-orientador: Ivan Muller.

1. Redes sem fio. 2. WirelessHART. 3. Redes industriais. 4. Gerente de Rede. I. Netto, João Cesar. II. Muller, Ivan. III. Título.

AGRADECIMENTOS

Gostaria de agradecer aos meus pais, Juarez e Rejane, e à minha irmã, Caroline, por me apoiarem em todos os momentos, incondicionalmente. À minha namorada, Karen, pela dedicação, companheirismo e paciência.

Aos colegas de faculdade, em especial Henrique e Leonardo, pela amizade e apoio mútuo em todos esses anos de Engenharia.

Aos colegas e professores do Laboratório de Processamento de Sinais e Imagens, pela grande oportunidade de aprendizagem que me proporcionaram.

Ao professor João Cesar Netto, Ivan Muller e Jean Winter, que tiveram grande contribuição na realização deste trabalho.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

WH	WirelessHART
NM	Network Manager
FD	Field Device
NAP	Network Access Point
TDMA	Time Division Multiple Access
DSSS	Direct-Sequence Spread Spectrum
EIRP	Equivalent Isotropically Radiated Power
CCA	Clear Channel Assessment
FSK	Frequency Shift Keying
HART	Highway Addressable Remote Transducer
MAC	Medium Access Control
MIC	Message Integrity Check
NPDU	Network Packet Data Unit
DLPDU	Data Link Packet Data Unit
OSI	Open Systems Interconnection
RCP	Radio Coprocessor
ISM	Idustrial, Scientific and Medical

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

O protocolo WirelessHART tem se estabelecido como a principal escolha na implantação de redes sem fio em ambientes industriais desde a sua criação, em 2007. O modelo de comunicação centralizado previsto neste protocolo tem como seu principal componente o Gerente de Rede, responsável pela formação e escalonamento da rede. Este trabalho aborda a implementação de um Gerente de Rede seguindo o modelo presente na revisão 7.3 do padrão WirelessHART. Após uma introdução ao protocolo, a implementação será abordada em detalhes, desde o processo de configuração do Gerente à inicialização completa da rede WirelessHART.

Palavras-chave: Redes sem fio, WirelessHART, redes industriais, Gerente de Rede.

Development of a WirelessHART Network Manager

RESUMO

The WirelessHART protocol has estabilished itself as the main choice for wireless networks applications in industrial environments sice its creation, in 2007. The centralized communication model described in this protocol has as its main component the Network Manager, responsible for network formation and scheduling. This work approaches the implementation of a Network Manager following the model provided on the 7.3 revision of the WirelessHART standard. After an introduction to the protocol, the implementation will be approached in detail, from the Network Manager configuration process to the complete initialization of the WirelessHART network.

Keywords: wireless networks, WirelessHART, industrial networks, Network Manager.

1 INTRODUÇÃO

As redes de comunicação em plantas industriais tem passado por uma completa transformação na última década, sendo as redes sem fio um dos principais fatores dessa mudança. Protocolos como **ZigBee** e **Bluetooth**, apesar de efetivos em outros segmentos, deixam a desejar quando se trata de redes industriais, onde características desejáveis incluem alta confiabilidade e segurança.

Ambientes industriais caracterizam-se pelo alto nível de ruído eletromagnético originado pelos mais variados tipos de máquinas elétricas, além de muito frequentemente apresentarem altas temperaturas e alto nível de umidade. Um protocolo de redes industriais sem fio deve garantir a integridade e a segurança da comunicação mesmo em ambientes desse tipo.

Desenvolvido na década de oitenta, o protocolo HART surgiu como mais um candidato para comunicação cabeada nos campos de automação industrial e instrumentação inteligente. Atualmente, é um dos protocolos mais amplamente suportados por instrumentos de campo produzidos em todo o mundo.

O padrão WirelessHART foi introduzido em 2007 pela HCF (*HART Communication Foundation*) com o objetivo de expandir a capacidade do padrão HART às redes de comunicação sem fio, mantendo o suporte ao protocolo de comunicação cabeada. As vantagens da utilização de comunicação sem fio em um abiente industrial são a redução de custos na instalação de novos dispositivos e a diminuição de riscos de danos à instalação, além de maior flexibilidade na manutenção da rede.

Uma rede WirelessHART segue uma topologia centralizada, onde toda a organização e escalonamento da rede ao longo do tempo é realizado na estação central da planta, mais especificamente pelo Gerente de Rede, ou *Network Manager*. O objetivo deste trabalho é implementar um Network Manager como previsto na revisão 7.3 do protocolo, capaz de inicializar e manter a rede WirelessHART com o auxílio de um Gateway e de um Ponto de Acesso sem fio.

1.1 HART Communication Foundation

Criado em 1993, o HART Communication Foundation, uma organização internacional sem fins lucrativos, é o proprietário da tecnologia e autoridade central do protocolo HART.

O HCF mantém e controla as normas do protocolo, incluindo melhorias no padrão já existente e o desenvolvimento de novas tecnologias.

O foco principal de todas as atividades do HCF é promover a aplicação da tecnologia HART, além de fortalecer sua posição no mercado global e auxiliar os usuários do padrão a maximizarem seus investimentos em instrumentação inteligente.

1.1.1 Criação do HCF

Em setembro de 1990, um grupo de profissionais da indústria reuniu-se em Bloomington, Minnesota, para o primeiro encontro oficial daquilo que mais tarde seria chamado de *HART Users Group* (Grupo de Usuários HART). A despeito de ter durado dois dias e requerido apenas cinco páginas de registro, esse encontro marcou o início do HART como uma tecnologia de comunicação aberta.

Representantes de vinte e seis companhias fizeram-se presentes no primeiro encontro, onde discutiu-se sobre a tecnologia HART - suas camadas de Aplicação, Enlace e Física - e a organização de um grupo que manteria as normas do protocolo e proveria suporte mundial à tecnologia.

Entre as companhias participantes, pode-se citar:

- Exxon
- Hitachi
- Siemens
- SMAR

Em um segundo encontro, em dezembro de 1990, as companhias-membro estabeleceram quatro grupos de trabalho: Definição, Teste de Conformidade, Interface Homem-Máquina e Interoperabilidade. Três encontros foram realizados em 1991 e, em novembro, outros dois grupos de trabalho foram criados: Saída e PID.

Em março de 1993, o Grupo de Usuários HART votou à criação do grupo de apoio que havia sido discutido na primeira reunião: uma organização independente e sem fins lucrativos para gerenciar e dar apoio ao protocolo HART. Em junho, a HART Communication Foundation era criada.

1.2 Protocolo HART

Com uma base global instalada de mais de trinta milhões de dispositivos, o protocolo de comunicação HART é reconhecido mundialmente como um padrão pela indústria. Uma de suas principais características, sendo a que o diferencia dos demais protocolos *fieldbus*, é a sobreposição sem interferência do sinal analógico 4-20mA padrão por um sinal digital. Com isso, o uso do protocolo HART não é restrito apenas à instrumentação analógica, podendo ser estendido para uso em todos os sistemas de controle de processos de plantas, tornando a monitoração dos processos mais eficaz. Os dispositivos capazes de trabalhar com esta comunicação híbrida são chamados de *smart*, ou inteligentes.

1.2.1 Funcionamento do Protocolo HART

O protocolo HART foi baseado no padrão **Bell 202**. Em um sinal analógico 4-20mA padrão é sobreposto um sinal digital modulado em **FSK** (Figura 1.1). Assim, os bits 0 e 1 são transmitidos modulando uma onda senoidal de 1mA de valor pico a pico nas frequências de 2400Hz e 1200Hz, respectivamente. Como o valor médio de uma onda senoidal é 0, nenhum componente DC é adicionado ao sinal original de 4-20mA, sendo normalmente utilizado um filtro passa-baixas para a recuperação do sinal analógico.

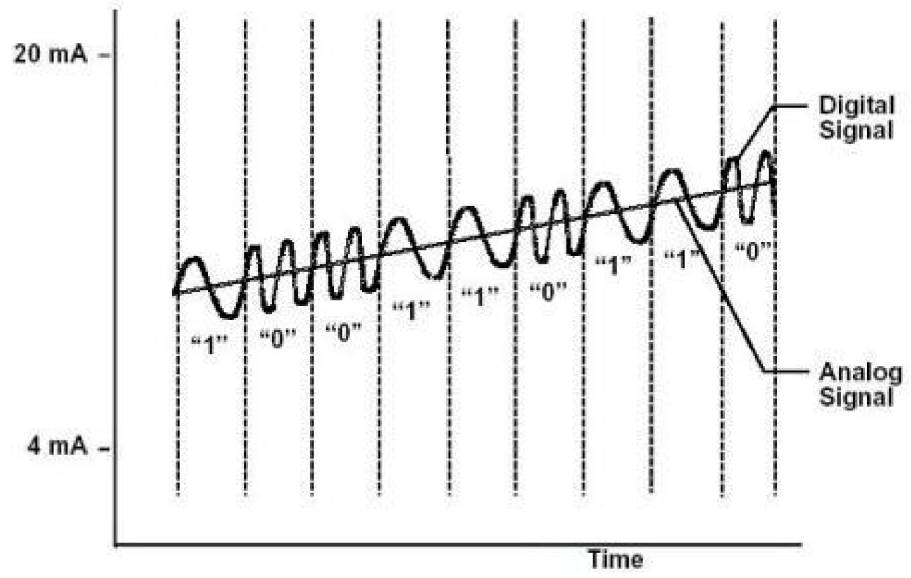

Figura 1.1: Sinal digital modulado por FSK sobreposto ao analógio de 4-20mA

2 O PROTOCOLO WIRELESSHART

Este capítulo contém as informações teóricas necessárias para a compreensão do desenvolvimento do Network Manager WirelessHART. O texto a seguir foi produzido com base no conteúdo das especificações do protocolo (HCF, 2009) e de bibliografia complementar (CHEN et al., 2010), além das conclusões obtidas de forma experimental ao longo do período de desenvolvimento.

2.1 Entidades da rede

O conceito de entidade no padrão WirelessHART está diretamente relacionado com o de função ou papel desempenhado dentro do sistema de comunicação. A delimitação de uma entidade é meramente lógica, não implicando restrição alguma à implementação, embora seja fundamental para a definição do modo de operação da rede. Podemos enxergar uma entidade como uma unidade funcional, que deve comportar-se como definido na especificação do padrão. No que tange à implementação, é possível que tenhamos entidades como o *Network Manager* (Gerente de Rede, NM), Gateway e Ponto de Acesso (NAP) idealizados de diversas formas. Todos podem ser implementados em software e compartilhar o hardware (em uma plataforma de propósito geral), por exemplo, ou rodarem em plataformas diferentes, tendo sua comunicação encapsulada em outro protocolo. Um exemplo de rede WirelessHART pode ser visto na Figura 2.1, onde temos Network Manager, Security Manager, Gateway e Access Point implementados em um único dispositivo de hardware. Os dispositivos de campo podem ser sensores, atuadores, etc.

2.1.1 Dispositivos de Campo

São os responsáveis por caracterizar e controlar o processo industrial de interesse. Produzem e consomem pacotes WirelessHART e devem ser capazes de rotear esses pacotes em função dos outros dispositivos da rede.

2.1.2 Adaptadores

São os responsáveis por conectar um dispositivo de campo HART à rede WirelessHART.

2.1.3 Handhelds

São dispositivos portáteis controlados por operadores da planta usados no monitoramento e diagnóstico da rede WirelessHART.

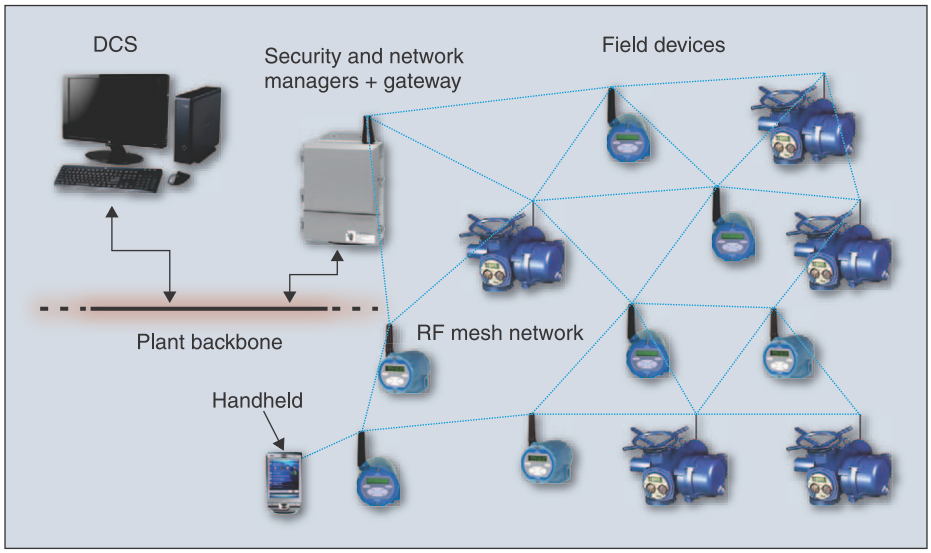

Figura 2.1: Exemplo de topologia de uma rede WirelessHART

2.1.4 Gateway

O gateway possibilita a comunicação das entidades presentes na estação central da planta com a rede sem fio, através de um Access Point. Tais entidades podem ser tanto aplicações de usuário quanto o próprio Network Manager.

2.1.5 Access Point

Conecta o Gateway à rede WirelessHART. Possui, de um lado, uma conexão WirelessHART, e do outro, uma conexão externa, podendo ser Ethernet, Wi-Fi ou outra proprietária. O padrão WirelessHART não define essa conexão.

2.1.6 Network Manager

É a entidade central de gerenciamento da rede. Pode existir de forma distribuída e redundante, mas apenas um gerente de rede pode estar ativo em qualquer instante do tempo. Gerencia o escalonamento da rede, ou seja, quando ocorrerá a comunicação de quaisquer dois dispositivos em uma rede WirelessHART. Além do escalonamento, o gerente é responsável pela formação e manutenção da rede. Isso consiste em realizar *Join* (processo de incorporação) de novos dispositivos de campo e detectar dispositivos ausentes e excluí-los da rede, por meio da interpretação de pacotes de *health report*. Outra atribuição do gerente é organizar o roteamento de pacotes na rede sem fio. O roteamento propriamente dito é realizado pelos dispositivos de campo, baseando-se na consulta de uma tabela local cuja construção e manutenção é responsabilidade do Gerente de Rede.

2.1.7 Security Manager

O gerente de segurança WirelessHART é a entidade responsável pela geração e armazenamento das chaves utilizadas em diversos processos da rede, como a autenticação ou decriptação de um pacote. Pode estar implementado junto ao Gerente.

Camada OSI	Função	WirelessHART
Aplicação	Provê ao usuário aplicações com capacidade de rede	Orientada a comandos. Tipos de dados pré-definidos e procedimentos de aplicação
Apresentação	Converte dados de aplicação do formato da rede para o formato local	
Sessão	Servicos de gerência de conexão para aplicações	
Transporte	Provê transferência de mensagens de modo transparente e independente da rede	Transferência auto-segmentada de grandes conjuntos de dados, stream de transporte confiável, tamanhos de segmento negociados
Rede	Roteamento fim-a-fim de pacotes Resolução de endereços de rede	Otimizado para baixo consumo, caminhos redundantes, rede em malha autorregenerativa
Enlace	Estabelece a estrutura dos pacotes de dados, enquadramento, detecção de erros	Segura e confiável, TDMA/CSMA, ARQ
Física	Conexão mecânica/elétrica Transmite stream de bits puro	2.4GHz Wireless, rádios 802.15.4, Tx de 10dBm

Figura 2.2: Protocolo WirelessHART visto em camadas OSI

2.2 Camada Física

Introduzida na revisão de número 7 do protocolo HART, a camada física sem fio expande o campo de aplicações do padrão, tornando possível a sua utilização em ambientes que apresentem condições adversas à comunicação cabeada. Além disso, a comunicação sem fio simplifica a implantação e a manutenção de uma rede em ambientes industriais.

