UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

RENATO FERREIRA FERNANDES JUNIOR

Protocolo assíncrono de acesso ao meio iniciado pelo receptor para redes de
sensores sem fio

São Carlos
2018

RENATO FERREIRA FERNANDES JUNIOR

Protocolo assíncrono de acesso ao meio iniciado pelo receptor para redes de
sensores sem fio

Tese apresentada à Escola de Engenharia de
São Carlos da Universidade de São Paulo,
como requisito para a obtenção do Título de
Doutor em Ciências do Programa de
Engenharia Elétrica.

Área de Concentração: Sistemas Dinâmicos

Orientador: Prof. Dr. Dennis Brandão

São Carlos

2018

**Trata-se da versão corrigida da tese. A versão original se encontra disponível na
EESC/USP que aloja o Programa de Pós-Graduação de Engenharia Elétrica**

AUTORIZO A REPRODUÇÃO TOTAL OU PARCIAL DESTE TRABALHO,
POR QUALQUER MEIO CONVENCIONAL OU ELETRÔNICO, PARA FINS
DE ESTUDO E PESQUISA, DESDE QUE CITADA A FONTE.

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da
EESC/USP com os dados inseridos pelo(a) autor(a).

F363p Fernandes Jr, Renato Ferreira
Protocolo assíncrono de acesso ao meio iniciado
pelo receptor para redes de sensores sem fio / Renato
Ferreira Fernandes Jr; orientador Dennis Brandão. São
Carlos, 2018.

Tese (Doutorado) - Programa de Pós-Graduação em
Engenharia Elétrica e Área de Concentração em Sistemas
Dinâmicos -- Escola de Engenharia de São Carlos da
Universidade de São Paulo, 2018.

1. MAC assíncrono iniciado pelo receptor. 2. Redes
de sensores sem fio. 3. Controle de acesso ao meio
assíncrono multicanal. 4. Internet das Coisas. I.
Título.

FOLHA DE JULGAMENTO

Candidato: Engenheiro **RENATO FERREIRA FERNANDES JUNIOR.**

Título da tese: "Protocolo assíncrono de acesso ao meio iniciado pelo receptor para redes de sensores sem fio".

Data da defesa: 29/06/2018.

Comissão Julgadora:

Resultado:

Prof. Associado **Dennis Brandão**
(Orientador)
(Escola de Engenharia de São Carlos/EESC)

aprovado

Prof. Dr. **João Paulo Pereira do Carmo**
(Escola de Engenharia de São Carlos/EESC)

APROVADO

Prof. Dr. **Danilo Hernane Spatti**
(Instituto de Ciências Matemáticas e de Computação/ICMC-USP)

aprovado

Prof. Dr. **Paolo Ferrari**
(Università degli Studi di Brescia/Unibs)

APROVADO

Prof. Dr. **Rodrigo Palucci Pantoni**
(Instituto Federal São Paulo/IFSP - Sertãozinho)

APROVADO

Coordenador do Programa de Pós-Graduação em Engenharia Elétrica:
Prof. Associado **Luís Fernando Costa Alberto**

Presidente da Comissão de Pós-Graduação:
Prof. Associado **Luís Fernando Costa Alberto**

AGRADECIMENTOS

Agradeço primeiramente a minha esposa pelo enorme apoio, carinho, auxílio nas horas de crise, e nas revisões de texto que tanto precisei neste trabalho.

Ao orientador e amigo Dr. Dennis Brandão, pela orientação, dedicação e amizade. Não fosse sua orientação, este trabalho não se realizaria.

Ao amigo Marcelo Barros e outros do grupo, que me ajudaram nos testes, na ajuda com o projeto dos módulos, e nos auxílios no conhecimento de redes e de sistema embarcado.

Aos amigos da Universidade Federal de Uberlândia pelo incentivo e apoio no término da tese.

A minha filha Maria Beatriz que me ajudou nos testes, e também no apoio e carinho quando sempre precisei.

A meus pais que sempre me deram total apoio e força para conseguir chegar até aqui;

E a Deus que sempre esteve presente dentro de mim.

RESUMO

FERNANDES Jr, R. F. **Protocolo assíncrono de acesso ao meio iniciado pelo receptor para redes de sensores sem fio**. 2018. 157 f. Tese (Doutorado) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2018.

A internet das coisas é considerada um novo sistema de comunicação que promete otimizar e melhorar diferentes áreas de aplicação com base em módulos sensores e objetos unicamente interligados através da internet. Em aplicações de redes de sensores sem fio em larga escala, as redes possuem características peculiares, como grande quantidade de módulos sensores de baixa potência, consumo limitado e perdas de comunicação intermitentes. Estas redes precisam operar com protocolos escaláveis e eficientes em termos de consumo de energia. Desta forma, esta tese propõe um protocolo multicanal assíncrono iniciado pelo receptor de acesso ao meio para redes de sensores sem fio de baixa potência para aplicações de internet das coisas. Através de uma comparação com soluções já existentes, o protocolo apresentado procura mitigar a colisão de mensagens e as perdas de energia com ociosidade na espera pela comunicação de um transmissor, através de um mecanismo de reconhecimento inicial eficiente. Adicionalmente, é proposto um diagnóstico efetivo de detecção de falha na comunicação ainda no ciclo de comunicação, de forma a auxiliar a economia de energia. Complementarmente, é proposto um mecanismo multicanal baseado no conhecimento do canal da vizinhança, além de serviços de inicialização e manutenção da rede. Para validação da proposta, o protocolo proposto foi comparado tanto com protocolos assíncronos multicanais iniciado pelo receptor quanto com protocolo síncrono relevantes na literatura científica. Os critérios de avaliação utilizados foram medição do consumo, latência e taxa de entrega da rede em diferentes cenários. Os resultados mostraram que o protocolo proposto minimiza o consumo de energia em relação aos protocolos assíncronos, além de melhorar a comunicação quando comparado aos protocolos analisados. Na comparação com o protocolo síncrono, demonstrou desempenho e consumo compatíveis, quando em período de trabalho menor, e consumo reduzido com períodos de trabalho maiores.

Palavras-chave: MAC assíncrono iniciado pelo receptor. Redes de sensores sem fio. Controle de acesso ao meio assíncrono multicanal. Internet das Coisas.

ABSTRACT

FERNANDES Jr., R. F. **Asynchronous receiver-initiated media access protocol for wireless sensor networks**: 2018. 157 f. Thesis (Ph.D.) – Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2018.

The Internet of Things is considered a new communication system that promises to optimize and improve different application areas. It is based on sensor modules and intelligent objects only interconnected through the internet. In large-scale wireless sensor network applications, networks have own specific characteristics such as many low-power, limited-power sensor modules with intermittent communication losses. These networks need to operate with scalable, energy-efficient protocols. Thus, this thesis proposes an asynchronous multichannel receiver-initiated MAC protocol for low power wireless sensor networks and internet of things applications. Through a comparison with already existing solutions, the proposed protocol tries to mitigate message containment and the effect of idle listening through an efficient initial recognition mechanism. It is also proposed an effective diagnosis of communication failure detection in the communication cycle, which also helps to save energy. In addition, a multichannel mechanism is proposed based on the knowledge of the neighborhood channel in addition to services of initialization and maintenance of the network. To validate the proposed protocol, evaluations were made for the consumption of each node sensor, the network traffic for each link, the latency and the network delivery rate in a web application. Tests were performed using asynchronous multichannel receiver-initiated and synchronous protocols based on literature scientific. The results show that the proposed protocol minimizes the energy consumption in relation to the asynchronous protocols, besides improving the communication when compared to the analyzed protocols. In the comparison with the synchronous protocol the proposed protocol showed performance and consumption compatible, when in a smaller duty cycles, and reduced consumption with longer duty cycles.

Keywords: Asynchronous receiver-initiated MAC. Wireless sensor networks. Multichannel asynchronous media access control. Internet of Things.

LISTA DE ILUSTRAÇÕES

Figura 1 – Arquitetura de um sistema IoT	27
Figura 2 – Exemplo de um escalonamento de rede no protocolo TSCH.....	38
Figura 3 – Mecanismo de transmissão no protocolo X-MAC	42
Figura 4 – Comunicação do rádio entre dois módulos no protocolo RIT	43
Figura 5 – Exemplo de compressão do cabeçalho do 6LoWPAN.....	46
Figura 6 – Exemplo de uma rede utilizando protocolo RPL	48
Figura 7 – Exemplo de uma mensagem RPL.DIO encapsulado no pacote ICMPv6 ..	49
Figura 8 – Exemplo de comunicação do RPL	50
Figura 9 – Exemplo da arquitetura CoAP.....	53
Figura 10 – Exemplo da mensagem CoAP	54
Figura 11 – Exemplo de troca de mensagens no protocolo CoAP	54
Figura 12 - Comunicação entre dois módulos sensores no protocolo ARM	66
Figura 13 – Comunicação entre dois módulos no protocolo A-MAC.....	69
Figura 14 – Comunicação entre dois módulos no protocolo RITMC	75
Figura 15 – Exemplo de detecção de colisão no protocolo RITMC.....	79
Figura 16 – Comunicação de três vizinhos usando RITMC multicanal	81
Figura 17 – Algoritmo de transmissão e recepção do serviço de LiveList.....	84
Figura 18 – Algoritmo do serviço de mudança de melhor canal	86
Figura 19 – Cenário 1 dos serviços na comunicação do RITMC	88
Figura 20 – Cenário 2 dos serviços na comunicação do RITMC	89
Figura 21 – Cenário 3 dos serviços na comunicação do RITMC	90
Figura 22 – Arquitetura OpenWSN de IoT	91
Figura 23 – Arquitetura do Firmware do OpenWSN com o RITMC.....	94
Figura 24 – Diagrama de Estados do serviço de transmissão do RITMC	99
Figura 25 – Diagrama de Estados do serviço de recepção do RITMC.....	101
Figura 26 – Algoritmo unslotted CSMA/CA.....	103
Figura 27 – Captura do envio do H utilizando CSMA	104
Figura 28 – Envio de dados com e sem CSMA no protocolo RITMC.....	104
Figura 29 – Hardware dos módulos sensores utilizados no projeto	108
Figura 30 – Ambiente Web de aplicação do OpenWSN	109
Figura 31 – Exemplo de aplicação e logs utilizado nos testes.....	110

Figura 32 – Diagrama de blocos da comunicação entre aplicação e a rede IEEE802.15.4	111
Figura 33 – Cenário de teste dos protocolos assíncronos com 6 módulos	113
Figura 34 – Cenário de teste dos protocolos assíncronos com até 16 módulos e somente 1 caminho para cada módulo	114
Figura 35 – Cenário de teste da rede com até 11 módulos e dois caminhos para cada módulo	115
Figura 36 – Cenário com somente um caminho para cada módulo e diferentes obstáculos	116
Figura 37 – TxHello com RxIL. CH1 - tensão em mV e CH2 - estados do rádio.....	118
Figura 38 – Transmissão de 127 bytes no protocolo RITMC. CH1(acima) - consumo e CH2 (abaixo) - estados do rádio	119
Figura 39 – Recepção de 127 bytes de dados no protocolo RITMC. CH1(acima) - consumo e CH2 (abaixo) - estados do rádio	120
Figura 40 – Comunicação dois módulos a cada 1s no protocolo RITMC	125
Figura 41 – Comparação entre energia medida e estimada para protocolo RITMC ..	125
Figura 42 –Consumo medido e estimado para período de dados de 2s no protocolo RITMC	127
Figura 43 – Gráfico do custo de Energia para transmissão de dados para os diferentes protocolos	128
Figura 44 – Consumo de transmissão e recepção de um RPL.DIO no protocolo TSCH	129
Figura 45 – Consumo do rádio no protocolo TSCH em um período de RPL.DIO de 2s	130
Figura 46 – Comparação consumo medido e calculado no protocolo TSCH.....	131
Figura 47 – Comparação consumo entre RITMC e TSCH.....	132
Figura 48 – Avaliação da taxa de entrega da rede com canal único.....	134
Figura 49 – Análise da variância com canal único para período de 3000ms	134
Figura 50 – Avaliação da taxa de entrega da rede com multicanal	135
Figura 51 – Análise da Variância dos protocolos assíncronos multicanal com seis módulos por canal.....	136
Figura 52 –Taxa de entrega de pacotes com e sem diagnóstico no protocolo RITMC.	137
Figura 53 – Análise da variância da taxa de entrega com e sem diagnóstico.....	138

Figura 54 – Taxa de erro das mensagens enviadas pelo receptor.....	138
Figura 55 – Taxa de entrega e taxa de erro com período fixo de 1000ms.....	139
Figura 56 – Latência da rede em aplicação IoT no cenário com 6 módulos.	141
Figura 57 – Taxa de entrega em Aplicação IoT no cenário com 6 módulos	141
Figura 58 – Latência da rede em Aplicação IoT no cenário com 16 módulos	142
Figura 59– Taxa de entrega de aplicação IoT no cenário com 16 módulos	143
Figura 60 – Latência da aplicação IoT em um cenário com 11 módulos	144
Figura 61 – Taxa de entrega da aplicação IoT em um cenário com 11 módulos	146
Figura 62 – Roteamento do protocolo RITMC com 11 módulos sensores	147
Figura 63 – Latência da rede com protocolo RITMC com até 12 saltos	147
Figura 64 – Taxa de entrega da rede com protocolo RITMC com até 12 saltos	148
Figura 65 – Ocupação de canais em uma rede com 10 módulos.....	150
Figura 66 – Ocupação de canais em uma rede com 50 módulos.....	151
Figura 67 – Ocupação de canais em uma rede com 200 nós e 15 canais	151

LISTA DE TABELAS

Tabela 1 - Pilhas de protocolos da rede WSN para IoT.....	32
Tabela 2 - Principais protocolos iniciados pelo receptor pesquisados.....	71
Tabela 3 - Identificação e tratamento dos diversos casos do evento ECW	78
Tabela 4 - Campos da mensagem de ECW.....	79
Tabela 5 - Campos da mensagem de LiveList.....	83
Tabela 6 - Campos da mensagem de Mudança de Canal	87
Tabela 7 – Custo de Energia para transmissão e recepção dos protocolos assíncronos	120
Tabela 8 – Cálculo Tempo RxWaitTimeout.....	122
Tabela 9 – Energia média em diferentes períodos de RIT para protocolo RITMC....	126
Tabela 10 – Medição do consumo de cada serviço dentro do slot de tempo no protocolo TSCH.....	129
Tabela 11 – Resulta da simulação de cada módulo na 1000a iteração	150

LISTA DE ABREVIATURAS E SIGLAS

ACE	Autenticação e Autorização para Ambientes com Restrições (<i>Authentication and Authorization for Constrained Environments</i>)
AES	Padrão Avançado de Criptografia (<i>Advanced Encryption Standard</i>)
ADB	Transmissão de ciclo de tarefas assíncronas (<i>Asynchronous duty cycle broadcasting</i>)
AL-MAC	MAC de conexões assimétricas (<i>Asymmetric Links MAC</i>)
A-MAC	MAC assíncrono (<i>Asynchronous MAC</i>)
AMCA	Adaptação multi-canal assíncrona (<i>Asynchronous Multi Channel Adaptation</i>)
ARM	Assíncrono multicanal iniciado pelo receptor (<i>Asynchronous Receiver-Initiated Multi-Channel</i>)
ARQ	Requisição de repetição automática (<i>Automatic Repeat-Request</i>)
ARIB	Associação de indústrias e empresas de rádio (<i>Association of Radio Industries and Businesses</i>)
ATIS	Aliança para as soluções da indústria de telecomunicações (<i>Alliance for Telecommunications Industry Solutions</i>)
BW	Janela de recuo (<i>Backoff window</i>)
CC	Canal de controle (<i>Channel Control</i>)
CCA	Limpeza da avaliação do canal (<i>Clear Channel Assessment</i>)
CCSA	Associação de normas de comunicação chinesa (<i>China Communications Standards Association</i>)
CoAP	Constrained Application Protocol
CORE	Ambientes RESTful restritos (<i>Constrained RESTful Environments</i>)
CSMA/CA	Sentido da transportadora com múltiplos acessos com evitação de colisão (<i>Carrier Sense Multiple Access with Collision Avoidance</i>)
CW	Janela de contenção (<i>Contention Window</i>)
DA	Reconhecimento de dados (<i>Data-Ack</i>)
DAG	Grafo acíclico direto (<i>Directed Acyclic Graph</i>)

DAO	DODAG Objeto de Anúncio de Destino (<i>DODAG Destination Advertisement Object</i>).
DB-MAC	MAC de atraso limitado (Delay bounded MAC)
DC	Canal de Dados (<i>Data Channel</i>)
DICE	DTLS em ambientes restritos (<i>DTLS In Constrained Environments</i>)
DIO	DODAG de objeto de informação (<i>DODAG Information Object</i>)
DIS	DODAG de solicitação de informação (<i>DODAG Information Solicitation</i>)
DODAG	Grafo acíclico direto orientado ao destino (<i>Destination Orientated Directed Acyclic Graph</i>)
DTLS	Segurança da Camada de Transporte do Datagrama (<i>Datagram Transport Layer Security</i>)
ECW	Janela de contenção de erro (<i>Error Contention Window</i>)
ED	Energia detectada (<i>Energy Detected</i>)
EE-RI-MAC	Energy Efficient RI-MAC
EH-MAC	Enhancement MAC
EM-MAC	Efficient Multi-channel MAC
ERI-MAC	Energy Harvested Receiver Initiated MAC
ETSI	Instituto de padronização de telecomunicações Europeu (<i>European Telecommunication Standards Institute</i>)
EXI	Efficient XML Interchange
H	Olá (Hello)
HA	Reconhecimento do olá (Hello Ack)
HTTP	Protocolo de transferência de hipertexto (<i>Hypertext Transfer Protocol</i>)
HTTPS	Segurança de protocolo de transferência de hipertexto (<i>Hyper Text Transfer Protocol Secure</i>)
IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos (<i>Institute of Electrical and Electronic Engineers</i>)
IETF	Força-tarefa de engenharia da Internet (<i>Internet Engineering Task Force</i>)
IL	Escuta ociosa (<i>Idle Listening</i>)

IoT	Internet das coisas (<i>Internet of Things</i>)
IP	Protocolo de Internet (<i>Internet Protocol</i>)
ISM	Industrial, científico e médico (<i>Industrial, Scientific e Medical</i>)
ITU-T	União de telecomunicações internacional (<i>International Telecommunication Union</i>)
JSON	JavaScript Object Notation
LBR	Low-Power and Lossy Network Border Router
LL	Lista de vivos (<i>LiveList</i>)
LNA	Amplificador baixo ruído (<i>Low Noise Amplifier</i>)
LPL	Baixa potência de escuta (<i>Low Power Listening</i>)
LQI	Indicador de qualidade de conexão (<i>Link Quality Indicator</i>)
M2M	Máquina para máquina (<i>Machine to Machine</i>)
MAC	Controle de acesso ao meio (<i>Media Access Control</i>)
MHR	Cabeçalho MAC (<i>MAC Header</i>)
MPLS	Comutação de rótulo multi-protocolo (<i>Multi Protocol Label Switching</i>)
MR-FSK	Chaveamento de Mudança de Frequência Multi-Regional e Multi-Regional (<i>Multi-Rate e Multi-Regional Frequency Shift Keying</i>)
MR-OFDM	Multiplexação por Divisão de Frequências Ortogonais Multi-Regionais e Multi-Regionais (<i>Multi-Rate e Multi-Regional Orthogonal Frequency Division Multiplexing</i>)
MR-O-QPSK	Multiplexação por Divisão de Frequências Ortogonais Multi-Regionais e Multi-Regionais (<i>Multi-Rate e Multi-Regional Offset-Quadrature Phase-shift Keying</i>)
NFC	Comunicação por campo de proximidade (<i>Near-Field communication</i>)
OC-MAC	MAC de Cooperação oportunista (<i>Opportunistic Cooperation MAC</i>)
OMA	Aliança de celular aberto (<i>Open Mobile Alliance</i>)
PCE	Entidade de Computação de caminho (<i>Path Computation Entity</i>)
PRADC-MAC	<i>Pseudo-random asynchronous duty cycle MAC</i>
PW-MAC	Despertar preditivo (<i>Predictive Wake-up</i>)

QoS	Qualidade de serviço (<i>Quality of Service</i>)
RC-MAC	MAC centralizado no receptor (<i>Receiver-Centric MAC</i>)
REST	Transferência de Estado Representacional (<i>Representational State Transfer</i>)
RFC	Requisição para comentários do IETF (<i>Request for Comments</i>)
RFID	Identificação por rádio frequência (<i>Radio-Frequency Identification</i>)
RI-MAC	Receiver-Initiate MAC
RIT	Transmissão iniciada pelo receptor (<i>Receiver Initiated Transmitter</i>)
RITMC	Controle de acesso ao meio com transmissão multicanal iniciado pelo receptor (<i>Receive Initiate Multichannel MAC</i>)
ROLL	Roteamento por rede com baixo consumo de energia e com perdas (<i>Routing Over Low-Power and Lossy network</i>)
RPL	Protocolo de Roteamento para Redes de Baixa Potência e Perdas (<i>Routing Protocol for Low Power and Lossy Networks</i>)
RSVP	Protocolo de reserva do recurso (<i>Resource Reservation Protocol</i>)
WSN	Rede de sensor sem fio (<i>Wireless Sensor Network</i>)
RSSI	Indicador de força do sinal recebido (<i>Received Signal Strength Indicator</i>)
RSSF	Rede de sensor sem fio
RSVP	Protocolo de reserva de recurso (<i>Resource Reservation Protocol</i>)
RTS/CTS	Retorno para o envio/Limpeza para o envio (<i>Return to Send / Clear to Send</i>)
RW-MAC	<i>Receiver Wake-up MAC</i>
RxIL	Escuta ociosa na recepção (<i>Rx Idle Listening</i>)
SARI-MAC	<i>MAC de inicialização pelo receptor com auto adaptação (Self Adapting Receiver Initiated MAC)</i>
SFD	Delimitador de início de mensagem (<i>Start Frame Delimiter</i>)
SMT	Transmissor múltiplo simultâneo (<i>Simultaneous Multiple Transmitter</i>)
SOA	Arquitetura orientada a eventos (<i>Service-Oriented Architecture</i>)
SRI-MAC	Synchronous Receiver-Initiated MAC

TA	Tempo do Ack (<i>Time Ack</i>)
TCP	Protocolo de controle de transmissão (<i>Transmission Control Protocol</i>)
TD	Atraso do tempo (<i>Time delay</i>)
TDMA	Acesso múltiplo de divisão de tempo (<i>Time Division Multiple Access</i>)
TI	Tecnologia da Informação
TSCH	Salto de canal com slot de tempo (<i>Time Slot Channel Hopping</i>)
TTA	Associação de tecnologia de telecomunicação (<i>Telecommunications Technology Association of Korea</i>)
TTC	Comitê de Tecnologia de Telecomunicações (<i>Telecommunications Technology Committee</i>)
TxIL	Escuta ociosa na transmissão
UANs	Redes Acústicas Subaquáticas
UDP	Protocolo de datagrama do usuário (<i>User Datagram Protocol</i>)
URI	Uniform Resource Identifier
WLAN	Redes locais sem fio (<i>Wireless Local Area Network</i>)
WPAN	Redes pessoais sem fio (<i>Wireless Personal Area Network</i>)
WWAN	Rede extensão sem fio (<i>Wireless Wide Area Network</i>)
XML	Extensible Markup Language
YA-MAC	Yet Another MAC
6LoWPAN	IPv6 sobre WPAN de baixa energia (<i>IPv6 over Low power Wireless Personal Area Networks</i>)
6TiSCH	IPv6 sobre TSCH (<i>IPv6 over the TSCH</i>)

