UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

PROJETO DE DIPLOMAÇÃO II

MÉTODO PARA DETECÇÃO DE FALHA DE ENLACE DE TRANSMISSÃO EM RSSFI

LEOMAR MATEUS RADKE

Porto Alegre, Julho de 2019

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

MÉTODO PARA DETECÇÃO DE FALHA DE ENLACE DE TRANSMISSÃO EM RSSFI

Leomar Mateus Radke

Projeto de Diplomação II entregue ao Departamento de Engenharia Elétrica da Universidade Federal do Rio Grande do Sul como parte dos requisitos para a Graduação em Engenharia Elétrica.

ORIENTADOR: Prof. Dr. Ivan Müller

Porto Alegre, Julho de 2019

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

Leomar Mateus Radke

MÉTODO PARA DETECÇÃO DE FALHA DE ENLACE DE TRANSMISSÃO EM RSSFI

Este projeto foi julgado adequado para fazer jus aos créditos da Disciplina de "Projeto de Diplomação" do Departamento de Engenharia Elétrica e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____
Prof. Dr. Ivan Müller
Doutor pela Universidade Federal do Rio Grande do Sul – Porto Alegre, Brasil

Banca Examinadora:

Prof. Ivan Müller, UFRGS
Doutor pela Universidade Federal do Rio Grande do Sul - Porto Alegre, Brasil.

Prof. João Cesar Netto, UFRGS
Doutor pela Université Catholique de Louvain, Louvain-la-Neuve - Bélgica.

Eng. Max Feldman, UFRGS
Engenheiro pela Universidade Federal do Rio Grande do Sul - Porto Alegre, Brasil.

Porto Alegre, Julho de 2019

AGRADECIMENTOS

Agradeço primeiramente aos meus pais, Leonel e Roseli pelo apoio incondicional nesses 5 anos de caminhada. A minha irmã Lilian, que apesar de distante, sempre me incentivou. Agradeço a minha namorada, Patricia, por todo o suporte em ocasiões adversas.

Agradeço também aos meus amigos da faculdade, Gabriel da Costa, Lucas Alves, Pedro Morgan e Vinícius Ceriotti, por todos os momentos de estudo e brincadeiras, que fizeram da minha graduação algo inesquecível.

Aos amigos do Laboratório de Sistemas de Controle, Automação e Robótica, Gustavo Cainelli, Christian Alan Krötz e Max Feldman que sempre estiveram disponíveis para me apoiar nesse trabalho.

Agradeço ao meu orientador, professor Dr. Ivan Muller pela compreensão e oportunidade de realizar esse trabalho.

Agradeço, por fim, todos aqueles que de alguma forma contribuíram para a chegada desse momento.

RESUMO

Sistemas de comunicação sem fio vem sendo cada vez mais empregados na indústria. Nesse contexto, os protocolos *Wireless*HART e ISA 100.11a são os mais utilizados atualmente, devido ao pioneirismo e elevada robustez. Este trabalho tem como objetivo geral realizar a detecção de falha de enlace de transmissão em uma rede sem fio industrial, utilizando *Wireless*HART, visando apresentar um diagnóstico de possíveis dispositivos de campo que possam conter falhas de hardware. Para realizar este trabalho, uma aplicação é anexada a um dos comandos da pilha do protocolo, que será executada toda vez que esse comando for solicitado pelo gerenciador. Com a integração de três camadas de software, o estudo de caso contempla o roteiro de validação do algoritmo descrito na aplicação. Os resultados obtidos revelam que o método é capaz de analisar e obter de forma a lista de dispositivos que apresentaram alguma falha, sendo a sua utilização adequada as mais diversas aplicações em comunicação de redes industriais sem fio.

Palavras-chave: *Wireless*HART, Amplificador de Potência, Enlace, Transmissão.

ABSTRACT

Wireless communication systems have been increasingly employed in the industry. In this context, the WirelessHART and ISA 100.11a protocols are the most used today due to the pioneering and high robustness. This work has the general objective of performing transmission link failure detection in an industrial wireless network using WirelessHART, in order to present a diagnosis of possible field devices that may contain hardware failures. To perform this work, an application is attached to one of the protocol stack commands, which will be executed every time this command is requested by the manager. With the integration of three layers of software, the case study contemplates the validation script of the algorithm described in the application. The results show that the method is capable of analyzing and obtaining in a way the list of devices that presented some failure, and its proper use is the most diverse applications in communication of industrial wireless networks.

Keywords: *Wireless*HART, Power Amplifier, Link, Transmission.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

ACK	*Acknowledge*
AP	*Access Point*
ASN	*Absolut Slot Number*
CMOS	*Complementary Metal Oxide Semiconductor*
CSV	*Comma-separated Values*
DDL	*Device Description Language*
DLL	*Dynamic Link Library*
DLPDU	*Data Link Protocol Data Unit*
FD	*Field Device*
HART	*Highway Addressable Remote Transducer*
IDE	*Integrated Development Environment*
IIR	*Infinite Impulse Response*
IP	*Internet Protocol*
ISO	*International Organization for Standardization*
LLC	*Logical Link Control*
LNA	*Low-Noise Amplifier*
MAC	*Medium Access Control*
NPDU	*Network Protocol Data Unit*

OSI	*Open System Interconnection*
PA	*Power Amplifier*
RF	Rádio Frequência
RSL	*Received Signal Level*
RSSFI	Redes de Sensores Sem Fio Industriais
SNR	*Signal-to-noise*
TCP	*Transmission Control Protocol*
TDMA	*Time Division Multiple Access*
UDP	*User Datagram Protocol*
WH	*Wireless*HART

1 INTRODUÇÃO

1.1 Motivação

Os avanços tecnológicos na área da eletrônica das comunicações sem fio trouxeram diversos benefícios para a indústria e para a sociedade em geral. Com esses progressos pode-se, por exemplo, controlar e automatizar processos industriais com maior confiabilidade e robustez, integrando processadores e circuitos cada vez mais complexos. Com um maior poder de processamento de dados, surge a demanda de haver um transporte dessas informações de forma confiável e em uma velocidade que corresponda às necessidades de determinada aplicação. Por isso, desde meados dos anos 80, existem esforços da indústria para o desenvolvimento de protocolos que atendam a requisitos de segurança e durabilidade, ao qual os dispositivos são expostos – ruídos eletromagnéticos, variações de temperatura e altos níveis de umidade.

Dentro desse processo de expansão e busca por novas soluções, o protocolo HART (*Highway Addressable Remote Transducer*) surgiu como um ótimo candidado para a comunicação cabeada (CASSIOLATO, 2011). Atualmente, ele é amplamente suportado por dispositivos de campo (*field devices*). Tendo em vista a grande importância das redes sem fios e a demanda da indústria pela diminuição de custos nos processos produtivos, a HART Foundation lançou, em 2007, o *Wireless*HART (WH).

Esse foi o primeiro padrão aberto de comunicação sem fio especificamente desenvolvido para ambientes industriais. Seu protocolo já rendeu diversos estudos e melhorias, resultando em evoluções e novos desafios. Conhecer e investigar todas as variáveis que influenciam no seu desempenho colaboram no aprimoramento desse padrão.

O protocolo WH produz enlaces com topologias do tipo *mesh*, nas quais todos os dispositivos são capazes de rotear mensagens (CHEN; NIXON; MOK; 2010). Como outros padrões *wireless*, esse protocolo sofre dos desafios de robustez sob falhas ou interferências. Não seria incomum em um cenário industrial que utiliza o WH como ferramenta de comunicação entre processos, que alguns resultados sejam

interpretados de forma equivocada, visto as diversas nuances que envolvem a pilha WH.

Para que haja diagnósticos mais precisos, desenvolveu-se algumas ferramentas que simulam falhas na rede, a fim de se obter o comportamento do protocolo em situações de bloqueios, interferências e até falhas de hardware. A interpretação dessas falhas é fundamental para um bom desempenho da rede, considerando-se que qualquer problema enfrentado em ambientes industriais, que venham a acarretar em possíveis perdas de produtividade, podem trazer prejuízos ou até causarem algum dano ainda maior.

1.2 Objetivo

Tendo em vista a motivação descrita, esse trabalho consiste em apresentar e elaborar um método para avaliar falhas de enlace de transmissões no protocolo WH, estabelecendo um algoritmo capaz de interpretar possíveis falhas na rede e apresentar seu diagnóstico. Especificamente, buscou-se implementar uma rotina que verifique os níveis de RSL das transmissões entre vizinhos de uma rede WH, afim de encontrar falhas no hardware dos dispositivos de campo.

1.3 Estruta do Trabalho

O capítulo 2 apresenta os fundamentos teóricos relevantes para a compreensão desse trabalho, trazendo um panorama geral do protocolo WH. O capítulo 3 expõe todos os materias utilizados no ambiente de testes e obtenção de resultados, bem como a descrição do estudo de caso e roteiro de testes. Já o capítulo 4, contém todos os resultados obtidos, bem como um validação de um intervalo de confiança analisado. O trabalho é finalizado com o capítulo 5, onde estão as conclusões e sugestões e melhorias.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta todos os conceitos estudados e utilizados para a realização da metodologia do estudo de caso, estruturação da proposta, bem como a obtenção e análise dos resultados obtidos.

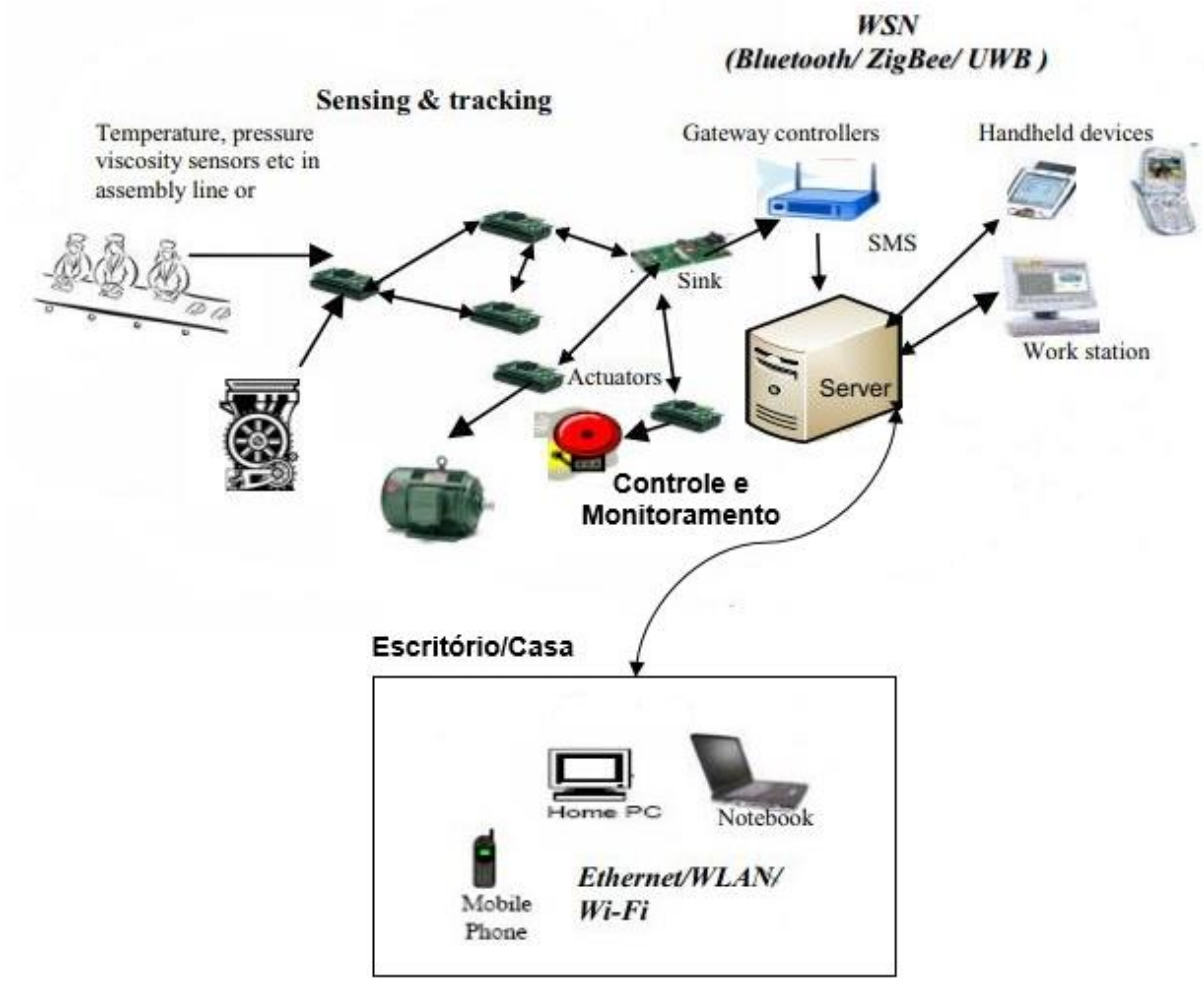
2.1 Redes de Sensores Sem Fio Industriais

Redes de sensores sem fio industriais (RSSFI) podem ser usadas vantajosamente para o monitoramento e controle por meio da coleta de dados. No monitoramento de rede, sensores são usados para identificar e classificar eventos incomuns, aleatórios e efêmeros, como notificações de alarme e detecção de falhas devido a alterações importantes na máquina, processo, segurança da planta, ações do operador ou instrumentos que são usados de forma intermitente. Por outro lado, a coleta periódica de dados é necessária para operações como o rastreamento dos fluxos de materiais e monitoramento da integridade do equipamento/processo. Tais aplicações de monitoramento e controle reduzem o custo de mão-de-obra, erros humanos e evitam o tempo ocioso de manufatura (LOW; WIN; ER; 2005).

A utilização das RSSFI requer atenção em função dos seus requisitos e desafios: oferecem alternativas atraentes com relação as redes cabeadas, mas também, sofrem dos problemas relacionados à propagação de sinais de RF. Ajudam ainda a melhorar a qualidade do produto, agilizando operações, aceleraram a produção, facilitam a instalação, aumentam a flexibilidade e mobilidade nas fábricas. Com isso, podem reduzir os gastos com infraestrutura e danos causados nos cabos em chão da fábrica, podendo ainda, serem utilizadas em máquinas que se movimentam. (MULLER, 2012).

Como mostrado na Figura 1, os sistemas de controle e manutenção de processo em tempo real são equipados com redes de sensores/atuadores sem fio nas linhas de produção e podem ser integrados ao software corporativo de *back-end*, bem como à Internet.

Figura 1 - Redes de sensores/atuadores industriais.



Fonte: Adaptado de (LOW; WIN; ER, 2005).

Com a internacionalização e com o rápido desenvolvimento das RSSFI, diversos protocolos de comunicação para esse tipo de rede foram desenvolvidos, tais como o WH, o WIA-PA, o ISA100.11a e o ZigBee (WANG; JIANG, 2016). Como todos esses protocolos são baseados no IEEE 802.15.4, existem muitas semelhanças entre eles. No entanto, existem também diferenças significativas, que promovem vantagens e desvantagens de cada protocolo. Para o desenvolvimento desta proposta, optou-se pela utilização do protocolo WH para a sua validação, uma vez que este está disponível em laboratório, assim como a pilha do protocolo. Alia-se isso ao fato de que atualmente esse é o protocolo mais empregado na indústria.