Esta camada é a responsável por definir as características do rádio, tais como métodos de sinalização, potêncial do sinal e sensibilidade do dispositivo. É baseada numa versão do padrão IEEE 804.15.4-2006 2,4GHz DSSS. Dessa maneira, o WirelessHART opera na faixa ISM 2400-2483,5MHz livre de licença, com uma taxa de dados de até 250 kbps. Seus canais são enumerados de 11 a 26, com uma banda de guarda de 5MHz entre canais consecutivos. Quinze canais estão disponíveis na banda 2450 MHz ISM (Tabela 2.1), uma vez que o canal 26 não é suportado pelo protocolo visto que seu uso não é permitido em alguns países. O centro de frequência dos canais, em MHz, é definido por

$$F_c = 2405 + 5(k - 11), \text{ com } k \in \{11..26\},$$

onde k é o número do canal.

É válido afirmarmos que a função da camada física WirelessHART é a de receber e transmitir dados de forma compatível com a prevista no padrão IEEE 802.15.4. Outras características de interesse da camada física incluem (CHEN et al., 2010):

- **Potência de Transmissão**: o padrão IEEE 802.15.4 é definido para uma *Personal Area Network*, ou rede de área pessoal, com um espaço operacional pessoal (*Personal Operating Space*) de dez metros. No entanto, a malha WirelessHART pode cobrir áreas maiores, com um alcance máximo de 100m por nodo. Todos os dispositivos devem fornecer um EIRP de +10dBm (10mW) $\pm$ 3dB. A potência de transmissão é programável na faixa de -10dBm a +10dBm;

- **Salto de Canais**: Afim de utilizar toda a banda disponível, o canal de frequência utilizado em uma comunicação é alterado a cada transmissão. Tal prática é chamada de "salto de canais".

Tabela 2.1: Frequência dos canais.

Índice	Canal 802.15.4	Frequência (Mhz)
0	11	2405
1	12	2410
2	13	2415
3	14	2420
4	15	2425
5	16	2430
6	17	2435
7	18	2440
8	19	2445
9	20	2450
10	21	2455
11	22	2460
12	23	2465
13	24	2470
14	25	2475
15	26	Não utilizado

2.3 Camada de Enlace

Uma caracterísitica distinta do protocolo WirelessHART é a sua camada de enlace sincronizada no tempo, onde é definido um intervalo (slot) de tempo rigorosamente preciso de 10ms utilizando tecnologia TDMA. O conceito de superframe é apresentado, sendo este uma sequência consecutiva de slots de tempo. Um superframe é periódico, com período igual à quantidade de slots de tempo que o compõe. Todos os superframes numa rede WirelessHART iniciam com um **ASN** (Absolute Slot Number) igual a 0 no momento em que a rede é criada.

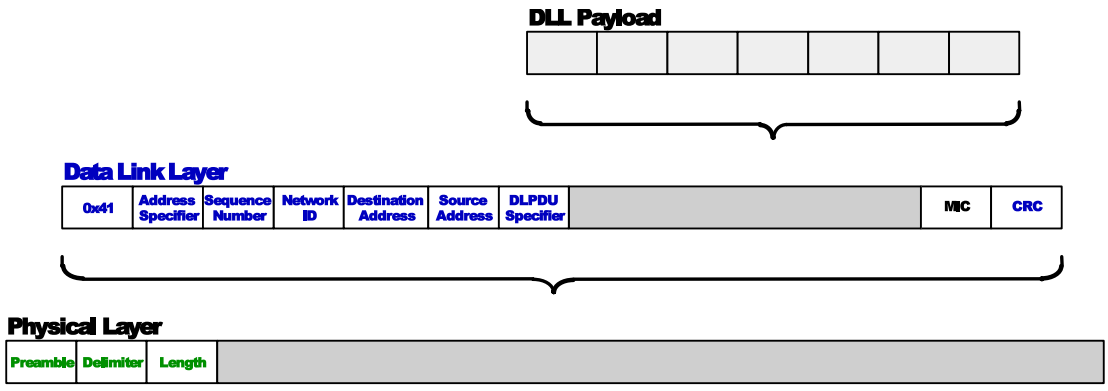

Figura 2.3: Estrutura do *Data Link Packet Data Unit* (DLPDU)

A estrutura do pacote de enlace (DLPDU) pode ser vista na Figura 2.3. É possível observarmos no cabeçalho de enlace os campos de especificação de endereço, número de sequência do pacote, ID da rede, endereço destino, endereço fonte e especificador do DLPDU.

Como o funcionamento correto de uma malha WirelessHART é altamente dependente

da sincronia temporal entre os dispositivos, o padrão prevê diversos mecanismos para a manutenção de um base de tempo sincronizada em toda a extensão da rede. Um deles, por exemplo, funciona da seguinte maneira: sempre que um nodo destino recebe um DLPDU o seu instante de recebimento é registrado, comparando-o com o instante em que o nodo destino esperava que tal comunicação acontecesse segundo o seu próprio relógio; essa variação é então incluída em todos os pacotes de ACK (confirmação) enviados ao nodo fonte para que o dessincronismo dos relógios possa ser medido.

No protocolo WirelessHART, uma transação em um slot de tempo é descrita por um vetor composto pelos seguintes campos:

- **Frame ID**: indica um superframe específico;
- **Index**: índice do slot no superframe;
- **Type**: indica o tipo de slot. Pode ser de transmissão, recepção ou inativo;
- **Source Address**: endereço do dispositivo de origem;
- **Destination Address**: endereço do dispositivo de destino;
- **Channel Offset**: fornece o canal lógico a ser usado na transmissão.

O uso dos canais de rádio pode ser ajustado pelo administrador da rede por meio de *channel blacklisting*, ou seja, é possível proibir o uso de um determinado conjunto de canais na comunicação da rede como um todo (escolhidos por meio de *channel hopping*, ou salto de canais). Os critérios de bloqueio podem variar desde a ocorrência frequente de interferências até à proteção de redes coexistentes operando na faixa de 2,4GHz. O *blacklisting* pode ser implementado aplicado-se uma máscara ao valor de 16 bits chamado de Mapa de Canais. Este valor é programado pelo gerente em todos os dispositivos de campo (e ponto de acesso) durante a inicialização dos mesmos, e sua função é exatamente a de indicar quais canais podem ser escolhidos para a comunicação, ou seja, quais os canais ativos.

O canal resultante do *channel hopping*, para um *offset* **OffsetCanal** e número de canais ativos **NumCanais** é definido por:

$$\textbf{Canal} = (\textbf{OffsetCanal} + \textbf{ASN})/\textbf{NumCanais}$$

Os quadros (*frames*) utilizados na comunicação são criados e gerenciados pela camada de enlace. Além disso, é responsabilidade dessa camada garantir a transferência confiável de dados entre os nós da rede, procurando sempre detectar e corrigir possíveis erros nos dados que são recebidos da camada física. A camada de enlace pode ser dividida em duas subcamadas: LLC (*Logical Link Control*), cujas principais funções são o controle de erros, controle de fluxo e montagem dos quadros de endereçamento, e MAC (*Media Access Control*), onde é realizado o controle temporal da comunicação como um todo. Em outras palavras, a LLC garante a *integridade* e o *formato* dos dados, enquanto a MAC determina *quando* os dados são transmitidos e/ou recebidos.

2.4 Camada de Rede

Esta camada lógica tem a função do encaminhamento dos pacotes WirelessHART a seus destinos finais interpretando informações de roteamento, provendo um canal seguro de comunicação fim-a-fim, além de disponibilizar serviços de transporte às camadas de nível superior.

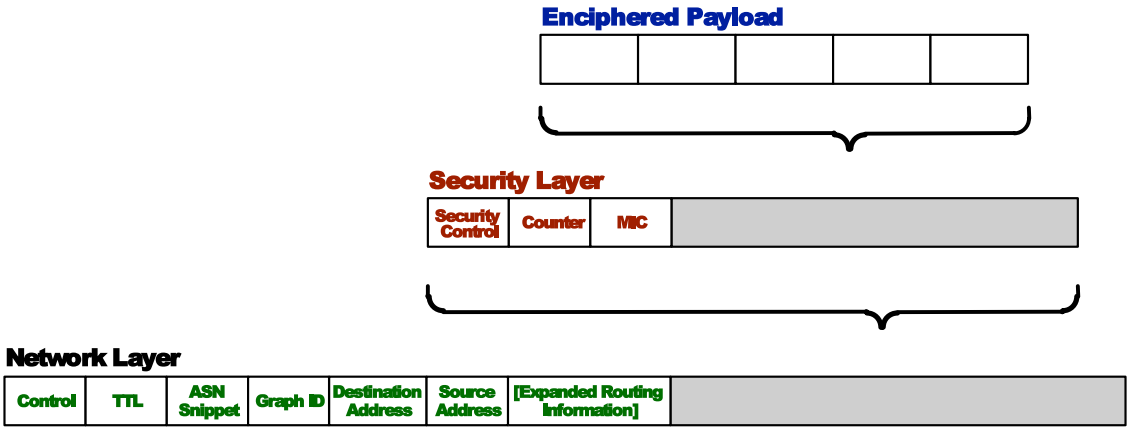

Figura 2.4: Estrutura do pacote de Rede

2.4.1 Sessões

Devido à questões de segurança, o padrão WirelessHART é orientado a sessões, e todos os dispositivos devem suportar sessões múltiplas. Através do seu uso, é possível garantir uma comunicação segura e privada entre dois endereços de rede. Geralmente, quatro sessões são criadas no momento em que um dispositivo faz Join:

- Sessão *Unicast* entre o Network Manager e dispositivo de campo, utilizada pelo NM para gerenciar o dispositivo;
- Sessão *Broadcast* do Network Manager para todos os dispositivos da rede, usada para gerenciamento global da rede (envio de informações a todos os dispositivos);
- Sessão *Unicast* entre o Gateway e o dispositivo, para a tráfego normal de dados;
- Sessão *Broadcast* do Gateway para todos os dispositivos.

Uma sessão pode ser do tipo Unicast, Broadcast ou Join. Com exceção da sessão de Join, que deve ser criada já na incialização do dispositivo ou a cada *reset*, todas as sessões são criadas pelo Network Manager através do comando **963** (Write Session). Somente o NM tem autoridade para criar, alterar ou excluir uma sessão e sua respectiva chave.

Sessões adicionais podem ser criadas, para um Handheld, por exemplo. É importante observar que, embora seja tecnicamente possível, a criação de uma sessão entre dois dispositivos de campo é proibida por motivos de segurança. O controle das sessões e das suas chaves deve sempre ser centralizado no Network Manager (em conjunto com o Security Manager).

2.4.2 Roteamento

É previsto no padrão que, em um ambiente real, nem todos os dispositivos de campo estejam ao alcance dos Pontos de Acesso da central da planta. Dessa forma, é necessário rotear os pacotes de comunicação através de nodos intermediários. Os critérios para a escolha da rota podem variar conforme a necessidade do usuário, como menor atraso, tempo real, etc.

Todo o dispositivo de campo WirelessHART deve ser capaz de encaminhar pacotes para os quais não seja o destino final, ou seja, deve ser capaz de atuar como um roteador. Para que isso aconteça, as informações de endereçamento em um pacote de rede não podem ser encriptadas com uma chave de sessão, pois só os nodos fonte e destino a conhecem.

O protocolo WirelessHART prevê dois tipos de roteamento:

- **Graph routing**: a malha WirelessHART pode ser representada como um grafo, com seus nodos conectados por arestas direcionadas (representando os links de um dispositivo a outro). Um Grafo (*Graph*) nada mais é do que um subconjunto dos links direcionais e seus respectivos nodos na rede, definindo rotas redundantes para a comunicação entre um par de dispositivos. A escolha da rota acontece no momento da transmissão/recepção, dependendo das condições da rede naquele instante;

- **Source routing**: uma rota do tipo *Source* consiste em um caminho único, direcionado ao nodo destino partindo do nodo fonte. Esse tipo de rota é especificada no próprio pacote de maneira estática. Condições desfavoráveis da rede no momento da comunicação podem impedir a entrega do pacote.

No caso de roteamento por grafo, é necessária a configuração prévia do grafo (ou de parte do grafo) em cada nodo. Nos dispositivos, as informações de roteamento são armazenadas em tabelas, que são consultadas conforme necessário para o encaminhamento do pacote. Além disso, é esperado que o dispositivo tenha sido provisionado com links necessários para encaminhar o pacote a qualquer um de seus vizinhos. Este tipo de roteamento confere maior confiabilidade à comunicação por ser redundante, e deve ser utilizado para a comunicação normal (alarmes, requisições/resposta, publicação, etc), tanto para tráfego *upstream* quanto *downstream*. Já o roteamento por fonte deve ser utilizado apenas em casos de teste e depuração devido à sua baixa confiabilidade.

Os caminhos (*paths*) associados a cada grafo devem ser instanciados explicitamente pelo Network Manager em cada dispositivo de campo. É possível a existência de múltiplos grafos em uma única rede WirelessHART, inclusive com nodos em comum. Cada dispositivo de campo pode fazer parte de múltiplos grafos, inclusive para os mesmos vizinhos, lembrando que grafos são unidirecionais.

Todo o grafo é associado a um identificador chamado *Graph Id*. Quando ativo o roteamento por grafo, o dispositivo inclui a informação de Graph Id no cabeçalho de rede (Figura 2.4). Quando o pacote é recebido no nodo seguinte, o dispositivo usa o identificador para determinar o endereçamento.

Um caso especial de roteamento por grafo é o roteamento por superframe, ou *Superframe Routing*. Quando ativo, o dispositivo de campo insere um identificador de superframe no NPDU, ao invés do Graph Id. O dispositivo seguinte, não encontrando um grafo com tal identificador, procura por um superframe respectivo. Caso exista, o pacote pode ser encaminhado para qualquer vizinho com que possua links naquele superframe. Graph

Ids com valores inferiores a 256 indicam roteamento por superframe, enquanto valores superiores indicam o roteamento por grafo.

2.4.3 Segurança

A aunteticação e encriptação dos dados é realizada com o auxílio de chaves providas ao Network Manager e aos disposisitivos de campo pelo Security Manager. O padrão WirelessHART adota o algoritmo de encriptação CCM (*Counter with CBC-MAC*) definido na especificação do padrão 802.15.4.

São três os tipos de chaves utilizados na rede WirelessHART:

- **Join Key**: utilizada para a encriptação do pacote de Join Request. Deve ser informada ao dispositivo por meio da Porta de Manutenção ou armazenada em memória não-volátil;
- **Network Key**: utilizada na autenticação a nível de Enlace. É uma chave pública, comum a todos os dispositivos;
- **Session Key**: utilizada para a autenticação e encriptação a nível de Rede. Cada sessão pode fazer uso de uma Session Key única, e é informada ao dispositivo na criação de uma sessão.

É importante salientar que o algoritmo de encriptação leva em conta diversos parâmetros além das chaves de segurança. No nível de enlace, o payload da camada física (sem os campos MIC e CRC), endereço de origem e ASN também são processados. No nível de rede, o cabeçalho de Rede (com os campos TTL, Counter e MIC zerados), o payload de Rede, endereço de origem e NonceCounter também contribuirão na encriptação.

2.5 Camada de Aplicação

Por ser a mais próxima do nível de usuário, a camada de aplicação apresenta o maior nível de abstração de todas as camadas OSI. Quando trabalhamos nesta camada, não estamos interessados em como a informação vai ser transportada, como será endereçada, ou em qualquer medida necessária para garantir a integridade ou correção dos pacotes, e sim única e exclusivamente nos próprios dados e no seu significado para o "mundo real".