LISTA DE SÍMBOLOS

E	Número de detecções esperada
K	Número de slots da janela de contenção
N	Número de competidores
f	Fator de velocidade
λ	parâmetro de ajuste de carga medido por cada módulo em pacotes/segundo
e	Fator de erro de pacotes
E_b	Energia consumida pelo envio da anúncio
E_w	Energia consumida pelo transmissor pela espera da anúncio do receptor
E_{tx}	Energia gasta para envio de dados
Rank	Qualidade do canal
NV	Número de vizinhos no canal
TOTV	Número total de vizinhos vivos
TER	Taxa de erro de pacotes
E_{Total}	Energia total consumida
ET_{RxIL}	o custo de energia total para anúncio com timeout
ET_{Tx}	custo total para transmissão
ET_{Rx}	custo total para recepção
f_H	Frequência de mensagem hello
E'_{RxIL}	custo de energia médio de uma anúncio com IL
E'_{TxIL}	custo de energia média da espera de uma comunicação do receptor
f_{Tx}	Frequência de transmissões no período
f_{Rx}	Frequência de recepções no período
E'_{Tx}	custo de energia médio da transmissão
E'_{Rx}	custo de energia médio da recepção

E'_{TxD}	energia consumida pelo restante do serviço de transmissão após terminado o TxIL
α	Constante de ajuste da frequência de IL
E_{TSCH}	Custo de energia do protocolo E_TSCH
f_{SIL}	Frequência do IL no período
f_{SRxDIO}	Frequência de recepção da mensagem RPL DIO
f_{STxDIO}	Frequência de transmissão da mensagem RPL DIO
E_{SIL}	Custo de energia médio de um slot em IL
E_{SRxDIO}	Custo de energia médio de um slot recebendo uma mensagem RPL.DIO
E_{STxDIO}	Custo de energia médio de um slot transmitindo uma mensagem RPL.DIO
P_{tx}	período da transmissão
P_M	período do macrociclo
$NRSlots$	Número de slots

SUMÁRIO

1 INTRODUÇÃO	19
1.1 Objetivos da tese	21
1.2. Organização da tese	22
1.3. Trabalhos Publicados	22
2 REDES DE SENSORES SEM FIO PARA IOT	23
2.1 Definição de um sistema IoT	23
Arquitetura de rede	26
Especificações de rede	28
2.2 Protocolos de uma WSN utilizando tecnologia IEEE802.15.4	31
Camada física	32
Camada MAC	35
Camada de Adaptação	44
Camada de Rede	47
Camada de Transporte	51
Camada de Aplicação	52
Segurança da rede	55
3 ESTADO DA ARTE DOS PROTOCOLOS ASSÍNCRONOS BASEADOS NA TRANSMISSÃO INICIADA PELO RECEPTOR	57
3.1 Protocolos RIT de canal único	59
3.2 Protocolos RIT multicanal	65
3.3 Comentários	71
4 PROPOSTA PROTOCOLO RITMC	75
4.1 Mecanismo básico de acesso ao meio	75
4.2 Mecanismo de Tratamento de Erro	77
Identificação de um Evento de ECW	77
Tratamento do Evento de ECW	78

4.3 Mecanismo de tratamento Multicanal.....	80
4.4 Serviços Multicanal	82
5 MÉTODOS E MATERIAIS	91
5.1 Arquitetura do OpenWSN	91
5.2 Pilha de Protocolos do OpenWSN.....	93
5.3 Implementação do mecanismo do CSMA/CA	102
5.4 Limitação da camada de Rede do OpenWSN.....	105
5.5 Plataforma de Hardware	106
5.6 Plataforma de Software.....	108
5.7 Aspectos de configuração dos cenários de teste	111
5.8 Metodologia de validação	112
6 RESULTADOS	117
6.1 Análise do consumo.....	117
Consumo protocolos assíncronos	117
Consumo protocolo síncrono	128
6.2 Taxa de entrega dos protocolos assíncronos.....	133
6.3 Diagnóstico do protocolo RITMC	136
6.4 Aplicação de IoT.....	139
Aplicação protocolos assíncronos.....	140
Aplicação protocolo síncrono	143
6.5 Mecanismo multicanal.....	148
6.6 Comentários e discussões	152
7 CONCLUSÃO	155
Trabalhos futuros	157
REFERÊNCIAS	159

1 INTRODUÇÃO

Um crescente interesse sobre o tema de Internet das Coisas (IoT) emergiu como um novo paradigma de tecnologia no qual se prevê que bilhões de elementos, entre sensores, atuadores serão interconectados (GAZIS, 2017). As redes de comunicação para esta finalidade estão sendo desenvolvidas ativamente e suportarão um grande número de nós utilizando sensores de baixo custo espalhados por uma extensa área e atendendo diversos cenários de aplicações. Muitas destas aplicações são voltadas ao ambiente onde vivemos e tem como objetivo melhorar a qualidade de vida como aplicações residenciais, prediais, hotelaria, aplicações urbanas de segurança no tráfego de locomoção, aplicações de saúde, trabalho, atividades físicas, aplicações de diagnóstico e manutenção em ambientes industriais e urbanos, entre várias outras aplicações (GAZIS, 2017; ALVI, et al., 2016 ; YAQOOB et al., 2017).

A IoT está baseada em uma grande rede de objetos exclusivamente endereçáveis e de serviços que podem se interagir com estes objetos (PALLATELA, et.al, 2013). Estas redes tiveram início com a tecnologia RFID, que atualmente é utilizada em inúmeros segmentos como logístico, farmacêutico, indústria, entre outros (LI, XU e ZHAO, 2015). Posteriormente ao RFID, a rede IoT englobou também as redes sem fio com o objetivo de processar sensores em uma escala maior. Atualmente, uma rede IoT engloba muitas tecnologias, como redes de sensor sem fio (WSN), leitores de código de barra, sensores como RFID, NFC, comunicação de baixo consumo, computação nas nuvens, *big data*, entre outras, e é objeto de estudo de centros de pesquisas do mundo todo (GAZIS, 2017).

A arquitetura de uma IoT envolve muitos papéis como infraestrutura, comunicação, interfaces, protocolos e padrões. Com relação às tecnologias de IoT, Li, Xu e Zhao (2015) declaram que as soluções ainda estão em um estágio inicial de definição e desenvolvimento, porém existe um grande esforço de empresas no desenvolvimento de tecnologias e soluções nos mais diversos níveis.

As redes de IoT dependem do cenário de uso e dos requisitos da aplicação, como confiabilidade, taxa de entrega, segurança e consumo (CONTI E GIORDANO, 2014). No cenário de aplicações de redes urbanas, por exemplo, segundo o documento RFC 5548 (DOYLER et.al., 2009), os requisitos para uma rede urbana são definidos por módulos sensores e atuadores posicionados em ambientes urbanos externos com o objetivo de melhorar a condição de vida da população, assim como para fiscalizar o cumprimento de leis ambientais. A operação esperada dos módulos é a medição e o relatório de uma ampla gama de dados como,

por exemplo, dados extraídos de aplicações que executam medição inteligente, que monitoram condições meteorológicas, poluição, entre outros. A maioria dos módulos se comunica por meio de tecnologias de rede sem fio, porém são capazes de formar redes em larga escala com a utilização de protocolos de roteamento adequados. O projeto destes protocolos deverá ser, principalmente, influenciado pelos limitados recursos dos módulos tais como memória, capacidade de processamento, bateria, e particularidades dos ambientes de aplicações urbanas. Desta forma, para a solução de roteamento em redes com perdas de comunicação intermitentes e de módulos de baixo consumo, tais redes precisam operar com protocolos escaláveis, eficientes em termos de consumo e autônomos.

Para as WSN, o aumento da eficiência energética tornou-se um critério chave de projeto. Em particular, a eficiência de energia na comunicação entre nós sensores alimentados por baterias é essencial, pois o rádio é um dos elementos que mais consome energia (KIM, et al., 2015).

No contexto de padronização das tecnologias de IoT, instituições como *Internet Engineering Task Force (IETF)* (WINTER, et.al, 2012; HUI, 2012; PUNNET, 2013), *European Telecommunication Standards Institute (ETSI)* (ETSI, 2013; ETSI, 2014), e também o IEEE, tem movido esforços para desenvolver padrões de comunicação e viabilizar os diferentes cenários de aplicações de IoT. Especificamente para WSN, as tecnologias IoT utilizam principalmente o padrão IEEE802.15.4, que define as camadas física e de acesso ao meio (MAC) de rede. Para as camadas superiores, devido aos equipamentos da WSN possuírem características de baixo consumo e baixa potência, é necessária a utilização de uma tecnologia internet baseada no protocolo IPV6 com características reduzidas de rede chamada 6LoWPAN. Para as tecnologias da WSN, a camada de transporte é baseada nos protocolos TCP e UDP. Para possibilitar a comunicação web por meio de aplicativos já existentes adotou-se um padrão de comunicação para as camadas superiores chamado de CoAP, correspondente ao HTTP na rede internet (PALLATELLA et al., 2013; GAZYS, 2017).

Dentro da pilha de protocolos de uma WSN a camada MAC visa otimizar a taxa de entrega da rede, diminuir atrasos de comunicação e, principalmente, reduzir o consumo de energia. Os protocolos MAC para WSN podem ser divididos em dois grupos principais: protocolos síncronos, onde a rede é gerenciada por um mecanismo rígido de sincronismo, e protocolos assíncronos, onde os módulos sensores atuam independentemente de sincronismo. Os protocolos assíncronos podem suportar a adaptação local para uma carga de tráfego variável em diferentes partes da rede, sem depender de um escalonamento global de transmissão. Também os protocolos assíncronos possibilitam mecanismos de menor consumo de energia

permitindo que os dispositivos comuniquem-se mantendo baixos ciclos de trabalho. Desta forma, os protocolos assíncronos são mais indicados para redes com características de consumo de energia reduzido e escalabilidade de módulos sensores (CANO et al., 2011; DUTTA et al., 2012). Entre os protocolos assíncronos da camada MAC os iniciados pelo receptor (RIT) podem ter um ganho maior de economia de energia do que os métodos iniciados pelo transmissor devido ao fato dos protocolos iniciados pelo receptor somente comunicar-se com o vizinho quando o vizinho anunciar o início da comunicação (PEREZ, et.al., 2015; ZHUO et.al, 2016).

Quanto à característica de comunicação das redes IoT, as WSN são peças essenciais na arquitetura da IoT devido à flexibilidade e à facilidade proporcionadas. Porém a comunicação sem fio enfrenta problemas devido a sua natureza, e que dificulta a comunicação. Entre os problemas enfrentados pela comunicação sem fio se destacam a interferência do canal, devido à coexistência com outros equipamentos comunicando na mesma frequência, espelhamento e reflexão que ocorrem em ambientes internos, diminuição da atuação devido a obstáculos, além das características intrínsecas do rádio. No caso de utilizar somente um canal, a comunicação pode ser um gargalo onde qualquer problema pode inviabilizá-la. Uma rede sem fio geralmente prevê mais de um canal de comunicação, onde para o protocolo IEEE802.15.4 comunicando a 2.4 GHz são 16 canais disponíveis. Para garantir os requisitos de qualidade de serviço, latência, robustez da rede, os protocolos WSN devem comunicar em ambiente multicanal (PALATTELA, et.al., 2013; NAMBOOTHIRI, SIVALINGAM, 2013; ZHUO et.al., 2016).

1.1 Objetivos da tese

O objetivo principal que caracteriza a tese é a análise de viabilidade das tecnologias de rede sem fio em aplicações de IoT com baixa transferência de dados e baixo consumo e que não possuem características rígidas de determinismo, usando protocolos assíncronos de acesso ao meio. Para isso é proposto um protocolo multicanal assíncrono de camada MAC iniciado pelo receptor para WSN chamado de RITMC, que está baseado em algoritmos existentes na literatura científica. O protocolo RITMC propõe um mecanismo de comunicação que minimiza o consumo de energia por meio do reconhecimento inicial, com um mecanismo multicanal e um diagnóstico mais robusto.

Para o atendimento a estes objetivos, será analisada uma rede de IoT com padrão aberto em seus diferentes níveis. Será considerada, desde a rede de sensores sem fio até o nível de aplicação remota usando tecnologias web e redes com protocolo IPV6. Para isso, será utilizada

a plataforma do OpenWSN, que é um ambiente de software aberto, e utiliza as principais tecnologias de IoT encontradas na literatura.

Para proceder à análise, será necessário o desenvolvimento de estratégias de comunicação que deverão seguir os padrões vigentes de IoT. Como objetivos específicos, propõe-se os itens abaixo:

- Proposta de um protocolo de camada de acesso ao meio assíncrono e multicanal para redes de larga escala com requisitos de economia de energia, além de latência e taxa de entrega compatíveis com o cenário de uso;
- Validação da proposta multicanal e estudo comparativo do protocolo proposto com outros protocolos multicanais de acesso ao meio.

1.2. Organização da tese

Esta tese será dividida em sete capítulos e está estruturada da seguinte forma: O primeiro capítulo é destinado a situar o leitor ao contexto do trabalho e apresentação dos objetivos. O segundo capítulo visa descrever os conceitos de IoT utilizando WSN, tendo como base as características do padrão IEEE802.15.4 e outras especificações correlatas. No capítulo três é realizado o estado da arte dos protocolos assíncronos de camada MAC iniciados pelo receptor, que é o tema desta tese. No quarto capítulo, são apresentadas, detalhadamente as estratégias desenvolvidas nesta tese. O capítulo cinco descreve as metodologias de desenvolvimento e de validação dos mecanismos propostos. No sexto capítulo são apresentados todos os resultados obtidos. No capítulo sete são apresentados a conclusão e sugestões de trabalhos futuros.

1.3. Trabalhos Publicados

FERNANDES, R. F.; DE ALMEIDA, M. B.; BRANDÃO, D. “An Energy Efficient Receiver-Initiated MAC Protocol for Low-Power WSN.”, *Wireless Personal Communications*, p.1517-1536 - 1536, 2018

BERTELLI, G.; SANTOS, A.; SILVA, I.; FERNANDES, R.; Brandao, D.; MULLER, I.; NETTO, J.; WINTER, J.; PEREIRA, C. E. “Research activities on industrial wireless instrumentation: Brazilian perspective.” *IEEE INSTRUMENTATION & MEASUREMENT MAGAZINE*, v.20, p.21 - 30, 2017.

FERNANDES, R. F.; BRANDÃO, D. “Proposal of Receiver Initiated MAC Protocol for WSN in urban environment using IoT”, *IFAC-PAPERSONLINE*, v.49, p.102 - 107, 2016

2 REDES DE SENSORES SEM FIO PARA IOT

Este capítulo destina-se a descrever as definições, a arquitetura, e os padrões envolvidos em uma rede IoT. Para isso, ele é dividido em duas seções. A primeira seção aborda as características gerais e os níveis básicos de um sistema IoT. Na seção seguinte será mostrada a pilha de protocolos padrões dentro de uma rede de sensor sem fio utilizando padrão IEEE802.15.4, que é o tema de estudo desta tese.

2.1 Definição de um sistema IoT

Atualmente, a IoT é considerada um novo sistema de comunicação onde a internet está conectada ao mundo físico por meio de uma ampla rede de dispositivos autônomos distribuídos espacialmente. Com o desenvolvimento das tecnologias IoT, dispositivos com microcontroladores de tamanho reduzido foram implantados de forma massiva em uma variedade de aplicações, e várias alianças de padronização se formaram com base nos interesses de tecnologia e dos mercados comerciais (SHENG et.al., 2013). Geralmente, os dispositivos compartilham características comuns, como recursos de energia restrita, capacidade de processamento limitada, condições de rádio vulneráveis, a natureza em tempo real das aplicações e a pouca necessidade de interação humana. Ao interconectar esses dispositivos usando tecnologias de comunicação sem fio de baixo custo como WSN, é formado um novo ambiente com grande potencial de aplicações com diferentes naturezas utilizando a mesma base de equipamentos (YAQOOB et.al., 2017).

Outro conceito similar a IoT é a comunicação entre máquinas (M2M - *Machine to Machine*). O M2M trata-se da comunicação entre um dispositivo conectado à rede e uma central de controle. Um sistema IoT utiliza a comunicação M2M para permitir a agregação de valor em aplicações mais complexas como tecnologias de nuvem, computação cognitiva, big data (GAZYS, 2017).

As aplicações de IoT podem ser utilizadas em diferentes cenários, como aplicações urbanas na área de saúde, transporte, monitoramento urbano por meio de câmeras, sensores, aplicações automotivas e de mobilidade como segurança e monitoramento de tráfego, aplicações na agricultura, pecuária através de monitoramento de plantações, automação residencial e predial por meio de monitoramento, segurança e conforto das instalações, automação industrial através do monitoramento, controle e segurança de processos e ambientes

industriais internos e externos, entre outras (YAQOOB et.al., 2017). Estes diferentes cenários de uso estão relacionados ao acesso fácil que a tecnologia IoT proporciona na comunicação de uma variedade de equipamentos como sensores, atuadores, displays, câmeras, etiquetas de identificação, entre outros (LI et al., 2015; ZANELLA et.al., 2015).

Em relação à tecnologia empregada, o IoT possui uma estrutura de rede flexível e autoconfigurável baseada em protocolos padrões de comunicação que permitem uma maior interoperabilidade. A tecnologia permite que sensores, atuadores e qualquer outro objeto sejam conectados à internet (YAQOOB et.al., 2017).

Segundo Li et.al. (2015) um dos problemas da IoT é que a rede de equipamentos precisa ser interligada dentro de uma estrutura onde seja garantida a correta operação da rede, de forma amigável e com qualidade de serviço. Esta interligação envolve várias camadas de software e hardware. Para atender estes requisitos de projeto devem ser considerados a extensibilidade, a escalabilidade, e a interoperabilidade entre diferentes equipamentos.

A característica da comunicação IoT introduz uma série de desafios de rede. A maioria dos aplicativos e cenários baseados em comunicações IoT geralmente envolvem uma grande quantidade de dispositivos. Uma questão fundamental é o gerenciamento eficiente de recursos de rede. O desenvolvimento de um projeto de IoT deve ser designado com um conjunto de requerimentos satisfatórios para cobrir as necessidades dos diferentes cenários de aplicações. Os principais requerimentos para o projeto de uma rede IoT são: (PALATTELA et al., 2013; KHALIFA et al., 2011; RAJANDEKAR et al., 2015, YAQOOB et.al., 2017):

- **Taxa de entrega:** uma rede de informação usando IoT deve ter uma adequada taxa de entrega para atender as aplicações de forma eficiente. Devido às peculiaridades da rede sem fio, com canais de comunicação limitados, uma quantidade grande de equipamentos acessando o mesmo meio físico, e uma arquitetura com eventuais obstáculos, é importante que os protocolos de rede minimizem os atrasos devido às colisões, ruídos, perdas de dados, e controle da troca de informação;
- **Confiabilidade:** as aplicações de Internet utilizando redes sem fio necessitam que a informação seja íntegra e confiável. Também, o fato da falta de acessibilidade de alguns ambientes torna-se difícil a substituição de dispositivos danificados ou sem energia. Por este motivo, é necessário providenciar módulos sensores com capacidades de adaptação ao meio para poderem lidar com situações em que o ambiente sofra alterações. Em relação às falhas ou danos permanentes de dispositivos, a rede deve ser tolerante e buscar formas alternativas para atingir o objetivo;

- **Consciência energética:** geralmente, uma rede de baixo custo e baixo consumo possui restrição de recursos, e desta forma, uma consciência quanto ao consumo de energia ajuda na otimização destes recursos. Em alguns casos, quando a carga da rede não é pesada, os dispositivos devem colocar-se no modo de suspensão ou de inatividade. Portanto, é importante que se tenha um protocolo de comunicação e um hardware que sejam conscientes energeticamente;
- **Acesso à internet:** a internet tem um papel primordial na rede IoT. No contexto das redes M2M, a internet está requisitando novas características com comunicação bidirecional. Desta forma, é extremamente importante que os protocolos permitam esta comunicação entre objetos de forma facilitada. Além disso, a internet deve permitir (por meio do conceito de nuvem) que qualquer nó da rede, em qualquer lugar, possa ser acessado através de uma linguagem universal de rede, que no caso seria o IP. Este requisito se debruça na utilização extensiva de padrões de comunicação;
- **Controle de recursos:** os dispositivos microprocessados que participam de um ambiente IoT devem ser acessíveis e configuráveis de maneira remota. Em algumas situações, quando os administradores não estão disponíveis em seus locais particulares, o controle dos recursos de fora pode ajudar a resolver o problema. Além disso, os sistemas IoT devem ser capazes de equilibrar a carga em caso de disponibilidade redundante de recursos, o que pode levar a uma utilização adequada dos recursos;
- **Segurança:** o sistema IoT deve ser suficientemente seguro para impedir que os dispositivos sejam acessados por elementos externos não autorizados. A segurança deve atingir os diferentes níveis da arquitetura do sistema. A falta de segurança no ambiente IoT pode fazer com que diminua a confiança dos usuários no sistema. A diversidade de aplicativos e a heterogeneidade das infraestruturas de comunicação da IoT resultam em uma variedade igualmente numerosa de desafios de segurança;
- **Latência:** a latência reflete para as aplicações que utilizam deste tipo de comunicação a efetividade e a utilidade dos serviços oferecidos pela rede. Algumas aplicações podem ser sensíveis à latência e necessitar de tempos de resposta curtos, em torno de milissegundos (como, por exemplo, controle de processos industriais rápidos). Outras não são tão dependentes do tempo de reação (por exemplo, monitoração da estabilidade de uma ponte). Desta forma, a rede deverá atender de forma satisfatória aos requisitos temporais da aplicação;
- **Escalabilidade:** no contexto de comunicação sem fio em redes urbanas, um ponto chave é a escalabilidade da rede. O aumento da densidade de módulos sensores é diretamente

proporcional ao aumento de cenários de aplicações. Deve-se também considerar a dinâmica da rede, onde módulos sensores podem estar entrando e saindo da rede. Desta forma, é imprescindível que o protocolo de rede consiga administrar tanto um aumento na densidade da rede como a dinâmica da rede, na qual a informação deve conseguir chegar ao destino mesmo com a mudança nas rotas;

- **Qualidade de serviço:** um dos requisitos das arquiteturas IoT é que eles devem poder fornecer serviços de qualidade aos usuários. A QoS no IoT pode ser assegurada priorizando os serviços e a recuperação. Os aplicativos que exigem processamento em tempo real devem ter alta prioridade para melhorar seu desempenho. Além disso, em resposta a uma consulta, devem ser processadas apenas as informações necessárias;
- **Coexistência:** atualmente as redes sem fio operam nas bandas não licenciadas do espectro, por motivos como custos de produtos associados e da facilidade de utilização da banda. Porém, com o aumento de aplicações e de equipamentos utilizando o mesmo meio físico com redes independentes, haverá a concorrência pelo espaço das bandas não licenciadas. Portanto, as redes WSN devem coexistir com outras redes do mesmo tipo e também com outros tipos de redes com diferentes codificações e sinais de rádio como Wifi e Bluetooth;
- **Interoperabilidade:** a comunicação de equipamentos de diferentes fabricantes é um requisito de extrema importância nas redes de sistemas abertos. A arquitetura deve suportar internet com comunicação transparente entre todos os dispositivos e aplicações de forma simples.