Do sensoriamento ambiental ao monitoramento de condições e automação de processos, as RSSFI atendem a uma ampla gama de aplicações. Embora o *ZigBee* e o MiWi geralmente atendam aplicativos de automação residencial, o WH e o

ISA100.11a são projetados especificamente para ambientes industriais. As arquiteturas industriais tradicionais com fio experimentam um nível maior de determinismo e um nível de escalabilidade industrial. Ainda assim, os RSSFI superam qualquer rede com fio em modularidade, facilidade de uso e custo-benefício.

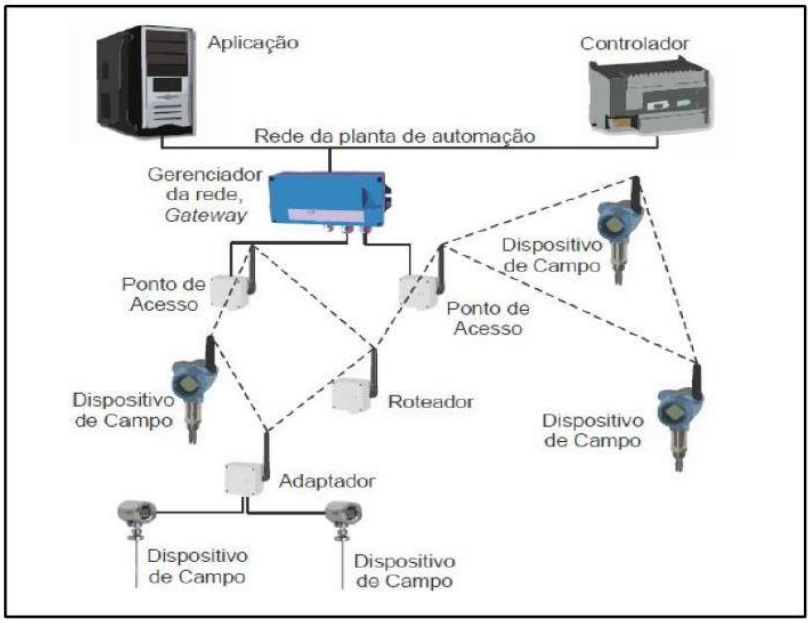
2.2 O Protocolo *Wireless*HART

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

As redes WH são chamadas de redes *wireless mesh networks*, ou redes sem fio em malha, nas quais cada dispositivo deve ser capaz de se comunicar com qualquer outro membro da rede por meio de outros dispositivos ou por um *gateway*. Com isso, faz-se necessário que cada dispositivo de campo seja capaz de agir como um nodo roteador, ou seja, ser apto a interpretar informações de endereçamento e encaminhar pacotes destinados a outros nodos.

Além disso, uma rede WH 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 2 apresenta a topologia de uma rede *mesh*.

Figura 2 - Dispositivos de uma rede WH



Fonte: Adaptado de (KUNZEL, 2012).

O protocolo WH foi desenvolvido com o objetivo de estabelecer um padrão de comunicação sem fio para uso em aplicações industriais (HART COMMUNICATION FOUNDATION, 2008). Por ser um protocolo seguro, sincronizado em tempo e de baixo consumo, é adequado ao controle de processos industriais. A compatibilidade é definida basicamente pela estrutura de comandos DDL (*device description language*), anteriormente desenvolvida pela organização HART (SMAR, 2018).

2.3 Modelo OSI

O Modelo OSI é um modelo de rede de computador referência, dividido em camadas de funções, criado em 1971 e formalizado em 1983, com objetivo de ser um padrão, para protocolos de comunicação entre os mais diversos sistemas em uma rede local (*Ethernet*), garantindo a comunicação entre dois sistemas computacionais (*end-to-end*) (OTERO CYSNE, 2016).

Esse modelo divide as redes de computadores em 7 camadas, de forma a se obter camadas de abstração. Cada protocolo implementa uma funcionalidade assinalada a uma determinada camada. O Modelo OSI não é uma arquitetura de redes, pois não especifica os serviços e protocolos exatos que devem ser usados em cada camada. Ele apenas informa o que cada camada deve fazer. Ainda, permite

comunicação entre máquinas heterogêneas e define diretivas genéricas para a construção de redes de computadores (seja de curta, média ou longa distância) independente da tecnologia utilizada (PINHEIRO, 2004). A Figura 3 ilustra de forma simplificada o que cada camada representa dentro deste modelo.

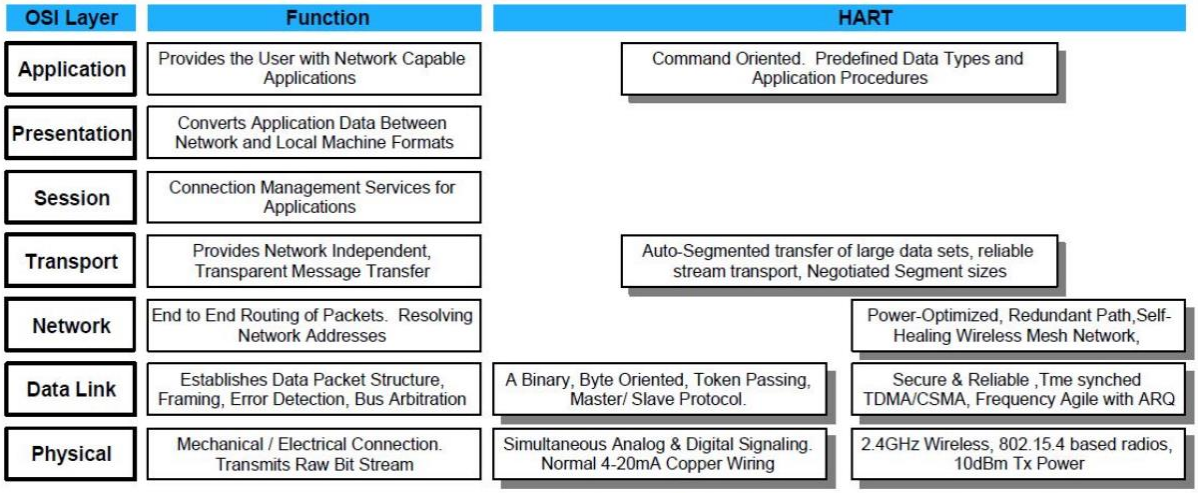
Figura 3 - Camadas do Modelo OSI

Fonte: (PINHEIRO, 2004).

O WH, visando compatibilidade com sua versão cabeada, utiliza a mesma camada de aplicação do HART. Apresenta-se na Figura 4 a definição de cada camada OSI com sua respectiva aplicação nos protocolos em questão.

Tanto no HART quanto no WH, as camadas de rede, transporte e sessão são tratadas como uma única camada – responsável pelas funções de roteamento de rede e entrega de dados; enquanto o mesmo pode ser dito das camadas de apresentação e aplicação – uma única camada responsável pelo gerenciamento dos comandos utilizados (HCF, 2007).

Figura 4 - Modelo OSI e camadas WH.



Fonte: (HART COMMUNICATION FOUNDATION, 2008).

2.4 Camada Física

A camada física tem como objetivo fornecer serviços para a camada de dados e controlar a interação entre dispositivo e meio de transmissão. Para isso, utilizam-se parâmetros bem definidos nos documentos da norma. Para o tráfego das informações é utilizada a faixa de frequência 2,4 GHz (SPEC-065, 2007). Todas as mensagens WH são do tipo IEEE 802.15.4. Algumas características da camada física podem ser citadas e descritas:

- **Canais**: São 15 canais (11-25) onde o canal 26 não é utilizado por não ser autorizado em algumas regiões do planeta.
- **Tempo entre *time slots***: na rede WH, esse tempo é de 10 ms.
- **Tempos entre mensagens**: o menor tempo entre o fim do envio do pacote e o inicio confirmação do recebimento é de 800 us e o tempo máximo é 1ms.
- **Interferências**: utiliza técnica de salto de canais para evitar interferências.

A camada física define o relacionamento de nível elétrico e físico entre um nó e um meio físico. Ele define características como antena, meio de ar, nível de potência de transmissão, etc. A camada física WH é um subconjunto simplificado do definido no padrão IEEE 802.15.4 (faixa de 2,4 GHz apenas) (CHEN; NIXON; MOK, 2010).

2.5 Camada de Enlace

Uma característica distinta do padrão WH é o tempo de sincronização da camada de enlace. Empregando a técnica de acesso ao meio *Time Division Multiple Access* (TDMA) é utilizada para fornecer comunicações sem colisão e determinísticas. O conceito de superframe é apresentado como uma sequência de intervalos de tempo *(time slots)* consecutivos.

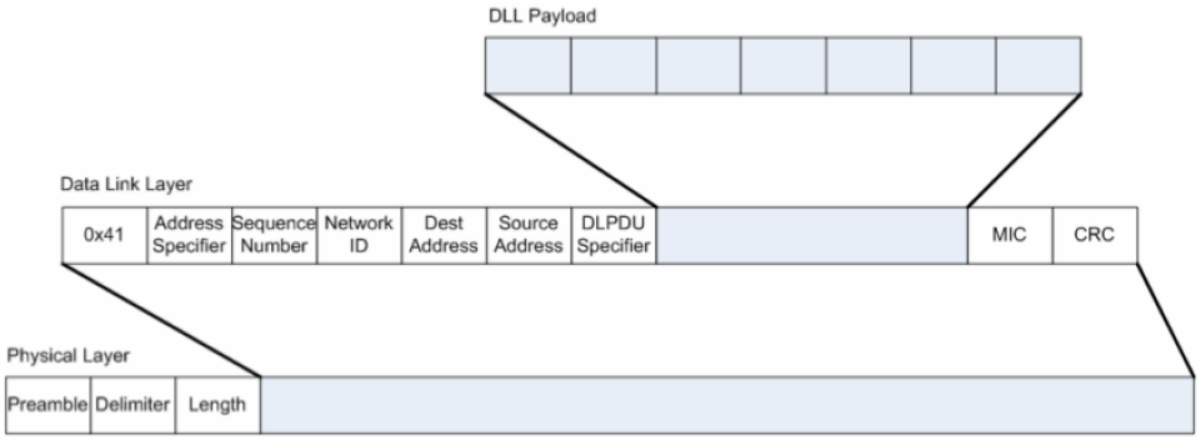
Um superframe é periódico, com o comprimento total do conjunto de slots como o período. Todos os superframes em uma rede WH começam com o *Absolut Slot Number* (ASN) 0, no momento em que a rede é criada pela primeira vez. Cada superframe então se repete ao longo do tempo com base no seu período.

No protocolo WH, uma transação em um intervalo de tempo é descrita por um vetor: {*frame id, index, type, src addr, dst addr, channel offset}* em que o frame ID identifica um superframe específico, *index* é o índice do slot no *superframe*; *type* indica o tipo do *slot* (transmissão/recepção/inativo); *src addr* e *src dst* são os endereços do dispositivo de origem e do dispositivo de destino, respectivamente; *channel offset* fornece um parâmetro da equação do canal ativo (WINTER, 2010).

O número do canal atual é usado como um índice para a tabela de canais ativos a fim de se obter o número do canal físico. Sendo que o ASN está aumentando constantemente, o mesmo deslocamento de canal pode ser mapeado para diferentes canais físicos em slots diferentes. Dessa forma, obtém-se a diversidade de canais e se aumenta a confiabilidade da comunicação.

A estrutura do pacote de enlace, o *Data Link Protocol Data Unit* (DLPDU) pode ser vista na Figura 5. 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.

Figura 5 - Estrutura DLPDU do WH.



Fonte: (SPEC-065, 2007).

A camada de enlace de dados fornece os meios confiáveis para transferir dados entre os nós da rede, detectando e, possivelmente, corrigindo erros que possam ocorrer na camada física. Essa camada tem a importante tarefa de criar e gerenciar quadros de dados. Geralmente, há duas subcamadas, camada *Logical Link Control* (LLC) e camada *Medium Access Control* (MAC). A camada LLC define o serviço para a camada de rede e o MAC define como o meio de comunicação é acessado por vários nós (CHEN; NIXON; MOK, 2010).

2.6 Camada de Aplicação

É na camada de aplicação que os comandos HART são implementados, definindo, portanto, o comportamento dos comandos e os tipos de dados que devem ser carregados nas mensagens. Cada comando especifica de forma única e não ambígua o pacote de dados e seu tamanho.

Os comandos são divididos nas classes:

- ***Universal Commands***: devem ser suportados por todos dispositivos HART. (SPEC-127, 2008);
- ***Common Practice Commands***: implementados por um grande número de dispositivos e devem ser suportados sempre que possível. (SPEC-151, 2008);

- ***Non-public Commands***: comandos especiais que têm seu uso limitado para ambiente de desenvolvimento dos dispositivos;
- ***Wireless Commands***: são a base da comunicação das redes WH sendo mandatório para produtos WH. (SPEC-155, 2008);
- ***Device Family Commands***: grupo de comandos usados para a configuração de dispositivos de campo, sem a necessidade de implementações individualizadas;
- ***Device-Specific Commands***: Comandos definidos pelos fabricantes segundo as necessidades e propósitos de seus dispositivos.

Os comandos da aplicação utilizados para o desenvolvimento desse trabalho serão listados a seguir.

2.6.1 Comando 771 - *Force Join Mode*

Esse comando permite que um sistema *host* ou dispositivo portátil force um dispositivo de campo para que entre no modo de *join*. O dispositivo deve permanecer no modo de pesquisa ativa por, pelo menos, o tempo de *join*, que é definido como um dos parâmetros de *input* do comando. (SPEC-155, 2008). O outro dado necessário é qual o modo a ser aplicado, três são as possibilidades:

- Impossibilitar o *join*;
- Realizar *join*;
- Tentar *join* imediatamente ao ligar ou reset.

2.6.2 Comando 780 - *Report Neighbor Helth List*

O comando 780 é um comando de aplicação sem fio. Este comando fornece dados estatísticos dos vizinhos que apresentam link com o dispositivo (SPEC-155, 2008). As variáveis de maior interesse no comando 780 são:

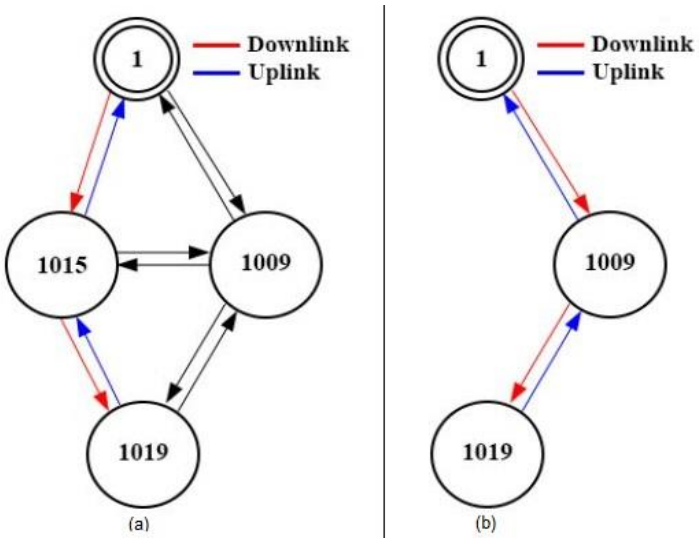
- Número total de vizinhos: informa número de dispositivos que possuem link com o dispositivo interrogado;
- Apelido dos vizinhos;

- *Received Signal Level* da origem para o destino: informa o nível de sinal recebido em dBm;
- Número de pacotes transmitidos para vizinhos;
- Pacotes recebidos do vizinho.