A definição da camada de aplicação WirelessHART é apresentada na forma de comandos, afim de dar suporte à comunicação baseada em requisição-resposta utilizado no padrão. Os dois principais casos de uso dos comandos de aplicação são a publicação de informações por parte dos dispositivos de campo e o envio de requisições originadas no Network Manager, Gateway ou aplicação de usuário. Um resumo dos grupos de comandos pode ser observado na Tabela 2.2.

2.5.1 Categorias de Comandos

Os comandos são divididos em seis categorias:

- **Universal**: comandos básicos cuja implementação é obrigatória;
- **Common Practice**: conjunto de comandos aplicável a uma gama abrangente de dispositivos, que devem ser suportados sempre que possível;

Tabela 2.2: Divisão dos comandos HART.

Commandos	**Tipo**
0 - 30, 38, 48	Universal
31	Flag adicional
21 - 121 (Exceto 38 e 48)	Common Practice
122 - 126	Non-public
127	Reservado
128 - 253	Device-Specific
254 - 511	Reservados
512 - 767	Common Practice - adicionais
768 - 1023	Wireless
1024 - 33791	Device family
33792 - 64511	Reservados
64512 - 64763	Device Specific - sem fio
64766 - 64767	Reservados
64768 - 65021	Device Specficic - adicionais
65022 - 65535	Reservados

- **Non-Public**: categoria especial de comandos que só devem ser utilizados durante o processo de desenvolvimento, para fins de teste;

- **Wireless**: conjunto de comandos destinados à parte sem fio do padrão HART. Todo e qualquer produto compatível com o padrão WirelessHART deve implementar esse grupo de comandos;

- **Device Family**: comandos utilizados para a parametrização e serviço de dispositivos de campo que se encaixem em um perfil generalizado, sem a necessidade de conhecimento sobre uma tecnologia específica;

- **Device Specific**: comandos desenvolvidos pelo fabricante do produto, que deem suporte a características e necessidades específicas de um dispositivo;

2.6 Inicialização da Rede

O processo de formação de uma rede WirelessHART começa no Network Manager. De posse de uma *network ID* (que pode ser fornecida em tempo de execução ou configurada previamente) e das demais chaves de segurança (*network key, session key* e *join key*) o gerente de rede cria um canal seguro e confiável de comunicação com o Gateway. O provisionamento inicial do Ponto de Acesso será realizado através desse canal, e consiste na passagem das seguintes informações:

- O superframe de gerenciamento, que garante banda mínima para a execução das funções básicas de monitoramento e manutenção da rede;

- O grafo de rede para tráfego *upstream*, ou seja, em direção ao Network Manager;

- O superframe e links de *Join*, que possibilitarão a entrada de novos dispositivos na rede;

- Links dedicados e compartilhados (tanto de transmissão quanto de recepção) para o gerenciamento de dispositivos, tráfego de *health reports* e comunicação de alarmes (link perdido, por exemplo).

Afim de maximizar a vazão de pacotes de gerenciamento e de anúncio (*advertisement*), o Ponto de Acesso deve ser configurado como ativo em todos os slots TDMA. A rede WirelessHART é considerada ativa no momento em que o Gerente de Rede habilita o primeiro superframe e o ASN de número 0 é estabelecido. Assim, os primeiros dispositivos de campo podem entrar na rede a partir do momento em que o Ponto de Acesso começar a publicar pacotes de anúncio, iniciando o processo de formação da rede WirelessHART.

O processo de formação da rede pode ser dividido em três etapas: *advertisement, Join* e negociação de parâmetros. O *advertisement* é realizado por dispositivos de campo que já passaram pelo processo de Join, e sua função é informar a presença da rede e possibilitar a incorporação de novos dispositivos que estejam ao seu alcance de sinal. O pacote de anúncio deve conter as informações de Network ID e ASN, além dos links e superframe de Join. Tal pacote é esperado por dispositivos passíveis de Join, cuja resposta será em forma de um pedido para o dispositivo em questão juntar-se à rede, ou seja, um *Join Request*. O processo de Join de um dispositivo pode ser observado na Figura **??** .

Assim que um novo dispositivo for agregado à rede, tanto o Gateway quanto o dispositivo farão requisições de banda ao Gerente. O Gateway necessitará da banda para suporte ao tráfego de requisições/respostas, enquanto o dispositivo fará uso da banda para publicar informações sobre a variável de processo, por exemplo. O Gerente de Rede estimará a banda necessária através dessas requisições de serviço, de modo a otimizar a gerência da rede. Caso exista banda suficiente, o Gerente alocará superframes e links de acordo com as requisições. Caso contrário, o Gerente pode alocar menos banda do que o dispositivo requeriu, ou até mesmo recusar totalmente o pedido.

2.6.1 O Processo de *Join*

Join é o nome dado à etapa da inicialização onde um novo dispositivo junta-se à rede WirelessHART, tendo acesso aos recursos disponibilizados pelo Gerente e desempenhando as funções esperadas de um dispositivo de campo. Os principais passos que formam o processo de Join são os seguintes, segundo a especificação de gerenciamento de rede do padrão WirelessHART (HCF, 2009):

- O anúncio periódico realizado por membros já existentes permite a identificação da rede por parte de possíveis novos dispositivos. Além disso, o pacote de anúncio contém as informações necessárias para um novo dispositivo sincronizar-se à rede e comunicar-se utilizando os links (ou timeslot e canal) corretos;

- Monitoramento constante por parte do novo dispositivo para localizar e sincronizar-se à rede;

- Estabelecimento de um canal de comunicação seguro entre o dispositivo e o Gerente. Isso é realizado fazendo-se uso da Join Key para a encriptação dos pacotes iniciais. Cada dispositivo pode ter uma Join Key associada, embora neste projeto a chave de Join utilizada seja a mesma para todos os dispositivos de campo.

- Verificação das "credenciais"do novo dispositivo. Tal verificação é realizada avaliando-se a *Identity* (conteúdo do comando 0) e a Long Tag do dispositivo, ambas enviadas

no pacote de Join Request, cifrado com a Join Key. A validação deste pacote determina se o dispositivo será aceito ou não na rede WirelessHART;

- Uma vez que o dispositivo é aceito na rede, o Gerente deve fazer seu provisionamento inicial com superframes e links do tipo Normal para que se torne operacional.

3 DESENVOLVIMENTO DO NETWORK MANAGER

Neste capítulo são detalhados todos os passos do processo de implementação do Network Manager, além do software auxiliar utilizado (implementação de outras entidades).

3.1 Plataforma de Hardware

O software que compõe a implementação do padrão WirelessHART utilizada neste trabalho está dividido principalmente em duas partes: uma delas implementa o *stack* propriamente dito e é executada no dispositivo de campo (firmware), e a outra é executada em uma máquina de propósito geral, que é a implementação da central de controle da planta (software).

O firmware utilizado foi desenvolvido para a plataforma **MC1322x**, fabricado pela Freescale, também bastante utilizado em implementações dos padrões ZigBee e 802.15.4. A Figura 3.1 mostra um diagrama de blocos da plataforma. O processador principal do sistema implementa a arquitetura ARM7, de 32 bits.

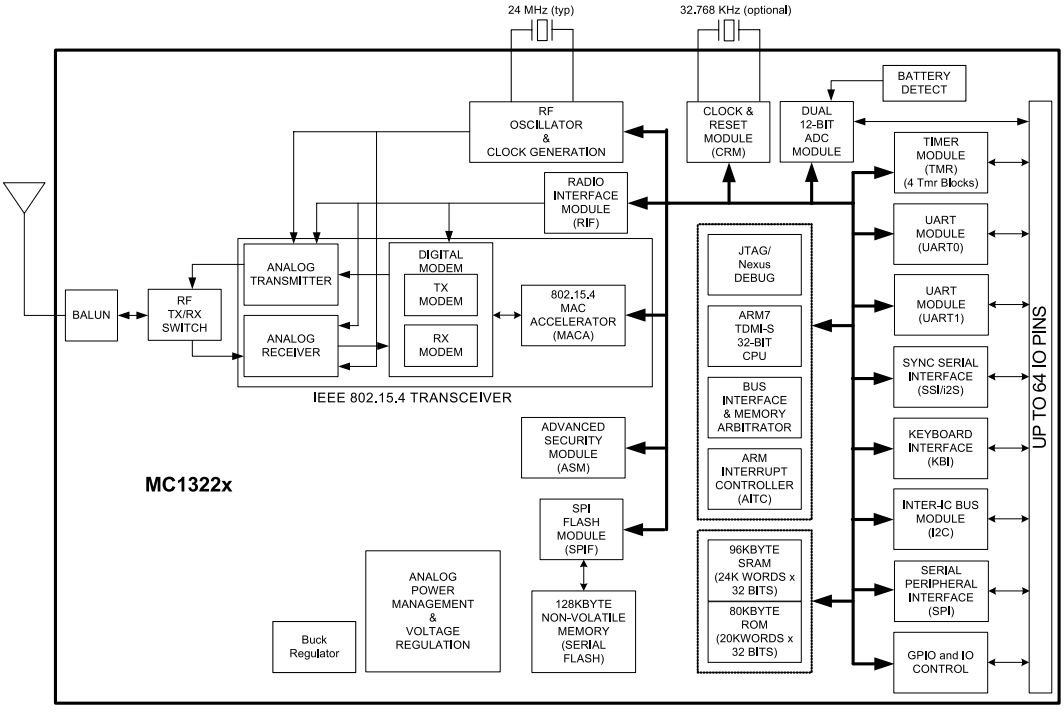

Figura 3.1: Diagrama de blocos da platforma MC1322x

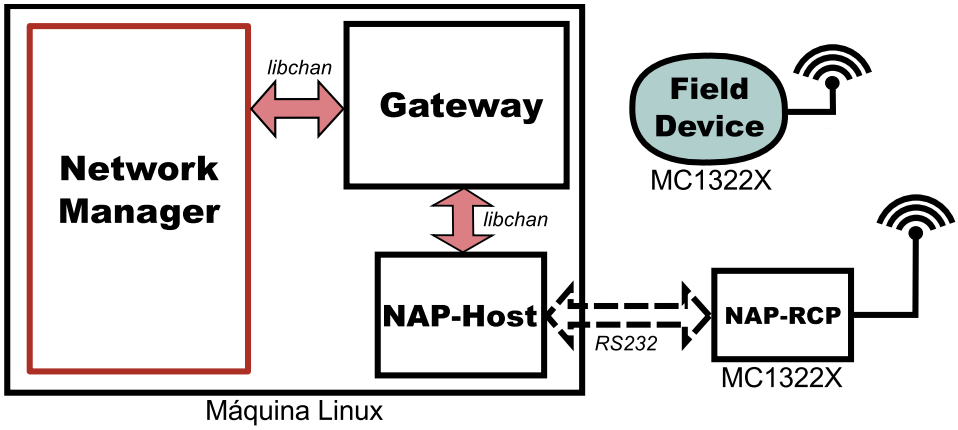

Figura 3.2: Organização das entidades na implementação

A central de processamento da planta foi desenvolvida sobre uma plataforma híbrida. Tanto o Gateway quanto o Network Manager foram implementados como processos individuais em uma máquina Linux. O Ponto de Acesso é composto pelo **NAP-RCP**, que consiste em um rádio MC1322X executando um *firmware* WirelessHART modificado, e por um software de interface chamado **NAP-Host**, responsável pela comunicação com os outros dois processos no sistema Linux. O diagrama da Figura 3.2 mostra a organização da implementação na forma de blocos funcionais. Pode-se observar na Figura 3.3 os nodos MC1322X utilizados no desenvolvimento.

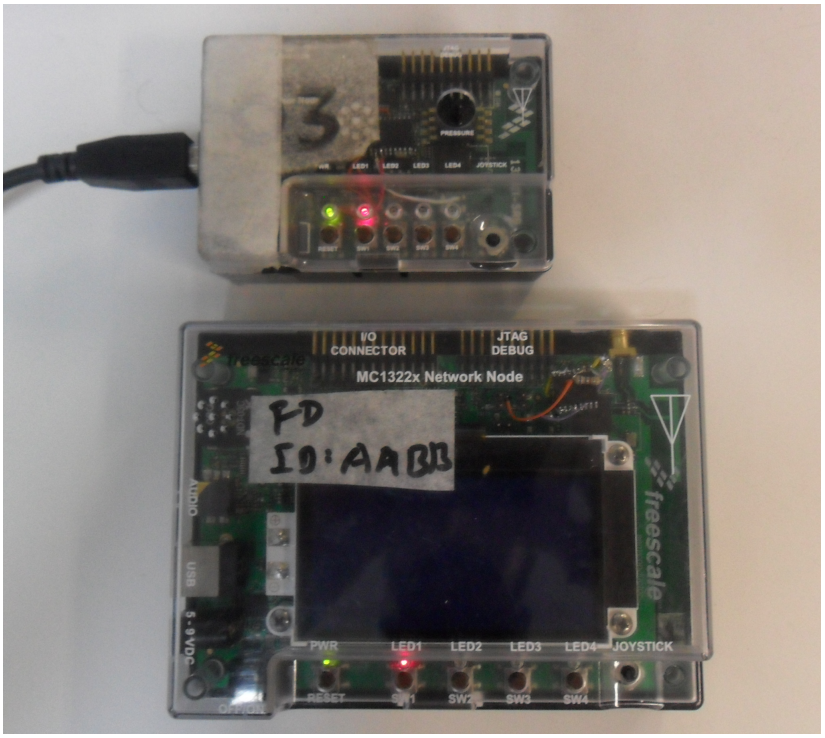

Figura 3.3: Nodos MC1322X utilizados nos testes. **a)** acima, NAP-RCP, conectado à máquina Linux; **b)** abaixo, dispositivo de campo

A diferença entre um dispositivo de campo comum e o NAP-RCP é a de que este atua como um canalizador de dados das camadas física, de enlace e de rede, direcionando todo o conteúdo de aplicação (no caso de recepção de dados) para o NAP-Host

através de uma interface serial. O NAP-Host, por sua vez, direciona o tráfego recebido ao Gateway, desempenhando assim a sua função de Ponto de Acesso. A comunicação no sentido contrário é realizada de forma análoga. Neste trabalho, para simplificação, refere-se ao conjunto NAP-RCP e NAP-Host apenas como **NAP**. Desta forma, três processos diferentes são executados na estação principal: NAP, Gateway e Network Manager. As três entidades foram implementadas na linguagem de programação **C++**, estruturadas de forma modular, orientada a objetos. Na próxima seção, a estrutura e o contexto da implementação do Network Manager serão abordados de forma detalhada.

3.2 Trabalhos Anteriores

O ponto de partida para este Trabalho de Diplomação, no que diz respeito à implementação do Network Manager, compreende os resultados do trabalho de implementação do NAP (HAHN, 2011), também desenvolvido no LASCAR (Laboratório de Sistemas de Controle, Automação e Robótica). Ambos os trabalhos foram baseados em uma implementação inicial das entidades existentes no LASCAR. No entando, os códigos-fonte não eram funcionais, e serviam apenas como base para o entendimento do protocolo e para pesquisas sobre o padrão WirelessHART. Até a realização de ambos os trabalhos, não tínhamos conhecimento do quanto poderia ser aproveitado da implementação original. Os resultados do trabalho de (HAHN, 2011) mostraram que o código referente ao Gateway pôde ser utilizado quase sem modificações; o código do NAP-RCP, por outro lado, necessitou ser reimplementado devido à diferença na plataforma-alvo para o qual tinha sido idealizado originalmente.

Para a validação do seu trabalho, o autor utilizou scripts Perl para emular o comportamento do Network Manager em uma situação específica. Os scripts, de execução puramente sequencial, apresentam um nível de complexidade muito menor do que a implementação completa do Network Manager, e cumpriram o propósito de validação do Ponto de Acesso sem a necessidade da compreensão e empenho envolvidos em depurar, corrigir e modificar a implementação do Gerente.