Arquitetura de rede

A rede IoT prevê que os equipamentos ou objetos estejam disponíveis dentro da rede para ser acessado por qualquer outro componente. Desta forma, é pressuposto que estes objetos estejam interligados. A arquitetura de um sistema de IoT precisa prover esta interligação entre os diferentes componentes (YAQOOB et.al., 2017).

A arquitetura de um sistema de IoT envolve muitos fatores como comunicação, segurança, negócios, processos. É necessário também haver a possibilidade dos elementos da rede estarem geograficamente distantes uns dos outros, e caso necessitem interagir com qualquer outro elemento do sistema seja possível a comunicação. A arquitetura da rede deve ser adaptativa e poder ser dinamicamente alterada. Adicionalmente, possuir processos

descentralizados e heterogêneos (YAQOOB et.al., 2017; ZANELLA et.al., 2014; LI et. Al., 2015).

Para atender estes requisitos, o sistema IoT tem uma arquitetura orientada a eventos (SOA - *Service-Oriented Architecture*). A arquitetura SOA divide um sistema complexo em objetos menores e bem definidos. Desta forma, estes objetos ou subsistemas permitem o reuso de componentes. A arquitetura SOA tem como característica o atendimento de requisitos, como escalabilidade, modularidade, interoperabilidade e intercambiabilidade (ZANELLA et.al., 2014; LI et. Al., 2015). Baseado neste conceito de SOA, a arquitetura de um sistema IoT pode ser dividida em quatro níveis principais: sensores, rede, serviços e aplicação. A Figura 1 mostra um exemplo desta arquitetura (LI et al., 2015).

Figura 1 – Arquitetura de um sistema IoT

Fonte adaptada: YAQOOB et.al. (2017)

No primeiro nível da arquitetura está o **nível de aplicação ou de interface** que está relacionada diretamente com o usuário final do sistema. Neste contexto, como IoT está relacionada com diferentes tecnologias de diferentes tipos de equipamentos, a interface fornece mecanismo de interconexão e o gerenciamento da comunicação entre estes equipamentos de forma simplificada e padronizada (LI et.al., 2015; ZANELLA et al., 2014).

No **nível de serviços** são as tecnologias de *middleware* que habilita os serviços e aplicações em IoT. Este *middleware* é formado por todos serviços de troca de informação e

armazenamento de dados, gerenciamento de dados e serviços de busca (ZANELLA et.al., 2014).

O **nível de rede ou de gateway** é constituído das tecnologias que servem de ponte entre as diversas redes de sensores e as aplicações. Desta forma, esta camada é capaz de fornecer serviços para a infraestrutura de tecnologia da informação (TI), onde esta informação pode ser acessada para unidades de processamento e tomada de decisão (LI et. al., 2015).

No nível mais baixo da arquitetura está a **rede de sensores**, que é a camada onde as informações são detectadas e podem ser acessadas remotamente. Nesta camada, os equipamentos possuem características de automaticamente tirar informações dos sensores e enviar para a rede para servir a outros equipamentos ou aplicações. Além disso, os objetos devem ser identificados unicamente dentro da rede. Atualmente, tem sido desenvolvido soluções de equipamentos e tecnologias de hardware e software para atender os requisitos de baixo consumo de energia e tamanho dos equipamentos e melhoria em performance.

Especificações de rede

Dentro dos vários níveis de comunicação de IoT, os protocolos devem existir para permitir a interoperabilidade entre os diferentes dispositivos e a comunicação de forma padronizada.

Para fomentar o desenvolvimento dos ecossistemas de IoT, várias atividades de padronização foram lançadas globalmente. A *International Telecommunication Union* (ITU-T) (ITU-T, 2012) discute as tecnologias, potenciais cenários de aplicação, desafios emergentes e as implicações da IoT. A organização oneM2M (oneM2M, 2014) procura garantir o alinhamento global dos padrões M2M. Entre os membros estão a Associação de Indústrias e Empresas de Rádio (ARIB), o Comitê de Tecnologia de Telecomunicações (TTC) do Japão, a Aliança para as Soluções da Indústria de Telecomunicações (ATIS), a Associação da Indústria de Telecomunicações (TIA), a Associação de Normas de Comunicação da China (CCSA), o Instituto Europeu de Normas de Telecomunicações (ETSI) e Associação de Tecnologia de Telecomunicações (TTA) da Coreia. Existe também uma interação com as principais alianças da indústria como Open Mobile Alliance (OMA, 2013) e organismos de padronização da Internet (IETF) (LI et.al., 2015; GAZIS, 2017). O Comitê Técnico do Instituto Europeu de Normas de Telecomunicações (ETSI) desenvolveu especificações que se concentram em facilitadores de integração e suporte de aplicativos por meio de interfaces como em redes móveis (GAZIS, 2017; ZANELLA et. Al., 2014; PALATTELA et al., 2013).

Algumas atividades de padronização abordam particularmente a camada de aplicação e seus requisitos na infraestrutura M2M/IoT. Por exemplo, o oneM2M define serviços e interfaces comuns através das quais as aplicações podem consumir serviços. O *Open Geospatial Consortium* (OGC, 2011) define uma arquitetura IoT para habilitar a criação de padrões para observações e medidas e também um conjunto de serviços de Internet para monitoramento e controle de redes de sensores de acordo com caso de uso específico. A associação TIA (*Telecommunications Industry Association*) desenvolve padrões de interface de acesso para sistemas no IoT. Os institutos OMA (Open Mobile Alliance) e o BBF (Fórum de banda larga) fornecem toda a infraestrutura para permitir o gerenciamento de aplicativos e dispositivos IoT (GAZIS, 2017).

Para habilitar a conectividade entre dispositivos microprocessados heterogêneos, diferentes redes e tecnologias de comunicação são usadas, como Sigfox, 6LoWPAN, LoRaWAN, tecnologia móvel. O Sigfox (SIGFOX, 2018) é uma tecnologia de comunicação baseada em software, onde a complexidade da rede é gerenciada na nuvem e não nos dispositivos. No nível do *gateway* a tecnologia Sigfox permite comunicação ampla como 802.11 e tecnologia GSM, 3G e 4G. A rede de sensores é baseada em equipamentos com modulação de rádio de banda ultra estreita na faixa de comunicação de 200 kHz. A comunicação é baseada em mensagem pequenas, no máximo de 26 bytes. 6LoWPAN (HUI,2012) é um protocolo de rede baseado em IP que define novos mecanismos de compressão de encapsulamento e cabeçalho projetado para redes WPAN. LoRa (LORA, 2018) é uma rede sem fio de baixo consumo trabalhando na faixa de Sub-GHz, que foi projetado para atingir redes de áreas extensa, além de suportar comunicação bidirecional, serviços de segurança, mobilidade e localização de ponta a ponta para IoT. Outras soluções de longa distância são WiMax e tecnologias móveis como GSM, 3G, 4G,5G (YAQOOB et.al., 2017).

No aspecto de comunicação de redes, é abordado por tecnologias diferentes, fornecendo soluções em um ambiente metropolitano (WWAN), local (WLAN) e em ambiente pessoal (WPAN). Em relação a rede de tecnologia móvel, o ETSI define o padrão 3GPP que envolve redes móveis e aplicações IoT em grande escala. Com relação aos protocolos de comunicação das WSN, as redes WLAN são baseadas no padrão IEEE802.11, enquanto que as redes WPAN são baseadas no padrão IEEE802.15. As características dos equipamentos de rede, que utilizam o padrão IEEE802.11, são de taxa de comunicação mais alta comparada com o IEEE802.15, porém com um maior consumo de energia. Segundo Cano, et.al. (2011), os protocolos de camada MAC projetados para redes WLAN não atendem as restrições das WSN. Normalmente, os protocolos WLAN assumem que as redes são compostas de dispositivos de alta capacidade

de processamento com alto consumo de memória e necessidade de sincronismo, o que aumenta muito a complexidade do hardware e o consumo de energia. Outra razão para sua inadequação é que os protocolos MAC tradicionais são focados em taxa de transferência e latência, enquanto que, em WSN muitas vezes trabalham com cargas extremamente baixas não necessitando de um alto rendimento. Portanto, o padrão IEEE 802.15 é constituído de equipamentos de baixo consumo e baixa potência, e devido a estas características tem sido cada vez maior o número de dispositivos IoT sendo desenvolvidos utilizando este padrão (GAZIS, 2017; PALATTELA et al., 2013).

Entre as especificações IEEE802.15, o IEEE 802.15.1 é a base para a tecnologia de comunicação sem fio Bluetooth, que apesar de poder comunicar em rede, é principalmente utilizada para comunicação ponto a ponto entre dispositivos em pequenas distâncias. A especificação IEEE802.15.4 define uma camada física de baixa potência e uma camada de acesso ao meio para redes WPAN. Ele tem sido a base de padrões de redes de sensores sem fio em diferentes áreas e principalmente na área industrial com os padrões WirelessHART, ISA100 e ZigBee (TOKOIGNON et.al., 2017), (RAJANDEKAR & SIKDAR, 2015). Para atender diferentes aplicações, foram propostas diferentes extensões do padrão de acordo com o cenário de aplicação. Entre eles, destacam-se o IEEE802.15.4e (IEEE802.15.4e, 2012), que propõe funcionalidades da camada MAC para suportar principalmente o ambiente industrial. A extensão IEEE802.15.4f possui uma relação com a definição da camada física para RFID. A extensão IEEE802.15.4g (2011) define regras para a camada física de ambiente urbano para aplicações de medição inteligente. Existem ainda outras extensões para áreas médicas (extensão j), áreas de monitoração de áreas críticas (extensão k), e para redes metropolitanas com transmissão de imagem (extensão m) (IEEE802.15.4, 2015).

No caso específico da extensão IEEE 802.15.4g (2012), esta foi criada para definir o padrão da camada física em redes urbanas. A extensão IEEE 802.15.4e (2012) que trata especificamente da camada MAC em diferentes cenários atende as funcionalidades determinadas para redes urbanas (CHANG,2013). Estas duas extensões do padrão IEEE802.15.4 (e) e (g) trazem um importante papel para interoperabilidade para os diversos cenários. Elas ainda estão em fase de desenvolvimento e aperfeiçoamento.

As camadas superiores das WSN são dependentes da aplicação e do cenário de uso. Importantes instituições trabalham em soluções de protocolos para atender aos diferentes cenários. Exemplo destas instituições são ISA (*International Society of Automation*), o IEC (*International Electrotechnical Commission*) e o IETF (*Internet Engineering Task Force*). Os

padrões industriais foram os primeiros a terem um padrão bem definido, como, por exemplo, WirelessHART, ISA100.11a e IEC 62601/WIA-PA (SILVA, 2013; GAZIS, 2017).

O IETF tem realizado vários trabalhos para atender aos requisitos técnicos relacionados ao M2M e IoT. O grupo ROLL (*Routing Over Low-Power and Lossy network*) desenvolveu uma estrutura de roteamento para o protocolo IPv6 adaptada para dispositivos com recursos limitados. O grupo CORE (*Constrained RESTful Environments*) desenvolveu uma estrutura para aplicativos orientados a recursos, que são destinados a executar o protocolo IP em redes com recursos restritos como uma rede de sensores sem fio baseada no padrão IEEE 802.15.4 (SHENG et.al., 2013). O grupo ACE (*Authentication and Authorization for Constrained Environments*) desenvolve casos de uso e especificações de requisitos para autenticação e preocupações de autorização em ambientes restritos, ao mesmo tempo em que identifica mecanismos de autenticação e autorização adequado para acesso a recursos. O grupo DTLS (*Datagram Transport Layer Security*) em ambientes com restrições DICE (*DTLS In Constrained Environments*) está definindo um perfil adequado para IoT que possa ser implementado em um amplo espectro de ambientes restritos, tais como, espaço de memória, opções de algoritmo, tamanhos de pacotes, taxas de perda, taxas de erro. O grupo DICE trabalha em prol da definição de mecanismos que permitam o uso da camada de registro DTLS para comunicação *multicast* segura (GAZIS, 2017). O grupo de trabalho 6TiSCH (*IPv6 over the TSCH mode of IEEE 802.15.4e*) foi criado para desenvolver um padrão de conexão que ofereça desempenho industrial em termos de confiabilidade e consumo de energia, em uma camada superior para IPV6 para o protocolo TSCH da especificação IEEE802.15.4e. Este grupo de trabalho está padronizando mecanismos para que as redes 6TiSCH operem gerenciando diferentes redes de sensores sem fio mantendo qualidade de serviço para aplicações como ambiente industrial e das chamadas *SmartCities* (DUJOVNE et.al., 2014).

Como mostrado anteriormente, as redes de IoT podem assumir diferentes tecnologias e padrões. Na próxima seção, serão detalhadas as WSN utilizando padrão IEEE802.15.4, pois é o objetivo de análise desta tese.

2.2 Protocolos de uma WSN utilizando tecnologia IEEE802.15.4

O padrão IEEE802.15.4 define uma camada física de baixa potência e também uma camada de acesso ao meio para redes WPAN. Para as aplicações de IoT, importantes padronizações têm sido propostas pelo IETF como o padrão 6LoWPAN, RPL e CoAP

(PALATTELLA et. al., 2013) (SHENG et.al., 2013). Na Tabela 1 estão descritos os principais protocolos utilizados em IoT, que posteriormente serão detalhados.

Tabela 1 - Pilhas de protocolos da rede WSN para IoT

Camada	Modelo ISO/OSI	Protocolo
7	Aplicação	HTTP/CoAP
4	Transporte	UDP/TCP
3b	Rede	IPv6 / RPL
3a	Adaptação	6LoWPAN
2	Enlace (MAC)	IEEE802.15.4
1	Física (PHY)	

Fonte: (PALATELA et.al., 2013)

Camada física

A camada física define o acesso ao meio que é realizado por meio do rádio. Esta camada é responsável por uma série de serviços, como operação do transceptor do rádio, seleção de frequências, geração da frequência da portadora, detecção de níveis de energia do canal (ED), modulação e codificação para transmissão e recepção de dados da rede (IEEE802.15.4,2015).

O padrão IEEE802.15.4 utiliza banda ISM, com taxa de dados de até 250 Kbps com canais variando de 11 ao 26 e com intervalos de 5MHz (IEEE802.15.4, 2015). A potência de transmissão pode ser, geralmente, programada de -25 a 20dBm.

A medida da detecção de energia do receptor (ED) é usada pela camada de rede como parte do algoritmo de seleção de canal. Corresponde a uma estimativa da potência do sinal recebido dentro da largura de banda do canal IEEE 802.15.4 (IEEE 802.15.4, 2015).

Outra medição realizada na camada física é o indicador de qualidade de enlace, chamado de LQI (*Link Quality Indicator*). A medida LQI é um indicador da qualidade do pacote recebido. A medida pode ser implementada usando a ED, uma estimativa da relação sinal/ruído ou uma combinação desses métodos. Os valores máximos e mínimos de LQI são associados com os valores de mais baixa e alta qualidade dos sinais IEEE 802.15.4 detectáveis pelo receptor, enquanto os outros valores estariam uniformemente distribuídos entre esses dois limites. O indicador RSSI (*Received Signal Strength Indicator*), indica a potência do sinal recebido no pacote. Ele está relacionado com a probabilidade de recepção de pacotes. A diferença entre o RSSI e o LQI é que o RSSI varia pouco se comparado ao LQI (IEEE 802.15.4, 2011).

Segundo Chang e Mason (2012) o IEEE 802.15 tem padronizado de forma positiva muitos padrões de rádios que são utilizados para aplicações com baixas taxas de transferência e gasto energético. O padrão IEEE 802.15.4 admite a operação nas faixas de frequência usando a banda livre de rádio ISM (*Industrial, Scientific and Medical*), as quais estão isentas de licenciamento. Especificamente a extensão IEEE802.15.4g cobre as aplicações de longas distâncias que inclui:

- Operação em bandas de frequências isentas de licenças disponíveis ISM, que cobre as faixas de 868MHz, 915 MHz e 2.4 GHz;
- Taxa de dados da faixa de 40 Kbps e não mais que 1000 Kbps;
- Tamanho da mensagem de até 1500 octetos, que permite a transmissão de uma mensagem com IP sem fragmentação;
- Garantia da coexistência com outros sistemas na mesma banda incluindo protocolos IEEE802.11, IEEE802.15 e IEEE802.16.

O padrão IEEE802.15.4g basicamente apresenta três diferentes tipos de camada física: MR-FSK, MR-OFDM, MR-O-QPSK (CHANG; MASON, 2012), (IEEE802.15.4g, 2012). As características de cada uma das camadas são:

- MR-FSK (*Multi-Rate e Multi-Regional Frequency Shift Keying*) contempla um sistema já instalado atualmente. A maioria dos sistemas implantados nos EUA são baseados em modulação FSK, principalmente nas faixas de frequência de 902 a 928 MHz com *Frequency Hopping Spread Spectrum* (FHSS). Para as bandas não licenciadas são utilizadas as bandas chamadas sub GHz também utilizando FSK;
- MR-OFDM (*Multi-Rate e Multi-Regional Orthogonal Frequency Division Multiplexing*) suporta altas taxas de dados;
- MR-O-QPSK (*Multi-Rate e Multi-Regional Offset-Quadrature Phase-shift Keying*) é o padrão até então empregado pelo IEEE802.15.4 que foi largamente utilizado por todo o mundo nas diversas aplicações de WSN, controle remotos sem fio e redes prediais, residenciais e industriais.

Segundo Chang e Mason (2012) as mudanças necessárias na camada MAC para suportar estes padrões, para o caso da faixa de frequência de 2.4 GHz, são cobertas pelo padrão IEEE 802.15.4e (2012). Nesta tese será adotado o padrão de 2.4 GHz pois é a banda atualmente mais utilizada e com maior disponibilidade de soluções de equipamentos comerciais.

No caso específico da banda de 2.4 GHz, a faixa de frequência prevista no padrão IEEE802.15.4 varia entre 2.4 a 2.485 GHz, constituído de 16 canais de frequência com distância

de 5 MHz entre cada canal e com 2 MHz de largura para cada canal. O rádio pode arbitrariamente enviar e receber em qualquer um destes canais.

Um rádio de baixa potência normalmente tem um sinal de saída da ordem de 0 dBm, ou 1 mW. Os rádios comerciais para aplicações em baixa potência permitem o ajuste da potência de transmissão. Os equipamentos de WSN enfrentam problemas na susceptibilidade aos fenômenos de propagação de ondas eletromagnéticas, como espalhamento e sombra causando o enfraquecimento do sinal. Quando o sinal de RF é capturado pela antena do receptor, este é amplificado pelo LNA sem degradar a relação Sinal/Ruído para posterior demodulação e processamento. A sensibilidade do receptor representa a potência mínima de RF do sinal na saída da antena com a qual ainda é possível o recebimento do sinal da mensagem com sucesso. Um rádio típico para comunicação em baixa potência tem uma sensibilidade -90 dBm (PALATTELLA, et.al, 2013).

Segundo Watteyne et.al (2012) o circuito do rádio geralmente é o componente que mais consome energia quando ligado. Quando desligado, não consome energia. Neste caso, para os equipamentos de baixa potência, o desafio é haver um protocolo de comunicação que permita a transmissão de dados com o uso restrito da banda de comunicação, possibilitando o rádio ficar desligado a maior parte do tempo e, desta forma, poupar energia. Um protocolo de comunicação eficiente no consumo de energia tem um ciclo de trabalho inferior a 1%.

Na grande maioria das aplicações de baixa potência, os módulos sensores são conectados a bateria, e neste caso são imprescindíveis soluções para aumentar o tempo de vida do equipamento diminuindo seu consumo de corrente. Isto pode ser feito escolhendo um rádio que drene pouca corrente e a escolha de um protocolo que utilize o rádio em um menor ciclo de trabalho possível.

Ainda na camada física, quando um rádio envia um pacote, inicialmente ele transmite um preâmbulo de 128 microssegundos para permitir que o receptor entenda o sinal do transmissor. Após o preâmbulo, ele envia um byte de SFD (*Start Frame Delimiter*) para indicar o início da mensagem. Então, o próximo byte indica o tamanho em bytes da mensagem. Seu valor máximo é de 127 bytes, que limita a mensagem em 128 bytes quando inclui o byte do tamanho. Quando nenhum módulo sensor está transmitindo, o rádio do receptor ouve um ruído branco e fica aguardando por um preâmbulo físico para travar a escuta e aguardar a mensagem. Uma vez que recebe o preâmbulo, ele espera pelo SFD e pelo tamanho da mensagem, posteriormente, todos os bytes da mensagem. Em geral, o componente transceptor do rádio armazena os dados internamente e somente após o sucesso da recepção, interrompe o microprocessador da chegada de uma nova mensagem.

Para transmissão de pacotes e gerenciamento de acesso ao meio, o padrão IEEE802.15.4 define o método de acesso múltiplo com verificação de portadora com prevenção de colisão CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance*), que reduz a ocorrência de colisões de pacotes na comunicação e é de grande importância em WSN (DINH & KIM, 2012).

No mecanismo básico do CSMA/CA, o transmissor da mensagem, antes de iniciar a transmissão, verifica se o canal está ocupado. No caso afirmativo, o transmissor espera um tempo aleatório e novamente verifica se o canal está livre indicando que pode iniciar a transmissão. Este mecanismo fornece uma interface entre a camada MAC e o canal físico do rádio. Neste caso, o CSMA/CA utiliza o CCA (*Clear Channel Assessment*) para verificar a ocupação do canal. Para execução do CCA pode ser utilizado o método de detecção de energia acima do nível ou detecção somente da portadora (IEEE802.15.4, 2015). Quando configurado para detectar somente energia acima do nível, o CCA reportará o estado intermediário como ocupado após detectar um nível de energia acima do nível ED. Quando configurado para detectar somente a portadora, o CCA reportará o estado intermediário como ocupado após a detecção do sinal da portadora. Neste caso, este sinal pode estar acima ou abaixo do nível ED. No caso de configuração detectar portadora com energia acima do nível, o CCA reportará o estado intermediário como ocupado após a detecção da portadora com energia acima do nível ED (PATEL & UPADHYAY, 2013).

Camada MAC

A camada MAC é a camada que interage diretamente com o rádio. O papel principal da camada MAC é coordenar o acesso e a transmissão por meio de um meio comum para vários módulos. Esta coordenação é complicada pois qualquer transmissão em curso interfere em qualquer outra transmissão dentro do alcance da comunicação. A interferência pode levar a perdas de pacotes que precisam ser detectadas e utilizar os mecanismos de retransmissão adequados (KIM et.al., 2015). A especificação IEEE802.15.4e (2012) define um protocolo MAC com regras diferentes dependendo do cenário de uso. Estas regras devem ser colocadas em prática para minimizar a interferência e colisões de pacotes.