2.6.3 Comando 960 – *Disconnect Device*

Esse comando permite que o gerenciador de rede force um dispositivo a sair da rede, limpe todas as informações e reingresse nela (SPEC-155, 2008). A desconexão do dispositivo da rede pode emular uma falha de hardware do dispositivo, fazendo com que o dispositivo tenha que se desconectar da rede. A figura 6 ilustra a utilização do comando, removendo um dispositivo da rede por completo.

Figura 6 - (a) Antes do comando 960. (b) Após o comando 960 ser enviado para o dispositivo de campo.

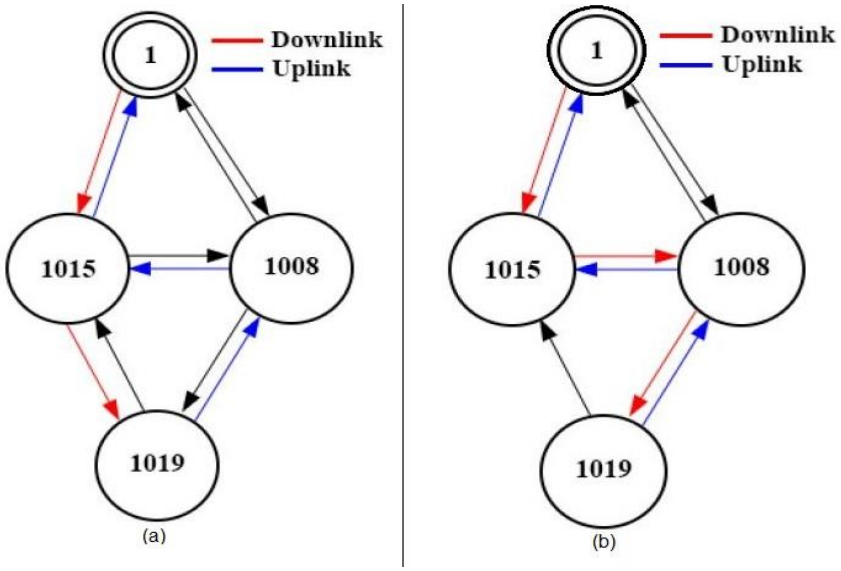


Fonte: Adaptado de (KRÖTZ, 2019).

2.6.4 Comando 968 - *Delete Link*

Esse comando permite que o gerenciador de rede exclua uma atribuição de link em um dispositivo de rede. Tal comando é utilizado para simular bloqueios de enlace que, em grandes plantas industriais, podem ocorrer quando um grande obstáculo (veículo, maquinário) interfere no enlace dos dispositivos (SPEC-155, 2008). A figura 7, ilustra o comando 968 sendo aplicado à rede, onde um *link* é removido entre os dispositivos de campo 1019 e 1015.

Figura 7 - (a) Antes do comando 968. (b) Após o comando 968 ser enviado pelo gerenciador.



Fonte: Adaptado de (KRÖTZ, 2019).

2.7 Latência

Latência é o tempo total gasto por um quadro desde a origem até o destino. Esse tempo absoluto é a soma dos atrasos do processamento nos elementos da rede e o atraso de propagação ao longo do meio de transmissão (BURGESS, 2004).

Para medir a latência, um quadro de teste contendo uma marca de tempo (*timestamp*) é transmitido pela rede. A marca de tempo é então analisada quando o quadro é recebido. Para que isso ocorra, o quadro de teste precisa voltar ao testador original por um laço de retorno (atraso de ida e volta). Uma grande latência não indica que ocorrerá degradação da voz, o que pode ocorrer é uma perda de sincronização.

Figura 8 - Exemplificação de uma mensagem, no qual o tempo descorrido entre seu envio e recebimento é denominado de latência da rede.

2.8 *Received Signal Level*

O *Received Signal Level* (RSL) é uma indicação do nível de potência recebido pela antena. Portanto, quanto maior o número de RSL, mais forte é o sinal recebido. Representado em dBm (decibéis em relação ao miliwatt), em geral, negativo. O menor sinal detectável é a sensibilidade do rádio, também representada em dBm. Por exemplo, o WH é 250 kbps e o IEEE802.15.4 demanda sensibilidade mínima de -85 dBm.

No protocolo WH, a capacidade do dispositivo de se comunicar com um vizinho é uma métrica fundamental na formação e preparação da rede em malha. Consequentemente, as estatísticas são mantidas nas tabelas de vizinhos. Esses incluem o nível médio de sinal recebido (RSL); estatísticas sobre os pacotes transmitidos e recebidos e o *timestamp* da última comunicação com o vizinho. Para vizinhos vinculados, o RSL é calculado usando um filtro *Infinite Impulse Response* (IIR) usando a seguinte equação (1):

$$RSL = RSL - \left(\frac{RSL}{RSL_{Damp}}\right) + \left(\frac{RSL_{Medido}}{RSL_{Damp}}\right) \quad (1)$$

Onde RSL_{Medido} é o RSL para o pacote atual e RSL_{Damp} é o fator de amortecimento. O RSL_{Damp} deve ter uma potência de 2 e o padrão é 64. Para vizinhos descobertos (ou seja, vizinhos com os quais o dispositivo não se comunica), o valor de RSL mais alto é retornado. O valor é ponderado para as comunicações mais recentes. Periodicamente, um dispositivo envia relatórios de saúde dos vizinhos para o gerenciador.

O valor de integridade da rede é principalmente o RSL de seus vizinhos. O valor é redefinido após cada relatório. Isso é importante para o gerenciador de rede otimizar a configuração da rede. Existem dois comandos para relatar a integridade do vizinho. A informação nesses dois comandos é exclusiva e o dispositivo deve reportar ambos periodicamente (787 e 780). O comando 780 fornece estatísticas para vizinhos ligados, isso é, existem ligações configuradas com os vizinhos. Outras informações, como estatísticas de comunicação, também são relatadas no Comando 780 (item 2.7.2) (CHEN; NIXON; MOK, 2010) .

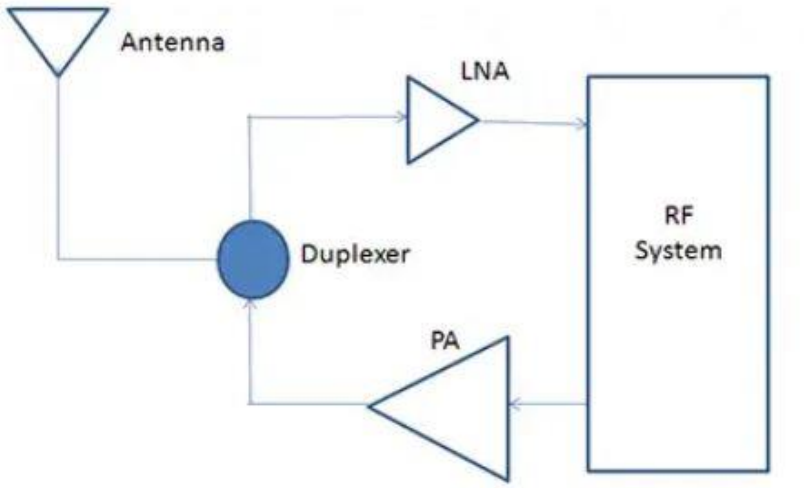
2.9 Power Amplifier (PA) e *Low-Noise Amplifier* (LNA)

A busca por desempenho, miniaturização e operação de frequência mais alta está desafiando os limites de dois componentes críticos conectados à antena de um sistema sem fio: o amplificador de potência – *Power Amplifier* (PA) – e o amplificador de baixo ruído – *Low-Noise Amplifier* (LNA).

Essas aplicações possuem requisitos que incluem menor ruído (para o LNA) e maior eficiência (para o PA), bem como operação em frequências mais altas. A função do LNA é levar o sinal extremamente fraco e incerto da antena, que pode ser da ordem de microvolts ou abaixo de -100 dBm, e amplificar para um nível maior, da ordem de dezenas a centenas de miliVolts.

Embora fornecer esse ganho em si não seja um grande desafio com a eletrônica moderna, ele é seriamente comprometido por qualquer ruído que o LNA possa adicionar ao sinal de entrada fraco. Esse ruído pode sobrecarregar qualquer benefício da amplificação que o LNA adiciona (SCHWEBER, 2013).

Figura 9 - O LNA do caminho de recepção e o PA do caminho de transmissão conectam-se à antena através de um duplexador.



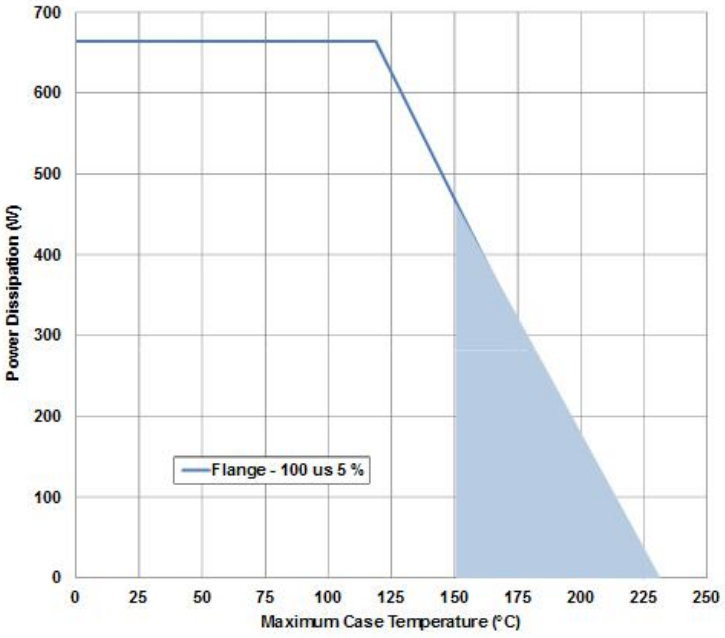
Fonte: (SCHWEBER, 2013).

Em contraste com o desafio de captura de sinal do LNA, o PA recebe um sinal relativamente forte com um *Signal-to-noise Ratio* (SNR) muito alto do circuito e, assim, deve aumentar sua potência. Todos os fatores gerais sobre o sinal são conhecidos, como amplitude, modulação, forma, ciclo de trabalho, entre outros. Esse

é o quadrante de sinal/ruído conhecido do mapa de processamento de sinal e o mais fácil de gerenciar. O parâmetro primário para o PA é sua saída de energia na frequência de interesse, com ganho típico de PA entre 10 e 30 dB. Juntamente com o ganho, a eficiência é o parâmetro importante na avaliação de um PA, mas qualquer avaliação de eficiência é influenciada pelo sistema de RF, ou seja, pela modulação a ser amplificado (SCHWEBER, 2013).

Igualmente importante, entre as muitas tabelas de especificação e curvas de desempenho de um PA, está a curva de redução de dissipação de energia Figura 10. Isso mostra a classificação da potência de saída disponível versus a temperatura do gabinete e indica que a potência máxima permitida é constante até 115 °C, depois diminui linearmente até a temperatura máxima de 150 °C.

Figura 10 - Curva de desclassificação de um PA mostra a redução na potência de saída permitida à medida que a temperatura do gabinete aumenta.



Fonte: (SCHWEBER, 2019).

As eficiências de PA para RF estão na faixa de 30 a 80%, mas isso depende de diversos fatores. Muitos PAs usam tecnologia *Complementary Metal Oxide Semiconductor* (CMOS) em níveis mais baixos de energia (até cerca de 1 a 5 W). Nos últimos anos, outras tecnologias amadureceram e também estão em uso

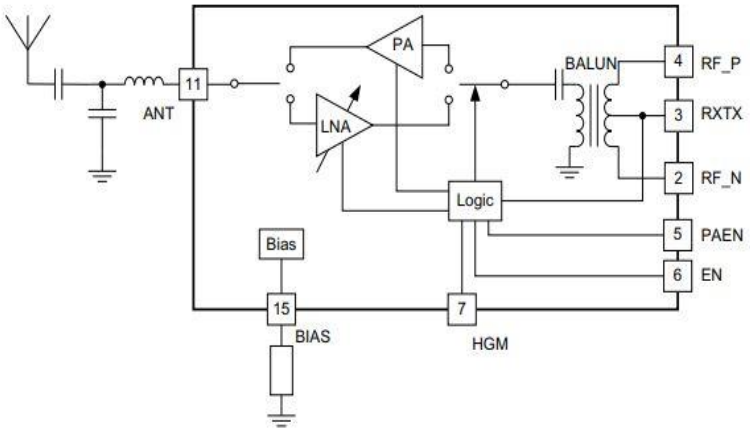
generalizado, especialmente em níveis mais altos de energia, em que a eficiência é crítica tanto para a vida útil da bateria quanto para considerações térmicas.

PAs feitos de GaN (nitreto de gálio) oferecem melhor eficiência em níveis de potências e frequências mais altas - tipicamente acima de 1 GHz, onde vários watts são necessários. PAs de GaN são competitivos, especialmente quando a eficiência e a dissipação de energia são consideradas (SCHWEBER, 2019).

Por exemplo, o CC2591 é um *front-end* de RF de baixo custo e alto desempenho para aplicações sem fio de 2,4 GHz, baixa potência e baixa voltagem. O dispositivo é um extensor de alcance para todos os RF atuais e futuros de 2,4 GHz de baixa potência. O diagrama interno deste componente pode ser visto na Figura 11.

O CC2591 fornece um PA para aumentar a potência de saída e um LNA com baixo ruído para melhorar a sensibilidade do receptor (TEXAS INSTRUMENTS, 2019).

Figura 11 - Diagrama do circuito interno do CC2591.



Fonte: (TEXAS INSTRUMENTS, 2019).

3 MÉTODOS E MATERIAIS

A descrição detalhada dos componentes que fazem parte do desenvolvimento e testes da proposta deste trabalho seguem a seguir.

3.1 Hardware

Como ilustrado na Figura 2, a rede WH possui uma série de dispositivos que a compõe. Para a realização do experimento, utilizou-se a estrutura disponível no Laboratório de Sistemas de Controle e Automação e Robótica (LASCAR), localizado na Universidade Federal do Rio Grande do Sul, Porto Alegre.

3.1.1 Gateway

Como *Network Manager*, *gateway* e *access point*, foi empregado o dispositivo comercial Emerson *Wireless* 1420A Gateway (EMERSON ELECTRIC, 2018). Ele possui a capacidade de conexão para com até 100 dispositivos, configuração automática de rede WH com rotas otimizadas e garantia de confiabilidade acima de 99%. Ainda, com a inserção de falhas por meio do software desenvolvido por (KRÖTZ, 2019), é possível, além da observação normal dos eventos de comunicação, obter informações sobre o desempenho da rede de forma automatizada. A Figura 12 apresenta o modelo utilizado.