3.3 Estrutura e organização do código

A implementação do Network Manager segue uma estrutura modular. As classes C++ são determinadas de forma a representar conceitos-chave, cuja existência é obrigatória para que a entidade desempenhe as funções previstas na especificação do protocolo. As principais classes do projeto podem ser observadas na Figura 5.1, na seção de Anexos.

3.4 Configuração do Gerente

A configuração do Gerente de Rede é realizada através do *parsing* de um arquivo de configuração do tipo **XML**. O analisador foi implementado com o auxílio dos softwares Flex (analisador léxico) e Bison (analisador sintático).

O arquivo de configuração foi adotado na implementação anterior como uma maneira simples de prover ao Gerente todas as informações necessárias sobre a rede em questão. Tais informações devem ser as seguintes (um exemplo de arquivo de configuração pode ser visto na Figura 3.5):

- Mapa de canais;

```
// Superframes
linklist<superframe> *sprfrms;

// Network devices
linklist<netdevice> *netdevs;
```

Figura 3.4: Listas de superframes e de dispositivos, ambas de escopo global

- Superframes;
- Links;
- Lista completa de Dispositivos de Rede (nenhum dispositivo fora desta lista será aceito como membro da rede);
- Uma lista de possíveis vizinhos referente a cada dispositivo listado;
- Grafos;
- Rotas;

Apesar de termos um modo de inicialização bastante direto, sua principal desvantagem é impor restrições ao crescimento da rede de forma dinâmica, o que pode ser interessante quando a plataforma é embarcada e possui recursos restritos. A maior parte das implementações comerciais , apesar de encaixarem-se no conceito de embarcada, oferecem uma unidade de maior poder computacional para as entidades de gerenciamento (Gerente de Rede e Gateway). No entanto, na hipótese de haver uma solução que implemente um Gerente de Rede distribuído, a preocupação com o consumo de recursos de hardware é justificada.

A falta de documentação do software existente tornou necessária a compreensão do código dos analisadores léxico e sintático para que pudéssemos gerar um arquivo de configuração válido. A etapa de configuração é realizada na função **configure()**, onde o *parser* é chamado para percorrer o arquivo e construir as listas encadeadas que armazenam todas as informações sobre os dispositivos da rede e sobre os superframes que serão utilizados no escalonamento. As listas formadas a partir da leitura do arquivo são mantidas em memória como instâncias da classe **linklist** e são definidas no arquivo **netobjs.cc**, como pode ser observado na Figura 3.4.

Um dos objetivos deste trabalho foi o de tornar o processo de configuração mais flexível. A primeira modificação realizada foi a implementação da configuração dinâmica dos dispositivos de campo. Essa providência torna possível o crescimento indefinido da rede, sem limitações pré-estabelecidas durante a inicialização no que diz respeito a ao número e ao endereçamento dos dispositivos de campo. Para efetuar essa modificação, o primeiro passo foi observar em detalhes como era realizado o processo de registro dos dispositivos a partir do arquivo de configuração. Como visto na Figura 3.4, uma lista de escopo global é utilizada para armazenar as informações sobre os dispositivos, que é construída e inicializada pelo próprio *parser*. Após a etapa de configuração, nenhum outro nodo (dispositivo) é adicionado à lista. Assim, na configuração dinâmica será necessário o uso do método construtor da classe **netdevice** seguido da inserção da instância recém-criada na lista global de dispositivos.

```
<ChannelMap>
    <Channel channelNumber=11>enabled</Channel>
</ChannelMap>
<SuperFrames>
    <Superframe Id=1 NumTimeSlots=100 ActiveFlag="Active">
        <Link Id=0>
            <TimeSlot>30</TimeSlot>
            <ChOffset>5</ChOffset>
            <LinkType>Normal</LinkType>
            <LinkOpts>Transmit Shared</LinkOpts>
        </Link>
    </Superframe>
</SuperFrames>
<NetworkDevices>
    <NetworkDevice address=3 type="Gateway" Uid="0x1b1ef980000002">
    </NetworkDevice>
    <NetworkDevice address=4 type="NetworkManager" Uid="0x1b1ef980000001">
    </NetworkDevice>
    <NetworkDevice address=5 type="Device" Uid="0x1b1ef982000001">
        <Neighbor address=6/>
    </NetworkDevice>
    <NetworkDevice address=6 type="Device" Uid="0x1b1ef98200aabb">
        <Neighbor address=5/>
    </NetworkDevice>
</NetworkDevices>
```

Figura 3.5: Arquivo de configuração simplificado;

O próximo passo para implementar o processo dinâmico é definir *quando* um novo dispositivo deverá ser adicionado, ou, em outras palavras, qual evento será aguardado para que o registro de um novo dispositivo aconteça. Tal evento deve ser a recepção de um pacote de Join Request, por ser o primeiro pacote enviado por um dispositivo sincronizado que queira fazer parte da rede.

No entanto, é interessante observar o comportamento do Gateway quando um Join Request de um dispositivo não declarado é recebido na Camada de Rede (lembrando que todo o tráfego vindo dos dispositivos de campo é processado primeiro no Gateway):

Como podemos ver na Figura 3.6, o processamento do Gateway é interrompido pelo seguinte motivo: o Gateway não possui uma sessão de Join para este dispositivo de campo. De fato, sessões de Join para o NAP e para os demais dispositivos declarados são criadas no processo de inicialização do Gateway, na função **gateway_join_network()**. É preciso que o Network Manager crie uma sessão de Join do Gateway com o novo dispositivo para que o processamento do Join Request possa prosseguir. Isso foi resolvido com a criação de um comando que informa o Gerente da necessidade de uma sessão de Join para o dispositivo com o UID informado. O número do comando é **123**, é do tipo *non-public* (mesma categoria do comando 122 utilizado na inicialização) e foi chamado de Request Join Session.

Uma solução mais simples seria fazer com que a sessão de Join fosse criada diretamente pelo Gateway assim que a Camada de Rede acusasse o erro. Entretanto, o padrão WirelessHART determina que somente a entidade Gerente de Rede tem autoridade para comandar a criação de sessões, fazendo necessária a implementação do comando auxiliar. Para tal, foi necessário modificar o código do Gateway e inserir uma função para a construção de um pacote contendo o comando 123, cujo payload é simplesmente o UID

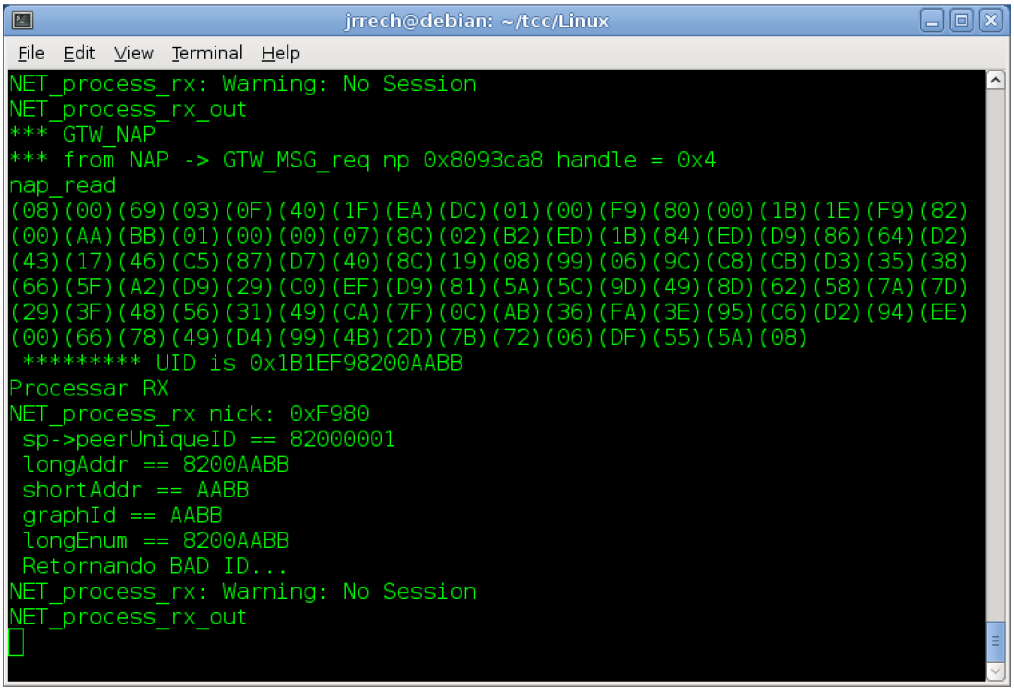

Figura 3.6: A falta da sessão de Join impede a inicialização do dispositivo

do dispositivo de campo para o qual a sessão de Join é necessária. No código do Gerente foram necessárias as alterações padrão (análogas àquelas realizadas na implementação dos comandos 122, 768 e 805) mais o código referente à interpretação do comando, já que é esperada uma ação do Gerente, diferente dos outros comandos onde a implementação resumia-se a "dar uma ordem"ao Gateway ou a outro dispositivo. A ação resultante do comando 123 foi implementada na função **proc_cmd_request_join_session()**. Essa função recebe como parâmetro o UID presente no payload do comando 123 e instancia um novo objeto da classe **netdevice** e insere-o na lista de dispositivos, para então criar uma sessão de Join com o Gateway através do comando 963 (Write Session). Um ponto importante é que, para a rede em questão, a Join Key é a mesma para todos os dispositivos, e por isso o Gerente só necessita do UID para criar a sessão.

Com a sessão criada, o processamento do Join Request pode prosseguir. Uma particularidade da implementação do Gateway é que, assim que a falta da sessão de Join é detectada, o pacote é descartado. Quando isso acontece, mesmo que o Gateway sinalize o Gerente através do comando 123 e a sessão seja criada corretamente, é necessário esperar o próximo Join Request (que é enviado periodicamente pelo dispositivo de campo) para que o Join seja iniciado.

3.5 Inicialização da Rede

Como visto no capítulo anterior, a primeira tarefa do Gerente de Rede, em ordem cronológica, é a de iniciar e formar a rede WirelessHART. A implementação inicial do Gerente apresentava inúmeras falhas neste processo. O setor de código referente a essa etapa foi totalmente reformulado, tendo como base tanto a especificação oficial do padrão quanto o script de testes referenciado anteriormente. Também foi necessária a implementação de alguns comandos WirelessHART no código-fonte do Gerente para que funcionasse corretamente em conjunto com o Gateway. Essas modificações serão abordadas

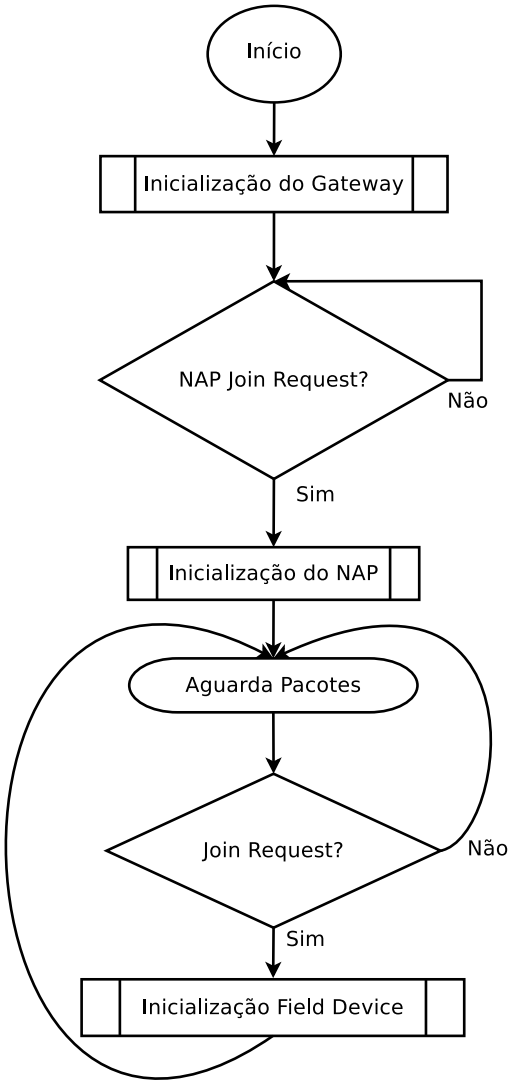

Figura 3.7: Fluxo de inicialização da rede pelo Network Manager

em detalhes nesta seção. O processo de inicialização executado pelo Network Manager pode ser visto no fluxograma da Figura 3.7.

3.5.1 Implementação de Novos Comandos

A implementação dos seguintes comandos foi necessária para o funcionamento correto em conjunto com o Gateway:

- Write Channel Map - 122;
- Write Join Key - 768;
- Write CCA Mode - 805;
- Correção no Write Link - 967: número do timeslot;
- Modificação no Write Route - 974: mudança nos parâmetros da função de construção do comando;

3.5.2 Inicialização do Gateway

O primeiro passo na formação da rede deve ser o estabelecimento da comunicação com o software que desempenha a função de Gateway. Isso é realizado através da biblioteca *libchan*, presente na implementação anterior. A *libchan* nada mais é que a abstração de uma comunicação via sockets UNIX, vistos pelas entidades como um "canal". Assim, o Gerente de Rede estabelece um canal de comunicação com o Gateway, que será mantido indefinidamente. Esse canal é indispensável, pois por ele trafegarão todos os dados entre o Gerente e os dispositivos de campo. Um canal análogo também é estabelecido entre o Gateway e o ponto de acesso.

3.5.3 Inicialização do NAP e dos Dispositivos de Campo

A inicialização e provisionamento do NAP são indispensáveis para possibilitar a comunicação dos dispositivos de campo com o Gerente de Rede. O processo de inicialização (o Join e o provisionamento inicial) é implementado no arquivo **formation.cc**.

O ponto de acesso sem fio é visto como um dispositivo de campo pelo Gerente de Rede, mas deve ser provisionado de forma especial. Todos os dispositivos de campo devem ser provisionados com links de transmissão e recepção com o NAP (e vice-versa) afim de se comunicarem com o Gerente, ao passo que a existência de links de um dispositivo de campo a outro não é obrigatória.

A implementação incial apresentava diversas falhas no processo de Join, impossibilitando a formação correta da rede. Foi necessária a reimplementação de todos os passos do processo, incluindo a diferenciação no tratamento do Join do NAP em relação a dispositivos de campo comuns na etapa de provisionamento. Em ambos os casos a solução foi implementada como uma máquina de estados, onde a transição para o estado seguinte é condicionada ao recebimento da confirmação dos comandos enviados no estado anterior. A implementação do processo Join pode ser observada na Figura 3.9.

Após o Join, é necessário que o dispositivo seja provisionado com superframes e links de modo a se comunicar normalmente dentro da rede WirelessHART. É responsabilidade do Network Manager organizar a rede de forma que a comunicação entre os dispositivos ocorra de forma correta e síncrona. O processo de organização da rede em superframes e links é chamado de **escalonamento**, e deve ser realizado de modo a atender os requisitos do processo em questão. Após a inicialização da rede, o escalonamento é a principal tarefa do Network Manager, e deve ser realizado constantemente, baseado em requisições de comunicação por parte dos dispositivos e no próprio estado da rede e suas capacidades.

Devido a sua alta complexidade, o escalonador não foi implementado neste trabalho. Como alternativa, a implementação foi testada com um provisionamento fixo. Assim como a função de Join, a função de provisionamento foi totalmente modificada, substituída por funções diferenciadas para o provisionamento do NAP e de dispositivos de campo. O provisionamento adotado para o NAP pode ser visto no fluxograma da Figura 3.8.