Segundo AlSkaif et.al (2017), de acordo com o escalonamento da rede, os protocolos da camada MAC para WSN podem ser classificados baseados na transmissão da mensagem em modo síncrono, assíncrono ou híbrido. O modo síncrono, também chamado de modo escalonado ou livre de contenção, depende de um sincronismo periódico enviado pelo

coordenador que controla o escalonamento da rede. O modo assíncrono é independente de sincronismo e cada módulo da rede possui seu próprio escalonamento interno. O modo híbrido reúne as duas técnicas síncronas e assíncronas de acordo com as características da rede.

Os métodos sincronizados diferem dos assíncronos no sentido de que cada módulo vizinho está programado para acordar ao mesmo tempo. Eles podem ser divididos em métodos sincronizados localmente ou globalmente. Os protocolos MAC sincronizados localmente adotam o mecanismo de ciclo de trabalho. Para economizar energia, eles permitem que os módulos desliguem o rádio quando nenhuma comunicação ocorre durante um determinado período. Exemplo deste tipo de mecanismo são os protocolos S-MAC e T-MAC. Os protocolos sincronizados globalmente ou com slot de mensagem dividem o tempo em quadros e atribuem intervalos de tempo aos módulos, de forma que nenhum módulo dentro da distância de dois saltos é alocado no mesmo intervalo de tempo. Exemplo deste tipo de tecnologia são L-MAC e TreeMAC (ALSKAIF et. al., 2017). O problema dos protocolos MAC síncronos é que eles precisam manter a rede sincronizada, o que implica uma sobrecarga alta de controle (KIM et.al., 2015).

Os métodos sincronizados são derivados do TDMA (*Time Division Multiple Access*). O TDMA é baseado em uma divisão do tempo em slots onde cada comunicação entre dois módulos ocupa um slot de tempo (IEEE802.15.4, 2015). Também, o protocolo prevê o uso dos n canais de comunicação disponível através do salto em frequência, onde n indica o número de canais disponíveis, dependendo da faixa de frequência adotada. Considerando o slot de tempo e os diferentes canais é formada uma estrutura matricial (linha e coluna) chamada de *superframe*. Dentro do mecanismo do *superframe*, cada nó da rede segue um escalonamento que informa o que fazer em cada slot. Em um dado slot de tempo o nó sensor pode transmitir, receber ou entrar em repouso. Para este último caso, chamado de modo inativo, o equipamento não precisa ligar o rádio. Para os outros casos, chamados de modo ativo, o escalonamento indica com qual vizinho ele deve transmitir ou receber e em qual canal. Devido esta estrutura de *superframe* e um escalonamento bem acertado e em perfeito sincronismo, as transmissões não sofrem colisões, garantindo qualidade na taxa de entrega e tempos fixos de comunicação (SMART et.al., 2016). Para manter o sincronismo da rede é necessário um coordenador. O coordenador da rede transmite mensagens de sincronismo ou *beacons* para os módulos associados a ele e toda comunicação ocorre dentro deste período entre os *beacons*, formando um ciclo de trabalho ou macrociclo.

Baseado neste mecanismo do TDMA vários outros protocolos e melhorias foram propostas. Por exemplo, o MLMAC que escalona a comunicação entre os módulos de forma

distribuída e randomicamente escolhe os slots em multicanais de forma fixa. No protocolo Y-MAC os módulos são sincronizados baseado em um canal de sincronismo. Quando múltiplos pacotes precisam ser enviados entre vizinhos, eles são enviados em diferentes canais, diminuindo a latência da rede (KIM et.al., 2015). Outros exemplos de mecanismos síncronos onde negociam o canal para a transmissão de dados são CAM-MAC, e o TSCH, este último incorporado na especificação IEEE802.15.4e (2012). O TSCH tem mostrado aplicabilidade em vários cenários de IoT tanto em ambiente fechado quanto em ambiente aberto (PALATTELA et.al.,2013) (SMART et.al., 2016).

O protocolo TSCH (*Time Slot Channel Hopping*) utiliza comunicação com sincronização de tempo e salto em frequência para prover robustez à rede através de redundância temporal. Ele é independente de topologia, ou seja, pode formar qualquer topologia de rede de estrela a rede mesh. A base da comunicação é uma divisão da comunicação em *slots* de tempo como mostrado na Figura 2. Os módulos podem se juntar à rede depois de receber um *beacon* de outro módulo. A sincronização do tempo é repassada do coordenador da rede para os módulos filhos ao longo de uma estrutura de diagrama acíclica direcionada (DAG). O *slot* de tempo é o tempo necessário para o envio da mensagem pelo transmissor e o reconhecimento pelo receptor, que é geralmente de 10 a 15 ms. O escalonamento do TSCH indica se um módulo deve transmitir, receber ou entrar em repouso naquele dado intervalo de tempo. Um intervalo de tempo em um *slot* é identificado pelo seu tempo relativo, a frequência de comunicação e a função do link de transmissão, recepção e sincronização de tempo. Os slots podem ser dedicados ou compartilhados. O *slot* dedicado é livre de contenção, enquanto que o *slot* compartilhado é baseado em contenção, desta forma necessita de mecanismo CSMA/CA para envio da mensagem (DUQUENNOY et.al., 2015).

O padrão IEEE802.15.4e apenas define o mecanismo da camada de enlace, porém não define como a rede deve realizar o escalonamento e o gerenciamento do tráfego da rede (THUBERT et al., 2013). O escalonamento da rede deve ser construído de forma a garantir a qualidade de serviço da rede para atender as características dos diferentes equipamentos. Os protocolos de áreas específicas como WirelessHART, ISA100 na área industrial, ZigBee como padrão de aplicações abertas, utilizam IEEE802.15.4e nas camadas inferiores e implementam características específicas do cenário nas camadas superiores (PALATTELA et. al., 2013).

Figura 2 – Exemplo de um escalonamento de rede no protocolo TSCH

Fonte adaptada: PALLATELLA et.al. (2013)

Dentro dos protocolos síncronos, o escalonamento pode ser construído de forma centralizada ou de forma distribuída. Na forma centralizada, um gerenciador da rede é responsável por construir e manter o escalonamento da rede. Cada nó da rede constantemente atualiza o coordenador da rede com a lista de outros módulos vizinhos que se encontram na sua redondeza, além de poder informar a estatística de comunicação sendo gerada. Uma vez que o escalonamento é construído, o coordenador da rede informa para cada módulo sobre as respectivas conexões entre os vizinhos do módulo como mostrado na Figura 2. Quando existe uma mudança na conectividade, por exemplo, um módulo perde um de seus vizinhos, o coordenador atualiza seu escalonamento e informa aos módulos afetados sobre a nova configuração. Porém, se a rede apresentar uma característica dinâmica, os módulos ficam constantemente em movimento, e a rede deve se reconstruir constantemente. Um exemplo de protocolo que utiliza o escalonamento sincronizado é o WirelessHART (PALLATELLA et.al., 2013).

Para o escalonamento distribuído os módulos decidem localmente em quais conexões devem ser escalonados e com quais vizinhos. Uma solução para uma rede pequena seria cada nó realizar um escalonamento de uma conexão para cada nó em posição específica. Em casos mais complexos com tráfegos e densidades de vizinhança maiores pode ser utilizado protocolos baseados na internet como é o caso do RSVP (*Resource Reservation Protocol*) e MPLS (*Multi Protocol Label Switching*) (WATTEYNE, et al., 2010).

Em relação à sincronização, de acordo com Vilajosana et.al (2014), existe o sincronismo baseado no reconhecimento e o sincronismo baseado na mensagem. No sincronismo baseado no reconhecimento, o módulo receptor calcula a diferença entre o tempo esperado e o tempo real do pacote que foi recebido, e passa esta informação para o módulo transmissor durante o reconhecimento. Isto permite que o transmissor sincronize seu relógio com o receptor. No

sincronismo baseado na mensagem o nó receptor calcula a diferença entre o tempo esperado e o tempo real da mensagem que foi recebida e ajusta seu relógio pela diferença. Se existir tráfego na rede, os módulos que estão comunicando se sincronizam baseados nas informações do tráfego. Se não existir tráfego na rede, então eles geram uma mensagem vazia para sincronização chamada de *keep-alive*.

Outro mecanismo dos protocolos sincronizados é o salto em frequência. Este mecanismo é importante pois implica em diversificar as frequências de forma a diminuir os efeitos de interferência e o enfraquecimento do sinal devido à reflexão ou espelhamento. Também, o uso de frequências variadas aumenta a capacidade da rede, desta forma, mais módulos podem transmitir ao mesmo tempo, usando diferentes canais. O mecanismo de salto em frequência combinado com o mecanismo de acesso por meio de slots aumenta também a confiança na rede. Alguns dos métodos utilizam canais de dados e de controle como o MLMAC e Y-MAC. O TSCH utiliza um mecanismo que constantemente salta em frequência de acordo com uma regra combinada entre os módulos (VILAJOSANA, et.al 2014).

No contexto das redes sincronizadas em redes urbanas, a característica das redes sincronizadas não permite a escalabilidade da rede quando se tem apenas um coordenador, isto devido à forte dependência do sincronismo da rede. Então, neste caso, é necessário ter várias sub-redes distribuídas onde houver somente um coordenador para cada sub-rede. Sendo assim, é indispensável um mecanismo de troca de informações entre as sub-redes. Para solucionar estas questões de arquitetura e aspectos gerais relacionados com o protocolo IPV6, foi criado o protocolo 6TiSCH (WATTEYNE et.al., 2014).

O 6TiSCH define uma arquitetura aberta de rede utilizando protocolo IPv6 e especifica como os pacotes que pertencem a uma rede determinística, que fazem uso do protocolo IPV6, são roteados e repassados para uma rede utilizando o protocolo IEEE802.15.4e como, por exemplo, uma rede TSCH. O 6TiSCH propõe a utilização de protocolos padrões de internet como 6LoWPAN e RPL, com um mínimo de adaptação requerida para atender aos requisitos de confiabilidade e determinismo dentro da rede *mesh*, e escalabilidade dentro do barramento principal utilizando protocolo IPV6 sob a ethernet (WATTEYNE et.al., 2014).

No contexto do protocolo IEEE802.15.4e, a arquitetura do 6TiSCH inclui uma nova camada situada acima da camada MAC, chamada de camada 6TiSCH, que é responsável pelo escalonamento e, fornecimento de subsídios para as camadas superiores como o roteamento. Também a camada 6TiSCH inclui o processo de monitoramento da rede, que pode decidir se um determinado elemento no escalonamento do TSCH não está sendo realizado como esperado. A arquitetura do 6TiSCH possibilita aplicações com uma escalabilidade maior de módulos

sensores, como em aplicações de rede urbana, devido à flexibilidade de clientes e distribuição de áreas de atuação (WATTEYNE et.al., 2014).

Com relação aos métodos assíncronos, os módulos sensores têm seu próprio escalonamento e eles não precisam estar sincronizados com os vizinhos. Os mecanismos de comunicação geralmente permitem os módulos operarem em um ciclo de trabalho reduzido, possibilitando um baixo consumo de energia. Porém, este fato faz com que a latência da rede seja não determinística e bem maior do que os protocolos síncronos. Em contrapartida, como não há a necessidade de sincronismo, o controle do sistema possa ser distribuído e a escalabilidade da rede superior ao encontrado no ambiente sincronizado. Desta forma, os métodos assíncronos são os métodos mais adequados para uma rede com uma grande quantidade de módulos sensores (BEAUDAUX, et.al., 2014).

Com relação ao desenvolvimento de código fonte para os módulos sensores, os protocolos assíncronos possibilitam códigos mais simples quando comparados aos protocolos síncronos, devido à independência de escalonamento rígido. Adicionalmente, os protocolos assíncronos podem reduzir complexidade de hardware e, assim reduzir o custo do sistema e a possibilidade de menor consumo de energia. Porém os protocolos assíncronos são mais vulneráveis à colisão de pacotes, tornando indispensável o uso de mecanismos de acesso ao meio como CSMA/CA (CANO et. al., 2011; ALSKAIF et.al., 2017; FAFOUTIS et.al., 2015).

De acordo com as características de transmissão, os protocolos assíncronos podem ser divididos em métodos baseados na comunicação iniciada pelo emissor e métodos baseados na comunicação iniciada pelo receptor. Nos métodos onde a comunicação é iniciada pelo emissor, quando o transmissor tem alguma mensagem pendente a ser enviada, ele inicialmente envia um preâmbulo até ser reconhecido pela checagem periódica do receptor e então ele envia o dado. Já nos métodos iniciados pelo receptor, o receptor determina a hora que o transmissor deve comunicar enviando periodicamente uma mensagem de aviso para o transmissor. Neste caso um transmissor com um pacote para transmitir precisa esperar o anúncio do receptor para começar a transmissão.

Nos protocolos baseados na comunicação iniciada pelo emissor ou métodos baseados em preâmbulo, os módulos sensores acordam em um período fixo chamado intervalo de checagem do canal comum a todos os módulos da rede. Como o início da comunicação é dependente do transmissor, para a correta recepção da mensagem, cada pacote é precedido de um longo preâmbulo que precisa se prolongar até o início do tempo de checagem do receptor (KIM et.al., 2015). Um dos primeiros protocolos baseados em preâmbulo foi em 2002 com o *Low Power Listening* (LPL), que foi a base para demais protocolos que seguiram a mesma

estratégia. A característica básica do LPL é o mecanismo do ALOHA com CSMA/CA. O ALOHA é um mecanismo de acesso ao meio mais simples baseado em divisão do tempo em quadros. Se um canal estiver ocupado, ele deve esperar pelo início do próximo quadro. O protocolo B-MAC é baseado no LPL e incluiu CCA para determinar o que seria ruído e o que seria um pacote real. Esta técnica reduz o número de falsos negativos. Outras características do protocolo B-MAC são mecanismos de habilitar e desabilitar CCA, e o reconhecimento (CANO et al., 2011). De acordo com Cano et al. (2011) o fato de precisar de um preâmbulo para iniciar a comunicação faz com que existam problemas, como: esforço excessivo da rede quando se tem um longo período inativo; colisão com custo elevado, pois todo o período do preâmbulo é perdido; escuta excessiva devido ao fato que o preâmbulo não traz a informação do destino.

Vários outros protocolos derivados do LPL e B-MAC foram propostos. Os protocolos baseados em preâmbulo podem ser divididos de acordo com o tratamento do preâmbulo em mecanismos que dividem um preâmbulo maior em pequenos pacotes, mecanismos que utilizam uma informação de sincronização, e protocolos que trabalham no ciclo de trabalho.

Nos protocolos que dividem o preâmbulo único em um trem de preâmbulos menores, geralmente inclui-se informações adicionais como um endereço do destino em pequenos pacotes, diminuindo o problema de escuta excessiva pois os módulos que não estão envolvidos na comunicação já podem voltar ao repouso. Outra informação importante incorporada neste preâmbulo é o tempo quando a transmissão de dados será iniciada, desta forma o receptor já sabe quanto tempo depois ele voltará a ligar seu rádio para receber a mensagem (CANO et al., 2011).

Os protocolos que combinam preâmbulo com um mecanismo de sincronização têm como objetivo diminuir o longo preâmbulo. Uma das técnicas utilizadas neste caso é guardar a informação do último tempo de início do estado ativo do nó receptor. O transmissor envia um preâmbulo de tamanho calculado a partir do menor valor entre o tempo da última sincronização e o período de amostragem. Neste caso, no momento que terminar o período de amostragem o transmissor começa a enviar o trem de preâmbulos menores. Quando o transmissor receber um reconhecimento do receptor, ele salva este tempo, correspondente à última comunicação entre os módulos. Um problema desta técnica é que ela pode sofrer colisões constantemente se for escolhido o mesmo tempo para todos os módulos (CANO et al., 2011).

Para os protocolos que trabalham na adaptação do ciclo de trabalho, existem diferentes abordagens, como alterar o preâmbulo devido à posição na topologia ou ao aumento da carga da rede. Por exemplo, quando o nó detecta que está ocorrendo um aumento no número de pacotes o receptor precisa aumentar seu ciclo de trabalho (CANO et al., 2011).

Mais especificamente, o protocolo X-MAC é um protocolo assíncrono baseado em preâmbulo, que utiliza uma combinação dos métodos mencionados anteriormente. O preâmbulo no protocolo X-MAC é gerado por um trem de elementos menores e inclui o endereço no preâmbulo. Desta forma, diminui o tempo que os vizinhos precisam esperar quando a comunicação não é para ele. Logo que o receptor reconhecer o seu endereço, ele envia um reconhecimento para o transmissor, que faz com que diminua também a ocupação da banda e do rádio (YANG & HEINZELMAN, 2012). Este mecanismo é mostrado na Figura 3.

Figura 3 – Mecanismo de transmissão no protocolo X-MAC

Fonte adaptada: BEADAUX et al., 2014

O exemplo da Figura 3 mostra o mecanismo básico de comunicação entre dois módulos sensores no protocolo X-MAC. Inicialmente, o transmissor aguarda um tempo de espera (*backoff*), e no seu período de trabalho ele envia um trem de preâmbulos (P) com o endereço do receptor. O receptor, após ligar o rádio no seu período de trabalho, recebe a mensagem de T1 e verifica que o preâmbulo é para ele. Então, o receptor envia um reconhecimento de preâmbulo (PA). O transmissor, então, envia o pacote de dados (DT) e caso a mensagem precise de confirmação, o receptor envia em seguida um reconhecimento de dados (DA) (BEADAUX et al., 2014).

No trabalho de Beadaux et al. (2014) foi feita uma análise do protocolo X-MAC em uma rede IoT em larga escala com 240 módulos usando a plataforma FIT IoT TestBed. Segundo Beadaux et al. (2014) os resultados mostraram que a taxa de transferência de dados do X-MAC pode ser comparada em performance aos protocolos síncronos S-MAC, bem como em relação a atraso e consumo de energia. E mostra também que os protocolos assíncronos são mais robustos que os protocolos síncronos em uma aplicação de larga escala.

Nos protocolos baseados na comunicação iniciada pelo receptor ou *Receiver Initiated Transmission* (RIT), os módulos sensores acordam em um período fixo chamado período de RIT comum a todos os módulos da rede. Como o início da comunicação é dependente do receptor, ele periodicamente liga o seu rádio e anuncia para os vizinhos que ele está esperando por mensagens. Desta forma, permite que os módulos receptores possam desligar seus rádios durante uma parte do ciclo de comunicação. O primeiro protocolo relacionado com este mecanismo foi o trabalho de Lin et.al., (2004), e foi incorporado na especificação IEEE802.15.4e (2012). O mecanismo do RIT é baseado em somente um canal de comunicação é a base para todos os outros métodos baseados no mecanismo de inicialização pelo receptor. Na Figura 4 é mostrado um exemplo básico da comunicação entre transmissor e receptor.

Figura 4 – Comunicação do rádio entre dois módulos no protocolo RIT

Fonte adaptada: IEEE802.15.4e, (2012)

De acordo com a Figura 4, quando o transmissor possui uma mensagem pendente, inicialmente liga o rádio para recepção (Rx) aguardando uma notificação do receptor por meio de uma mensagem de olá (H). Este estado é chamado *idle listening* (IL) ou escuta ociosa de transmissão (TxIL), e é aleatório porque o transmissor não conhece o momento em que o receptor anunciará. Quando o receptor envia H, isso indica que a janela de comunicação está aberta. Então, o transmissor envia os dados. O receptor envia um quadro de confirmação ou *Data-Ack* (DA) se a mensagem requer confirmação. As vantagens dos métodos iniciados pelo receptor em relação ao iniciado pelo transmissor é que o RIT tem uma menor sobrecarga da rede e um menor tempo ocioso de escuta, devido à necessidade de enviar a mensagem e depois esperar pelo vizinho para responder (DUTTA et.al., 2012, KIM et.al., 2015). Com relação à economia de energia e dentro do contexto assíncrono, os protocolos baseados no RIT são mais eficientes do que os protocolos baseados em um mecanismo de transmissão iniciada pelo

transmissor, devido ao controle pelo receptor do início da comunicação, porém ele tem um desempenho inferior quanto à latência da rede (FAFOUTIS et.al., 2015).

O RIT é formado basicamente por três parâmetros principais: *macRitPeriod* que é o período do RIT, onde o receptor liga o rádio, e envia uma mensagem de anúncio (H) na rede, indicando que começou um novo ciclo de recepção. O parâmetro *macRitDataWaitPeriod* indica o tempo de espera do receptor por uma mensagem do transmissor. E o parâmetro *macRITTxWaitDuration* que indica o tempo que o transmissor fica aguardando a mensagem de início da comunicação do receptor (IEEE802.15.4e, 2012).

No serviço de recepção, cada nó anuncia a disponibilidade de recepção de mensagem a cada *macRitPeriod* através do comando de requisição de dados (*RITDataReq*) usando um mecanismo de acesso ao meio CSMA/CA. Então o módulo sensor abre uma janela de recepção pelo período de *macRitDataWaitPeriod* onde ele espera pela recepção de alguma mensagem dos vizinhos, exceto o pacote *RITDataReq*, que é descartado. Ao fim do tempo da janela o módulo fecha a janela de recepção e retorna para o estado ocioso até a próxima transmissão do comando *RITDataReq* (IEEE802.15.4e, 2012).

Baseado neste mecanismo de RIT vários outros protocolos e melhorias foram propostas. No capítulo 3 desta tese será feito um estado da arte dos principais protocolos assíncronos baseados em RIT encontrados na literatura científica atual, pois estes protocolos serão o objetivo principal de estudos desta tese.

Camada de Adaptação

A camada de rede é responsável por conseguir encontrar o caminho até o ponto de destino, onde em uma rede de IoT, o acesso deve ser através da internet. Segundo Hui & Culler (2010) as soluções de rede de sensores que não são baseadas em IP (como ZigBee, Z-Wave, WirelessHART) não possuem uma camada de rede comum, portanto essas redes precisam de um *gateway* para interligação desses sistemas com outros sistemas ou com a internet. O papel deste *gateway* é de um tradutor entre os dois protocolos. Estes *gateways* são elementos complexos dentro da rede para se projetar, configurar e gerenciar. Também este *gateway* pode trazer um atraso na comunicação pelo fato de necessitar executar esta tradução entre estes dois protocolos. Desta forma, é desejável ter o mínimo de linguagens na duas pontas da comunicação. Por outro lado, haver uma linguagem comum na comunicação proporciona muito maior flexibilidade e robustez. Neste caso, a rede IP permite utilizar toda a infraestrutura já existente, isto inclui ferramentas de diagnóstico, gerenciamento e comissionamento. Novas

aplicações e protocolos podem ser introduzidos sem mudar a estrutura da rede. A rede pode também ser composta e integrada por diferentes tecnologias como IEEE802.15.4, IEEE802.11 e IEEE802.3.