Figura 12 - Emerson *Wireless* 1420A.

O dispositivo também apresenta uma interface homem-máquina, como mostra a Figura 13, onde muitas muitas informações da rede e estatísticas podem ser obtidas. Ainda, é possível enviar comandos para um ou mais dispositivos da rede. Isso auxiliará na metodologia para a construção do ambiente de teste desejado.

Figura 13 - Ambiente WEB disponível no gateway Emerson.

3.1.2 Dispositivos de campo

Como dispositivos de campo, foram utilizados os rádios elaborados no LASCAR, ilustrados na Figura 14, compatíveis com WH (MULLER *et al*., 2010). Os dispositivos são compostos por um microcontrolador *Freescale* MC13224, um transceptor de rádio IEEE 802.15.4 integrado e diversos periféricos responsáveis por demais características necessárias a um dispositivo WH. Ainda, o dispositivo conta com o PA analisado e descrito no item 2.9. A ferramenta utilizada para fazer as gravações de *firmware* é o software IAR *Embedded Workbench* 5.4.

Figura 14 - Dispositivo de campo disponível em laboratório.

3.1.3 Sniffer

O instrumento de captura dos sinais de comunicação para análise da rede é o *sniffer* Wi-Analys (HAN et al., 2009), presente na Figura 15. Trata-se de um dispositivo receptor de RF que captura as mensagens da rede WH dentro de seu raio de alcance e salva os logs/registros de cada mensagem. Sua utilidade no trabalho se deve à possibilidade de acompanhar o tráfego de mensagens da rede, bem como perceber que algum dispositivo não consta mais em seus registros.

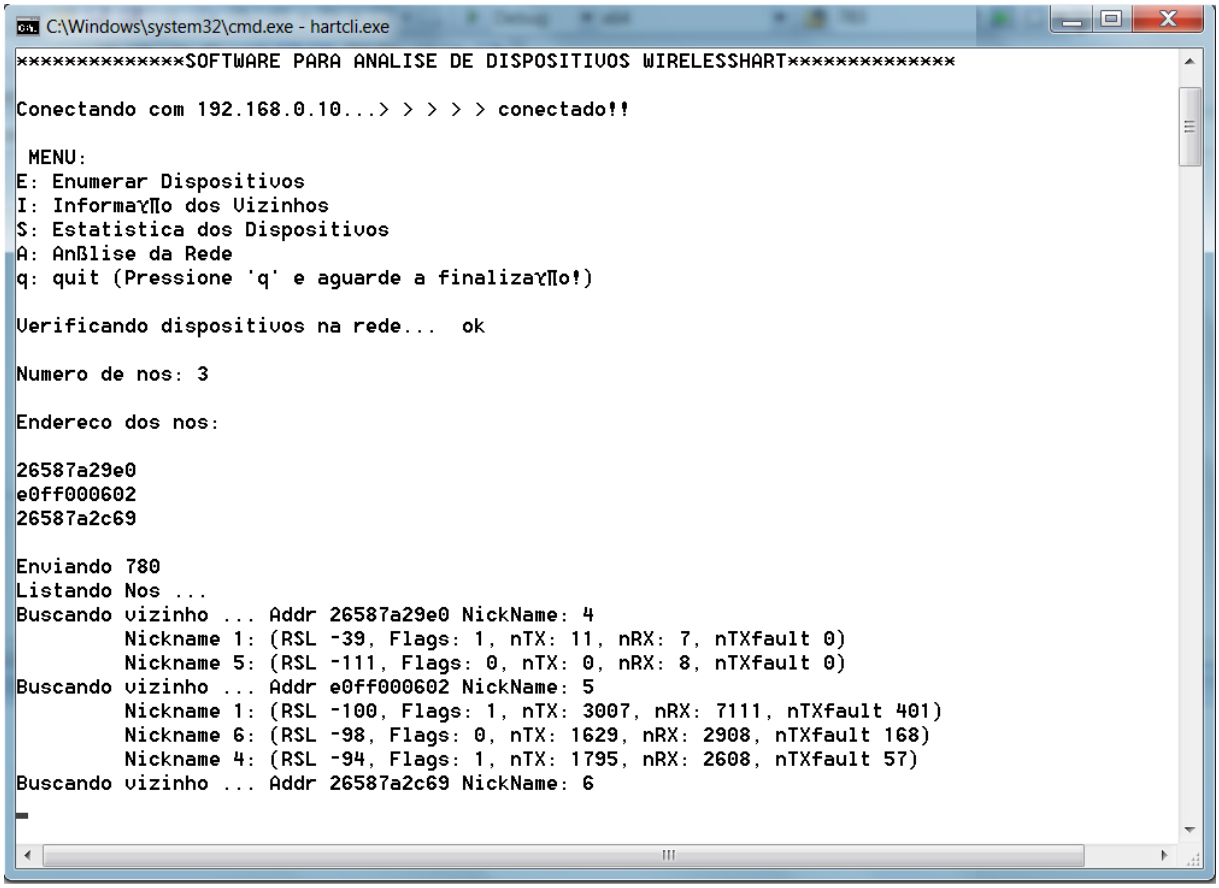
Figura 15 - *Wi-Analys Network Analyzer*.

3.2 Software

3.2.1 Ferramenta *Back-end*

Uma ferramenta para a obtenção de parâmetros de uma rede WH foi desenvolvida por (WINTER, 2010). Tal aplicação foi construída utilizando dois projetos: hartcli.cpp e hartip.cpp, entro do ambiente do Visual Studio. O projeto hartcli comporta a interface do programa e a formatação dos dados recebidos. Já o hartip gera uma *Dynamic Link Library* (DLL) que comporta as funções de montagem dos frames sobre *User Datagram Protocol* (UDP), as funções de envio de mensagem, validação de dados recebidos e estruturas dos comandos. Na Figura 16 é possível verificar a tela de console do início da aplicação (WINTER, 2010).

Figura 16 - Executável gerado ao compilar a solução.



Fonte: (WINTER, 2010)

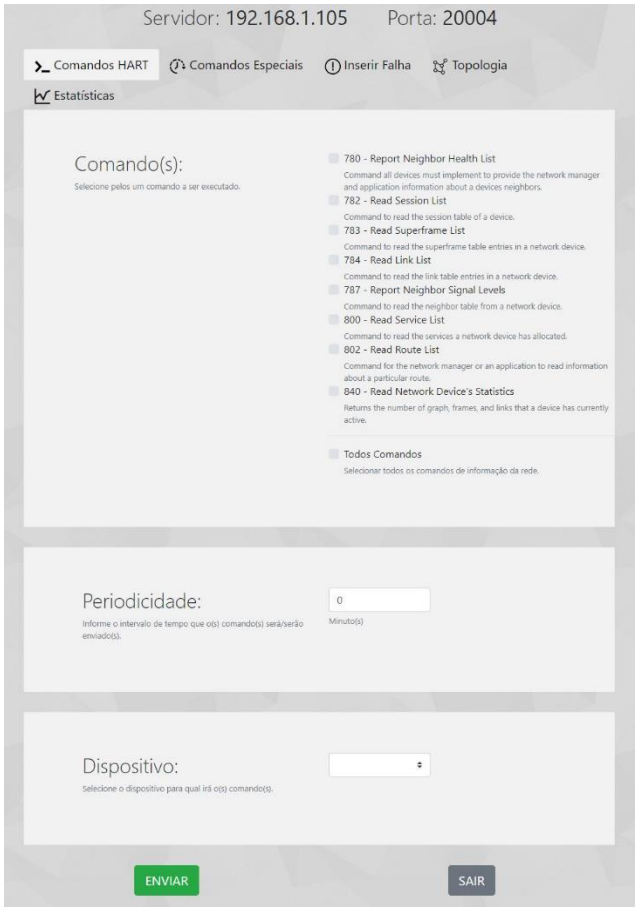
Essa solução foi desenvolvida há quase 10 anos e passou por algumas modificações que a deixaram mais completa e com um repertório de testes mais

amplo. No entando, essas modificações o deixaram com uma interface ainda pouco amigável ao usuário.

3.2.2 Ferramenta *Front-end*

Utilizando uma rede sem fio WH, a ferramenta desenvolvida utiliza o protocolo HART over IP para realizar a comunicação com o gateway. O *gateway* por sua vez, comunica-se com os dispositivos na rede por meio do ponto de acesso WH. A aplicação desenvolvida faz uso de comandos HART, implementados para a obtenção dos dados desejados para análise da rede e dos dispositivos. Os comandos HART são encapsulados e enviados para o gateway, que responde à requisição, enviando os dados solicitados (KRÖTZ, 2019). A interface da ferramenta desenvolvida em laboratório está presente na Figura 17.

Figura 17 - Interface de usuário da ferramenta.



A aba "Injeção de Falhas", contém os comandos implementados para a inserção de falhas na rede. As falhas implementadas foram elaboradas para refletir

situações reais no uso de RSSFI. Nesse sentido, pode-se simular problemas de hardware, cuja consequência é a sua desconexão da rede (comando 960), ocorrendo devido à bateria descarregada, mau contato ou antena quebrada.

Simular bloqueios de enlaces (comando 968) como, por exemplo, bloqueios que podem ocorrer quando veículos ou até mesmo uma pessoa se posicionam próximo a um dispositivo de campo, interferindo no enlace. Falhas que levam à assimetria do enlace entre dois pares, implementadas através do comando especial 129, podem ocorrer por dois motivos: comissionamento inadequado da potência de RF ou por falha no PA (KRÖTZ, 2019). O ambiente de testes elaborado neste trabalho faz uso desta aplicação.

3.3 Estudo de Caso

O estudo de caso que será apresentado é motivado por trabalhos anteriores de (WINTER, 2016), (SOUZA, 2013) e (KRÖTZ, 2019). Os dois primeiros autores notaram que, inserindo uma falha que leve à assimetria do enlace entre dois pares de uma rede WH, ocasiona um comprometimento do dispositvo de receber, reconhecer e transmitir mensagens de confirmação (ACK).

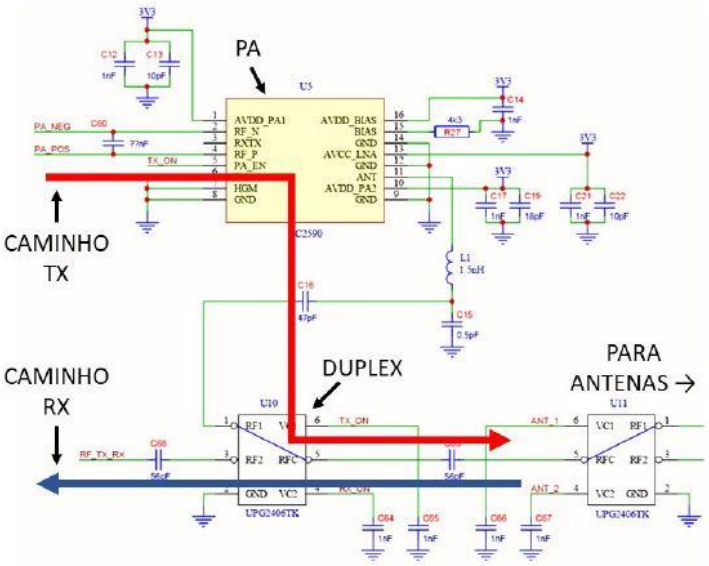
Esse tipo de falha pode ocorrer numa rede por dois motivos: comissionamento inadequado da potência de RF ou por falha no PA. A primeira, é decorrente do ajuste inadequado da potência de transmissão de RF, que pode ocorrer tanto no comissionamento como no ajuste de um dispositivo de campo já operacional. O problema ocorre quando o transceptor com a potência mal ajustada (muito baixa) está perto de um par, mas longe do outro.

Já a segunda, é decorrente de uma falha de hardware no PA, que pode ocorrer uma vez que esse componente é de potência e, por motivos diversos (como, por exemplo, excesso de calor ou sobretensão), pode queimar. KRÖTZ, 2019, implementou a simulação de falha no hardware dos dispositivos de campo presentes em laboratório, por meio de comando especial na camada de aplicação da pilha WH.

A Figura 18 mostra uma arquitetura de hardware usual para transceptores com PA externo. Pela análise da mesma, vê-se que o caminho para a recepção dos sinais

é diferente do caminho para a transmissão, que passa por um PA. Logo, uma falha no PA não compromete a sensibilidade da recepção de sinais de RF, mas sim, a transmissão. Nota-se que essa falha é diferente de um bloqueio de sinal, no qual há simetria (perda de potência transmitida e recebida). Para identificar essa falha no PA do dispositivo de campo, implementou-se uma rotina, que é executada toda vez que o comando 780 for solicitado ao gerenciador.

Figura 18 - Arquitetura de hardware usual para transceptores com PA externo.



Fonte: (KRÖTZ, 2019)

3.3.1 Implementação da aplicação

A detecção de uma falha pode, por muitas vezes, ser algo complexo em redes WH. Por haver diversas nuances presentes no protoloco, pode-se tomar conclusões precipitadas acerca de uma determinada falha na rede. Um dos casos, que é o cenário problema desse trabalho, apresentou-se em alguns trabalhos realizados em laboratório.

Com a ferramenta *front-end* desenvolvida, a injeção de falhas na rede foi facilitada, podendo-se, dentro das limitações de ambiente, criar topologias de redes a serem testadas. Uma dessas falhas que puderam ser injetadas na rede, simulando um problema, foi o desligamento do PA de um determinado dispositivo presente na

rede – escolhido via interface. Isso é algo que pode acontecer de fato em uma rede industrial, em que esse hardware pode apresentar algum tipo de problema, deixando de funcionar e, por consequência, comprometer a robustez da rede.

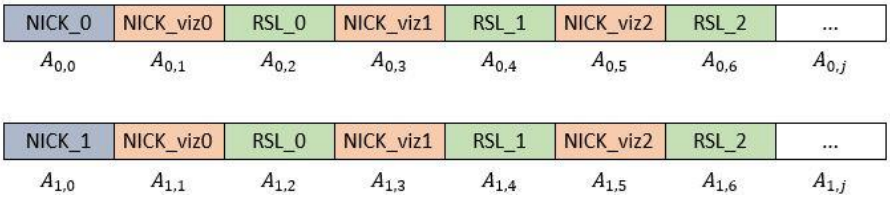
O mal funcionamento do PA acarreta na perda de potência de transmissão, não prejudicando a recepção de mensagens por parte do dispositivo. Com isso, os níveis de RSL que os vizinhos do dispositivo analisando irão detectar cairão substancialmente. Como estratégia de detectar essa falha, é utilizado o comando 780. Ele apresenta os valores de RSL e, por isso, será adotado como peça chave na solução proposta. O fluxograma apresentado na Figura 21 descreve a solução para verificar se houve ou não problema no PA.

Tendo em vista as ferramentas já apresentadas, criou-se uma estratégia baseada na estrutura presente na solução *back-end* de software, sendo ela toda descrita em linguagem C*.* Toda vez que é solicitado o comando 780 para um determinado dispositivo, ele irá retornar alguns dados, que podem ser observados no item 2.7.2. Nesse trabalho, considerou-se necessárias três informações:

- O dispositivo solicitado;
- Os *nicknames* dos seus vizinhos;
- O nível de RSL que o dispositivo solicitado percebe de seus vizinhos.