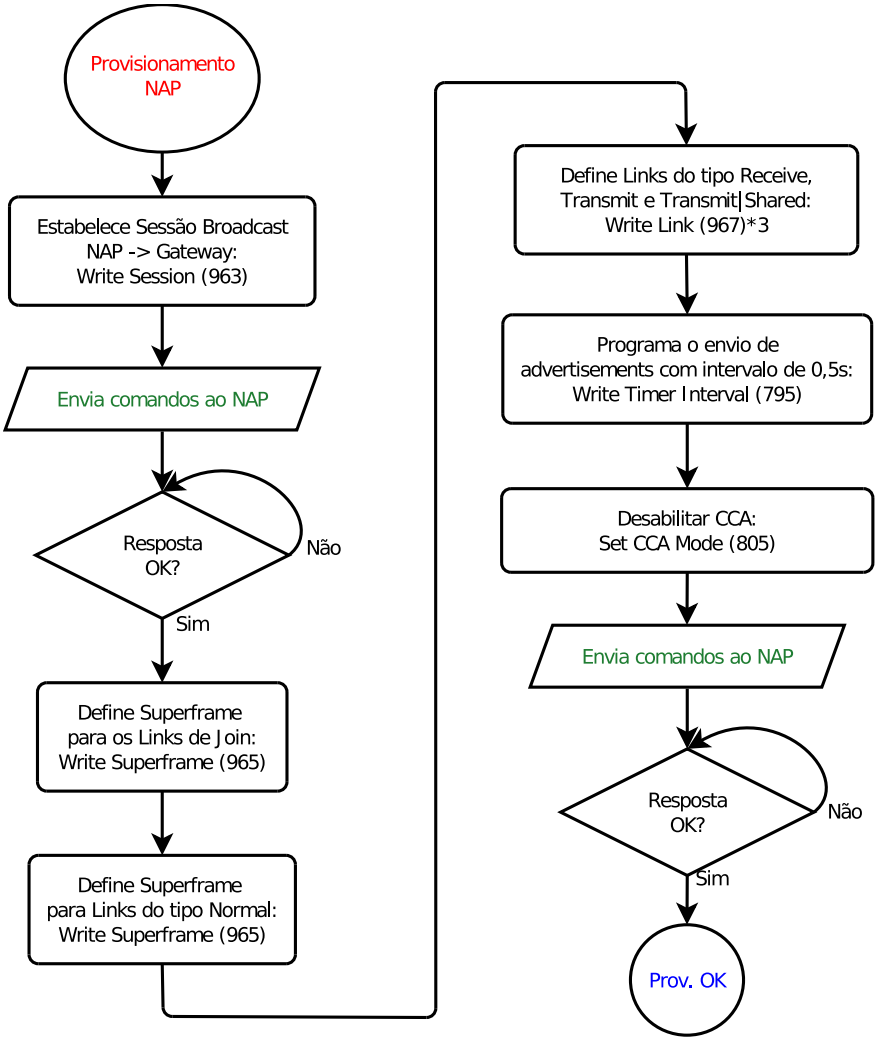

Figura 3.8: Fluxograma representando o processo de provisionamento do NAP

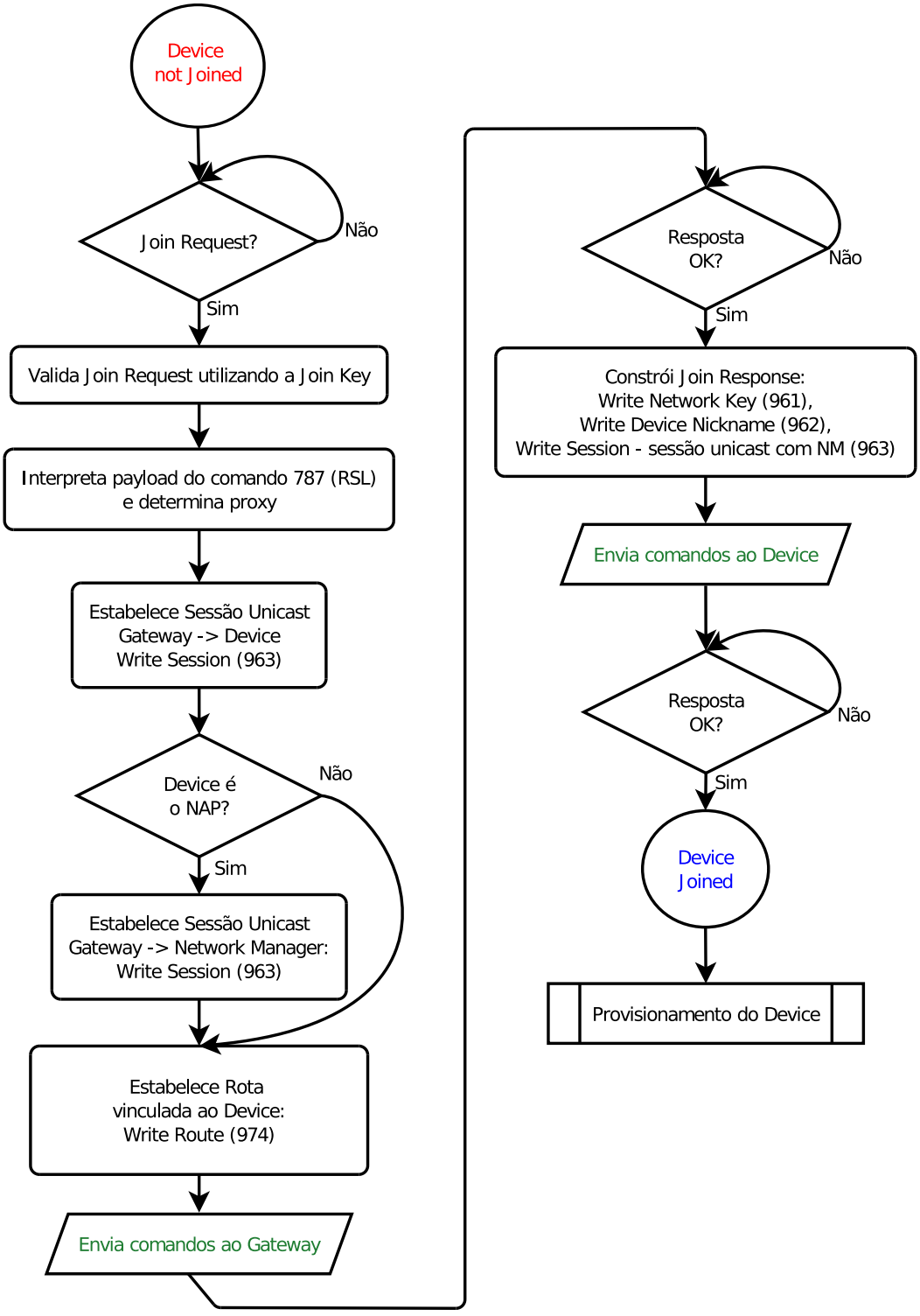

Figura 3.9: Fluxograma representando o *Join* de um dispositivo

4 RESULTADOS

Ao final da implementação descrita, temos um Gerente de Rede WirelessHART funcional, ainda que desprovido de um escalonador completo. O software atual é capaz de interagir totalmente com as outras entidades através de comandos previstos nas especificações do padrão, além de ser a peça principal na organização inicial e manutenção da rede. Além disso, é capaz de fazer o Join de novos dispositivos de campo de forma dinâmica, sem necessidade de configuração prévia.

Os testes de funcionamento foram limitados a um dispositivo de campo. No entanto, a implementação do Network Manager foi idealizada para suportar múltiplos dispositivos, o que pode ser testado no futuro com o auxílio de um escalonador. Após a inicialização, a comunicação do Network Manager com o dispositivo é mantida através do envio periódico de requisições de *Health Report* (comando 779). A comunicação do dispositivo com o NAP é mantida com a troca de mensages *keep alive*.

A rede manteve-se estável durante os testes realizados, exceto por falhas ocasionais na comunicação serial NAP-Host - NAP-RCP, fazendo com que pacotes fosse perdidos e a comunicação se encerrasse abruptamente. Tal falha também foi percebida algumas vezes durante a inicialização.

O arquivo de configuração mencionado na seção 3.5 pode ser totalmente suprimido em uma nova versão da implementação, pois o Join dinâmico dispensa a necessidade de declaração de dispositivos, além de os superframes e links poderem ser originados em um escalonador. A declaração dos dispositivos referentes ao Gateway, Network Manager e NAP podem ser movidos para um arquivo de cabeçalho (.h) sem perda alguma, restando apenas a declaração do Mapa de Canais.

As figuras 4.1, 4.2 e 4.3 apresentam a saída do software Network Manager executado através de linha de comando para um caso de teste. Na Figura 4.1 pode-se observar a inicialização e configuração do Gerente, criando os dispositivos básicos inicializando o Gateway (primeiro bloco de comandos), até o Join e o provisionamento do NAP. Na Figura 4.2 é possível observar o envio do comando 123 ao Network Manager referente a um novo dispositivo de campo, e os subsequentes Join e provisionamento com superframes e links. Por fim, pode-se observar na Figura 4.3 o envio dos comandos 779 (Health Report).

Também foi possível realizar a validação da comunicação entre o Network Manager e o dispositivo de campo com o auxílio do *sniffer* WirelessHART presente no LASCAR. Esse equipamento permite a visualização do conteúdo dos pacotes WirelessHART trocados entre dispositivos de campo (dentro de sua área de alcance) para uma determinada NetID, sendo necessário fornecermos a Join Key para a decriptação dos pacotes de Join. Um fragmento da saída do *sniffer* pode ser observado na Figura 4.4, enquanto uma captura detalhada pode ser vista no Anexo.

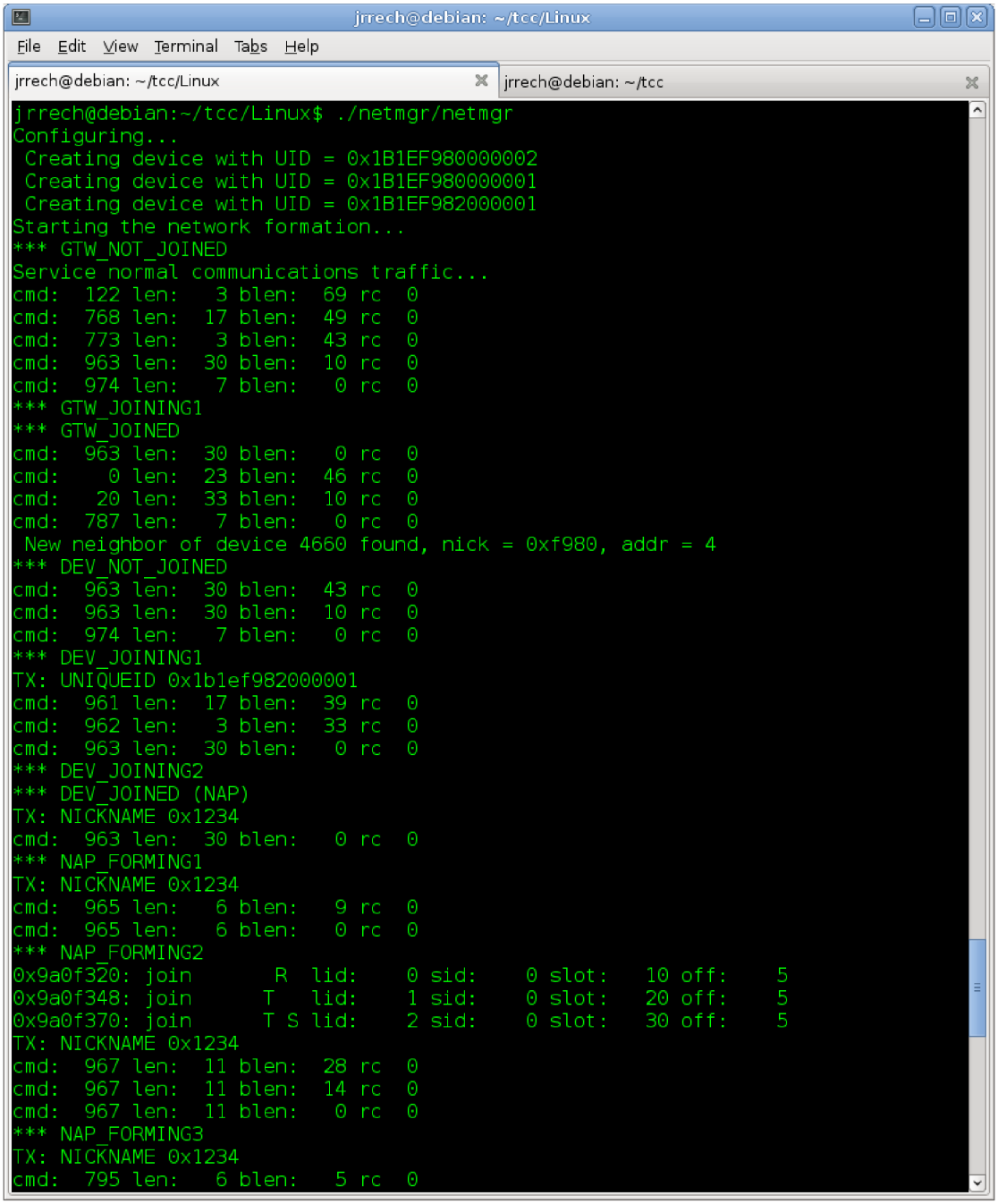

Figura 4.1: Execução do Network Manager: inicialização do gateway, join e provisionamento do NAP