Em relação às tecnologias relacionadas com IoT, as características do protocolo IPV6 como universalidade, extensibilidade e estabilidade, apontam como tecnologia dominante no futuro da internet (SHENG et.al., 2013). Porém, o padrão IPv6 possui tamanhos muito grandes para uma rede de sensores sem fio. O padrão IPv6 determina 40 octetos de cabeçalho e requer que os equipamentos suportem até 1280 bytes de tamanho máximo de mensagem. Já o protocolo IEEE802.15.4 determina que o limite da mensagem seja de 127 bytes. Para habilitar a conectividade IP em redes de sensores de recursos restritos, o grupo 6LoWPAN do IETF definiu o protocolo 6LoWPAN como otimização de protocolo do IPv6 em WSN usando o IEEE 802.15.4 (PALATTELA et.al., 2013).

Segundo Hui e Thubert, (2011), as mensagens 6LoWPAN são prefixadas por uma pilha de cabeçalhos, cada um identificado por um tipo de campo. Os tipos de cabeçalho podem ser logicamente agrupados em quatro categorias, dependendo da função que desempenham na estratégia de adaptação:

- Cabeçalho 6LoWPAN - é utilizado para especificar que o pacote recebido não é compatível com as especificações 6LoWPAN e, portanto, deve ser descartado, o que permite a convivência com outros módulos na mesma rede;
- Cabeçalho de despacho - é usado para comprimir um cabeçalho IPv6 ou para gerenciar a camada de ligação;
- Cabeçalho de endereçamento *Mesh* - permite que mensagens IEEE802.15.4 sejam encaminhadas para a camada de enlace, transformando WSN de um salto para múltiplos saltos;
- Cabeçalho de fragmentação - é usado quando uma mensagem é superior ao tamanho máximo da mensagem IEEE 802.15.4. de 127 bytes.

A Figura 5 mostra um exemplo de compactação do 6LoWPAN em três diferentes cenários. O cenário 1 mostra a comunicação entre dois dispositivos dentro da mesma rede. O cenário 2 mostra a comunicação entre um dispositivo da WSN e um dispositivo fora da rede, onde é conhecido os prefixos da rede externa. Por fim, o cenário 3 mostra a comunicação entre um dispositivo da WSN e um dispositivo fora da rede, onde não é conhecido o prefixo da rede externa (OLSSON, 2014). No exemplo, é considerado somente o cabeçalho da mensagem, que no caso do IPV6 possui 40 bytes.

De acordo com a Figura 5, o cenário 1 tem o máximo de compressão na comunicação entre dois dispositivos (DEV1 e DEV2). Neste caso, os dois módulos estão dentro da mesma rede 6LoWPAN, usando endereços locais de conexão. O cabeçalho do protocolo IPv6 pode ser compactado para apenas 2 bytes. O cenário 2 mostra a comunicação entre um módulo interno da rede e um dispositivo fora da rede 6LoWPAN, onde o prefixo para a rede externa são conhecidos. Neste caso, o cabeçalho do protocolo IPv6 pode ser comprimido para 12 bytes. O cenário 3 é o pior caso, onde o módulo interno não conhece o prefixo do dispositivo externo. Neste caso, o cabeçalho do protocolo IPv6 pode ser comprimido para 20 bytes. Este pior caso mostra que houve uma taxa de compressão de 50% (OLSSON, 2014).

Figura 5 – Exemplo de compressão do cabeçalho do 6LoWPAN

IPV6 HEADER

Ver	Traffic Label	Flow Label	Payload length	Next Header	Hop limit	Source Addr 128 bits	Dest Addr 128 bits	40 bytes
-----	---------------	------------	----------------	-------------	-----------	-------------------------	-----------------------	----------

1. Comunicação na mesma rede IEEE802.15.4

DEV 1

DEV2

FE80::CAFE:00FF:FE00:0100 → FE80::CAFE:00FF:FE00:0200

Dispatch	Compressed Header	2 bytes
----------	-------------------	---------

2. Comunicação módulo IEEE802.15.4 e módulo externo com prefixo conhecido

DEV 1

DEV2

2001::DEC4:E3A1:FE24:9600 → 2001::4455:84C6:39BB:A2DD

Dispatch	Compressed Header	CID	Hop Limit	Dest Addr 64 bits	12 bytes
----------	-------------------	-----	-----------	----------------------	----------

3. Comunicação módulo IEEE802.15.4 e módulo externo sem prefixo conhecido

DEV 1

DEV2

FE80::CAFE:00FF:FE00:0100 → 2001::4455:84C6:39BB:A2DD

Dispatch	Compressed Header	CID	Hop Limit	Source Addr 64 bits	Dest Addr 128 bits	20 bytes
----------	-------------------	-----	-----------	------------------------	-----------------------	----------

Fonte adaptada: OLSSON, (2014)

Camada de Rede

As WSN são constituídas por equipamentos com topologia estrela ou mesh, taxa de dados e consumo de energia reduzidos, conexões instáveis, e consequentemente baixas taxas de entrega de pacotes. Para atender aos requisitos de roteamento da WSN, o grupo ROLL do IETF propôs uma solução de roteamento em redes de baixa potência chamado RPL (*Routing Protocol for Low Power and Lossy Networks*) (WINTER et.al., 2012).

O RPL é um algoritmo de roteamento baseado no IPV6 para redes mesh de baixa potência com características limitadas, potencialmente com perdas de comunicação, e utilizadas em conjunto com dispositivos hospedeiros e roteadores com recursos limitados, como na automação predial e residencial, ambientes industriais e aplicações urbanas. A proposta do protocolo é construir rapidamente as rotas de rede, para distribuir conhecimento de roteamento entre os módulos, e para se adaptar à topologia de uma forma eficiente (GADDOUR & KOUBAA, 2012).

O RPL é fundamentado em um mecanismo baseado em gradiente, no qual é construído um grafo acíclico orientado a distância chamado de DODAG (*Destination Orientated Directed Acyclic Graph*). A rede pode ser montada com um ou mais roteadores de borda (LBR) e com suporte à comunicação bidirecional entre dispositivos de rede baseados no protocolo IPv6. No cenário mais típico do RPL mostrado na Figura 6, os módulos da rede estão conectados por caminhos de múltiplos saltos a um pequeno conjunto de dispositivos que são responsáveis pela coleta de dados e tarefas de coordenação. Para cada um deles, um DODAG é criado e mapeado sua função objetivo. A função objetivo é determinada por um conjunto de métricas e restrições, e é identificada por um identificador (DODAG-ID). Exemplos de funções objetivos incluem minimização de energia e minimização da latência (KIM et.al., 2017) (PALATTELA et al., 2013).

Figura 6 – Exemplo de uma rede utilizando protocolo RPL

Fonte adaptada: KIM et.al., (2017)

Com relação ao grafo formado, o módulo raiz é situado geralmente no roteador de borda, que é o elemento da interface entre a internet e a rede de sensores. A topologia é configurada com base em uma métrica chamada de *rank*, que codifica a distância de cada módulo em relação ao módulo raiz, conforme especificado pela função objetivo (PALATTELA et al., 2013).

O RPL pode suportar diferentes tipos de tráfegos e troca de informações dependendo dos requisitos do fluxo de dados, conforme descrito a seguir (GADDOUR & KOUBAA, 2012):

- Tráfego convergente ou Multiponto-para-Ponto (MP2P) – a informação é roteada no sentido dos módulos da rede para a Internet ou para o centro da rede. Em geral, esses destinos são os DODAG raiz, e eles atuam principalmente como pontos de coleta de dados para aplicações de monitoramento distribuídas. Um exemplo deste cenário seria uma notificação de alarme de uma região que deve chegar ao operador;
- Tráfego divergente ou Ponto-para-Multiponto (P2MP) – as mensagens são enviadas a partir do DODAG para o(s) módulos de destino. Um exemplo deste cenário seria um acionamento de um determinado ponto da rede, enviado pelo operador;
- Ponto-a-Ponto (P2P) – este tráfego é necessário para permitir a comunicação entre dois dispositivos pertencentes ao mesmo WSN, por exemplo, um sensor e um atuador. Um exemplo neste caso seria a troca de informação entre dois dispositivos.

Quanto aos tipos de mensagens, o protocolo do RPL é composto por três mensagens encapsuladas em um pacote ICMPv6: DIO (*DODAG Information Object*), DIS (*DODAG Information Solicitation*) e DAO (*DODAG Destination Advertisement Object*). O cabeçalho do ICMPv6 é formado por três campos que são o tipo, o código e a checagem de erro da mensagem. O corpo da mensagem é formado por uma base e opções do protocolo. A Figura 7 mostra a estrutura da mensagem ICMPv6 com o RPL. Dentro do cabeçalho do ICMPv6 o tipo 155 indica

o protocolo RPL. O campo código indica o tipo de mensagem RPL que pode ser o DIO, DAO ou DIS. (GADDOUR & KOUBAA, 2012).

Figura 7 – Exemplo de uma mensagem RPL.DIO encapsulado no pacote ICMPv6

```

> Frame 96: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
> IEEE 802.15.4 Data, Dst: Broadcast, Src: TexasIns_00:02:f4:ac:fd
> 6LoWPAN
> Internet Protocol Version 6, Src: bbbb::12:4b00:2f4:acfd, Dst: ff02::2
▼ Internet Control Message Protocol v6
  Type: RPL Control (155)
    Code: 1 (DODAG Information Object)
    Checksum: 0x3ff0 [correct]
    [Checksum Status: Good]
    RPLInstanceID: 0
    Version: 0
    Rank: 0
  > Flags: 0xb8, Grounded (G), Mode of Operation (MOP): Unknown
    Destination Advertisement Trigger Sequence Number (DTSN): 51
    Flags: 0x00
    Reserved: 00
    DODAGID: bbbb::12:4b00:2f4:acfd

```

0000	60 00 00 00 00 1c 3a 40	bb bb 00 00 00 00 00 00	@
0010	00 12 4b 00 02 f4 ac fd	ff 02 00 00 00 00 00 00	..K.....
0020	00 00 00 00 00 00 02 9b	01 3f f0 00 00 00 00 00	[?]....
0030	b8 33 00 00 bb bb 00 00	00 00 00 00 00 12 4b 00	.3.....K.
0040	02 f4 ac fd		

A mensagem DIO é responsável pela montagem e manutenção do DODAG. O DIO contém informações de *rank*, função objetivo, e o ID da rede. Quando a topologia está sendo iniciada, o elemento raiz ou DAGROOT envia uma mensagem DIO para todos os módulos vizinhos considerados “filhos”. Um nó vizinho que recebe uma mensagem DIO usa as informações da mensagem para decidir se muda e se associa a um novo DODAG, ou se mantém no atual, de acordo com a função objetivo e o rank dos seus vizinhos. Qualquer filho que deseja se juntar à rede deve enviar uma mensagem *multicast* DIS para sua vizinhança. O DIS dispara o envio de uma mensagem DIO da vizinhança. Quando uma mensagem DIO é recebida de um vizinho, um módulo calcula seu próprio rank para um valor que é uma função de ambos o rank vizinho e o custo de realizar o DODAG raiz através dele. O nó verifica o conjunto dos possíveis “pais” contendo apenas aquele vizinho, de acordo com um algoritmo de escolha. Neste caso o vizinho é adicionado para o conjunto dos possíveis “pais”. Caso não atenda aos requisitos o DIO não é considerado. Por fim, cada módulo da rede pode selecionar seu preferido “pai” dentro do conjunto de possíveis “pais” baseados nas regras do roteamento. Um módulo sensor pode ter vários “pais” para obter uma entrega de pacotes confiável através da diversidade do caminho. As mensagens DIO são transmitidas com base no parâmetro de ajuste para obter um

equilíbrio entre a carga e a convergência rápida, ou seja, deve ser feito um balanço entre consumo de energia e a velocidade da rede em mudanças na topologia (KIM et. al., 2017).

Quando os módulos já estão associados ao grafo, eles garantem uma rota a partir dos módulos para o LBR. Para estabelecer uma rota descendente, um nó anuncia seu endereço por meio da transmissão da mensagem DAO. O DAO também permite uma comunicação de cada nó entre si. No caso do DAO, o “filho” notifica seu próprio endereçamento para seus vizinhos e o DAGROOT usando *unicast*. No caso das mensagens DAO é necessário o reconhecimento. Como uma mensagem DAO é processada por cada nó e indica a rota até o LBR, ela pode ter dois modos de operação: modo armazenado ou não armazenado. No modo armazenado, cada nó armazena as informações de roteamento para todos os módulos descendentes em sua subárvore. No modo não armazenado apenas o LBR armazena essas informações para todos os módulos em sua rede. O mecanismo básico, válido para ambos os modos, é o processamento nos módulos superiores e posterior armazenamento das informações em mensagens DAO para criar entradas de roteamento para os módulos na subárvore (KIM et. al., 2017). A Figura 8 mostra um exemplo de comunicação do RPL.

Figura 8 – Exemplo de comunicação do RPL

Fonte adaptada: GADDOUR e KOUBAA, (2012)

De acordo com a Figura 8, o processo de construção da rede inicia-se com o nó raiz (DAGROOT) que envia uma mensagem DIO com o rank igual a zero. Os módulos M1 e M2 recebem a mensagem DIO em (1) e calculam o rank atual baseado nas informações do DIO

recebido e se associam a um pai. Se o módulo já possui um pai, ele já pertence à rede. Todos os módulos participantes da rede devem enviar mensagens DIO periodicamente informando o seu rank, mostrado em (2). Posteriormente, o módulo M3 deseja se associar à rede. Neste caso, em (3), M3 envia uma mensagem DIS que representa uma requisição da mensagem DIO dos seus vizinhos. Ele então, calcula o seu rank para cada uma das mensagens dos seus vizinhos e então envia um DIO indicando qual é o “pai” escolhido. Cada nó da rede deve enviar uma mensagem DAO para indicar um caminho reverso do nó para o DAGROOT. Assim, ele envia a mensagem DAO com seu endereço para os vizinhos com o destino do DAGROOT.

Camada de Transporte

A camada de transporte é responsável por fornecer confiabilidade na comunicação fim-a-fim através de redes baseadas em IP. Desta forma, para a rede sob o padrão IP, os dois principais mecanismos de transporte disponíveis são o TCP (*Transmission Control Protocol*) e o UDP (*User Datagram protocol*).

Para aplicações que requerem comunicação confiável, TCP provê controle de tráfego e de congestionamento através da técnica da requisição de repetição automática ou ARQ (*Automatic Repeat-Request*). O TCP é orientado à conexão, onde a mensagem é segmentada em pacotes que são transmitidas individualmente sobre IP. Se um pacote for perdido, o protocolo TCP detecta a perda e retransmite o pacote perdido.

O protocolo UDP permite que a aplicação envie uma mensagem encapsulada no protocolo IPv4 ou IPv6, para o destino. Este protocolo UDP pode ser usado por aplicações que necessitam de uma forma mais direta para transporte de dados, mas como UDP não tem confirmação, o UDP não fornece uma garantia de que os dados devem ser entregues de forma confiável. Como visto anteriormente na camada de adaptação, o 6LoWPAN remove alguns dos campos do cabeçalho do protocolo IPV6 e do protocolo UDP considerados valores bem conhecidos ou redundantes e que podem ser inferidos de alguns campos do cabeçalho do IEEE802.15.4.

O TCP é conhecido por sofrer uma redução da taxa de transferência quando executado através de uma rede sem fio, devido à perda de pacotes no canal. Porém, muitas aplicações de WSN têm quantidades limitadas de dados, e o efeito da perda não é tão significativo. Segundo Kim et.al., (2015), apesar do TCP ser o protocolo dominante na Internet, devido ao gasto de energia imposto pelo TCP em WSN ele não é tão utilizado. Para WSN opta-se pelo UDP na

maioria das vezes e incluindo um mecanismo de controle de retransmissão na camada de aplicação o que tem mostrado bons resultados.

Camada de Aplicação

A camada de aplicação da rede WSN fornece as informações dos sensores para as aplicações IoT através da Internet. Na internet as três principais tecnologias web são: HTML, HTTP / REST, e URIs. No entanto, o HTML não é adequado para aplicações de IoT devido a seu propósito básico de definição de páginas *web* e de sua necessidade de recursos, que são limitados nas aplicações para IoT. Formatos de dados especiais para substituir o HTML para aplicações embarcadas com recursos limitado são definidos, muitas vezes, baseadas em XML (*Extensible Markup Language*) e sua representação mais compacta como EXI (*Efficient XML Interchange*), ou JSON (*JavaScript Object Notation*) (BORMANN et. al., 2012).

O protocolo de transferência de hipertexto (HTTP) é um protocolo poderoso e muito adequado para utilizar em aplicações na internet. Porém, o HTTP é relativamente complexo para utilizar em equipamentos que possuem restrições como a WSN, tanto em relação ao tamanho do código fonte, quanto no uso de recursos e de rede. O REST (*Representational State Transfer*) é um estilo de arquitetura que define um conjunto de restrições e propriedades baseados em HTTP. O REST tem características importantes para um sistema IoT como arquitetura distribuída, escalabilidade, baixo carregamento, e simplicidade para ambientes limitados, e é compatível com o HTTP. Os serviços *web* que são baseados em REST são chamados de *RESTful* (BORMANN et. al., 2012).

O grupo CORE (*Constrained RESTful Environments*) do IETF definiu o protocolo para aplicações limitadas chamado CoAP (*Constrained Application Protocol*). O CoAP é um protocolo para ambientes limitados em termos de comunicação, baseado na especificação REST, porém ele é interoperável com o HTTP.

O CoAP é um protocolo assíncrono de pergunta e resposta sobre uma mensagem de transporte UDP. Ele é diferente do HTTP, onde a arquitetura cliente/servidor deste último não assume uma regra clara (SHELBY, 2012). A Figura 9 mostra um exemplo de mensagem do protocolo CoAP.

A arquitetura do CoAP é dividida em duas camadas: requisição/resposta e mensagem. A camada de mensagem é responsável pelo controle da troca de mensagens sobre o UDP. A camada de requisição/resposta compartilha um formato comum de comunicação com a

aplicação do usuário. As mensagens são identificadas por um identificador (ID) usado para confiabilidade da informação.

Figura 9 – Exemplo da arquitetura CoAP

No.	Time	Source	Destination	Protocol	Length	Info
3	4.506015	bbbb::1	bbbb::12:4b00:40e:fc87	CoAP	80	CON, MID:26380, GET, End of Block #0, /.well-known
4	4.507560	bbbb::1	bbbb::12:4b00:40e:fc87	CoAP	179	CON, MID:26380, GET, End of Block #0, /.well-known
5	5.006427	fe80::12:4b00:40... fe80::1		CoAP	199	ACK, MID:26380, 2.05 Content
6	5.006950	bbbb::12:4b00:40... bbbb::1		CoAP	113	ACK, MID:26380, 2.05 Content

<p>Frame 6: 113 bytes on wire (904 bits), 113 bytes captured (904 bits) on interface 0</p> <p>Ethernet II, Src: 00:ff:df:8f:dd:d4 (00:ff:df:8f:dd:d4), Dst: 00:ff:de:8f:dd:d4 (00:ff:de:8f:dd:d4)</p> <p>Internet Protocol Version 6, Src: bbbb::12:4b00:40e:fc87, Dst: bbbb::1</p> <p>User Datagram Protocol, Src Port: 5683, Dst Port: 56244</p> <p>Constrained Application Protocol, Acknowledgement, 2.05 Content, MID:26380</p> <p>01... = Version: 1</p> <p>..10... = Type: Acknowledgement (2)</p> <p>.... 0000 = Token Length: 0</p> <p>Code: 2.05 Content (69)</p> <p>Message ID: 26380</p> <p>Opt Name: #1: Content-Format: application/link-format</p> <p>Opt Desc: Type 12, Elective, Safe</p> <p>1100... = Opt Delta: 12</p> <p>.... 0001 = Opt Length: 1</p> <p>Content-type: application/link-format</p> <p>End of options marker: 255</p> <p>[Request In: 612]</p> <p>[Response Time: -650.447929000 seconds]</p> <p>Payload: Payload Content-Format: application/link-format, Length: 44</p> <p>Payload Desc: application/link-format</p>	<pre> 0010 00 00 00 3b 11 40 bb bb 00 00 00 00 00 00 12 ...j.@. 0020 4b 00 04 0e fc 87 bb bb 00 00 00 00 00 00 00 K..... 0030 00 00 00 00 00 01 16 33 db b4 00 3b 24 c0 60 453...;\$.f 0040 67 0c c1 28 ff 3c 2f 2e 77 65 6c 6c 2d 6b 6e 6f g..(</. well-kno 0050 77 6e 2f 63 6f 72 65 3e 2c 3c 2f 73 74 6f 72 6d wn/core> ,</storm 0060 3e 2c 3c 2f 6c 3e 2c 3c 2f 69 3e 2c 3c 2f 36 74 >,</l>,</i>,</bt 0070 3e > </pre>
--	---

Dentro do CoAP são definidos quatro tipos de mensagens: confirmadas, não confirmadas, reconhecidas e *reset*. Em adição, mensagens *multicast* são suportadas, sendo apenas possível para mensagens não-confirmadas. A camada de pergunta e resposta inclui elementos como código do método, código de resposta e outras informações como o identificador de recurso URI (*Uniform Resource Identifier*) (BORMANN et.al., 2012). Tanto CoAP quanto HTTP identificam recursos usando o (URIs). As interfaces HTTP podem utilizar CoAP URI da forma: “coap://”.

Os dispositivos que suportam o CoAP oferecem serviços flexíveis em qualquer rede de IP usando UDP. O CoAP permite a iteração com HTTP através de *proxies* que se comportam como um servidor para um cliente e um cliente para outro servidor. Um servidor *proxy* reverso pode realizar a ligação entre COAP e HTTP sem precisar de mudanças no cliente ou servidor (PETROLO et.al., 2017).

A Figura 10 mostra um exemplo de mensagem CoAP. Os principais campos da mensagem são: versão do CoAP (Ver); tipo da mensagem (T) que admite os tipos (0) confirmado; (1) Não confirmado; (2) Ack; (3) Reset. Opção de Contagem: Indica se uma mensagem é uma requisição ou uma resposta. No caso da requisição, ele indica o método. Quando for resposta, será enviado o código de resposta. O campo de identificação da mensagem

(*Message ID*) indica um único identificador da mensagem. Por fim os campos de opções e campo de dados da mensagem completam a mensagem (SHELBY, 2012).

Figura 10 – Exemplo da mensagem CoAP

Fonte: SHELBY, (2012)

Os métodos básicos de requisição do CoAP são GET, POST, PUT e DELETE. O método GET requisita uma representação do recurso especificado. O método POST envia uma mensagem e requisita que o servidor a aceite como mensagem subordinada do recurso identificado. O método PUT requisita que uma mensagem seja armazenada baseado na URI fornecida. O método DELETE apaga o recurso especificado. Na Figura 11 é mostrado um exemplo de troca de mensagens do CoAP.

Considere um exemplo onde o módulo sensor está ligado a uma lâmpada. No exemplo da Figura 11.a o cliente envia uma requisição de um GET pedindo a informação da lâmpada (GET/Light). A mensagem tem um ID igual a 123. Após receber a requisição e interpretá-la, o servidor, ou o módulo sensor, envia a resposta com o valor do pedido para o ID 123. Enquanto o cliente não receber uma resposta de confirmação do servidor, ele continuará tentando até um número de retransmissões como mostrado na Figura 11.b (BORMANN et.al., 2012).