Para que se possa avaliar de forma recorrente os nívels de RSL, é necessário ter uma estrutura que armazene os dados acima citados. Para tal, adotou-se uma estrutura de armazenamento implementada na linguagem C, apresentada na Figura 19. O conceito de matrizes em C não será detalhado visto que esse não é o objetivo do presente trabalho. A matriz $A_{i,j}$ será usada para descrever o conceito adotado.

Figura 19 - Estratégia de adotada para armazenamento dos dados provenientes do comando 780.

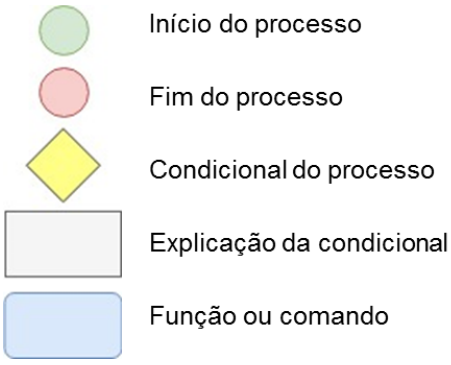


A posição $A_{i,0}$ sempre é ocupada pelo dispositivo solicitado. Na sequência da composição da linha da matriz, nas posições ímpares, estão armazenados os *nicknames* dos dispositivos vizinhos ao dispositivo solicitado.

Por outro lado, o RSL percebido é armazenado nas posições pares. Como há uma necessidade de cruzar dados – entre duas solicitações – são criadas duas estruturas iguais a essa, nas quais uma delas terá os dados referentes a primeira solicitação e, a segunda, a solicitação do comando 780 atual.

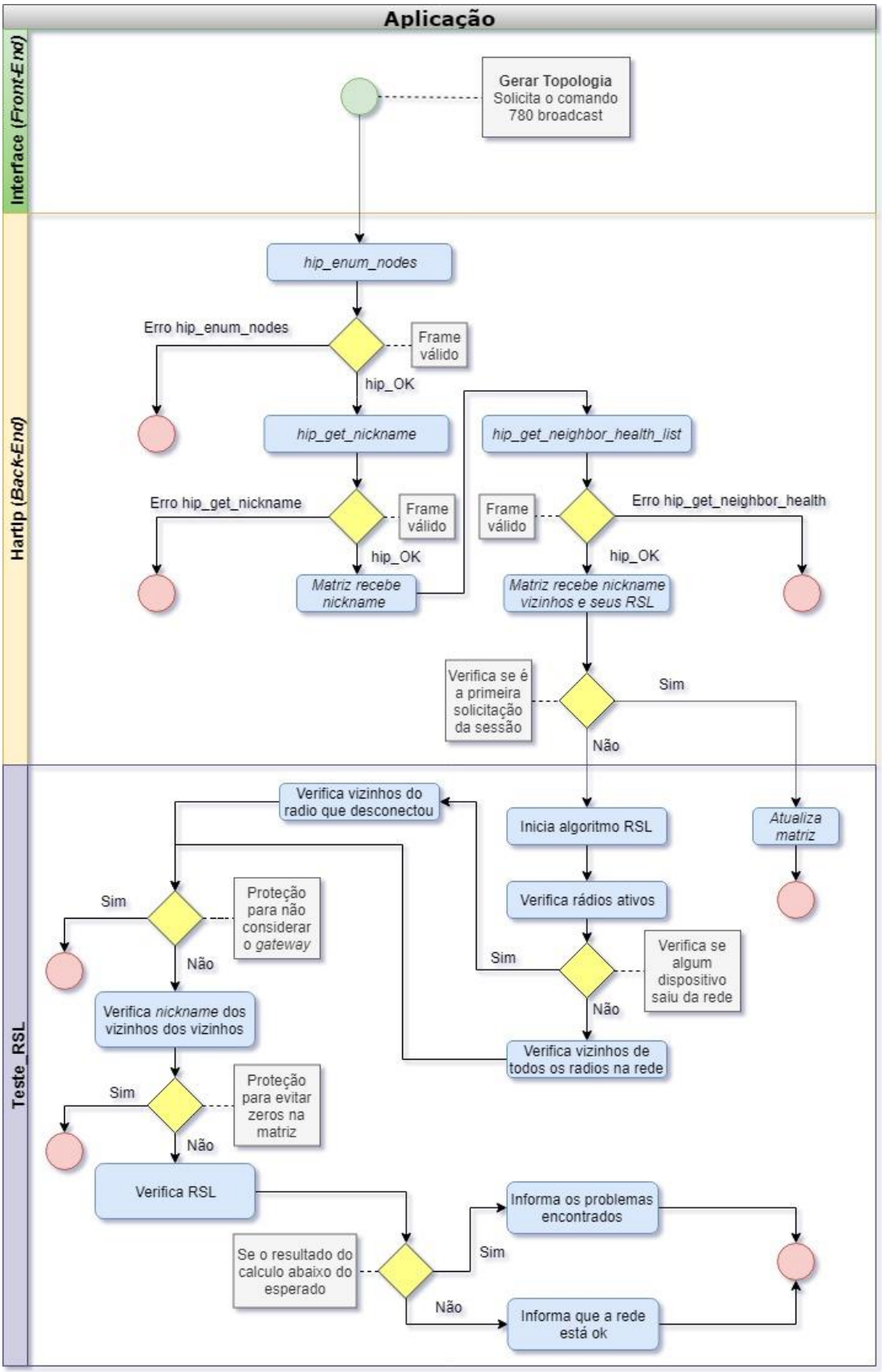
No fluxograma ilustrado na Figura 21, cada forma geométrica representa uma determinada tarefa, seja ela de início do processo, execução de comando ou função, condicional e final de processo. A legenda das represetações contidas no fluxograma, está na Figura 20.

Figura 20 - Legendas do fluxograma.



O início do processo se caracteriza pela solicitação do usuário para que o comando 780 seja enviado ao dispositivo. As condicionais acontecem nas 3 camadas de desenvolvimento, cada uma com sua particularidade. Igualmente para funções ou comandos, são dependentes da camada de software em que estão contidas.

Figura 21 - Fluxograma da Aplicação, dividida nas 3 camadas de software presentes no desenvolvimento.



Na camada do software *front-end*, é solicitado o comando 780, já adaptado (APÊNDICE B) para ser em modo *unicast* (enviado a todos os dispositivos presentes na rede, em laço). Já na camada *back-end,* os frames de comunicação são montados e enviados ao *gateway.* A ferramenta possui uma comunicação com o *gateway* via HART *over IP*, que suporta tanto a comunicação *Transmission Control Protocol* (TCP) quanto a *User Datagram Protocol* (UDP).

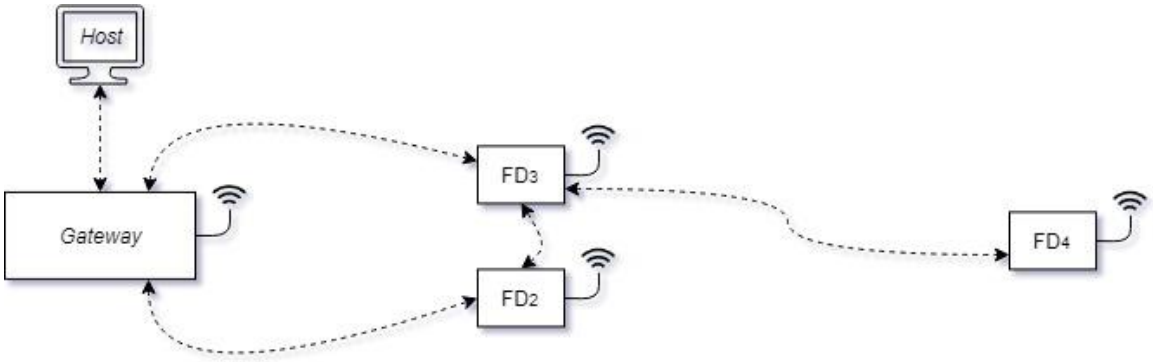
Nesse momento, são coletados os 3 dados importantes para a solução adotada. Preenchida a matriz com esses valores, inicia-se a verificação se há novos dispositivos ou se algum foi removido da rede. O código fonte com todas as funções presentes no método de análise desses parâmetros está no APÊNDICE A.

3.4 Criação do ambiente de testes

Os procedimentos para montar a rede WH que servirão de padrão para os testes do método de detecção de falha de enlance precisam seguir alguns passos importantes. Como não há acesso à forma de escalonamento das rotas no gerenciador (visto que é um dispositivo comercial) a rede, por vezes, há um longo período de *setup*, pois nem sempre os dispositivos estabelecem as conexões desejadas.

O *setup* desejado está presente na Figura 22. O gerenciador/*gateway* está conectado ao *host*. Os dispositivos de campo serão representados pela sigla FD (*field device*). A intenção de ter um dipositivo afastado do gerenciador e conectado a apenas um dispositivo – no caso, FD_4 afastado e conectado apenas em FD_3, demonstrará que, quando o PA de FD_3 for desligado, a comunicação entre o gateway e o dispotivo FD_4 será perdida. Isso pode fazer com que o operador da rede possa crer que há algum problema com o rádio mais afastado, entretanto, não é isso de fato o que ocorre.

Figura 22 - *Setup* da rede para avaliação do método.

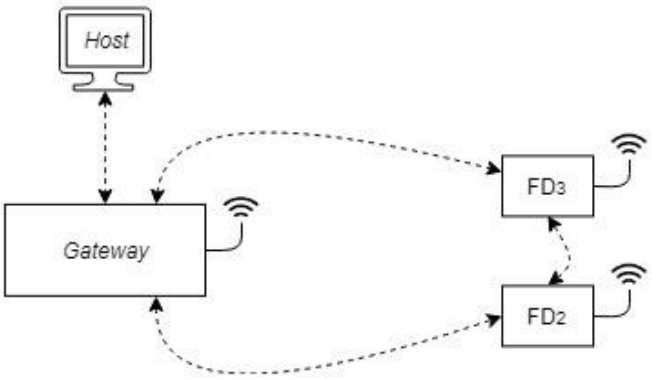


Os dispositivos FD_2 e FD_3 estão ligados mais próximos, distanciados em cerca de 30 cm. Tal distância é importante para que se possa receber os dados de transmissão do dispositivo afetado pela falha inserida. As linhas tracejadas com terminais em setas ilustram que há comunicação em ambas as direções (recepção e transmissão). Caso alguma linha apresente apenas uma seta, significa que há comunicação em um sentido. Já citado, o gerenciado/*gateway* possui algumas limitações, o que dificulta a criação do ambiente da Figura 22. Então, elaborou-se um roteiro de acionamento dos dispositivos, sequência de desligamento de *links* e, por fim, a injeção da falha na rede – o desligamento do PA no dispositivo FD_3. O passo a passo será descrito em tópicos a seguir e explicado posterioremente.

I. Ligar o *gateway* em modo *factory* e os dispositivos FD_2 e FD_3;
II. Após os 3 estarem interligados, liga-se o FD_4;
III. Deletar *links* (Figura 24) que não fazem parte da topologia ideal;
IV. Enviar o comando 771 aos dispositivos FD_4 e FD_2;
V. Fazer a primeira solicitação na ferramenta *front-end* do comando 780;
VI. Aplicar a falha no dispositivo FD_3 – desabilita o PA;
VII. Realizar a segunda solicitação do comando 780.

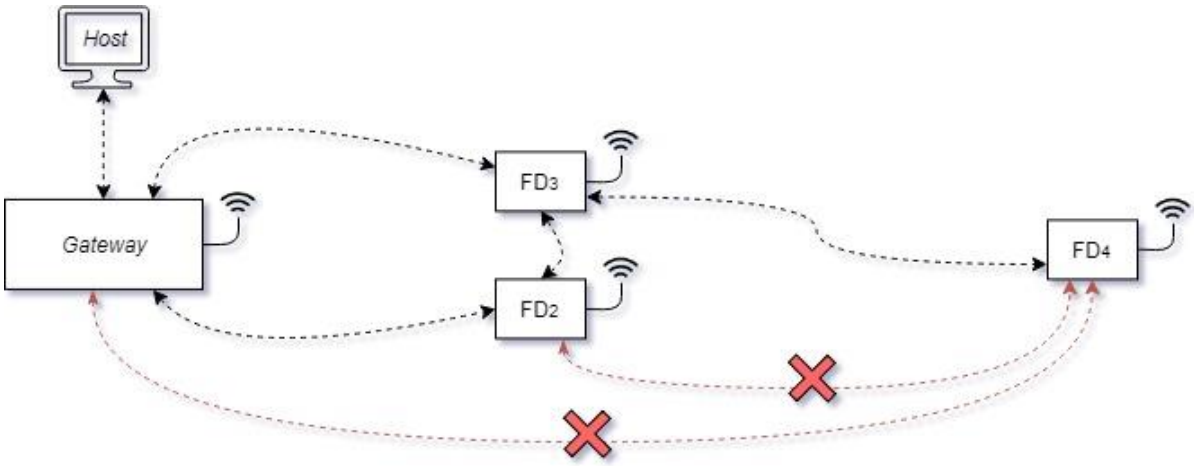
Apesar de poucos passos de criação do ambiente desejado, por possuir uma questão de segurança e roteamento do protocolo muito robusta, o procedimento pode levar até algumas horas para ser realizado. O primeiro passo, ligar o *gateway* em modo *factory,* via navegador *web,* traz a opção ao operador da rede de enviar comandos pela interface gráfica do gerenciador (Figura 12). Ainda na descrição do item I, ligam-se os dispositos que estão próximos.

Figura 23 - Topologia após o item I.



Com as conexões estabelecidas entre os dispositivos, deve-se ligar FD_4 (II). O Item III só se aplica caso existam *links* entre dispositivos que não sejam desejados, como mostra a Figura 24.

Figura 24 - Deletando links indesejados.



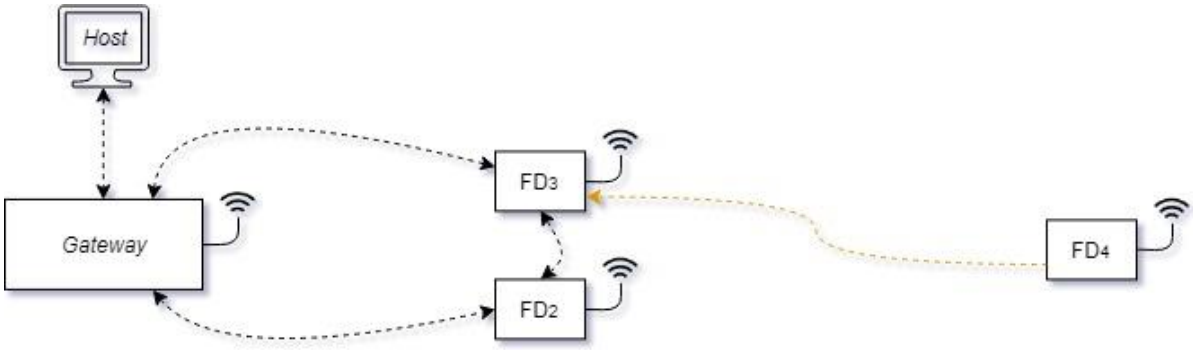
A proteção que o item IV irá trazer à topologia evita que os *links* deletados retornem na sequência do teste. Há aqui um porém: quando enviado ao dispositivo, o mesmo só voltará a realizar um *join* – uma nova conexão com outros dispositivos da rede, se o *firmware* do mesmo for regravado. Isso se deve ao fato de, na pilha WH, esse comando afetar a memória não-volátil do dispositivo.