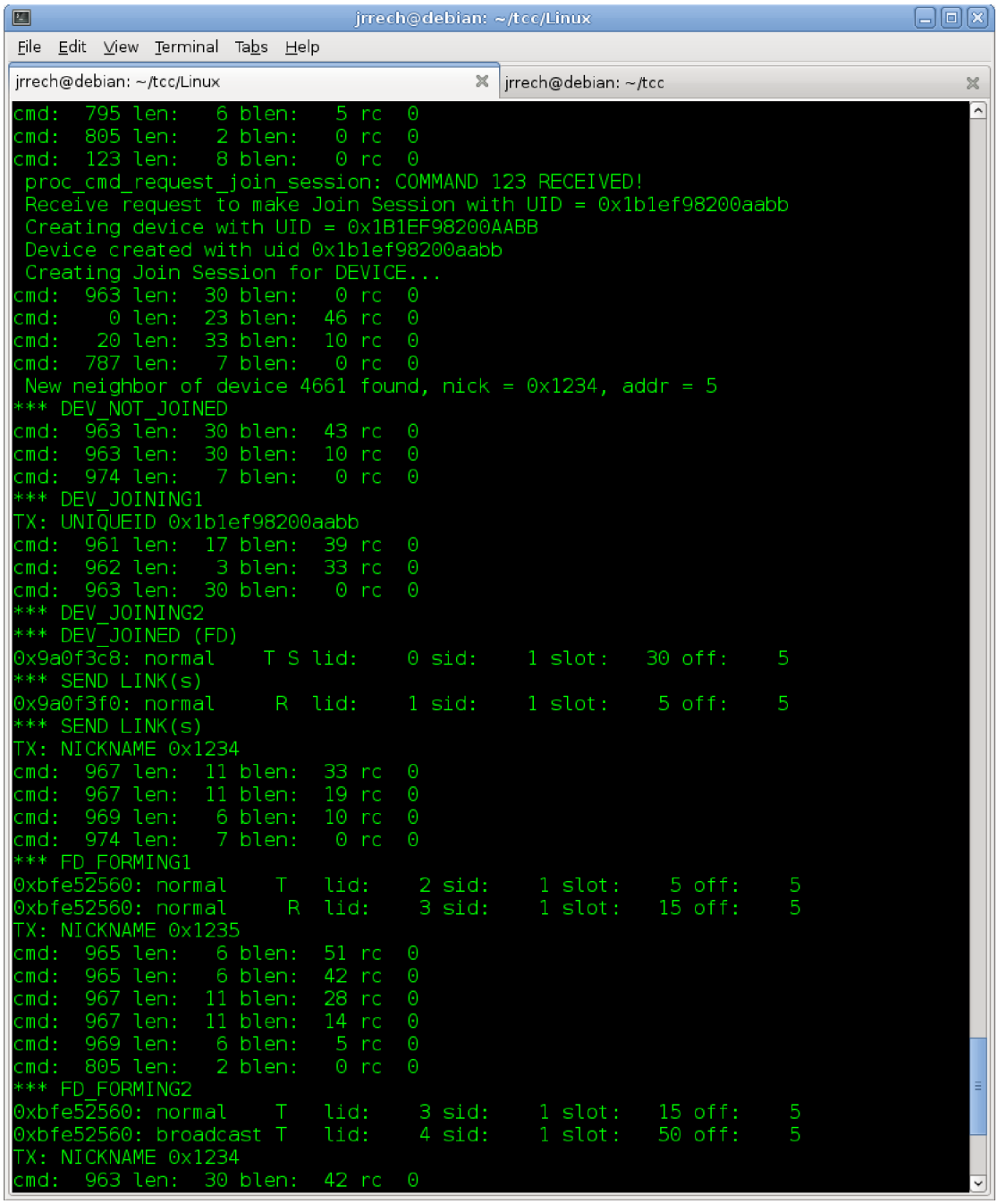

Figura 4.2: Join e provisionamento do dispositivo de campo

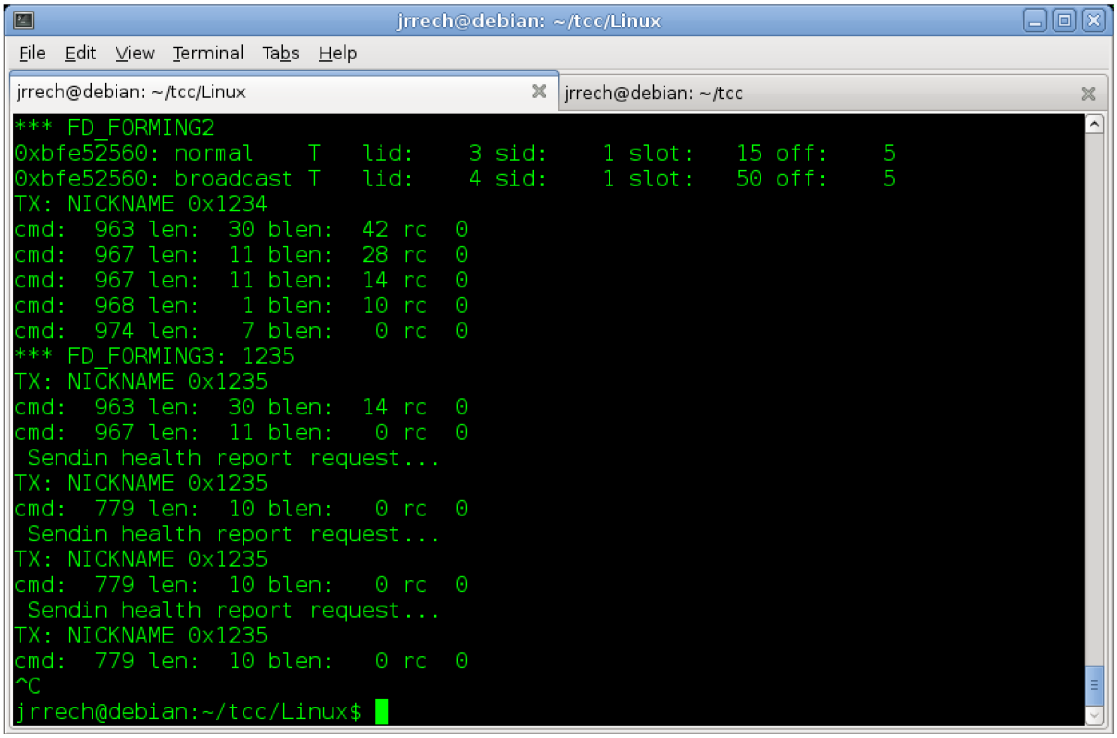

Figura 4.3: Envio de requisições de Health Report

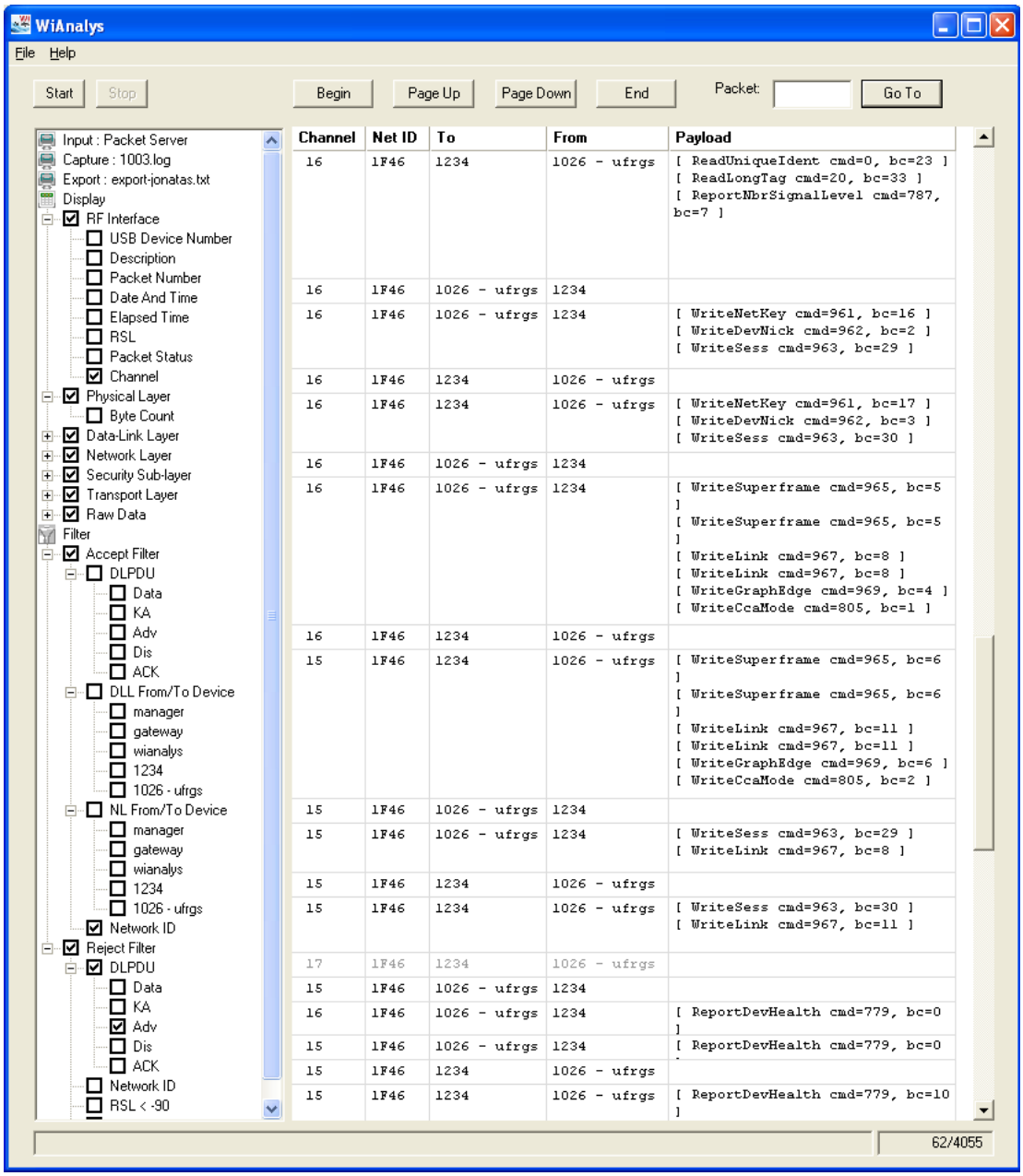

Figura 4.4: Saída do *sniffer*: captura referente ao processo de Join

5 CONCLUSÕES

Ao final deste trabalho, constatamos que o objetivo de implementação de um Gerente de Rede WirelessHART foi atingido. O software final é modular e totalmente expansível de acordo com as necessidades da aplicação.

Porém, a comunicação entre os módulos do NAP precisa ser melhorada. O padrão RS232, apesar de simples, não é confiável o suficiente para ser utilizado em um ponto onde todo o tráfego da rede é concentrado. As falhas (corrupção dos dados e consequente dessincronia dos dispositivos) ocorridas neste ponto da comunicação impediram testar mais a fundo a estabilidade do Network Manager.

O Join dinâmico, apesar de funcional, apresenta uma falha no que diz respeito a desempenho, pois é necessário o envio de dois Join Requests por parte do dispositivo para que o Join de fato aconteça. No estágio atual da implementação, somente o campo de endereço fonte do primeiro pacote de Join Request é analisado para a criação da sessão de Join, fazendo necessário o recebimento do Join Request seguinte para que o Join aconteça. Uma melhoria na implementação seria fazer com que somente um pacote de Join Request fosse necessário, otimizando o processo (descrito a partir da página 30).

A versão final do Network Manager possui 48 arquivos contendo 7522 linhas de código-fonte (fora comentários), o que representa um aumento de 57,1% em comparação à versão inicial (4788 linhas em 44 arquivos). Todo o código-fonte é compilado em um único arquivo binário, e sua execução é *single threaded*.

Para o início do desenvolvimento do Network Manager foi necessário um período de aproximadamente quatro meses dedicado ao estudo e à compreensão do funcionamento do protocolo WirelessHART, além do estudo dos códigos-fonte já existentes. Este período teve início já na primeira etapa do Trabalho de Diplomação, quando o objetivo do trabalho era o de adaptar ferramentas de código livre para a compilação do *stack*. O processo de desenvolvimento também foi de forte incentivo para um entendimento mais profundo do protocolo, o que pode ser aproveitado para trabalhos futuros envolvendo o padrão WirelessHART.

A realização deste trabalho foi de grande importância para o grupo de pesquisas em redes industriais sem fio do LASCAR, que até então só dispunha de implementações comerciais de Gerentes de Rede/Gateways WirelessHART. Uma implementação própria e aberta possibilita inúmeros estudos futuros, principalmente no que diz respeito a roteamento e a escalonamento, além de poder se tornar um produto competitivo no mercado brasileiro de redes industriais sem fio.

REFERÊNCIAS

CHEN, D.; NIXON, M.; MOK, A. **WirelessHART: Real-Time Mesh Network for Industrial Automation**. Springer, 2010. 282p.

HAHN, D. **Desenvolvimento de um ponto de acesso para redes WirelessHART**. 2011. 58p. Trabalho de Conclusão (Graduação em Engenharia Elétrica) — Departamento de Engenharia Elétrica, Universidade Federal do Rio Grande Sul, Porto Alegre.

WINTER, J. M. **Software de Análise de Roteamento de Dispositivos WirelessHART**. 2010. 117p. Trabalho de Conclusão (Graduação em Engenharia Elétrica) — Departamento de Engenharia Elétrica, Universidade Federal do Rio Grande Sul, Porto Alegre.

SÁNCHEZ, J. H. **WirelessHART Network Manager**. 2011. 77p. Dissertação (Mestrado em Engenharia Elétrica) — The Royal Institute of Technology, Suécia.

HART Communication Foundation. **Página da HART Communication Foundation**. Disponível em: <http://www.hartcomm.org/>. Acesso em: maio 2012.

HART Communication Foundation. **Network Management Specification HCF SPEC-085, Revision 1.2**. 2009. 98p.

HART Communication Foundation. **Command Summary Specification HCF SPEC-099, Revision 9.0**. 2007. 58p.

HART Communication Foundation. **Wireless Command Specification HCF SPEC-155, Revision 1.1**. 2008. 140p.