Figura 11 – Exemplo de troca de mensagens no protocolo CoAP

Fonte: BORMANN et.al., (2012)

Segurança da rede

Atualmente, a segurança da informação é uma preocupação fundamental na confiabilidade e no sucesso de uma tecnologia. Desta forma, mecanismos de segurança adequados devem ser aplicados em todos os níveis do sistema IoT. Os dispositivos nos diferentes níveis, devem ser capazes de reconhecer e neutralizar potenciais ameaças (YAQOOB, et.al., 2017).

Segundo Sciancalepore, et.al., (2016), as mensagens podem ser autenticadas ou criptografadas nos diferentes níveis da rede usando mecanismos padrões. Por exemplo, o trabalho de Vucinic, et.al. (2014) explora a segurança de um sistema IoT no nível de aplicação utilizando o conceito de segurança de objetos. Neste trabalho, são abordados os domínios de confidencialidade através de controle de acesso, e proteção através de criptografia da informação.

O Trabalho de Keoh, et. al. (2014) explora a segurança da rede nas camadas de transporte. Os trabalhos do IETF é de prover funcionalidades de segurança na camada de transporte CoAP similar ao protocolo de transferência de hipertexto seguro (HTTPS) através do DTLS (*Datagram Transport Layer Security*).

Na camada MAC do padrão IEEE802.15.4, a especificação fornece o mecanismo AES (*Advanced Encryption Standard*) 128 bits com criptografia de chave simétrica. O trabalho de Sciancalepore, et.al., (2016) discute as vantagens da aceleração de hardware para operações de segurança. Segundo o autor, embora o AES não seja o mais eficiente bloco em termos de tamanho de código e desempenho de tempo, suas conhecidas propriedades de segurança, bem como a possibilidade de realizá-lo em hardware, tornando-o a solução mais adequada na segurança da rede IEEE802.15.4.

Neste capítulo foi mostrado os principais conceitos de IoT e seus padrões. No próximo capítulo serão abordados os protocolos assíncronos de camada MAC utilizando o mecanismo de transmissão iniciada pelo receptor visando aplicações de IoT.

3 ESTADO DA ARTE DOS PROTOCOLOS ASSÍNCRONOS BASEADOS NA TRANSMISSÃO INICIADA PELO RECEPTOR

Este capítulo destina-se a descrever os principais protocolos assíncronos de camada MAC iniciado pelo receptor de uma WSN utilizando padrão IEEE802.15.4e, que caracteriza esta tese.

Os protocolos assíncronos utilizando o mecanismo de RIT são baseados na comunicação com o vizinho quando o receptor anuncia o início da comunicação. O mecanismo de comunicação se baseia nos estados periódicos de rádio ativo e em repouso para economizar energia. Segundo FAFOUTIS et. al., (2015), estas características dos protocolos assíncronos baseada em RIT de ciclos de trabalho produz desafios para a camada MAC, mostrados abaixo.

a) **Ouvir em excesso (*Overhearing*)**: Os protocolos assíncronos são vulneráveis a colisões, uma vez que um rádio que está transmitindo é incapaz de detectar outras transmissões no meio sem fio. A abordagem de espera e envio sem confirmação do protocolo RIT pode apresentar problemas na colisão de mensagem e fazer com que ocorra um atraso na comunicação devido a retransmissões. Neste caso, o tempo necessário para se recuperar da colisão é, pelo menos, um período RIT. Ouvir em excesso gera atrasos na rede junto com o desperdício de energia (MINH & KIM, 2016; PALLATELA, et.al., 2013; DUTTA et.al., 2012).

b) **Escuta ociosa (*IL-Idle Listening*)**: Os protocolos assíncronos possuem a característica do tempo ocioso. O receptor liga periodicamente o rádio para recepção (Rx) e mantém o rádio ligado durante uma janela de tempo enquanto aguarda a comunicação de um vizinho. As soluções para o problema da IL foram propostas com base na técnica de ciclagem do ciclo de trabalho, onde os estados de rádio são alternados entre ativo e repouso (KIM et.al., 2015).

c) **Terminais Escondidos e Fome**: Os protocolos assíncronos podem também sofrer de problemas de fome e do chamado terminais escondidos quando os módulos transmissores não conseguem comunicar. A fome ocorre em colisões repetidas entre módulos concorrentes, ou módulos que acordam ao mesmo tempo em instantes consecutivos. Problema de terminal escondido ocorre quando existe vizinhos na comunicação entre dois módulos sensores e estes vizinhos se comunicam com outro e interfere na comunicação destes módulos. Outro problema ocorre quando em uma rede carregada, dois transmissores estão esperando a anúncio um do outro ou de um outro vizinho (SUN et.al., 2008, LI et.al., 2010).

d) **Comunicação *Broadcast***: A comunicação *broadcast* constitui um desafio em WSN assíncronas devido às características da comunicação. Como os períodos de atividade e repouso dos módulos não estão sincronizados no tempo, é improvável que um transmissor encontre um momento onde todos os módulos sensores estão acordados e prontos para receber uma transmissão.

e) **Ciclo de trabalho adaptativo**: Dependendo das características da rede, quando existe uma adaptação dinâmica, estas contribuem com a eficiência do sistema. Um protocolo MAC com ciclos de trabalho adaptativos pode propor um uso mais eficiente da energia disponível, fazendo uso de informações como a estrutura da rede, as condições de tráfego ou dos recursos dos módulos sensores.

f) **Qualidade de serviço**: Diferentes tipos de mensagens podem coexistir dentro da rede. De acordo com os requisitos do aplicativo subjacente, ou mesmo o próprio protocolo, cada classe de mensagem pode exigir um tratamento diferente. Por exemplo, as mensagens de alta prioridade podem ser retransmitidas antes das prioridades de baixa prioridade, as mensagens podem ser reordenadas para minimizar o atraso ou as mensagens de controle de ganhos podem ter precedência sobre as mensagens de dados para garantir o funcionamento correto da rede.

g) **Segurança**: As redes de sensores são vulneráveis a ataques associados ao meio sem fio. Os canais sem fio podem ser facilmente espionados e o tráfego pode ser alterado. Os atacantes não estão limitados pelas restrições de recursos dos módulos sensores e podem interagir ao lado da rede, utilizando equipamentos muito mais poderosos. Além disso, as redes de sensores podem ser implantadas em ambientes precariamente inseguros e os módulos sensores são vulneráveis a ataques e alteração da informação.

Para tentar resolver estes desafios, vários métodos iniciados pelo receptor foram propostos nestes últimos anos. Os trabalhos de Cano, et.al. 2011, Fafoulis et.al., 2015 e Alskaf et.al., 2016 mapearam trinta e um protocolos iniciados pelo transmissor desde 2004. Esta seção apresenta os principais protocolos baseado em RIT e suas características básicas, primeiramente apresentando os protocolos baseados em canal único, e posteriormente mostrando os protocolos multicanais.

3.1 Protocolos RIT de canal único

A proposta do RI-MAC (*Receiver-Initiate MAC*) é de minimização do consumo de energia, através da diminuição do tempo de ocupação da banda e através do desacoplamento do tempo de comunicação entre o transmissor e o receptor (SUN et al., 2008). No protocolo RI-MAC os módulos periodicamente acordam e enviam mensagens de anúncio H anunciando a abertura da janela de recepção como mostrado na Figura 4. O transmissor com mensagem pendente espera a anúncio do receptor e envia a mensagem. Após receber o pacote de dados, o receptor envia novamente uma mensagem de anúncio H, que servirá de reconhecimento do dado para o transmissor. Caso o transmissor tenha mais alguma informação para o destino, a mensagem H do receptor indica que o receptor está disponível novamente. Desta forma, um outro transmissor pode voltar a comunicar. No caso de precisar receber mais de uma mensagem em sequência, o receptor permanece com o rádio ativo por um tempo extra após ter enviado um reconhecimento. Este tempo de espera é configurado de acordo com o número estimado de transmissores. O receptor emprega também o H para coordenar mensagens de dados dos transmissores através do ajuste do tempo de atraso BW (*backoff window*). Se um H não contém campo BW, significa que os transmissores podem iniciar a transmissão de dados assim que receber o H. Se existir o campo de BW, cada transmissor deve enviar um tempo randômico com o valor no máximo de BW. Este valor é aumentado quando o receptor detectar colisão. Quando uma colisão é detectada, o receptor envia um H para os vizinhos que tem dupla função: notificar o transmissor da colisão e anunciar uma janela de contenção. Nos dois casos o transmissor escolhe um slot randômico na janela de contenção e envia o dado novamente após um sinal positivo de CCA. Sucessivas colisões fazem com que aumente a janela de contenção.

O protocolo Wide-MAC (ROUSSELOT et.al., 2008) trabalha de forma a conhecer globalmente a vizinhança e os tempos de anúncio de cada vizinho. Este conhecimento é necessário para prever a próxima anúncio e tentar diminuir o IL. Cada vez que o transmissor recebe uma anúncio, ele atualiza seu relógio. Porém, devido aos atrasos do relógio, o valor dessa previsão diminui ao longo do tempo.

O protocolo ADB (*Asynchronous Duty cycle Broadcasting*) (SUN et. al., 2009) segue o protocolo RI-MAC e propõe uma melhoria na comunicação *broadcast*. O ADB evita transmissões em conexões pobres, e estabelece um mecanismo colaborativo entre os vizinhos. O transmissor possui duas listas de módulos vizinhos: uma relacionada aos módulos que receberam o *broadcast*, e outra lista de broadcast pendente. Por exemplo, se um transmissor A quer transmitir um quadro para B e C. De acordo com critérios do algoritmo, considere que a

qualidade da conexão entre A e B é ruim, enquanto que a conexão entre B e C é boa. Após a transmissão do pacote de A para C, o receptor C assume a responsabilidade de encaminhar o pacote para B.

A proposta de *Opportunistic Cooperation* MAC (OC-MAC) (WANG et. al., 2010) procura reduzir o tempo que um transmissor espera por uma anúncio (TxIL) através de uma comunicação colaborativa. O transmissor transmite um RTS no canal, contendo o endereço de destino, uma função de energia e uma solicitação para outros transmissores. Como esta informação é direcionada para os transmissores, isso garante que o canal não terá comunicação dos vizinhos diminuindo a carga da rede. O transmissor escuta o canal esperando a anúncio do receptor e se não receber uma resposta dentro de um período determinado, o transmissor passa a comunicação colaborativa para outro transmissor. Neste caso, é necessária uma anúncio inicial do transmissor para cada mensagem que o transmissor desejar enviar.

O protocolo RC-MAC (*Receiver-Centric* MAC) (HUANG et. al., 2010) é projetado para aplicativos orientados a eventos com tráfego pesado. O RC-MAC requer um tempo inicial de atraso aleatório antes do envio de dado. Em caso de colisão, os transmissores voltarão a tentar com uma reversão exponencial binária sempre que o pacote de reconhecimento não for recebido. O receptor neste caso fica acordado até uma próxima anúncio. A quantidade de tentativas de retransmissão é limitada por um número predefinido.

O protocolo PRADC-MAC (*Pseudo-Random Asynchronous Duty Cycle* MAC) (LEE et.al., 2010) utiliza uma função *hash* para determinar os próximos tempos de anúncio do receptor considerando o atraso do relógio do módulo, de modo a diminuir o IL. Este tempo pseudoaleatórios é distribuído uniformemente no intervalo médio de trabalho e um intervalo aleatório. O transmissor e o receptor compartilham seus tempos de anúncio. Assim, cada nó é capaz de estimar o próximo tempo de despertar de cada receptor.

O protocolo RW-MAC (*Receiver Wake-up* MAC) (YANG et.al.,2010) procura reduzir o IL do transmissor (TxIL) prevendo a anúncio do receptor. O transmissor usa o tempo de espera restante de um receptor, que é carregado na anúncio H, para estimar o tempo da próxima anúncio. Cada módulo mantém uma tabela com o tempo anterior da vizinhança. Inicialmente, o remetente deve permanecer acordado por um período de tempo para estimar o tempo da vizinhança. O transmissor deve inicialmente aumentar a sua janela de espera *Twait* e esperar pelo H do receptor. O *Twait* é calculado considerando a menor diferença de atraso do

relógio, o ciclo de trabalho estático e o tempo estimado anterior. O tempo máximo que o remetente escuta ao canal após o despertar é definido como T_{cycle} .

O protocolo EE-RI-MAC (*Energy Efficient RI-MAC*) (YONG et.al., 2011) é baseado no RI-MAC e se propõe a aumentar a eficiência energética na transmissão diminuindo o IL. Para isso, o EE-RI-MAC usa um mecanismo baseado no protocolo X-MAC, onde inicialmente o transmissor chaveia entre o estado ativo e o estado de repouso. Além disso, o transmissor pode entrar no estado de suspensão depois de ouvir o canal por um período W e acordar após uma duração S . Porém, o protocolo causa um aumento na latência da rede.

O protocolo DB-MAC (*Delay Bounded MAC*) (PENG et.al., 2011) procura fornecer garantias de entrega de mensagens baseado em anúncios dedicados para cada vizinho. A anúncio H é personalizada, portanto permite que o período de cada H seja adaptado para cada transmissor. O protocolo também introduz um mecanismo dinâmico de adaptação do ciclo de trabalho que visa ajustar os horários de repouso às condições de tráfego fornecidas. Os módulos possuem um período de trabalho menor e economizam mais energia quando o tráfego é leve. Enquanto que com tráfego pesado, eles aumentam o ciclo de trabalho para melhorar o desempenho. Segundo o algoritmo de adaptação do ciclo de trabalho, inicialmente todos os módulos funcionam com ciclo máximo de trabalho. De acordo com a estimativa do tráfego, eles aumentam ou diminuem este ciclo conforme estabelecido por um algoritmo interno.

O protocolo PW-MAC (*Predictive Wake-up*) proposto por (Tang et.al., 2011) é baseado no *WiseMAC* e procura reduzir o consumo de energia do transmissor (TxIL). O PW-MAC é baseado em uma sequência pseudoaleatória gerada de forma independente para controlar os tempos de ativação de cada nó, permitindo que os transmissores consigam prever o momento em que o receptor irá acordar. Se o transmissor A tiver um pacote para enviar para B , ele aguarda a anúncio de B . Então, o módulo A transmite a mensagem de dados, configurando no cabeçalho da mensagem uma bandeira de solicitação do Estado de Predição (EP) de B . Então, B envia um reconhecimento, seguido de um pequeno pacote de duração EP em que ele incorpora seu tempo atual e estado de previsão. O tempo atual de B é usado por A para calcular a diferença de tempo entre os relógios de A e B (*drift*). O tempo EP de B representa o tempo esperado no qual B irá acordar na próxima vez. Então, quando A quer comunicar com B , A acorda um tempo um pouco antes do tempo de ativação previsto de B . O intervalo de TxIL no

protocolo PW-MAC reduz significativamente. Além disso, os tempos de alerta previsíveis são usados para melhorar o desempenho em caso de colisões e erros de canal.

O protocolo YA-MAC (*Yet Another MAC*) (YADAV & MCCANN, 2011) se propõe a aumento da taxa de entrega para comunicação *unicast* e *broadcast*. Na inicialização do YA-MAC os módulos partem com ciclo de trabalho máximo. Durante esta fase, eles determinam sua vizinhança e estabelecem o intervalo de tempo de transmissão. Também, YA-MAC usa a quantidade de módulos vizinhos para selecionar a janela de contenção para evitar a colisão. Todos os módulos vizinhos são despertados durante o slot de transmissão, e desta forma estão sincronizados. O protocolo prevê também uma janela de tolerância a erros de sincronização (SETW). O SETW define um intervalo de tempo de proteção para o sistema de derivações de relógios menores. Se a sincronização for inferior a um nível desejado, os módulos são ativados para inserir uma fase de ciclo de trabalho maior durante a qual a sincronização é restabelecida.

A proposta do SARI-MAC (*Self Adapting Receiver Initiated MAC*) (LAMPIN et al., 2012) é reduzir o atraso na taxa de entrega que surgem em WSN, no qual operam sob cargas de tráfego variáveis. Para isso SARI-MAC tem um protocolo adaptativo do ciclo de trabalho através da estimativa de tráfego e das restrições de atraso. Para fazer isso, SARI-MAC combina o mecanismo do RI-MAC com um mecanismo de estimação de tráfego. A estimativa do tráfego no protocolo SARI-MAC utiliza o mecanismo de reserva do meio no cálculo da estimativa. Neste caso, o receptor conta o número de slots que ocorreram antes de receber uma determinada mensagem. Baseado neste cálculo, o receptor estima o número de competidores. O cálculo da estimativa é obtido da equação 1, onde (E) representa o número de detecções esperada, (K) é o número de slots da janela de contenção, N é o número de competidores.

$$E(\text{detecção}) = K \left(1 - \left(1 - \frac{1}{K} \right)^N \right) \quad (1)$$

No final do escalonamento, o receptor compara o número de mensagens realmente recebidas com o número de competidores estimados. Caso o número estimado seja menor que o número de mensagens recebido, conclui-se que houve colisão, e então o receptor repete a sequência incrementando o número de estimativas.

O protocolo FERI-MAC (*Traffic estimation for underwater networks*) (PU et.al., 2013) foi projetado especificamente para Redes Acústicas Subaquáticas (UANs) e aborda problemas que são característicos desta configuração, como o preâmbulo estendido de modems acústicos

e a atenuação de sinal aumentada. A técnica principal utilizada pelo FERI-MAC é a estimativa de tráfego usada para prever quando o transmissor terá dados para transmitir e então deve enviar uma anúncioção.

O protocolo ERI-MAC (*Energy Harvested Receiver Initiated MAC*) proposto por (NGUYEN et al., 2014) tem uma filosofia muito parecida com o RI-MAC. A diferença principal é que o reconhecimento da mensagem é a própria mensagem de anúncioção que já servirá para habilitar a comunicação, caso for necessária. Outra característica proposta no algoritmo é a capacidade de compactar mensagens. Quando existe mais de uma mensagem para ser enviada, ela é compactada em um pacote maior. Desta forma, há economia de energia no envio. Isto é útil principalmente no caso do caminho convergente, onde os módulos sensores enviam as mensagens para o sink e desta forma aumentam a taxa de entrega e diminuem o tempo de resposta.

O protocolo SRI-MAC (*Synchronous Receiver-Initiated MAC*) (BOULFEKHAR & BENMOHAMMED, 2014) é um protocolo de ciclo de trabalho síncrono com transmissão de dados iniciada pelo receptor. O pacote de anúncioção do receptor contém o ID do receptor e o período de alocação de duração (DAP), que depende do número de vizinhos do receptor. Esses valores são usados como um fator comum para gerar valores de tempos de atraso para evitar colisão. Ao receber uma anúncioção, cada remetente envia um pacote RTS contendo o endereço do receptor, o endereço do transmissor e o tamanho do dado. Então, o receptor transmite um pacote CTS que é usado para atribuir um intervalo de tempo a cada remetente que se registrou anteriormente através de um RTS. Neste ponto, o período de comunicação começa e cada remetente acorda de acordo com a ordem predeterminada. No caso de não haver nenhum remetente para um determinado receptor, a anúncioção ficará sem resposta, ou seja, nenhum CTS será enviado, e após o tempo igual ao DAP especificado na anúncioção, o canal será considerado ocioso.

O protocolo AL-MAC (*Asymmetric links*) (WON et.al., 2014) alterna dinamicamente entre dois modos de operação: iniciado pelo transmissor e pelo receptor, dependendo da assimetria do canal de comunicação. Segundo Won et.al. (2014), os protocolos MAC iniciados pelo receptor funcionam mal quando os canais assimétricos estão presentes. O modo de operação padrão do AL-MAC é o chamado modo R que opera usando o RIT padrão. Sempre que um transmissor não recebe uma anúncioção do receptor pretendido por mais de n vezes, ele

muda para o modo T e começa a enviar preâmbulos. Não existe uma comunicação explícita desta alteração, em vez disso, o nó receptor executa uma CCA por um pequeno período de tempo no final de cada transmissão, a fim de verificar se existe ou não uma transmissão de preâmbulo em andamento. Caso contrário, o nó receptor também muda para o modo T. As colisões podem acontecer entre as anúnciões de um receptor em modo R e os preâmbulos formam um remetente em modo T. Para resolver isso, após uma colisão, o receptor executa CCAs adicionais para aumentar a chance de receber pré-ajustes. Finalmente, a assimetria de um link é medida através da análise da Razão de Recepção de Pacotes (PRR) que é calculado baseado no período de anúncio H , o número de pacotes enviados, e o número teórico de pacote que deveria ter sido enviado.

O protocolo EH-MAC (*enhancement MAC*) (PEREZ et.al., 2015) propõe um mecanismo de sequência pseudorrandômica de adaptação do tráfego da rede procurando diminuir o TxIL. O EH-MAC utiliza o mecanismo de sequência pseudorrandômica do PW-MAC que é realizada por cada módulo para o escalonamento das anúnciões. O EH-MAC propõe um mecanismo dinâmico, de acordo com as características da rede. A proposta trabalha com anúnciões primárias e anúnciões secundárias. Entre as anúnciões primárias, devem haver anúnciões secundárias em f subintervalos de acordo com a equação 2.

$$f = \lambda * \frac{2*(E_w+E_{tx})-(1+e)E_b}{E_b+E_w+E_{tx}} \quad (2)$$

Sendo f é o fator de velocidade, λ é um parâmetro de ajuste de carga medido por cada módulo em pacotes/segundo, e é um fator de erro, E_b é a energia consumida para envio da anúncio, E_w é a energia consumida pelo transmissor pela espera da anúncio do receptor (TxIL), e E_{tx} é a energia gasta para envio do dado. A geração de anúnciões extras não tem impacto no desempenho da rede e obtém melhores resultados de taxa de entrega e consumo com diferentes tráfegos da rede.

O protocolo L-MAC (*wake-up time self-learning*) (DINH et.al., 2016) é baseado no RI-MAC e propõe um mecanismo de autoaprendizagem que permite que um nó coordene seu despertar com seu “pai” em uma topologia em árvore, sem requerer sincronização ou troca de informações de programação. De modo que sempre que um “filho” tenha pacotes de dados para enviar, ele pode enviar pacotes rapidamente para alcançar alta eficiência energética e baixa latência. Isso é feito através de um algoritmo de autoaprendizagem de tempo de ativação no

qual um “filho”, ao invés de operar com um intervalo de ativação fixo, adapta seu período de repouso com base no tempo relativo de ativação com seu “pai”. Assim, ele pode acordar de forma mais próxima ao seu “pai”. Em cada intervalo de ativação, um vizinho mede e compensa o deslocamento, recalculando seu período de repouso. Se um vizinho completar suas tarefas anteriormente em comparação com o intervalo anterior, pode repousar mais no próximo ciclo e vice-versa. Isso significa que um módulo pode adaptar seu período de repouso com base em sua carga de trabalho. Esta característica beneficia especificamente os módulos intermediários, nos quais o tráfego pode variar ao longo do tempo. Quando o módulo não detecta a anúncio de seu “pai”, ele reduz seu período de repouso. O L-MAC também suporta um modo de pacotes múltiplos que é acionado quando um módulo possui mais de um pacote para enviar. Neste caso, primeiramente o transmissor envia a primeira mensagem com uma bandeira de várias mensagens anexada ao cabeçalho do pacote, sinalizando para seus superiores o modo de pacotes múltiplos. Desta forma, um nó pode reduzir o número de janelas de contenção e mensagens de reconhecimento para transmissão de múltiplos pacotes. Além disso, os pacotes podem ser agrupados para reduzir o número de transmissões. Para evitar colisão entre mensagens é utilizado janelas de contenção para distribuir o tempo de suspensão dos vizinhos. Como resultado, cada vizinho em um determinado ramo acorda em um ponto de comunicação diferente em relação a outros ramos.