O item V gera a fonte de comparação dos níveis de RSL pré e pós falha na rede. Nesse ponto, a primeira matriz de dados é gerada, armazenando as 3 variáveis de interesse do comando 780. Já em VI, também via ferramenta *front-end*, desabilita-se o PA do rádio FD_3 – o hardware CC2591 possui um pino que possibilita habilitá-lo

ou não. Logo o *link* de entre FD_3 e FD_4 irá desaparecer. Por vezes, o *link* de recepção de FD_4 por FD_3 permanecerá por um tempo extra, visto que o PA não influencia na recepção dos sinais transmitidos, como observado na Figura 25.

Como o gerenciador, entretando, irá solicitar informações comuns à todos os dispositivos e, visto que não há mais a rota de recepção de dados por FD_4, a solicitação não chegará a ele, fazendo com que não haja mais transmissões de frames. Em breve, ele será dado como *offline.* Já a comunicação entre FD_3 e FD_2 será mantida nos dois sentidos, visto a proximidade entre os eles.

Figura 25 - O dispositivo FD_3 apenas recebe os dados do seu vizinho distante.



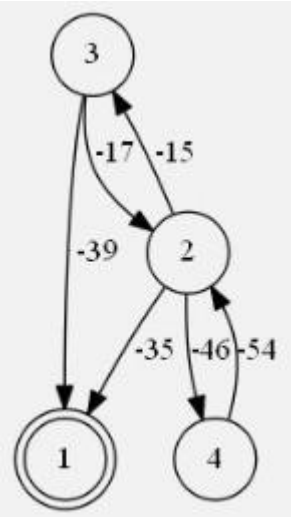
Por fim, após 10 minutos, envia-se novamente o comando 780 aos dispositivos, preenchendo a segunda matriz de dados. Com as duas matrizes compostas, inicia-se o algoritmo para verificação dos dados coletados.

4 RESULTADOS E DISCUSSÕES

Este capítulo apresenta os resultados e as discussões acerca dos mesmos. Para a implementação do método de análise do RSL em redes WH, necessitou-se estabelecer alguns padrões, visto que o protocolo possui diversas aspectos que devem ser levados em conta no momento da execução e elaboração dos testes. O estudo de caso apresentado em 3.3 supõe que o nível de RSL percebido pelos vizinhos do dispositivo afetado pela falha no PA irá diminuir, causado pela falha de enlace na transmissão.

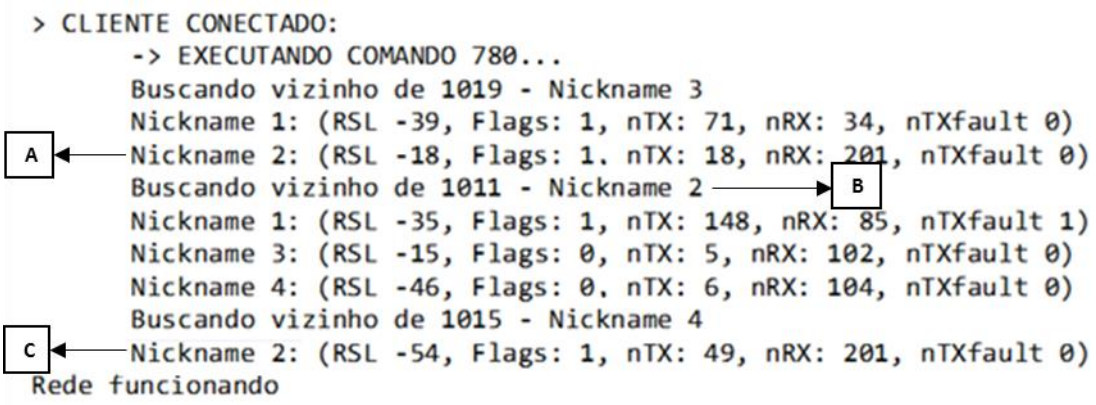
Após realizado o *setup* demonstrado no capítulo 3.4, a rede terá o formato da Figura 22, com dois dispositivos muito próximos e um terceiro distante desses dois, como mostra a Figura 26. Os resultados foram obtidos por meio da ferramenta *back-end*, que gerou arquivos .CSV com os dados coletados. Para facilitar a visualização, serão apresentados alguns testes e seus padrões obtidos de forma reduzida, pré e pós falha.

Figura 26 - Teste 1: Topologia gerada pré-falha



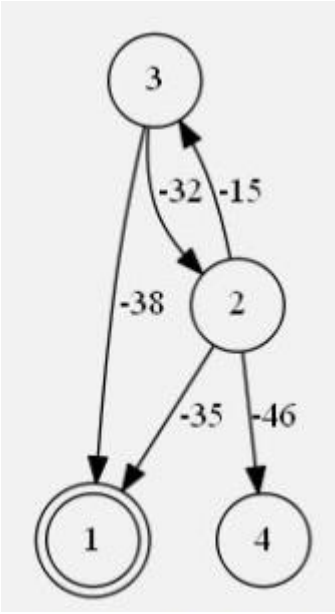
O *gateway* está representado pela circunferência de número 1. Já os demais (2, 3 e 4) são os *nicknames* dos dispositivos de campo testados. O par de rádios que está próximo é representado pelos números 2 e 3. As setas indicam os valores de RSL percebidos pelo dispositivo em relação ao seu vizinho, por exemplo: o dispositivo 4 percebe um nível de RSL no valor de -54 dBm do vizinho 2.

Figura 27 - Teste 1: Valores obtidos pré-falha na rede WH.



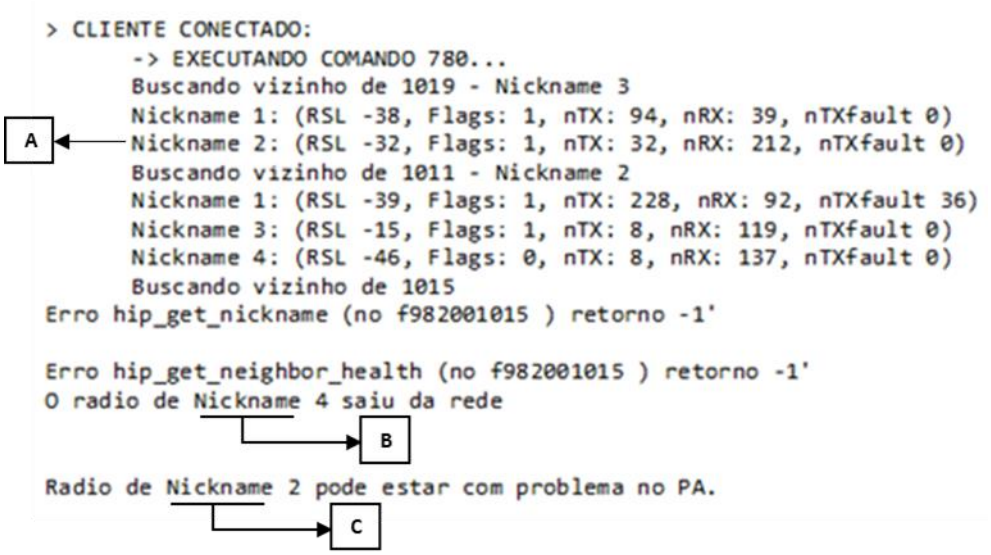
A primeira linha da Figura 27 indica que o cliente está conectado ao *host*. A segunda linha indica que o comando 780 da norma WH será executado. Após, há uma repetição do comando 780 para todos os dispositivos presentes na rede. O *nickname* 2 aparece destacado em 3 locais: primeiro, na linha indicada por "**A**", está o valor que representa o nível de RSL que o vizinho 1019 percebe de 1011. Já em "**B**", traz o dispositivo como sendo alvo do comando 780. Por fim, em "**C**", está o nível de RSL percebido pelo vizinho 1015. Ao final da execução, a aplicação de análise da rede já está operando, percebendo que não há problemas na rede já que nenhum dispositivo apresentou queda no nível de RSL ou saiu da rede.

Figura 28 - Teste 1: Topologia da Rede pós-falha.



Após a inserção da falha no sistema, no qual o PA do dispositivo 1011 foi desabilitado, a topologia ficou como ilustrado na Figura 28. Antes de gerar os novos dados, deve-se levar em conta dois aspectos do protocolo WH: o primero é considerar que o escalonamento da rede irá sempre priorizar a reabilitação de qualquer dispositivo presente nela, o que resulta numa certa latência em promover alterações bruscas em seu comportamento. O segundo aspecto, que decorre do primeiro, é ter por padrão aguardar cerca de 10 minutos para gerar a segunda estrutura de dados, mencionada no capítulo 3.3.1. Isso é necessário pois o valor de RSL recebido do comando 780 é fornecido como um valor médio (ANEXO 1). Os valores de RSL obtidos cerca de 10 minutos após a falha são vistos na Figura 29.

Figura 29 - Teste 1: Valores obtidos pós-falha na rede WH.



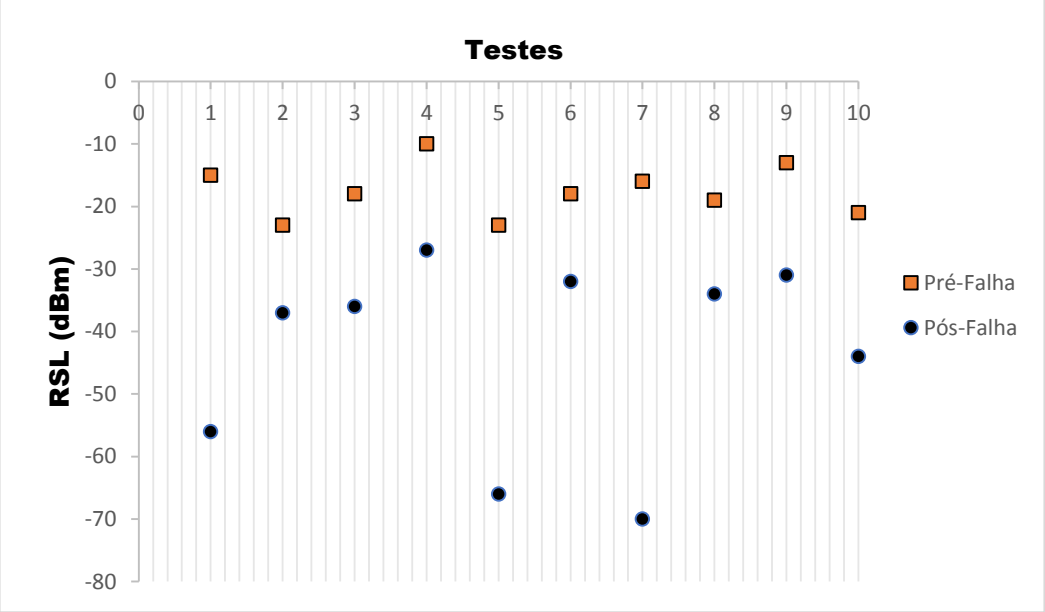
Em “**A**”, novamente é trazido o valor do RSL percebido do dispositivo 1011 pelo rádio 1019. Compara-se de imediato esse dado com o valor recebido, presente na Figura 27. Antes, o dispositivo 1019 recebia de 1011 com uma potência média de -18 dBm. Com a falha, o valor médio caiu para -32 dBm. Para fins de comprovação do método, no Teste 1, o cálculo de diferença de nível de RSL entre os momentos pré e pós falha foi calculado como mostra a equação (2). Além disso, o RSL_{Corte} foi setado no código para valores de 40% de degradação, para validar a capacidade do método em detectar a um problema no PA.

$$RSL_{Corte} = \frac{RSL_{pós}}{RSL_{pré}} - 1 \qquad (2)$$

Com os valores do Teste 1, notou-se que houve uma degradação do sinal em quase 80% em valores absolutos. Percebe-se que houve, em "**B**" a detecção da saída do dipositivo 1015 da rede e que o 1011 pode ter problemas, como apontado em **"C"**.

Para se obter uma validação estatística dos valores obtidos e para encontrar o intervalo de confiança, utilizou-se a distribuição T de *Student*, pois o número de amostras coletados foi baixo. A forma exata da distribuição T, que se assemelha muito com a normal, depende de um parâmetro chamado de graus de liberdade, que nada mais é do que o número de amostras menos um (n-1). O gráfico da Figura 30 apresenta 10 amostras de testes analisadas, as quais constam os valores de RSL pré e pós falha. Abaixo, os valores calculados da média (em absoluto) das diferenças de níveis de RSL pré e pós falha, o desvio padrão e a variância.

Figura 30 - Gráfico dos níveis de RSL pré e pós falha na rede WH



$$Média_{RSL_{dif}} = 24{,}2\ dBm$$

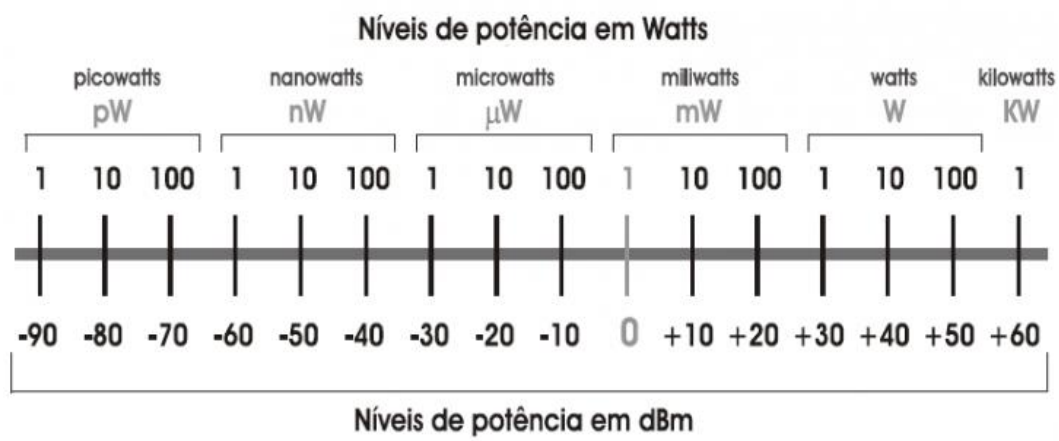
$$Desv\_Padrão_{RSL_{dif}} = 13{,}61\ dBm$$

$$Variância_{RSL_{dif}} = 185{,}29\ dBm^2$$

Utilizou-se a média das diferenças dos níveis de RSL por uma questão simples: apesar de o *setup* da rede estar exatamente igual – rádios e gerenciador na mesma posição tempos de latência da rede respeitados – existem outras várias inerentes aos

testes. Interferências de outras redes IEEE 802.11 presentes no ambiente, obstáculos no ambiente (pessoas, objetos). Com a diferença dos valores, pode-se perceber que o *gap* representa algo próximo de -25 dBm, um valor alto se considerarmos a Figura 31, que apresenta a potência em Watt e a relação com dBm.