ANEXO A <FIGURAS COMPLEMENTARES>

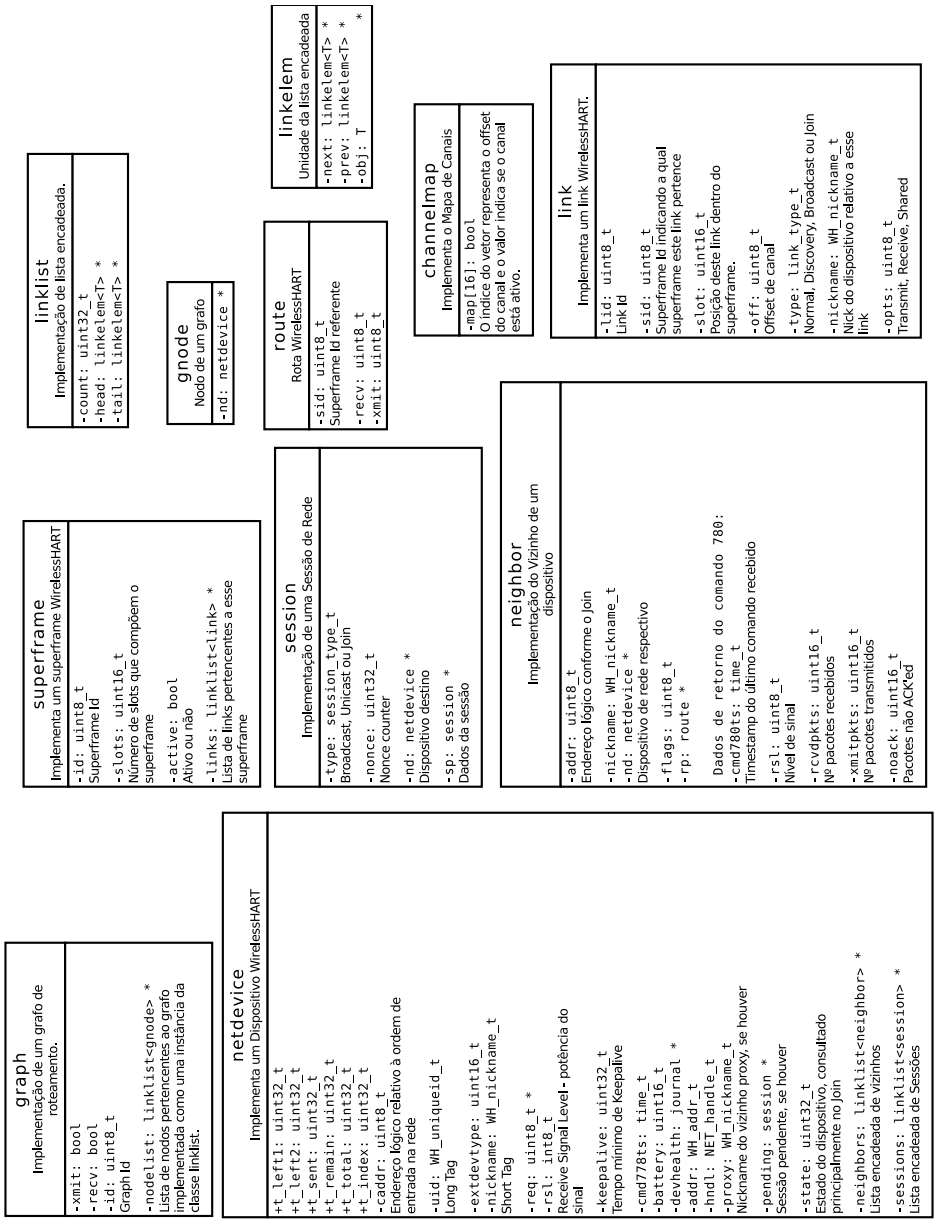

Figura 5.1: Diagrama com as principais classes do Network Manager

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	**Description**	**RSL**	**Channel**	**Byte Count**	**PDU**	**Priority**	**ASN (0))**	**Net ID**	**ASN Snippet**	**Graph ID**	**Dest**	**Src**	**Proxy**	**Payload**	**SL Ctl**
77	802.15.4-DATA	-35	16	122	Data	Cmd	2A	1F46	1BD2	100	F980	001B1EF98200AABB		[ReadUniqueIdent cmd=0, bc=23] [ReadLongTag cmd=20, bc=33] [ReportNbrSignalLevel cmd=787, bc=7]	Join
78	802.15.4-DATA	-53	16	25	ACK	Cmd	2A	1F46							
109	802.15.4-DATA	-35	16	122	Data	Cmd	EE	1F46	2597	100	F980	001B1EF98200AABB		[ReadUniqueIdent cmd=0, bc=23]*[ReadLongTag cmd=20, bc=33]*[ReportNbrSignalLevel cmd=787, bc=7]	Join
110	802.15.4-DATA	-55	16	25	ACK	Cmd	EE	1F46							
113	802.15.4-DATA	-55	16	108	Data	Cmd	5C	1F46	25FA	300	001B1EF98200AABB	F980	1234	[WriteNetKey cmd=961, bc=16]*[WriteDevNick cmd=962, bc=2]*[WriteSess cmd=963, bc=29]	Join
114	802.15.4-DATA	-34	16	25	ACK	Cmd	5C	1F46							
117	802.15.4-DATA	-35	16	94	Data	Cmd	B6	1F46	2660	100	F980	1235		WriteDevNick cmd=962, bc=3]*[WriteSess cmd=963, bc=30]	Session
118	802.15.4-DATA	-52	16	19	ACK	Cmd	B6	1F46							
119	802.15.4-DATA	-53	16	84	Data	Cmd	2E	1F46	26D6	300	1235	F980		[WriteSuperframe cmd=965, bc=5]*[WriteSuperframe cmd=965, bc=5]*[WriteLink cmd=967, bc=8]*[WriteLink cmd=967, bc=8]*[WriteGraphEdge cmd=969, bc=4]*[WriteCcaMode cmd=805, bc=1]	Session
120	802.15.4-DATA	-35	16	19	ACK	Cmd	2E	1F46							
121	802.15.4-DATA	-31	15	95	Data	Cmd	79	1F46	2732	100	F980	1235		[WriteSuperframe cmd=965, bc=6]*[WriteSuperframe cmd=965, bc=6]*[WriteLink cmd=967, bc=11]*[WriteLink cmd=967, bc=11]*[WriteGraphEdge cmd=969, bc=6]*[WriteCcaMode cmd=805, bc=2]	Session
122	802.15.4-DATA	-51	15	19	ACK	Cmd	79	1F46							
123	802.15.4-DATA	-49	15	78	Data	Cmd	E7	1F46	2798	300	1235	F980		[WriteSess cmd=963, bc=29]*[WriteLink cmd=967, bc=8]	Session
124	802.15.4-DATA	-31	15	19	ACK	Cmd	E7	1F46							
125	802.15.4-DATA	-33	15	82	Data	Cmd	41	1F46	27EA	100	F980	1235		[WriteSess cmd=963, bc=30]*[WriteLink cmd=967, bc=11]	Session
126	802.15.4-DATA	-51	15	19	ACK	Cmd	41	1F46							
170	802.15.4-DATA	-53	16	38	Data	Cmd	2A	1F46	301C	300	1235	F980		[ReportDevHealth cmd=779, bc=0]	Session
171	802.15.4-DATA	-32	15	19	ACK	Cmd	7F	1F46							
179	802.15.4-DATA	-31	15	48	Data	Cmd	D9	1F46	3080	100	F980	1235		[ReportDevHealth cmd=779, bc=10]	Session
180	802.15.4-DATA	-51	15	19	ACK	Cmd	D9	1F46							
252	802.15.4-DATA	-52	16	38	Data	Cmd	C2	1F46	38B3	300	1235	F980		[ReportDevHealth cmd=779, bc=0]	Session

Figura 5.2: Saída do *sniffer*: captura referente ao processo de Join

ANEXO B <ARTIGO: TRABALHO DE GRADUAÇÃO 1>

Estudo e Implementação de Protocolos de Rede Sem Fio para Automação Industrial

Jônatas Romani Rech[1]

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil

jrrech@inf.ufrgs.br

Abstract. *The objective of this paper is to describe and to specify the study and implementation of Wireless Networks protocols, such as WirelessHART and ISA100.11a. We will also specify the adaptation process of a WirelessHART protocol stack to an open source Real Time Operating System, which will eventually replace the proprietary system in an existant implementation. The software development environment and the utilized open source tools are also described, following a brief introduction to WirelessHART and to the referred implementation.*

Resumo. *O objetivo deste artigo é descrever e especificar o estudo e a implementação de protocolos de Redes Sem Fio, como WirelessHART e ISA100.11a. Também será especificado o processo de adaptação da pilha de protocolos WirelessHART a um Sistema Operacional de Tempo Real* open source*, que substituirá o sistema proprietário em uma implementação existente. O ambiente de desenvolvimento e ferramentas de código aberto utilizadas também são descritas, após uma breve introdução ao WirelessHART e à implementação em questão.*

1. Introdução

A crescente expansão do mercado de dispositivos móveis e de tecnologias de comunicação sem fio tem se refletido de modo bastante expressivo no ramo industrial durante a última década. A tecnologia de Redes de Sensores Sem Fio tem sido cada vez mais utilizada em aplicações de monitoramento e de controle de processos industriais, visto o aumento do poder de processamento dos dispositivos de campo e a uma melhora na confiabilidade das redes de um modo geral.

O padrão WirelessHART tem se mostrado uma das melhores opções para redes industriais sem fio, pois agrega aspectos como alta confiabilidade e determinismo na comunicação a baixos custos computacionais e a baixo consumo energético. Desde que surgiu, em 2007, o WirelessHART tem ocupado nichos de mercado em que tecnologias como a ZigBee, dentre outras, deixam a desejar em matéria de confiabilidade e segurança. Além do WiHART, existem outros padrões mais abrangentes e que utilizam a mesma infra-estrutura, como o ISA100.11a.

O projeto de Monitoração e Controle de Atuadores Sem Fio, desenvolvido no Laboratório de Sistemas de Controle, Automação e Robótica (LASCAR) da UFRGS em

parceria com empresas locais, dá a este trabalho de pesquisa uma aplicação imediata, o que motiva ainda mais o seu desenvolvimento. Se bem-sucedida, os resultados desta pesquisa poderão fazer parte de soluções desenvolvidas pelo Laboratório na área de redes sem fio.

Este trabalho descreverá, após uma introdução ao padrão WirelessHART, o processo de adaptação de um RTOS (Sistema Operacional de Tempo Real), de uma toolchain e de bibliotecas de software de código livre à implementação atual do padrão, substituindo o sistema e ferramentras proprietárias existentes. Isso aumentará a flexibilidade da atual implementação, permitindo que a mesma seja utilizada posteriormente em outros projetos, como uma implementação parcial do padrão ISA100.11a.

2. O Padrão WirelessHART

O estabelecimento do padrão WirelessHART aconteceu de forma a complementar o já existente e amplamente utilizado HART, protocolo de comunicação cabeada, com funcionalidades wireless. A especificação do WiHART prevê a operação das camadas de enlace, rede, transporte e aplicação sobre nível físico do tipo IEEE 802.15.4[1], a mesma tecnologia utilizada nos padrões ZigBee[2] e ISA 100.11a[3].

2.1. Características Gerais

As redes WiHART são chamadas de redes wireless mesh networks, ou redes semfio em malha, por normalmente apresentarem topologias difusas, onde cada dispositivo deve ser capaz de comunicar-se com qualquer outro membro da rede através de outros dispositivos ou através de um gateway . Isso implica a necessidade de qualquer dispositvo de campo ser capaz de agir como um nodo roteador, ou seja, ser capaz de interpretar informações de endereçamento e encaminhar pacotes destinados a outros nodos. Além disso, uma rede WiHART deve ser suficientemente robusta de modo que a saída de um nodo da rede não afete a comunicação de outros dispositivos que o utilizem como nodo intermediário, ao mesmo tempo que a entrada de novos dispositivos na rede seja também realizada de forma dinâmica e transparente. A Figura 1 exemplifica uma rede mesh.

As entidades básicas de uma rede WirelessHART são:

- **Field Device**: entidade básica da rede, é um dispositivo móvel cuja função é variável de acordo com a aplicação. É normalmente conectado a dispositivos de instrumentação, como sensores ou atuadores, e deve ser capaz de executar os comandos de aplicação recebidos do Network Manager. Também deve poder atuar como um nodo roteador para outros dispositivos de campo;
- **Gateway**: também chamado de ponto de acesso, é o dispositivo que concentra todo o tráfego de dados entrando ou saindo do Network Manager. Sua função é a de garantir o acesso do Network Manager à rede sem fio, alcançando o maior número de dispositivos de campo possível;
- **Network Manager**: responsável por todas as tarefas administrativas da rede, como estabelecimento de rotas, registro de novos nodos, envio de comandos,

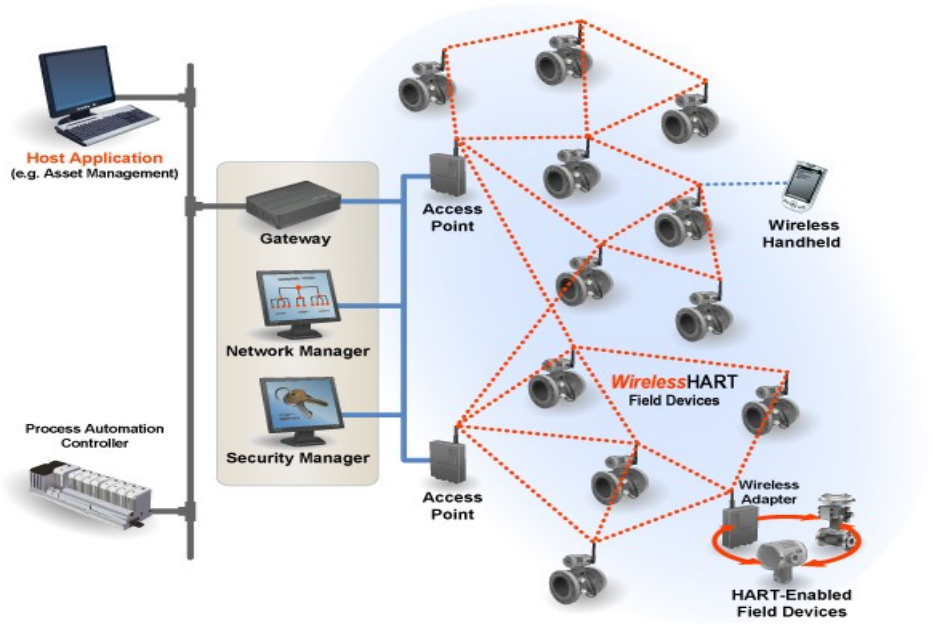

Figura 1. Exemplo de uma rede em malha

etc. Deve ser implementado em um dispositivo com maior capacidade de processamento, devido à carga computacional ao qual será submetido. O Network Manager deve ter acesso à rede principal da planta (geralmente cabeada), possibilitando o interfaceamento dos dispositivos de campo com um cliente de aplicações, por exemplo. Não raro, Gateway e Network Manager são implementados em um mesmo dispositivo.

Outro critério importante no contexto de redes sem fio é o consumo de energia (e consequente duração da bateria, se for o caso) do dispositivo. O uso de TDMA (Time Division Multiple Acess) para acesso ao meio é um fator chave, pois possibilita que os rádios só estejam de fato em funcionamento durante o seu slot de tempo reservado para a transmissão. Esse tipo de acesso ao meio também contribui para a diminuição de colisões no meio físico, pois cada dispositivo só transmitirá na sua fatia de tempo. Isso faz com que a necessidade de retransmissões diminua, o que também contribui para menores gastos de energia.

O padrão WiHART foi idealizdo para fins essencialmente industriais, diferentemente do padrão ZigBee, que foi originalmente idealizado prezando-se pela simplicidade e baixo custo. A pouca aceitação do ZigBee no meio industrial deveu-se provavelmente à sua falta de robustez e baixa confiabilidade, além de não ter um grande foco em segurança. É importante obsevar que o ZigBee precedeu o WiHART por cerca de quatro anos (ZigBee foi concebido em 2003 e WH em 2007), o que possibilitou o estudo e a observação dos pontos fracos do primeiro padrão. Desse modo, o WiHART foi capaz de cobrir muitos desses pontos, principalmente no que diz respeito a robustez, confiabilidade, segurança e flexibilidade. Tais aspectos são importantíssimos em um ambiente industrial, pois é pouco provável que possamos implementar sistemas de controle, acionamento ou monitoração sem fio sem um nível mínimo de cada um deles.

2.2. Nível Físico: IEEE 802.15.4

A norma IEEE 802.15.4 especifica tanto o nível físico quanto o nível de enlace para redes WPAN (Wireless Personal Area Networks) de baixa vazão e de baixa potência, ideal para aplicações residenciais e industriais. O framework básico prevê, num alcance de dez metros, uma taxa de transferência de 250 kbit/s. Taxas menores, de 20 kbit/s , 40 kbit/s e 100 kbit/s foram adicionadas nas últimas revisões do padrão, idealizadas para aplicações em que o consumo de energia seja ainda menor.

O nível físico do padrão WirelessHART opera em 2.4GHz, na faixa de frequências denominada ISM (Industrial, Scientific, Medical). Os canais suportados e suas respectivas frequências podem ser observados na Tabela 1.

Tabela 1. Canais do nível físico (IEEE 802.15.4)

Num. do Canal	Frequência (GHz)	Num. do Canal	Frequência (GHz)	Num. do Canal	Frequência (GHz)
11	2.405	16	2.430	21	2.455
12	2.410	17	2.435	22	2.460
13	2.415	18	2.440	23	2.465
14	2.420	19	2.445	24	2.470
15	2.425	20	2.450	25	2.475

2.3. Nível de Enlace

A camada de enlace, responsável pela formatação dos pacotes de dados e acesso à camada física, faz uso de TDMA e channel hopping para controlar o fluxo de dados e o acesso das camadas superiores ao meio físico. Time Division Multiple Access é uma técnica de Controle de Acesso ao Meio largamente utilizada quando necessitamos de uma comunicação determinística e livre de colisões, algo muito importante em um sistema de tempo real. Pelo conceito de TDMA, a comunicação entre dois dispositivos é segmentada em fatias de tempo, chamadas de time slots, enquanto diversos time slots compõem um superframe. Os superframes definem sequências de time slots ao longo do tempo, o que significa que a ordem das transferências de dados é conhecida previamente. Sempre há, no mínimo, um superframe ativo na rede, embora seja possível que vários superframes coexistam(por razões de aumento de flexibilidade na gerência da rede). Os dispositivos WiHART-compliant devem ser capazes de suportar múltiplos superframes.

É importante notar que, como a múltiplexação dos dados é baseada no tempo, a sincronização do relógio dos dispositivos é essencial para uma comunicação livre de erros. Este desafio fica a cargo da implementação.

De modo a aumentar a confiabilidade e a flexibilidade da rede, o padrão WirelessHART combina a técninca de channel hopping, ou salto de canais, com TDMA. Com channel hopping é possível variarmos a frequência da comunicação de um par de dispositivos. Assim, podemos otimizar cada time slot do TDMA fazendo com

que mais dispositivos comuniquem-se em uma mesma fatia de tempo, em diferentes canais. Um link entre dois dispositivos é definido pela tupla {Superframe, time slot, canal} e os dispositivos compatíveis com o padrão devem suportar múltiplos links.

Outro ponto interessante da camada de enlace é a função de channel blacklisting, que possibilita ao Network Manager manter uma "lista negra" de canais que apresentem maior interferência e consequente perda de pacotes, fazendo com que os links sejam realocados e tais canais sejam evitados.