3.2 Protocolos RIT multicanal

No contexto dos protocolos multicanais, o trabalho de Li et al. (2010) propõe uma arquitetura de comunicação assíncrona multicanal chamada de ARM (*Asynchronous Receiver-Initiated Multi-Channel*). O protocolo ARM é constituído de um canal de controle e canais de dados. A característica básica do protocolo ARM é um mecanismo de troca de dados inicial de RTS/CTS para estabelecer a comunicação entre transmissor e o receptor. Na Figura 12 é apresentado um exemplo de implementação do tratamento da transmissão e recepção de uma mensagem de dados dentro de um período da comunicação.

De acordo com a Figura 12 tanto o serviço de transmissão quanto o serviço de recepção são constituídos de 5 estados. O ciclo de transmissão, se inicia com o transmissor esperando a anúncio do receptor (TxIL). A segunda etapa ocorre assim que é recebida a anúncio H, onde o transmissor envia o RTS para o receptor com o endereço de destino, indicando que ele deseja comunicar. Na terceira etapa, o transmissor aguarda o CTS do receptor com o endereço

do destino. Na quarta etapa, o transmissor envia o dado. Por fim, na quinta etapa o transmissor deve aguardar uma mensagem de Data.Ack (DA) se necessário.

Figura 12 - Comunicação entre dois módulos sensores no protocolo ARM

Fonte adaptada: Li, et.al., (2010)

A proposta de comunicação assíncrona multicanal do protocolo ARM é constituída de um canal de controle dedicado (CC), e canais de dados (DC) para pacotes de controle e dados respectivamente. As informações podem ser transmitidas de forma *unicast* e *broadcast*. Para troca de comunicação entre transmissor e receptor é utilizado o mecanismo de RTS/CTS. A escolha do melhor canal no protocolo ARM é realizado utilizando um método baseado em uma seleção aleatória baseado em probabilidade para selecionar DC. Desta forma, quando um nó está pronto para receber, ele seleciona aleatoriamente um DC com probabilidade p , e volta ao repouso com uma probabilidade de $(1-p)$. Para o tratamento de mensagens broadcast o protocolo ARM tem um canal broadcast (BC) dedicado para a transmissão e recepção. Quando um remetente (S) tem um pacote broadcast para enviar, ele muda para o BC verifica se o canal está ocioso, e então envia o pacote de broadcast no canal em um intervalo de tempo determinado pelo período médio de envio.

O protocolo EM-MAC (*Efficient Multi-channel MAC*) (TANG et.al., 2011) é um protocolo multicanal que não utiliza canal de controle comum e utiliza um mecanismo de alocação dinâmica de canais baseados nas condições atuais do canal. O EM-MAC trabalha em melhorar a eficiência energética através da previsão de canal do transmissor e o tempo de ativação do receptor. No EM-MAC, cada módulo sensor gera um número de canal e um tempo para a próxima anúncio usando um gerador de números pseudoaleatórios compartilhados. Cada módulo é capaz de prever o próximo evento de despertar de qualquer outro vizinho apenas conhecendo o estado de previsão. O estado de previsão inclui a informação da semente

aleatória, o tempo anterior de despertar, um multiplicador e uma constante. Um módulo que não possui o estado de previsão de um determinado receptor, escuta uma anúncio no primeiro canal, que contém a informação correspondente. Além disso, cada módulo mantém o status de cada canal contando as falhas. Se a métrica de status exceder um limite pré-determinado, o canal entra em uma lista negra e não é mais utilizado. Se o gerador de números pseudoaleatórios escolher um canal na lista negra, o módulo permanece no canal anterior. A economia de energia é maximizada pelo mecanismo pseudoaleatório que prevê o próximo despertar do vizinho.

O protocolo AMCA (*Asynchronous Multi Channel Adaptation*), previsto na especificação IEEE802.15.4e (2012), propõe um mecanismo genérico de tratamento do multicanal para protocolos assíncronos. O AMCA pode ser utilizado tanto para mecanismos iniciados pelo transmissor quanto iniciados pelo receptor. No caso desta tese, será considerado o algoritmo baseado no mecanismo iniciado pelo receptor da Figura 4. No AMCA, cada equipamento seleciona seu melhor canal de recepção baseado na qualidade da conexão local, que deve ser publicado para todos os vizinhos, e toda comunicação ocorre baseada no melhor canal do receptor. Por exemplo, na comunicação entre o transmissor T e o receptor R, considerando que T e R já conhecem previamente os melhores canais de cada um. Inicialmente, quando o transmissor T tem uma mensagem pendente para R, o transmissor salta para o melhor canal de R e aguarda a anúncio de R. Logo após a anúncio, T transmite o dado. No caso de ser necessário um reconhecimento na mensagem de dados, após R receber a mensagem pelo seu melhor canal, ele deve saltar para o melhor canal de T e transmitir o reconhecimento (DA). Após isso, os dois módulos deverão voltar a anunciar nos seus respectivos melhores canais. Em relação à vizinhança, para determinar o melhor canal de cada vizinho, cada módulo deve fazer uma varredura em todos os canais através do serviço de varredura ativa assimétrica multicanal. Neste serviço, para cada canal disponível, o equipamento deve chavear para o canal desejado e enviar um comando de AMCA-Beacon.Request. Em seguida, o transmissor deve habilitar a recepção e esperar por uma resposta por um período de duas vezes o período de varredura. Durante este tempo, o equipamento deve somente aceitar mensagens de anúncio e responder aos vizinhos. Decorrido o tempo do canal, o equipamento deve passar para um próximo canal e repetir o mesmo procedimento. Segundo a especificação o equipamento deve fazer este procedimento pelo menos duas vezes para garantir um maior sucesso na obtenção das informações do canal. Quanto ao melhor canal de cada módulo, os módulos periodicamente verificam seu melhor canal utilizando o comando MLME-SCAN.request para verificar as

condições do canal com seus vizinhos. Quando o equipamento decidir mudar seu melhor canal ele deve então enviar a mensagem H multicanal informando seu novo canal para todos os vizinhos.

O trabalho de Dutta et al. (2012) apresenta um protocolo RIT chamado A-MAC onde é proposto um mecanismo multicanal utilizando um reconhecimento inicial da mensagem de anúncio. O protocolo possui um canal específico para troca de mensagens iniciais e o restante dos canais para troca de dados. A proposta do A-MAC em uma comunicação *unicast* é baseada em um primeiro reconhecimento de forma rápida e determinística, e somente após esta primeira transação é que será enviada a mensagem de dados. Segundo a proposta do A-MAC o transmissor espera pela anúncio do receptor H, então o transmissor reconhece usando um HHA (*Hello Hardware Ack*). Após esta primeira comunicação, o transmissor aguarda um tempo pequeno e randômico para transmitir o dado. O reconhecimento da anúncio, chamado de *backcast*, permite um sincronismo da mensagem de dados com a anúncio do receptor, além de permitir, no caso de colisão um tratamento especial. Quando houver colisão, o receptor reenvia a anúncio de *backcast* (P-CW) com uma janela de contenção mais larga.

A abordagem do HHA traz alguns problemas dependendo da implementação. Primeiramente, o HHA é dependente da plataforma de hardware, sendo que cada processador tem a sua implementação. Consequentemente, o algoritmo não é totalmente portátil caso ocorra uma mudança de plataforma. Além disso a característica de implementação pode mudar de acordo com o hardware. Um exemplo de implementação na plataforma CC2538 da Texas, o *Hardware Ack* possui somente 5 bytes sendo formado por: *Frame Control Field* (2 bytes), *Data Sequence Number* (1 byte), *Frame Check Sequence* (2 bytes).

Uma segunda dificuldade observada no protocolo A-MAC é a necessidade de habilitar o bit Ack no header da anúncio do receptor (H). O bit de Ack não pode ser ativado em uma mensagem do tipo broadcast, segundo a própria especificação IEEE802.15.4e, (2012), pois pode aumentar o tráfego da rede com múltiplos reconhecimentos desnecessários na vizinhança. Desta forma a proposta de Dutta et al., (2012) é a criação de um mecanismo onde o transmissor primeiramente faz o papel de receptor enviando a anúncio com uma máscara endereçada para o receptor (HM). Quando o receptor perceber a necessidade da comunicação então ele envia o (H) com o bit de ack habilitado na mensagem. Para a implementação deste algoritmo na camada MAC, são necessários vários períodos de RIT para transmitir uma mensagem pendente. Outra forma seria concentrar toda esta troca de mensagens em um único período de

RIT que é a forma que será utilizada nesta tese. Na Figura 13 é apresentado um exemplo de implementação do tratamento da transmissão e recepção de uma mensagem de dados dentro de um período da comunicação.

Figura 13 – Comunicação entre dois módulos no protocolo A-MAC

Fonte adaptada: Dutta et al., (2012)

Os serviços de transmissão e recepção do protocolo A-MAC mostrados na Figura 13 possuem seis etapas. Primeiramente, o transmissor aguarda a anúncio do receptor (TxIL). Quando o receptor envia a anúncio broadcast (H), o transmissor envia outra anúncio (H1-TxHelloWithMask), onde uma máscara é colocada no endereço de destino sinalizando para o receptor a mensagem pendente. Na terceira etapa, o receptor envia uma nova anúncio (H2-RxHelloWithBitAck) com a máscara enviada pelo transmissor e com o bit de reconhecimento no cabeçalho da mensagem IEEE802.15.4 ativado. Na quarta etapa, o transmissor envia o Hardware Ack (HAA-HelloAutoAck) instantaneamente com um tempo em média de 200us garantindo, assim, a abertura da conexão entre transmissor e receptor. Na quinta etapa, o transmissor envia o dado após a espera de um tempo aleatório. Finalmente, caso a mensagem obtenha sucesso e dependente de um reconhecimento, é enviada uma nova anúncio como no protocolo RI-MAC. Caso tenha havido alguma colisão no HAA ou no dado o receptor envia uma mensagem de contenção (CW) indicando a necessidade de retransmitir o quadro usando um tempo de atraso (*backoff*). Os mesmos 6 estados são verificados para o serviço de recepção.

O protocolo A-MAC propõe também um mecanismo de *broadcast* onde a mensagem deve esperar uma anúncio do receptor para ser enviada. Neste caso para o transmissor apesar de existir uma mensagem broadcast pendente para ser enviada, ele continua mandando suas anúncios de RIT periodicamente (isto é feito somente com mensagens broadcast). Isto previne que quando outros vizinhos estiverem esperando também para enviar o broadcast fiquem aguardando por muito tempo. Para detecção da vizinhança o A-MAC utiliza um

mecanismo de detecção da vizinhança chamado *Pollcast*. Primeiramente, é enviado um quadro com um “predicado” usando broadcast que será recebido e avaliado pela vizinhança. Então, após um tempo reduzido, é enviado um endereço especial que estava no primeiro quadro. Por fim, os vizinhos que avaliaram o “predicado” como verdadeiro respondem com um reconhecimento.

O protocolo MR-MAC(*Multi-receptor*) proposto por Liao et.al., (2015), é um protocolo iniciado pelo receptor com trem de mensagens para redes de sensores acústicos subaquáticos (UWSN), que apresenta alta latência, e incerteza do espaço e do tempo. O MR-MAC utiliza o RIT com mecanismo RTS/CTS. O algoritmo possui duas fases: a reserva do canal e a transmissão de dados. Durante a inicialização, todos os módulos se revezam para transmitir alguns pacotes de controle para seus vizinhos. De acordo com as medidas de tempo de mensagens de controle fim-a-fim, um nó pode calcular seu atraso de propagação comparando os dados de tempo recebidos do seu vizinho. Na fase inicial de reserva de canal, a comunicação entre dois módulos é realizada através do mecanismo de ATR/TNI/MSP/*Data*. Inicialmente, o transmissor deve esperar a anúncio do receptor. Então, o transmissor envia uma mensagem (ATR- *ask-to-receive*) informando o ID do transmissor para os receptores. O receptor responde com um TNI. Vários receptores são previstos neste cenário onde um receptor é considerado principal na vizinhança. Assim, o transmissor envia uma mensagem (MSP- *Main Schedule Packet*), indicando o início da fase de transmissão de dados. O MSP possui as informações do transmissor e sequência de todos os receptores. O receptor envia automaticamente o pacote de dados baseado no escalonamento recebido. Este protocolo é baseado no mecanismo de RTS/CTS, similar ao protocolo ARM, onde existe uma quantidade grande de negociação para transmissão de uma mensagem de dados.

O protocolo DURI-MAC (KHALIL et.al., 2017), que é baseado no RI-MAC e utiliza canal duplo para minimizar o número de terminais escondidos e reduzir o número de colisões, também utiliza um gerador de tempo CCA exclusivo que melhora a taxa de entrega do sistema. O DURI-MAC utiliza um canal de controle e canais de dados. O canal de controle executa todo tipo de troca de pacotes de controle. O transmissor ao receber a anúncio no canal de controle, executa CCA exclusivo no canal de controle para determinar se o canal está ocupado ou livre. Se o canal for livre, o remetente começa a enviar seu pacote de dados no canal de dados. O CCA exclusivo é gerado através do Gerador Congruente Linear (LCG) para evitar colisão entre módulos vizinhos.

3.3 Comentários

A Tabela 2 resume as principais características de cada um dos protocolos mostrados acima. Ela foi adaptada do trabalho de FAFOULIS et.al. (2015), incluindo novos protocolos pesquisados além de outras características relevantes para este trabalho. Na Tabela 2 são mostrados os principais paradigmas encontrados nos protocolos assíncronos baseados em RIT e um resumo das técnicas utilizadas para solucionar o dado paradigma.

Tabela 2 - Principais protocolos iniciados pelo receptor pesquisados

Paradigma	Técnica	Protocolo
Mecanismo comunicação	Baseado no RIT Básico	RI-MAC, ADB, RC-MAC, PRADC-MAC, RW-MAC, DB-MAC, PW-MAC, YA-MAC, EM-MAC, SARI-MAC, AMCA, FERI-MAC, L-MAC
	Mecanismo de RTS-CTS	OC-MAC, ARM, SRI-MAC, MR-MAC
	Mecanismo Híbrido (SIT, RIT,síncrono)	EE-RI-MAC, SRI-MAC, AL-MAC
Escuta ociosa	RxIL	PRADC-MAC, A-MAC, FERI-MAC, SRI-MAC, EH-MAC
	TxIL	OC-MAC, RW-MAC, EE-RI-MAC, PW-MAC, EM-MAC, EM-MAC, Wide-MAC
	Adaptação da anunciação	SARI-MAC, FERI-MAC, L-MAC
Evitar colisão	Tempo atraso (Backoff) randômico.	RI-MAC, Wide-MAC, RC-MAC, ARM, YA-MAC, A-MAC, ERI-MAC, SRI-MAC
	Multicanal	A-MAC, ARM, EM-MAC, AMCA, MR-MAC, DURI-MAC
	Cooperação e Agregação de dados	OC-MAC, FERI-MAC, ERI-MAC
	Adaptação da anunciação	SARI-MAC, L-MAC
	Reserva do tempo de slot	SARI-MAC
	Extensão do CCA	AL-MAC
Tráfego da rede	Estimativa do tráfego,	RC-MAC, FERI-MAC, ERI-MAC, EH-MAC
	Ciclo de trabalho adaptativo	DB-MAC, SARI-MAC
Broadcast	Mecanismos broadcasts próprios	ADB, ARM, YA-MAC, A-MAC

Fonte adaptada: FAFOULIS et.al., (2015)

De acordo com a Tabela 2, é possível verificar que a grande maioria dos protocolos utiliza uma adaptação do mecanismo básico do RIT mostrado na Figura 4. Também é possível

ver que o objetivo principal dos algoritmos é a diminuição do IL e dos tratamentos de colisão para melhoria na taxa de entrega.

No contexto das métricas de comparação dos protocolos assíncronos, elas dependem do cenário de aplicação da WSN. Contudo, os protocolos assíncronos atuam principalmente em cenários onde o consumo de energia desempenha papel primordial. Desta forma, a principal métrica a ser atingida em cenários de protocolos assíncronos baseados em RIT é a diminuição do consumo de energia (CANO, 2011; DUTTA, et.al, 2012; LI, et.al., 2010). Uma métrica utilizada para medição do consumo é com base na obtenção de um modelo de energia baseado na medição de alguns módulos comunicando e então estimado o consumo baseado no modelo obtido (DUTTA et.al. 2012; LAMPIN, et.al, 2012). Outra forma utilizada, é através da obtenção do ciclo de trabalho, e posterior estimativa do consumo baseado no ciclo de trabalho (ZHUO, et.al., 2016; SUN, et.al., 2008; YANG, et.al, 2010; TANG, et.al., 2011; PEREZ, et.al.,2015). Outras métricas comuns para comparação do desempenho dos protocolos são obtidas pela medição da taxa de transferência, latência da rede e a taxa de erro (ZHUO, et.al., 2016; SUN, et.al., 2008; DUTTA et.al. 2012; YONG, et.al., 2011; TANG,et.al, 2011; LAMPIN, et.al.,2012; PEREZ,et.al.,2015).

Com relação aos cenários de uso dos protocolos assíncronos, segundo Cano et.al., (2011) embora aumentos esporádicos na carga da rede possam ocorrer devido a, por exemplo, detecções de eventos, as aplicações de WSN que utilizam mecanismos baseados em RIT, geralmente trabalham com tráfego tolerante a atrasos e com carga reduzida.

Quanto à abordagem de reconhecimento inicial proposto pelo protocolo A-MAC, ela se apresenta interessante, pois é possível uma diminuição do RxIL na comunicação. Também, o reconhecimento da anúncio permite um sincronismo da mensagem de dados com o receptor que pode diminuir as colisões. Este mecanismo é muito similar à proposta do protocolo X-MAC baseado em mecanismo iniciado pelo transmissor. Porém, a implementação do reconhecimento automático baseado no mecanismo de hardware do microcontrolador traz alguns problemas apresentados nesta seção. O protocolo ARM utiliza o mecanismo RTS/CTS que a princípio, resolve o problema de terminal escondido quando em ambiente multicanal. No entanto, a comunicação entre os módulos se torna carregada, com muitos pacotes sendo tráfegados. Quanto ao mecanismo multicanal, a proposta de um canal único de controle proposto por A-MAC e ARM, embora a implementação possa ser mais simples, torna-se um gargalo quando se tem muitos nós sensores na rede. Portanto, a abordagem de tratamento multicanal do AMCA, baseado em um mecanismo de melhor canal apresenta maior viabilidade pois não é dependente de canal de controle único.

Baseado nestes conceitos abordados nos capítulos 2 e 3, o próximo capítulo mostrará em detalhes o protocolo assíncrono baseado em RIT proposto nesta tese.

4 PROPOSTA PROTOCOLO RITMC

Este capítulo destina-se a apresentar o protocolo assíncrono multicanal baseado em transmissão iniciada pelo receptor chamado de RITMC. Esta tese é uma continuação dos trabalhos já realizados no Laboratório de Automação Industrial da Faculdade de Engenharia de São Carlos, que tem pesquisas na área de WSN em ambiente urbano em cenários como iluminação pública, como no trabalho de Fernandes et al., (2014). Atualmente, tem-se desenvolvido esforços para soluções de IoT que se agreguem aos cenários e às soluções já existentes. O protocolo RITMC proposto, refere-se ao aperfeiçoamento de protocolos consolidados em termos de pesquisa utilizando o padrão IEEE802.15.4e, visando cenários de aplicações de sistemas IoT em WSN de baixo consumo e sem a exigência de características rígidas de determinismo.

O protocolo RITMC proposto é uma mudança no atual mecanismo do A-MAC de reconhecimento inicial da anúncio do transmissor proposto por Dutta et. al., (2012) e de um mecanismo multicanal baseado no conhecimento do melhor canal, de acordo com o AMCA de IEEE802.15.4e (IEEE802.15.4e ,2012).

4.1 Mecanismo básico de acesso ao meio

O modelo básico de comunicação do protocolo RITMC em uma comunicação *unicast* é mostrado na Figura 14. Na figura é possível ver os estados do rádio na transmissão e na recepção de dados na comunicação entre um transmissor e um receptor.

Figura 14 – Comunicação entre dois módulos no protocolo RITMC

No exemplo da Figura 14, quando o transmissor possui uma mensagem pendente para enviar ao receptor, ele inicialmente liga o rádio para recepção (Rx) aguardando a anúncio (H) do receptor (TxIL). A mensagem de anúncio H^1 do receptor indica que a janela de comunicação está aberta. Em seguida, o transmissor envia uma mensagem de confirmação (HA – *Hello Ack*). Logo após o transmissor envia os dados usando o mecanismo de CSMA/CA. Além disso, o transmissor pode aguardar um reconhecimento de dados (DA), quando a mensagem precisa de uma confirmação, ou aguarda uma mensagem de erro (ECW), quando uma colisão foi detectada.

No projeto RITMC, o tempo para o envio do HA, *Time Ack* (TA), deve ser rápido o suficiente para que o receptor se sincronize com o transmissor. Desta forma, o RITMC não utiliza o mecanismo CSMA/CA. A mensagem HA^2 requer o tempo 770 μs em média para transmitir. Depois disso, o transmissor espera um atraso de tempo (TD) fixo para que o receptor se prepare para os dados. O TD mínimo é 1ms e a transmissão de dados utiliza CSMA/CA. Assim, o tempo TD varia de 1 a 9 ms.

Após o envio de H, o receptor deve esperar o início de comunicação de algum vizinho. O receptor somente aceita mensagens diferentes de H. Caso contrário, ele reprograma novamente o rádio para a espera de uma mensagem. O RxIL ou o estado de espera na recepção ocorre quando não há vizinho para comunicar. Como mencionado anteriormente, o RxIL é um dos principais gastos de energia na comunicação assíncrona, pois ele ocorre periodicamente dentro da rede (KIM et.al.,2015).

O fato de usar o mecanismo de reconhecimento inicial traz uma vantagem para o receptor, onde a janela RIT pode ser pequena o bastante, diminuindo significativamente o (RxIL). Isto traz uma grande economia de energia em redes com pouco tráfego. Por outro lado, o rádio transmissor terá um pequeno aumento no consumo em relação ao AMCA, pois terá mais uma transmissão e recepção. No entanto, se a rede tiver pouco tráfego, essa perda não será tão significativa quanto à economia da recepção.

¹A mensagem H possui 12 bytes e é formada pelos seguintes campos: cabeçalho IEEE802.15.4e (MHR) com endereçamento de 16 bits e tipo de frame igual a Command (CMD) (9 bytes); CFI (Command Frame Identifier (1 byte) onde o valor 0x1E representa o Hello e o CRC (2 bytes).