Figura 31 - Relação Watt e dBm.



Para encontrar o valor que melhor caracterize a falha e seja usado como padrão no momento da comparação das estruturas $A_{i,j}$, calculou-se o intervalo de confiabilidade. Com um nível de significância de 0,05, 10 amostras e com o desvio padrão já calculado, pode-se obter um intervalo de confiança de ±9,73 dBm.

Isso significa que a diferença entre os níveis de RSL que estiverem dentro do intervalo de confiança de 24,2 ±9,73 dBm tem 95% de chances de estar com problema no PA. Por isso, o valor padrão da diferença entre os níveis de RSLs medidos, que será adotado, é de 14,46 dBm, já que para valores acima do outro intervalo (33,94dBm) se tem uma maior chance de que haja problema no hardware em questão – valores estão em absoluto.

Após o estabelecimento desse intervalo de confiança, realizou-se novos testes em outros dispositivos de campo. Os resultados são semelhantes aos anteriores, como a sequência de figuras 32 - 35 apresentam.

Figura 32 - Teste 2: Valores e topologia pré-falha na rede WH.

```
> CLIENTE CONECTADO:
       -> EXECUTANDO COMANDO 780...
       Buscando vizinho de 1019 - Nickname 3
       Nickname 1: (RSL -49, Flags: 1, nTX: 86, nRX: 32, nTXfault 0)
       Nickname 2: (RSL -16, Flags: 1, nTX: 2, nRX: 481, nTXfault 0)
       Buscando vizinho de 1005 - Nickname 2
       Nickname 1: (RSL -35, Flags: 1, nTX: 101, nRX: 43, nTXfault 0)
       Nickname 3: (RSL -15, Flags: 0, nTX: 5, nRX: 66, nTXfault 0)
       Nickname 4: (RSL -49, Flags: 0, nTX: 6, nRX: 50, nTXfault 0)
       Buscando vizinho de 1011 - Nickname 4
       Nickname 2: (RSL -54, Flags: 1, nTX: 18, nRX: 354, nTXfault 0)
Rede funcionando
```

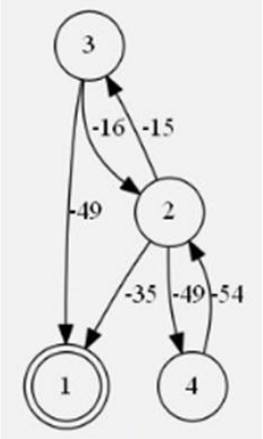


Figura 33 - Teste 2: Valores e topologia pós-falha na rede WH.

```
> CLIENTE CONECTADO:
       -> EXECUTANDO COMANDO 780...
       Buscando vizinho de 1019 - Nickname 3
       Nickname 1: (RSL -46, Flags: 1, nTX: 110, nRX: 36, nTXfault 0)
       Nickname 2: (RSL -70, Flags: 1, nTX: 3, nRX: 676, nTXfault 0)
       Buscando vizinho de 1005 - Nickname 2
       Nickname 1: (RSL -35, Flags: 1, nTX: 150, nRX: 49, nTXfault 11)
       Nickname 3: (RSL -15, Flags: 0, nTX: 5, nRX: 85, nTXfault 0)
       Nickname 4: (RSL -51, Flags: 0, nTX: 104, nRX: 75, nTXfault 97)
       Buscando vizinho de 1011
Erro hip_get_nickname (no f982001011 ) retorno -1'

Erro hip_get_neighbor_health (no f982001011 ) retorno -1'
O radio de Nickname 4 saiu da rede


Radio de Nickname 2 pode estar com problema no PA.
```

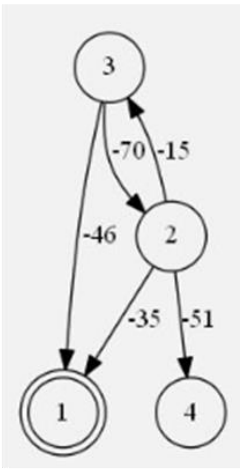


Figura 34 - Teste 3: Valores e topologia pré-falha na rede WH.

```
> CLIENTE CONECTADO:
       -> EXECUTANDO COMANDO 780...
       Buscando vizinho de 1019 - Nickname 3
       Nickname 1: (RSL -39, Flags: 1, nTX: 189, nRX: 143, nTXfault 1)
       Nickname 5: (RSL -20, Flags: 1, nTX: 7, nRX: 58, nTXfault 0)
       Nickname 4: (RSL -49, Flags: 0, nTX: 7, nRX: 78, nTXfault 0)
       Buscando vizinho de 1021 - Nickname 5
       Nickname 1: (RSL -37, Flags: 1, nTX: 32, nRX: 49, nTXfault 0)
       Nickname 3: (RSL -19, Flags: 0, nTX: 1, nRX: 42, nTXfault 0)
       Buscando vizinho de 1006 - Nickname 4
       Nickname 3: (RSL -53, Flags: 1, nTX: 32, nRX: 448, nTXfault 0)
Rede funcionando
```

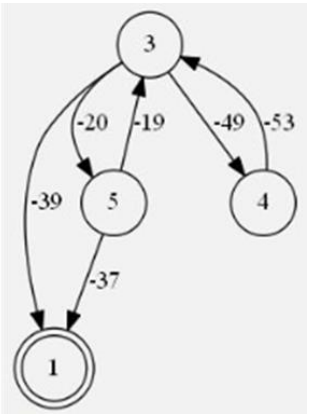
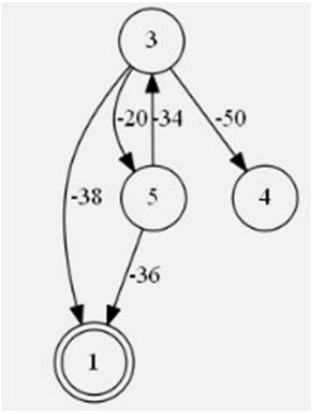


Figura 35 - Teste 3: Valores e topologia pós-falha na rede WH.

```
> CLIENTE CONECTADO:
      -> EXECUTANDO COMANDO 780...
      Buscando vizinho de 1019 - Nickname 3
      Nickname 1: (RSL -38, Flags: 1, nTX: 94, nRX: 39, nTXfault 22)
      Nickname 5: (RSL -20, Flags: 1, nTX: 16, nRX: 93, nTXfault 2)
      Nickname 4: (RSL -50, Flags: 0, nTX: 16, nRX: 100, nTXfault 7)
      Buscando vizinho de 1021 - Nickname 5
      Nickname 1: (RSL -36, Flags: 1, nTX: 51, nRX: 53, nTXfault 1)
      Nickname 3: (RSL -34, Flags: 0, nTX: 2, nRX: 63, nTXfault 0)
      Buscando vizinho de 1006
Erro hip_get_nickname (no f982001006 ) retorno -1'

Erro hip_get_neighbor_health (no f982001006 ) retorno -1'
O radio de Nickname 4 saiu da rede


Radio de Nickname 3 pode estar com problema no PA.
```



As Figuras 32 e 33 apresentam um segundo teste, realizado com o novo padrão de cálculo para avisar ao usuário sobre o problema na rede WH. Como é possível perceber, a algoritmo foi capaz de informar que o rádio pode estar com alguma falha de hardware, visto a discrepância analisada nos níveis de RSL.

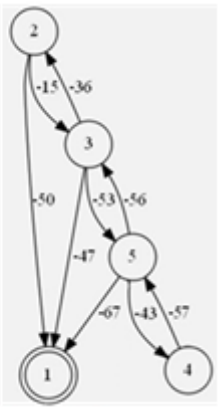
Também é perceptível nas Figuras 34 e 35, onde um terceiro teste foi realizado com outros dispositivos de campo, onde o mesmo resultado foi encontrado. Os dispositivos foram diagnosticados corretamente: o *device* 1006 acabou sendo desconectado da rede e o seu vizinho 1019 foi apontado para o usuário como um possível causador desta falha, devido a um problema em seu PA.

Para ter certeza que a estrutura funcionará com a adição de outros *devices* à rede WH, realizou-se outro teste. Inserindo-se mais um rádio a rede WH, como mostra a Figura 36.

Figura 36 - Teste 4: Topologia da rede pré-falha.

```
> CLIENTE CONECTADO:
      -> EXECUTANDO COMANDO 780...
      Buscando vizinho de 1019 - Nickname 2
      Nickname 1: (RSL -50, Flags: 1, nTX: 71, nRX: 34, nTXfault 0)
      Nickname 3: (RSL -15, Flags: 1, nTX: 18, nRX: 201, nTXfault 0)
      Buscando vizinho de 1006 - Nickname 3
      Nickname 1: (RSL -47, Flags: 1, nTX: 148, nRX: 85, nTXfault 1)
      Nickname 2: (RSL -36, Flags: 0, nTX: 5, nRX: 102, nTXfault 0)
      Nickname 5: (RSL -53, Flags: 0, nTX: 6, nRX: 104, nTXfault 0)
      Buscando vizinho de 1005 - Nickname 5
      Nickname 1: (RSL -67, Flags: 1, nTX: 148, nRX: 85, nTXfault 1)
      Nickname 3: (RSL -56, Flags: 0, nTX: 5, nRX: 102, nTXfault 0)
      Nickname 4: (RSL -43, Flags: 0, nTX: 6, nRX: 104, nTXfault 0)
      Buscando vizinho de 1015 - Nickname 4
      Nickname 5: (RSL -57, Flags: 1, nTX: 49, nRX: 201, nTXfault 0)

Rede funcionando
```



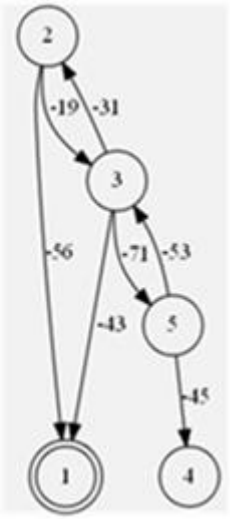
Na figura 37, pode-se observar que, após injetada a falha de enlace na rede (desabilitando o PA no dispositivo 1005). O algoritmo funcionou mesmo saindo da topologia característica dos testes 1, 2 e 3, conseguindo apontar o possível problema no dispositivo afetado.

Figura 37 - Teste 4: Topologia de rede pós-falha.

```
> CLIENTE CONECTADO:
      -> EXECUTANDO COMANDO 780...
      Buscando vizinho de 1019 - Nickname 2
      Nickname 1: (RSL -56, Flags: 1, nTX: 91, nRX: 36, nTXfault 0)
      Nickname 3: (RSL -19, Flags: 0, nTX: 27, nRX: 220, nTXfault 0)
      Buscando vizinho de 1006 - Nickname 3
      Nickname 1: (RSL -43, Flags: 1, nTX: 152, nRX: 101, nTXfault 1)
      Nickname 2: (RSL -31, Flags: 0, nTX: 13, nRX: 132, nTXfault 0)
      Nickname 5: (RSL -71, Flags: 0, nTX: 19, nRX: 112, nTXfault 0)
      Buscando vizinho de 1005 - Nickname 5
      Nickname 3: (RSL -53, Flags: 0, nTX: 5, nRX: 115, nTXfault 0)
      Nickname 4: (RSL -45, Flags: 1, nTX: 6, nRX: 132, nTXfault 0)
      Buscando vizinho de 1015
Erro hip_get_nickname (no f982001015 ) retorno -1'

Erro hip_get_neighbor_health (no f982001015 ) retorno -1'
O radio de Nickname 4 saiu da rede


Radio de Nickname 5 pode estar com problema no PA.
```



5 CONCLUSÕES

A comunicação sem fio industrial ainda apresenta diversos desafios para pesquisadores e desenvolvedores, visto o espaço hostil que muitas vezes é enfrentado nesses ambientes. Além das ferramentas disponíveis no protocolo WH, deve-se ter muita atenção em falhas associadas ao hardware utilizado para compor determinada aplicação, desde o *host* até os dispositivos de campo. Outro ponto importante, é ter posse de boas ferramentas de avaliação da rede, que possam assegurar bons parâmetros de desempenho.

Em diversos momentos, pode-se tomar decisões equivocadas caso não haja dados relativos a determinada falha. O presente trabalho apresentou um método para analisar falhas de enlace de transmissão em redes de sensores sem fio industrais, conseguindo validar tal método em um ambiente controlado, com uma topologia conhecida.

O estudo de caso descrito trouxe uma topologia com 3 dispositivos de campo, dos quais apenas um deles teria seu hardware afetado por meio de uma simulação de falha no seu amplificador de potência (PA). Tal falha prejudicou a transmissão de pacotes com o dispositivo mais afastado, porém manteve potência suficiente para se comunicar com seus vizinhos mais próximos. Essa falha fez com que o rádio mais afastado perdesse a conexão com a rede, levando a uma interpretação errônea.

O algoritmo descrito nesse trabalho pôde apresentar o verdadeiro diagnóstico, averiguado por meio dos níveis de RSL dos rádios da rede, apontando de fato qual era o dispositivo com problema. Encontrado o intervalo de confiança para a diferença entre os níveis de RSL pré e pós falha, apresentou-se que, para valores que estiverem dentro do intervalo de confiança de 24,2 ±9,73 dBm, tem-se 95% de chances de o rádio estar com problema no PA.

Ainda, para fins de validação do método, elaborou-se outra topologia, que apresentasse mais um dispositivo de campo conectado a rede. Saindo da topologia padrão, pode-se perceber que o método funcionou corretamente, trazendo o diagnóstico exato do dispositivo afetado pela falha simulada.

5.1 Trabalhos Futuros

Apesar das dificuldades que o WH apresenta como fonte de estudo, visto o grande esforço para compreender como o protocolo se estabelece e, também, dispor de tempo para realização de possíveis avaliações, como trabalhos futuros, pode-se sugerir a realização de novos testes do mesmo algoritmo. Sugere-se que, a inclusão de mais dispositivos a rede, inserindo a falha em mais de um desses dispositivos. Além disso, pode-se aplicar falhas além da de hardware. Um exemplo seria a utilização de objetos físicos entre os dispositivos, afetando os níveis de RSL colocando, assim, a prova o intervalo de confiança calculado para a topologia analisada. Outra sugestão, também, que pode ser válida para aprimorar ainda mais o algortmo, seria estabelecer uma correlação com a taxa de pacotes perdidos, podendo trazer ainda mais confiabilidade ao método.

6 REFERÊNCIAS

BURGESS, N. **Testing of Ethernet Services in Telecom Networks**: RFC 2544. [S.l.]: Agilent Technologies, 2004.

CASSIOLATO, C. Redes Industriais. Disponível em: <smar.com>. Acesso em 13 março, 2019.

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

EMERSON ELECTRIC. **Emerson Wireless 1420 Gateway**. Disponível em: <http://www.emerson.com/en-us/catalog/emerson-1420>. Acesso em 13 de março, 2019.

GUNANTARA N.; SADIARTA P. k.; PRESETYA A. I.; DHARMA A.; ANTARA I. N. **Measurements of the Received Signal Level and Service Coverage Area at the IEEE 802.11 Access Point in the Building.** 2018. Universitas Udayana. Journal of Physics: Conference Series.