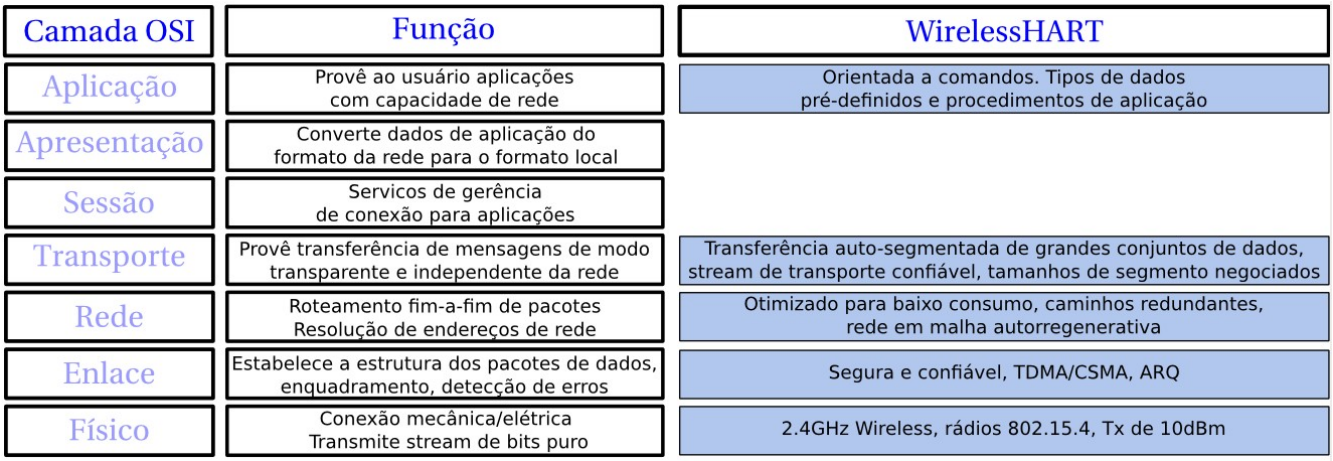

Figura 2. A rede WirelessHART vista em camadas OSI

2.4. Níveis de Rede e de Transporte

No padrão WirelessHART, as camadas OSI referentes ao nível de 3 (Rede) e 4 (Transporte) unem-se em uma camada única, com a função de garantir uma comunicação fim-a-fim segura e confiável aos níveis superiores.

As informações de endereçamento e de roteamento na rede são responsabilidade dessa camada. Os protocolos de roteamento em uma rede WirelessHART são dois:

- **Roteamento por Grafo**: é o método usual de roteamento adotado em uma rede WiHART. Os pacotes seguem rotas pré-definidas pelo Network Manager, que são carregadas em cada dispositivo antes do início da comunicação. Para um dispositivo endereçar um pacote, basta que adicione um identificador (Graph ID) no cabeçalho do nível de rede dizendo qual grafo aquele pacote deve seguir. Cada nodo intermediário deve ser capaz de interpretar tal identificador e encaminhar o pacote em direção ao destino conforme o grafo correto;
- **Roteamento por Fonte**: utilizado normalmente em operações de manutenção ou diagnóstico, este método funciona da seguinte forma: o pacote de dados é enviado pela fonte já possuindo no cabeçalho uma lista completa de nodos pelos quais o mesmo deve passar para atingir o destino. Assim, cada dispositivo intermediário simplesmente interpreta as informações e encaminha o pacote.

2.5. Nível de Aplicação

Camada mais superior da pilha, a camada de aplicação é o destino dos dados e informações que viajam na rede. É também a camada em que os comandos são

originados, requisitando dados de dispositivos remotos, informando leituras de sensores ou confirmando dados recebidos.

2.6. Compatibilidade com o ISA100.11a

A especificação ISA100.11a foi criada em 2009 pela International Society of Automation (ISA) como uma forma de padronizar os protocolos de comunicação sem fio utilizados em indústrias. O objetivo do ISA100.11a é o de garantir uma comunicação segura e altamente confiável para processos industriais, e foi idealizado para ser compatível com diversos padrões wireless existentes, incluindo WirelessHART e PROFIBUS. A especificação do padrão também prevê conectividade com a Internet por meio de uma tecnologia chamada 6LowPAN (IPv6 em redes de baixa potência), expandido a conectividade dos nodos até muito além dos níveis atuais.

Tabela 2. Comparação entre WirelessHART e ISA100.11a

Item	WirelessHART (HART 7.2)	ISA100.11a
Organization	HART Communication Foundation	ISA SP100 committee
Features	Wireless extension of HART specification	Industrial wireless system specification ISA100 family standard Standards for relational technologies are under consideration such as Factory automation, Backbone network, power source, RFID, Trustworthy wireless, etc.
PHY, MAC Layer	IEEE 802.15.4, 2.4GHz-band DSSS (Direct-Sequence Spread Spectrum) Channel hopping, TDMA, Channel Blacklisting	IEEE 802.15.4, 2.4GHz-band DSSS (Direct-Sequence Spread Spectrum) Channel hopping, TDMA, CSMA, Hybrid Channel Blacklisting
Network Layer	Extended HART address Mesh network	IPv6 addressing (6LowPAN) Mesh network
Upper Layer	HART protocol - Command & response - Burst mode	ISA100.11a Native protocol - Publishing / Subscribe, Client / Server, Bulk, Alert (event notification) Object Mapping、Tunneling protocol (Available existing protocol: HART, FOUNDATION Fieldbus, Profibus, Modbus, etc.)
Security	Encryption: AES128bit public symmetric key Join Key, Network ID, End to end security	Encryption: AES128bit public symmetric key Join Key, Network ID, End to end security Public key cryptosystem (option)

Um dos objetivos iniciais deste trabalho foi o de verificar a viabilidade de uma possível implementação parcial do stack ISA100.11a por meio da adaptação do stack WirelessHART disponível. Este tema foi levado em conta visto o fato de ambos os padrões fazerem uso da mesma tecnologia no nível físico (IEEE 802.15.4), além de o acesso ao meio no padrão ISA também poder ser realizado por TDMA (CSMA é outra alternativa prevista no ISA100.11a). Uma comparação mais completa dos dois padrões pode ser vista na Tabela 2[Hayashi et al., 2009]. Isso faria com que, havendo compatibilidade nos níveis físico e de enlace, bastassem implementações parciais das camadas de rede/transporte e de aplicação para confirmarmos a viabilidade de uma implementação completa. A motivação inicial, a de o ISA100.11a ser um padrão bastante novo e ainda bastante desconhecido, também foi o que nos fez mudar o foco do trabalho. Dispositivos ISA100.11a-compatíveis ainda são escassos e significativamente caros (um kit de desenvolvimento gira em torno de cinco mil dólares), o que

inviabilizaria a posterior validação da implementação.

3. A Implementação do Stack WirelessHART

O padrão especificado pela HART Communication Foundation nada mais é do que um conjunto de regras que definem como um sistema WiHART-compatível deve se comportar. Embora os documentos de especificação sejam minuciosos quanto a detalhes de funcionamento, não há sugestões ou restrições para uma implementação real do padrão. Assim, nos deparamos com uma vasta gama de opções no que diz respeito a componentes de hardware e de software, ficando a critério do projetista a combinação ideal. Em um contexto industrial, onde o sucesso de uma tecnologia se traduz em uma relação de custo/produtividade, é natural que sempre se deseje fazer uso do mínimo de recursos possível para atingir o objetivo da aplicação, utilizando-os da maneira mais otimizada.

3.1. Solução Atual

No projeto desenvolvido no Laboratório de Sistemas de Controle, Automação e Robótica (LASCAR), o modelo utilizado como base para testes e experimentos com o padrão tem sido um stack preliminar que utiliza ferramentas específicas de desenvolvimento para microcontroladores, além de uma biblioteca de funções para a implantação de um RTOS proprietário. As ferramentas necessárias para a compilação e geração de código executável também são propritetárias (IAR) e limitadas à plataforma Windows.

A parte de hardware da solução utiliza uma plataforma IEEE 802.15.4 Freescale MC13224, cujo cérebro é um microcontrolador ARM7 TDMI-S de 32 bits capaz de operar em frequências de até 26MHz. Além da memória interna de 96KB da MCU (MicroController Unit), a plataforma conta com módulos de memória flash de 128KB, SRAM de 96KB e ROM de 80KB.

4. Proposta do Trabalho e Motivação

O foco deste trabalho será a parte de software do sistema final, mais especificamente o que diz respeito ao Sistema Operacional, ambiente de desenvolvimento e bibliotecas de acesso à ROM do chip MC13224 sobre os quais a implementação do padrão WirelessHART será executada em um dispositivo de campo. Como visto anteriormente, a solução utilizada até agora conta com o um sistema operacional do tipo proprietário, ou seja, é preciso pagar para se ter acesso ao código-fonte e eventualmente realizar modificações. Além disso, as licenças do sistema limitam o número de pessoas com acesso ao código e são válidas apenas para o desenvolvimento de um único produto.

Porém, o software de código livre (open source) vem ganhando mais e mais espaço, principalmente quando falamos em software embarcado. A principal vantagem do software open source é, sem dúvida, a flexibilidade. O fato de podermos extender, corrigir, e, principalmente, adaptar o software livremente às nossas necessidades é a principal motivação deste trabalho de pesquisa.

4.1. Objetivos

O principal objetivo deste Trabalho de Graduação consiste primeiramente em substituir o sistema operacional da atual solução WirelessHART por um RTOS de código aberto, escolhido observando-se os seguintes critérios:

- **Desempenho;**
- **Consumo de energia;**
- **Consumo de memória;**
- **Suporte ao hardware atual;**
- **Ferramentas e ambiente de desenvolvimento;**
- **Qualidade da documentação.**

É esperado que, ao fim do projeto, tenhamos um sistema mais robusto, eficiente e que mantenha-se totalmente customizável e modular, afim de ser aproveitado para uma possível implementação do stack ISA100.11a em projetos futuros.

4.2. Levantamento de RTOSs open source

Existem diversas opções de RTOSs de código aberto disponíveis. Cinco deles foram escolhidos para os primeiros testes com o stack WirelessHART, por serem os principais representantes dos RTOSs open source existentes (e os mais bem documentados):

- **FreeRTOS**: portado para diversas arquiteturas, a principal caracterísitca do FreeRTOS é a simplicidade. Suporta funções e estruturas de dados básicas; bem documentado e testado;
- **RTEMS**: originalmente voltado para aplicações militares, o desenvolvimento do RTEMS foi iniciado no fim da década de 1980. Largamente testado, suporta diversas APIs;
- **eCos**: RTOS mais pesado que os anteriores, mas de configuração mais fácil e intuitiva. Suporte a diversas APIs incluindo POSIX;
- **RTLinux**: RTOS que é capaz de rodar um kernel Linux completo como uma thread do sistema “host”. Operações de entrada e saída são gerenciadas por drivers Linux, enquanto timers e interrupções críticas são gerenciadas pela parte real-time;

Podemos dividir os sistemas operacionais acima em dois grupos: FreeRTOS e RTEMS são sistemas mais leves, enquanto eCos e RTLinux requerem mais memória e geram imagens significativamente maiores quando compilados. Assim que forem realizados os primeiros testes de compilação e integração será possível decidirmos se isso será um fator limitante na escolha do sistema operacional, pois só aí conheceremos o consumo de memória em cada situação. O artigo do Trabalho de Graduação 2 conterá resultados de testes com cada um destes sistemas.

É importante lembrarmos que os principais critérios na escolha do sistema operacional serão, além da ocupação de memória e desempenho, o compromisso com

tempo real e a estabilidade do sistema como um todo. Outro conceito que será observado é se o RTOS é autocontido (apenas carrega a aplicação e possui bootloader próprio) ou se a tarefa de inicialização cabe à aplicação, que faz uso de bibliotecas disponibilizadas pelo SO. No caso da solução atual, o código de inicialização é customizado e a aplicação deve ser capaz de fazer o boot do sistema.

4.3. Ferramentas de desenvolvimento

Como abordado anteriormente, as ferramentas de desenvolvimento atualmente utilizadas pelo grupo de pesquisa são parte da solução fechada adquirida pela Universidade. A IDE utilizada é o IAR Embedded Workbench configurado para compilação de código ARM7, rodando em ambiente Windows. Todo o código-fonte do stack WirelessHART disponível contém diretivas específicas para o compilador IAR, além de fazer uso de algumas bibliotecas de código proprietário vinculadas ao sistema operacional CMX-RTX. Tais bibliotecas contém funções de tratamento de interrupções próprias do sistema CMX, compiladas para uso em conjunto com o compilador IAR.

Já temos um protótipo de ambiente de desenvolvimento baseado em ferramentas open source. A toolchain para compilação, linkagem e depuração do código-fonte é formada pelas seguintes ferramentas GNU (configuradas para cross-compilação com target arm-eabi, ISA ARM7):

- **GCC-4.6.1**: compilador C:
- **Binutils-2.21.1**: linker, assembler e outas ferramentas para manipulação de código binário;
- **GDB-7.3**: ferramenta de depuração;
- **Newlib-1.14.0**: biblioteca de funções padrão C.

A IDE Eclipse (configurada para desenvolvimento C/C++) está sendo utilizada como uma interface gráfica para as ferramentas e já é operacional. Ainda é necessário integrarmos uma ferramenta de programação via JTAG ao ambiente, uma vez que é indispensável para iniciarmos os testes em hardware.

A fim de compilarmos corretamente um protótipo do sistema, ainda é necessária a adaptação e adequação de cada RTOS ao código do stack, além da adaptação de bibliotecas de funções de baixo nível como tratamento de interrupções e inicialização de memória, por exemplo. Além disso, é necessário integrarmos corretamente as ferramentas do ambiente a estas bibliotecas para que a ligação do código do SO às funções de nível mais baixo seja realizada de forma correta.

5. Proposta e cronograma para a segunda etapa

Podemos estabelecer as seguintes atividades para a segunda etapa do Trabalho de Graduação:

1. **Ajuste do ambiente de desenvolvimento**: resolver as pendências relativas à integração compilador-bibliotecas-RTOS se possível de um modo geral, aplicável aos quatro sistemas propostos;

2. **Testes preliminares das opções de RTOSs**: teste dos sistemas em conjunto com uma implementação parcial ou wrapper do stack;
3. **Escolha do Sistema Operacional**: o RTOS que melhor atender aos critérios avaliados será escolhido para as etapas posteriores;
4. **Integração do stack WirelessHART**: integração da solução escolhida à implementação do stack WirelessHART e bibliotecas de acesso ao hardware;
5. **Testes preliminares do sistema integrado;**
6. **Otimização do RTOS para o stack WiHART**: otimização em termos de ocupação de memória e quantidade de processamento necessitado pelo stack;
7. **Testes do sistema otimizado;**
8. **Escrita da monografia** discutindo o desenvolvimento do trabalho nas etapas anteriores, os resultados alcançados e trabalhos futuros;
9. **Apresentação do Trabalho de Graduação**

Tabela 3. Cronograma de atividades

	Jan.	Fev.	Mar.	Abr.	Mai.	Jun.	Jul.
1. Ajuste do ambiente de desenvolvimento							
2. Testes preliminares das opções de RTOSs							
3. Escolha do Sistema Operacional							
4. Integração ao stack WirelessHART							
5. Testes preliminares do sistema integrado							
6. Otimização do RTOS para o stack WiHART							
7. Testes do sistema otimizado							
8. Escrita da monografia							
9. Apresentação							

6. Referências

[1] “802.15.4-2011 - IEEE Standard for Local and metropolitan area networks--Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)”, 2011

[2] “ZigBee Specification”, 2008

[3] “ISA100.11a Specification”, 2009

IAR Systems, http://www.iar.com (2011)

Chen, D., Nixon, M. and Mok, A. (2010) “WirelessHART – Real-Time Mesh Network

for Industrial Automation", Edited by Springer, EUA.

Jianping, S., Song, H., Mok, A., Chen, D., Lucas, M., Nixon, M. and Pratt, W. (2009) "WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control", em IEEE Real-time Embedded Technology and Applications Symposium

Gustafsson, D. (2009) "WirelessHART – Implementation and Evaluation on Wireless Sensors", Masters' Degree Project, KTH, Suécia.