HCF: **HART COMMUNICATION FOUNDATION**. HCF_SPEC-065, Rev. 1.0. Austin: HCF, 2007.

KRÖTZ, C. A. **Ferramenta e Método para Análise de Confiabilidade Fim-A-Fim de Redes Industriais Sem Fio**. 2019. Dissertação (Mestrado em Engenharia Elétrica) — Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS.

KUNZEL, G. et al. Passive monitoring software tool for evaluation of deployed wirelesshart networks. In: BRAZILIAN SYMPOSIUM ON COMPUTING SYSTEM ENGINEERING, 2012, Natal.

LIAO J-G.; GE J-L. **STUDY ON WIRELESS HART NETWORK LAYER.** 2010. University of Electronic Science and Technology of China, IEEE International Conference on Apperceiving Computing and Intelligence Analysis Proceeding (2010).

LOW K. S.; WIN N. N.; ER M. J. **WIRELESS SENSOR NETWORKS FOR INDUSTRIAL ENVIRONMENTS.** 2005. IEEE Computer Society.

MULLER, I. **Gerenciamento Descentralizado de Redes Sem Fio Industriais Segundo o Padrão Wirelesshart**. 2012. Tese (Doutorado em Engenharia Elétrica) — Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS.

MULLER, I. et al. Development of a WirelessHART compatible field device. In: INSTRUMENTATION MEASUREMENT TECHNOLOGY CONFERENCE PROCEEDINGS, 2010, Austin. Proceedings. . . Austin: IEEE, 2010. p.1430–1434.

OTERO CYSNE, L. F. **A NOVA BÍBLIA DO SOM.** Cysne Engineering LTD, USA. Ed.Cia do Book. 2016. ISBN – 9788555850448.

PINHEIRO, J. M. S. **O Modelo OSI**. Novembro de 2004. Disponível em: < www.projetoderedes.com.br/artigos/>. Acesso em: 20 Mar 2019.

SCHWEBER, B. **Understanding the Basics of Low-Noise and Power Amplifiers in Wireless Designs**. Outubro de 2013. Disponível em: < https://www.digikey.com/ >. Acesso em 13 de março, 2019.

SMAR. WirelessHART - **Características, tecnologia e tendências**. Outubro de 2017. Disponível em: <http://www.smar.com/>. Acesso em: 13 Mar 2019.

SOUSA, F. A. A. C. Testes de robustez em uma rede WirelessHART. 2013. Graduação (Bacharel em Engenharia Elétrica) — Programa de Graduação em Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre, RS.

SPEC-065: 2.4GHZ DSSS O-QPSK – **Physical Layer Specification**. HART Communication Foundation. SPEC-065, 2007.

SPEC-127. **Universal Command Specification**. HART Communication Foundation. SPEC-127, 2008

SPEC-151. **Common Practice Command Specification**. HART Communication Foundation. SPEC-151, 2008.

SPEC-155. **Wireless Command Specification**. HART Communication Foundation. SPEC-155, 2008.

TEXAS INSTRUMENTS**. CC2591 2.4-GHz RF Front End.** Setembro de 2014. Disponível em: < http://www.ti.com/lit/ds/symlink/cc2591.pdf>. Acesso em: 10 abril 2019.

WANG, Q.; JIANG, J. Comparative examination on architecture and protocol of industrial wireless sensor network standards. IEEE Communications Surveys Tutorials, New York, v.18, p.2197–2219, 2016.

WINTER, J. M. **Software de análise de roteamento de dispositivos WirelessHART**. 2010. Graduação (Bacharel em Engenharia Elétrica) — Programa de Graduação em Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2010.

WINTER, J. M. et al. Analysis of a radio physical layer fault in WirelessHART networks. In: INTERNATIONAL INSTRUMENTATION AND MEASUREMENT TECHNOLOGY CONFERENCE PROCEEDINGS, 2016, Taipei.

APÊNDICE A – CÓDIGO FONTE DA APLICAÇÃO EM C PARA ANÁLISE DA FALHA DE ENLACE EM REDES WIRELESSHART

```
/*********************************************************************
*                                                                    *
*                       teste_algoritmo_RSL.h                        *
*                                                                    *
*********************************************************************/
#ifndef TESTE_ALGORITMO_RSL_H
#define TESTE_ALGORITMO_RSL_H

#ifdef __cplusplus
extern "C"
{
#endif

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>


        void atualiza_matriz(void);
        void verifica_vizinhos_radio_desconectado(int i);
        int verifica_radios_ativos(int x);
        int verifica_RSL_vizinhos_dos_vizinhos(int x);
        int verifica_vizinhos_dos_vizinhos(int x);

        int inicia_algoritmo_rsl(void);

        char dados_RSL[100][512];
        char dados_RSL_old[100][512];


#ifdef __cplusplus
}
#endif



#endif // !TESTE_ALGORITMO_RSL_H
```

```
/*********************************************************************
*                                                                    *
*                      teste_algoritmo_RSL.c                         *
*                                                                    *
*********************************************************************/

#include "Teste_algoritmo_RSL.h"

int radio_desconectado=0;
float calculo_diferenca_RSL;

int i, j, viz_viz;

int inicia_algoritmo_rsl(void)
{
        //preenche os valores com zeros
        //estrutura-> posicoes impares estao os nicknames dos vizinhos
        //            posicoes pares estao os RSL's de cada um
        do
        {
                radio_desconectado =
verifica_radios_ativos(radio_desconectado);
                if (radio_desconectado != -1)
                {
                        printf("O radio de Nickname %d saiu da rede\n\n\n",
dados_RSL_old[radio_desconectado][0]);

        verifica_vizinhos_radio_desconectado(radio_desconectado);
                }
                else
                {
                        printf("Nenhum Dispositivo Saiu da rede!\n\n\n");
//fazer varedura igual
                        for (i = 0; i < 100; i++)
                        {
                                verifica_vizinhos_radio_desconectado(i);
//vou verificar apenas os niveis de RSL!
                        }
                }
        }while(radio_desconectado!=-1);

        atualiza_matriz();
        return 1;
}

int verifica_vizinhos_dos_vizinhos(int x)
{
    int y,z;
    int aux_posicao;
```

```
    for(y=0; y<100; y++)
    {
        if(x==dados_RSL_old[y][0])
        {
            aux_posicao=y;
            for(z=0;z<250;z++)
            {
                if(dados_RSL_old[aux_posicao][2*z+1]!=0 &&
dados_RSL_old[aux_posicao][2*z+1] != 1)
                    return dados_RSL_old[aux_posicao][2*z+1];
            }
        }
    }
    return 0; //nao ha problemas com o radio.
}

void atualiza_matriz(void)
{
       memset(dados_RSL_old, 0, sizeof(dados_RSL_old)); //preenche os
valores com zeros
       for (i = 0; i<100;
i++)                                                    //atulizar matriz
       {
              for (j = 0; j<512; j++)
                     dados_RSL_old[i][j] = dados_RSL[i][j];
       }
}

//verifica se todos os radios estao na rede
//retorna -1 se todos estao, ou o valor da posiçao na matriz do radio que
saiu
int verifica_radios_ativos(int x)
{
       if(x!=0) //proteçao para quando for fazer a varedura completa
              x++;
       for (i = x; i<100; i++)
       {
              if (dados_RSL_old[i][0] != dados_RSL[i][0] &&
dados_RSL_old[i][0] != 0) //protecao para entrada de novos radios
                     return i; //retorna o nickname do radio que nao esta
mais na rede
       }

       return -1;//nao ha nickname negativo
}



int verifica_RSL_vizinhos_dos_vizinhos(int x)
```

```
 {
    int y,z;
    int aux_posicao;
    for(y=0; y<100; y++)
    {
        if(x==dados_RSL_old[y][0])
        {
            aux_posicao=y;
            for(z=0;z<250;z++)
            {

if(dados_RSL_old[aux_posicao][2*z]!=dados_RSL[aux_posicao][2*z] &&
dados_RSL[aux_posicao][2*z]!=0) //valor da nova matriz pode ser zero -->
proteçao
                {
                                        printf("antigo %d e novo %d\n\n\n",
dados_RSL_old[aux_posicao][2 * z], dados_RSL[aux_posicao][2 * z] );
                                        //calculo o quanto eles sao
diferentes

        calculo_diferenca_RSL=(dados_RSL_old[aux_posicao][2*z])+(float)(dad
os_RSL[aux_posicao][2*z]);
                                        if(fabs(calculo_diferenca_RSL) >=
14.46) //o valor indicar que o RSL atual eh pior (maior em absoluto)
                                                return 1; //retornar que o
radio esta com problema
                }
            }
        }
    }
        return 0; //nao ha problemas com o radio.
}

void verifica_vizinhos_radio_desconectado(int i)
{
        int mensagem,radio_aux;
        for (j = 1; j<100; j++) //começa de 1 para pular a posicao do
nickname lido
        {
                viz_viz = dados_RSL_old[i][2 * j - 1]; //pego o nickname do
vizinhos do radio que saiu da rede
                if (viz_viz != 1 && viz_viz != 0)
                {
                        radio_aux = verifica_vizinhos_dos_vizinhos(viz_viz);
//verifico o desses vizinhos com seus vizinhos
                        if (radio_aux != 0)
                                mensagem =
verifica_RSL_vizinhos_dos_vizinhos(radio_aux);
                                if(mensagem!=0)
```

```
                                        printf("Radio de Nickname %d pode
estar com problema no PA.\n\n\n", viz_viz);
                }
        }
}
```

APÊNDICE B – COMANDO 780 ADAPTADO

```
static char * comando780broadcast(struct hip_sess *sess)
{
 char resp[2048] = "";
 int rv = -1;
 {
  int rv;
  hip_addr_t *nodes;
  size_t nnodes;
  struct hip_node_vizinhos_linked *tabela_vizinho[10];
  unsigned int nickname[10];
  FILE *arq;
  time_t agora;
  struct tm *ts;
  char timestamp[30];
  int aux, aux2, k=0;
  nodes = NULL;
  for (aux = 0; aux<10; aux++)
   {
    tabela_vizinho[aux] = NULL;
   }
  arq = fopen("Comando780b.csv", "w");
  fprintf(arq, "Date;Hour;Nickname from;Nickname destination;
               RSL; Packets transmitted;Packets received;
               nTXfault;Neighbor Flags;Ciclo");
  fclose(arq);
  printf("\n\t -> EXECUTANDO COMANDO 780b...\n");
  fflush(stdout);
  rv = hip_enum_nodes(sess, &nodes, &nnodes);
  memset(dados_RSL, 0, sizeof(dados_RSL)); // Limpa Matriz dado_RSL
  if (rv == HIP_OK)
   {
    for (aux = 0; aux<nnodes; aux++)
     {
      printf("\t Buscando vizinho de ");
      printf("%x", nodes[aux] & 0xFFFF);
      rv = hip_get_nickname(sess, nodes[aux], &nickname[aux]);
      if (rv == HIP_OK)
        {
         _tprintf(_T(" - Nickname %u\n"), nickname[aux]);
         //armazena nickname do dispositivo
         dados_RSL[aux][0]=nickname[aux];
        }
      else
        {
         printf("\nErro hip_get_nickname (no ");
         print_addr(nodes[aux]);       // Escreve endereco
```

```
         printf(" ) retorno %d'\n", rv);
        }
      rv = hip_get_neighbor_health_list(sess, nodes[aux], &tabela_vizinho[aux]);
      if (rv == HIP_OK)
        {
          char line[512];
          char * k_string;
          arq = fopen("Comando780b.csv", "a");
          // Cria timestamp
          agora = time(NULL);
          ts = localtime(&agora);
          strftime(timestamp, sizeof(timestamp), "%d/%m/%Y;%H:%M:%S", ts);
          for (aux2 = 0; aux2 < tabela_vizinho[aux]->vizinhos_lidos; aux2++)
           {
           _tprintf(_T("\t Nickname %d: (RSL %d, Flags: %d, nTX: %d, nRX: %d,
nTXfault %d)\n"),
                tabela_vizinho[aux]->lista[aux2].nickname,
                tabela_vizinho[aux]->lista[aux2].RSL,
                tabela_vizinho[aux]->lista[aux2].flags,
                tabela_vizinho[aux]->lista[aux2].pacotes_okTX,
                tabela_vizinho[aux]->lista[aux2].pacotes_okRX,
                tabela_vizinho[aux]->lista[aux2].pacotes_falhaTX);

                dados_RSL[aux][2*aux2+1] = tabela_vizinho[aux]->lista[aux2].nickname;
                dados_RSL[aux][2*aux2+2] = tabela_vizinho[aux]->lista[aux2].RSL;

                fprintf(arq, "%s;%d;%d;%d;%d;%d;%d;%d;%d\n",
                timestamp,
                nickname[aux], // nodes[aux] & 0xFFFF,
                tabela_vizinho[aux]->lista[aux2].nickname,
                tabela_vizinho[aux]->lista[aux2].RSL,
                tabela_vizinho[aux]->lista[aux2].pacotes_okTX,
                tabela_vizinho[aux]->lista[aux2].pacotes_okRX,
                tabela_vizinho[aux]->lista[aux2].pacotes_falhaTX,
                tabela_vizinho[aux]->lista[aux2].flags,
                k);
              }
             fclose(arq);
         }
        else
         {
           printf("\nErro hip_get_neighbor_health (no ");
           print_addr(nodes[aux]);     // Escreve endereco
           printf(" ) retorno %d'\n", rv);
         }
        }

        if (flag_inicia_algoritmo != 1)
        {
```

```
          flag_inicia_algoritmo = 1;
          atualiza_matriz();
        }
        else
        {
        flag_inicia_algoritmo = inicia_algoritmo_rsl();
        }
     }
    else
     {
       printf("\nErro hip_enum_nodes: retorno %d\n", rv);
     }
 if (nodes)
   {
    hip_free_node_list(sess, nodes);
   }
  for (aux = 0; aux<9; aux++)
   {
     if (tabela_vizinho[aux])
      {
        free(tabela_vizinho[aux]->lista);
        free(tabela_vizinho[aux]);
      }
   }
 }
 return resp;
}
```

ANEXO 1 – HCF SPEC-075 – CÁLCULO DO RSL

Subsection 9.2.3
Details on averaging the RSL must be clarified. After paragraph 4 these details must be inserted as follows:

The device's ability to communicate with a neighbor is a key metric in forming and grooming the mesh network. Consequently, statistics are maintained in each neighbor table entry. These include average Received Signal Level (RSL); statistics on the packets transmitted and received and the timestamp of the last communication with the neighbor.

For linked neighbors, RSL is calculated using an IIR filter using the following equation:

$$RSL = RSL - (RSL / RSLDamp) + (MeasuredRSL / RSLDamp)$$

Where MeasuredRSL is the RSL for the current packet and RSLDamp is the damping factor. RSLDamp must be a power of 2 and defaults to 64. For discovered or un-linked neighbors (i.e., neighbors the device does not communicate with) the highest RSL value is returned.

If a link to that neighbor exists, the LastTimeCommunicated is used to trigger transmission of Keep-Alive packets. A Keep-Alive must be transmitted to the neighbor (see Subsection 9.3) whenever the LastTimeCommunicated is greater than the keepAliveInterval. Keep-Alive transmissions are repeated until a new DLPDU is received from the neighbor.