²A mensagem HA possui 12 bytes com os seguintes campos: MHR (IEEE802.15.4e header) 16 bits endereçamento (9bytes); CFI – Command Frame Identifier (1 byte) e CRC (2 bytes).

O algoritmo do RITMC *unicast* diverge do algoritmo do A-MAC principalmente usando o HA no nível da tarefa em vez do nível de hardware proposto por Dutta, et al, (2012). Esta abordagem do RITMC permite recursos importantes. Primeiro, o RITMC permite que seja enviada uma mensagem pendente para as camadas superiores do protocolo mais rapidamente do que no protocolo A-MAC. Quanto aos serviços de transição e recepção, o RITMC tem quatro etapas, enquanto o A-MAC tem seis. Outro fator relevante é o RITMC não possuir dependência de hardware. O quadro HA pode ser personalizado, diferente da HHA de A-MAC. Em seguida, o HA pode incluir o endereço de origem na mensagem, enquanto no auto-reconhecimento é restrito a somente 5 bytes e, portanto, o RITMC possui um melhor reconhecimento da mensagem quando ocorrer colisão.

4.2 Mecanismo de Tratamento de Erro

O mecanismo de erro (ECW – *Error Contention Window*) permite verificar possíveis falhas na transmissão da mensagem. O ECW é importante, pois pode indicar uma colisão e uma recuperação ainda na janela (*slot*) de transmissão. O mecanismo é realizado no receptor e pode detectar falha no HA ou no envio do dado. O algoritmo do tratamento do ECW pode ser separado em duas etapas, a identificação de um Evento ECW e o tratamento do ECW.

A detecção de colisão baseia-se no mecanismo proposto por Dutta et. al. (2012). O mecanismo de erro é executado sobre o receptor e pode detectar uma falha no HA, bem como na transmissão de dados como mostrado na Figura 14. O ECW é uma mensagem enviada pelo receptor para os transmissores indicando que ocorreu algum evento de erro. O transmissor, então deve inferir se deve reenviar ou não a mensagem pendente.

Identificação de um Evento de ECW

Durante a comunicação entre dois módulos sensores no padrão IEEE802.15.4, muitos fatores podem afetar a comunicação. O sinal pode sofrer influência de outros vizinhos, conhecido como problema do terminal escondido, ou a interferência no sinal causado por outras fontes, como o IEEE802.11.

Desta forma, o algoritmo de recepção deve detectar o erro e realizar uma análise da mensagem recebida. Desta forma, em vez de simplesmente recusar uma mensagem com erro, é possível verificar a qualidade das informações, tentando extrair informações úteis desta

mensagem dentro de uma janela de comunicação. O mecanismo de erro RITMC pode detectar entre dois eventos de erro: ECW1 e ECW2.

O ECW1 é ativado quando o tipo da mensagem é de reconhecimento (ACK). A informação do tipo está no primeiro byte do cabeçalho da mensagem IEEE 802.15.4 (IEEE802.15.4e, 2012). Sendo assim, mesmo se houver uma colisão da mensagem nos bytes restantes, inviabilizando partes da mensagem, o ECW1 geralmente poderá ser ativado, gerando uma sinalização de que possivelmente ocorreu uma colisão no HA.

O ECW2 é ativado quando o tipo de mensagem é de dados e já passou pelo HA. Neste caso, é desejável que a mensagem seja do endereço de origem que enviou o HA. Porém, caso não tenha sido ainda detectado um endereço de origem válido, é aceito qualquer dado.

Tratamento do Evento de ECW

Uma mensagem ECW somente será enviada se ocorrer um evento ECW1 ou ECW2 nas condições estabelecidas na Tabela 3. A coluna de ação na tabela mostra se a mensagem de erro precisa ser enviada e o status correspondente.

Tabela 3 - Identificação e tratamento dos diversos casos do evento ECW

	Condição	Significado	Ação
1	Ocorreu ECW1 e não ocorreu ECW2	Houve colisão no HA. Porém foi recebida uma mensagem de dados com sucesso.	Envio da mensagem ECW. Endereço de destino = endereço origem da mensagem de data. Status = OK
2	Ocorreu ECW1 e ECW2	Houve colisão tanto no HA quanto no DATA.	Envio da mensagem ECW. Endereço de destino = broadcast. Status = Reenvio com CW
3	Não ocorreu ECW1 e ocorreu ECW2	Houve colisão no DATA ou não foi recebido DATA dentro do tempo.	Envio da mensagem ECW. Endereço de destino = endereço de origem da mensagem HA. Status = Reenvio com CW
4	Não ocorreu ECW1 e ECW2	Não houve colisão.	Não enviar mensagem de ECW.

A mensagem ECW consiste basicamente de uma mensagem de 12 bytes enviada pelo receptor para os transmissores informando da ocorrência do erro. A Tabela 4 mostra a descrição da mensagem de ECW. O receptor deve tomar a ação de enviar o ECW somente quando foi detectado algum problema conforme descrito na Tabela 3. E neste caso deve aguardar uma nova mensagem dos transmissores. Quando o transmissor receber uma mensagem ECW, ele verifica

se deve reenviar, ou não a mensagem pendente, através do campo do endereço de destino e o status da mensagem.

Tabela 4 - Campos da mensagem de ECW

Campos	Tamanho (Bytes)	Especificação
FCS	2	IEEE802.15.4eMHR com FrameType=ACK
SeqNumber	1	
PAN	2	
AddrDest16bits	2	
AddrSource16bits	2	
Status	1	0 = Msg OK Maior que 0 → Msg com Erro
CRC	2	

Figura 15 – Exemplo de detecção de colisão no protocolo RITMC

A Figura 15 mostra um exemplo do algoritmo de tratamento de erro quando ocorre uma colisão dentro de um período de comunicação. Considere três módulos vizinhos M1, M2 e M3. Para cada módulo é mostrada a mensagem nas camadas superiores e também na camada MAC. Primeiramente, no instante (1), apenas M1 tem uma mensagem pendente para M2. Então, M1 aguarda a anúncio H de M2 e envia a mensagem de reconhecimento HA. No instante seguinte, M1 envia os dados e aguarda a resposta de M2. Então, M2 envia o reconhecimento

(DA) indicando que não ocorreu nenhuma colisão. No instante (2), M1 e M3 possuem mensagens pendentes para M2. Quando M2 envia H, tanto M1 como M3 enviam HA em (3), e neste caso, ocorre uma colisão em (4). No entanto, se o M2 puder descobrir a origem da mensagem, ele envia a mensagem ECW com o endereço de destino de M1 e status OK ou erro. Neste caso, M3 escuta a mensagem, descobre que não é para ele, e aguarda um novo H para reenviar a mensagem pendente. Para o M1, o status OK indica que ele não deve reenviar a mensagem. Quando a mensagem ECW não tiver endereço de destino, ambos os módulos sensores enviam dados novamente usando um período de atraso.

4.3 Mecanismo de tratamento Multicanal

A estratégia de multicanal RITMC proposta utiliza um mecanismo independente de canal de controle e com o conhecimento do melhor canal da vizinhança. Neste método, cada módulo descobre o melhor canal de cada vizinho e armazena esta informação internamente. Assim, quando um transmissor possui uma mensagem pendente, o módulo utiliza do seu conhecimento prévio do melhor canal do vizinho para se comunicar com ele. A estratégia de melhor canal é proposta por IEEE802.15.4e, (2012). Outros métodos, como A-MAC (DUTTA, et al., 2012) e ARM (LI, et al., 2010), são baseados no canal de controle, que pode se tornar um gargalo quando o número de módulos e o tráfego na rede aumentam. O algoritmo RITMC proposto traz melhorias no método AMCA.

Na inicialização da rede, é necessário descobrir a vizinhança e os respectivos melhores canais dos vizinhos. Os serviços de inicialização da rede serão detalhados na próxima seção. Após a inicialização da rede e todos os vizinhos descobrirem os melhores canais de sua vizinhança, a comunicação consistirá em cada vizinho anunciar H periodicamente no seu melhor canal e esperar por um reconhecimento HA neste canal. Quando um transmissor tem uma mensagem pendente, ele deve saltar para o melhor canal do receptor e aguardar a anúncio. Em seguida, o transmissor enviará um HA e, posteriormente, os dados.

A Figura 16 mostra um exemplo de manipulação multicanal com três módulos no protocolo RITMC. No exemplo, M1, M2 e M3 são vizinhos e já conhecem o melhor canal (BC) da vizinhança. Primeiramente, no instante (1), M1 tem uma mensagem pendente para M2. Então, M1 liga o rádio no melhor canal de M2 ($BC = 18$) e aguarda pela mensagem de anúncio de M2. No instante (2), M2 anuncia H no canal 18. No instante seguinte, em (3), M1 envia uma mensagem HA, e posteriormente em (4), depois de decorrido o tempo T_d , M1

envia a mensagem de dados. Caso a mensagem precise de reconhecimento, M2 envia a mensagem DA para M1 em (5). Caso houver colisão, M2 envia a mensagem de ECW. Por fim, M1 e M2 voltam a se anunciar em seus respectivos melhores canais. Nos instantes seguintes, de (6) até (10) M1 tem uma mensagem pendente para M3. Então o procedimento se repete agora no canal 19.

Figura 16 – Comunicação de três vizinhos usando RITMC multicanal

Com relação à comunicação *broadcast*, ela é importante dentro do protocolo para atender uma variedade de serviços, como a descoberta de vizinhos, rotas de atualização e anúncios comuns como dados de alarme. Para os protocolos assíncronos não é possível enviar uma mensagem *broadcast* uma única vez para cada vizinho, e a própria especificação IEEE802.15.4e (2012) não recomenda isso. Desta forma, o mecanismo de transmissão é semelhante ao *unicast*, ou seja, a mensagem é enviada repetidamente para cada um dos vizinhos conhecidos. A forma adotada neste trabalho é somente mandar mensagem *broadcast* para os elementos que estão ativos na *LiveList*.

4.4 Serviços Multicanal

O algoritmo RITMC proposto traz melhorias no método AMCA (IEEE802.15.4e, 2012). Os serviços de descoberta da vizinhança e escolha do melhor canal já são previstos por IEEE802.15.4e (2012), porém de forma genérica, sem detalhes específicos para o tratamento do protocolo assíncrono baseado em RIT. Desta forma, esta tese define completamente os principais serviços para protocolos assíncronos multicanais baseados em RIT com o mecanismo de escolha do melhor canal. Os serviços propostos nesta tese são *LiveList*, *DeadList*, a descoberta da vizinhança e escolha do melhor canal.

A *Livelist* é a lista atual de módulos “vivos” na vizinhança do módulo. O serviço de *Livelist* é importante em vários aspectos dentro do algoritmo. Primeiramente, garante a lista de módulos ativos para a execução dos comandos broadcast e nos serviços do protocolo, como na mudança de canal. Desta forma, somente os módulos ativos, ou que se mostraram ativos nos últimos ciclos de trabalho, precisam ser tratados normalmente pelo respectivo módulo sensor. Segundo, é possível, na comunicação entre transmissor e receptor, verificar o melhor canal do vizinho e seu respectivo *rank*. Também, este mecanismo permite uma atualização dos parâmetros da rede, como o tempo do HÁ, e adicionalmente o atraso da comunicação entre transmissor e receptor devido ao envio do delta entre transmissão e recepção.

No serviço de *LiveList*, o módulo sensor analisa apenas os vizinhos que já são conhecidos. Neste caso, não só o status é atualizado, mas também o melhor canal do transmissor e também do receptor. O serviço é realizado periodicamente a cada 10 segundos para cada módulo.

O serviço de *LiveList* é realizado por um comando de *LiveList* (LL) com confirmação enviada periodicamente por cada nó da rede para verificar a atividade da vizinhança. O comando usado para executar este serviço é baseado no comando de anúncio com confirmação (H) proposto por IEEE802.15.4e (2012). A mensagem de requisição possui 15 bytes e é mostrada na Tabela 5. Os primeiros 9 bytes são o cabeçalho IEEE802.15.4e da mensagem. O campo CMD indica que é um comando de *LiveList* (0x2E). Para o campo HelloReq, o valor 1 indica que é uma requisição com confirmação enquanto o valor 0 significa que este comando de *LiveList* não necessita de confirmação. Os campos BC e chRank indicam o atual melhor canal e o respectivo rank do transmissor. O *rank* é um índice da qualidade do canal e será mostrado no serviço de escolha do melhor canal. A mensagem de resposta é análoga à requisição do transmissor.

O mecanismo da *LiveList* é baseado em somente um comando por período de RIT. A implementação do mecanismo dentro do OpenWSN foi realizada através de uma tabela de *LiveList* gerenciada por um sistema de pesos, onde a mensagem é enviada para o vizinho com menor peso naquele ciclo.

Tabela 5 - Campos da mensagem de LiveList

Campos	Tamanho (Bytes)	Especificação
FCS	2	IEEE802.15.4e MHR Com FrameType=CMD
SeqNumber	1	
PAN	2	
AddrDest16bits	2	
AddrSource16bits	2	
CMD=0x2E (LL - Livelist)	1	IEEE802.15.4e e RITMC
HelloReq = 1 ou 0.	1	IEEE802.15.4e
BC=melhor canal	1	RITMC
ChRank=Rank	1	RITMC
CRC	2	

O algoritmo da transmissão e recepção é mostrado na Figura 17. O funcionamento do serviço de transmissão de *LiveList* começa quando ocorre o tempo de tratamento. Inicialmente é procurado pelos vizinhos que estão com menor peso. Isto por que somente um comando LL é enviado por ciclo. Então, o módulo envia o LL para o vizinho com menor peso no melhor canal deste vizinho. Quando o vizinho enviar o reconhecimento da mensagem, é incrementado o peso do respectivo módulo. Se o vizinho não responder, somente é incrementado o número de retransmissões. Neste caso, este módulo terá prioridade sobre os demais no próximo ciclo. Caso seja ultrapassado o número de retransmissões, o módulo passa a não fazer mais parte da *LiveList* e passa a fazer parte da *DeadList*. Caso seja recebido o comando de *LiveList* de outro módulo vizinho, isto também atualiza a tabela de *LiveList* incrementando o peso, fazendo com que diminua o número de comandos da rede.

Figura 17 – Algoritmo de transmissão e recepção do serviço de LiveList

Um serviço complementar à *LiveList*, é a *DeadList* ou lista dos equipamentos “não vivos”. A *DeadList* é uma lista de equipamentos que não estão na *LiveList*, porém já tiveram alguma comunicação com o módulo sensor. Ele é composto dos módulos que falharam na comunicação da *LiveList* ou que já se anunciaram alguma vez e não estão ainda na *LiveList*. O mecanismo da *DeadList* segue o mesmo princípio da *LiveList*, com o mesmo comando conforme mostrado na Tabela 5.

O funcionamento do serviço de *DeadList* ocorre em um período bem mais longo que a *LiveList*. No momento da execução do serviço, é enviada a mensagem LL para o endereço de vizinho que está na tabela *DeadList*. Quando um módulo responde à mensagem LL, ele passa a fazer parte da *LiveList*. Caso o módulo não responda, é incrementado o número de

retransmissões. Quando for ultrapassado o número de retransmissões, o módulo é removido da *DeadList* e precisará passar novamente pelo serviço de descoberta da vizinhança para entrar na rede. No momento de executar a *DeadList*, também serão zerados os pesos da *LiveList*. Desta forma, todos os módulos voltam a ter o mesmo peso na comunicação. A *DeadList* é executada a cada dez comandos de *LiveList*.

O serviço de descoberta da vizinhança é essencial para construção da rede e para inclusão de novos vizinhos na lista de cada módulo. O serviço, consiste em o módulo saltar para cada canal disponível e aguardar pelo anúncio de todos os módulos vizinhos daquele canal. Durante a execução do serviço, cada anúncio de vizinho, que não faz parte ainda da *LiveList* do correspondente módulo, é colocado na *DeadList*. Na inicialização do módulo, logo após a energização ou o *reset*, o serviço de descoberta da vizinhança é executado em sequência para todos os canais da rede. Após esta primeira fase, o serviço passa a ter uma periodicidade reduzida, executando um canal por vez e com periodicidade a cada quinze períodos de *LiveList*.

O serviço de escolha do melhor canal (BC) coordena a escolha e a notificação na mudança de canal para a vizinhança do módulo. Ele pode ser dividido em três etapas distintas mostradas a seguir, mostradas na Figura 18.

A primeira etapa do serviço de escolha do melhor canal é a verificação da necessidade da mudança de canal. A checagem do melhor canal é realizada na transmissão da *LiveList*. Toda vez que for enviado um comando de *LiveList*, é recalculado o *rank* do correspondente módulo. Neste momento é verificada a necessidade de trocar de canal. Como mostrado anteriormente na Tabela 5, no serviço de *LiveList* os vizinhos publicam o seu BC e o *rank*. Esta informação é utilizada como subsídio para estimativa da qualidade do canal.

O *rank* informa a qualidade atual do canal, visto pelo módulo. O *rank* é função do número de vizinhos no canal e da taxa de erros de pacotes enviados dentro do tempo da *LiveList*. O cálculo do *rank* é realizado através da seguinte equação:

$$Rank = (NV/TOTV) * p1 + (TER) * p2 \quad (3)$$

Sendo, NV corresponde o número de vizinhos no canal, TOTV é o número total de vizinhos vivos do módulo na tabela da *LiveList*. O parâmetro TER representa a taxa de erro de pacotes no período de *LiveList*. Os parâmetros $p1$ e $p2$ são pesos para a taxa de vizinhos no canal e taxa de erro respectivamente.

Figura 18 – Algoritmo do serviço de mudança de melhor canal

Para o cálculo da taxa de erro é considerado todo comando que tiver reconhecimento, incluindo RPL.DAO, COAP e LL. Para uma eventual mudança de canal é desejável que a taxa de erro tenha uma influência maior do que a quantidade de vizinhos no canal. Os valores de peso utilizados foram $p1$ igual a 2 e $p2$ igual a 10. De acordo com a equação (3), quando houver pouco erro para o módulo em um determinado canal, mesmo se tiver uma concentração de módulos no canal, isto não indicará que a comunicação esteja ruim, desta forma o valor do *rank* é pequeno. Porém, quando existirem muitos módulos no mesmo canal, a taxa de erro aumenta, aumentando o valor do *rank*. A mudança de canal ocorre quando o valor do *rank* é superior a

um limite. Valores menores que o limite indica que o canal está estável e não tem necessidade de alteração. A necessidade da mudança é baseada no *rank* do próprio módulo. O valor de limite de *rank* estabelecido foi 5.

Após o módulo decidir que deve ser realizada a mudança de canal, o módulo precisa determinar qual o melhor canal para a mudança. Neste caso, é montada uma tabela com todos os canais e a soma de todos os *rank*s dos vizinhos para cada canal. Valores de *rank* menores indicam canais menos ocupados. Para o caso de todos os módulos que estão em um canal ruim desejarem mudar de canal ao mesmo tempo e não optarem pelo mesmo canal, uma escolha randômica é realizada com todos os canais que tiverem valores menores dentro de uma faixa de *rank*. Nesse momento, apesar do módulo já ter decidido qual o canal ele deseja mudar, ainda ele não pode mudar de canal antes de anunciar este novo canal.

A próxima etapa do serviço de mudança de canal é a fase de notificação. Isto é realizado através do comando de mudança de canal (CBC) mostrado na Tabela 6. Neste caso, o transmissor envia a mensagem CBC para cada vizinho da *Livelist*, informando qual é o novo melhor canal que ele passará a comunicar (NewBC). Este comando precisa de confirmação, pois deve ser enviado para todos os vizinhos ativos do módulo antes da troca de canal.

Tabela 6 - Campos da mensagem de Mudança de Canal

Campos	Tamanho (Bytes)	Especificação
FCS	2	IEEE802.15.4e MHR Com FrameType=CMD
SeqNumber	1	
PAN	2	
AddrDest16bits	2	
AddrSource16bits	2	
Comando=0x2F (CBC - Mudança de canal)	1	IEEE802.15.4e e RITMC
LastBC	1	RITMC
NewBC	1	RITMC
CRC	2	

Por fim, a última etapa do serviço de mudança de canal é a mudança do canal propriamente dita, onde o módulo passa a se anunciar no novo melhor canal.

A seguir é mostrado um exemplo de funcionamento dos serviços de LiveList, DeadList, descoberta da vizinhança e a escolha do melhor canal do protocolo RITMC para inicialização de uma rede.

Considere uma rede com três sensores e três canais conforme mostrado na Figura 19. Os sensores são os módulos M1, M2 e M3. Os canais disponíveis são 17, 18 e 19. No exemplo

somente será mostrada a análise do módulo M1, porém os demais seguem o mesmo princípio. Para cada módulo existem três listas internas: a *LiveList*, responsável pelas informações dos módulos ativos atuais; a *DeadList*, que é a lista de equipamentos existentes que não estão na *LiveList*, e a *ChannelList*, que é uma tabela auxiliar com as informações de todos os canais. Por facilidade de compreensão, as informações do correspondente módulo estão na *LiveList*.

Na inicialização do módulo sensor, ele escolhe um melhor canal (BC) de forma aleatória segundo o critério do menor *rank*. Considerando que, no início, os canais têm *rank* zero, e os módulos escolheram aleatoriamente canais 17 e 19 conforme mostrado na Figura 19 (M1.BC = 19, M2.BC=19, M3.BC = 17).

Figura 19 – Cenário 1 dos serviços na comunicação do RITMC

O serviço de descoberta da vizinhança, varre todos os canais ouvindo todos os módulos que estão em cada canal, e inclui na *DeadList*. De acordo com a Figura 19, M1 descobriu os módulos M2 e M3 e incluiu os seus respectivos endereços e o BC na *DeadList*. Após a inicialização do módulo, este serviço é realizado com maior prioridade antes de qualquer outro serviço, e em sequência para cada canal, para construção das tabelas da vizinhança. Após a primeira inicialização, este serviço passa a ter periodicidade menor.

Quando ocorre o evento da *LiveList*, não é enviado comandos pois não existe elementos na *LiveList*. Decorrido o tempo do serviço da *DeadList*, o vizinho, no caso M1 envia o comando LL para cada elemento na rede. Caso o vizinho responda, este vizinho passa a fazer parte da *LiveList* conforme mostrado na Figura 20. No exemplo, tanto o módulo M2 quanto o módulo M3 responderam ao comando, passando então para a *LiveList*.

Figura 20 – Cenário 2 dos serviços na comunicação do RITMC

Os próximos eventos de *LiveList* que ocorrerem, gerarão uma atualização no *rank* dos módulos. Ainda na Figura 20, o módulo M1 no tempo 2 apresentou uma taxa de erro de 10%. Isto fez com que o *rank* do módulo aumentasse, já que existiam dois módulos no canal. Esta combinação gera um *rank* de 2.33, inferior ao limite de 5 e não é suficiente para a mudança de canal. Porém, no tempo 3 da Figura 25, M1 apresentou uma taxa de erro de 40% e então o valor do *rank* foi para 5.33.

Como o *rank* de M1 é superior a 5, resulta na necessidade de mudança de canal. A escolha do melhor canal segue o critério de comparação dos *ranks* dos vizinhos para cada canal. Baseado na informação dos vizinhos, é calculado o *rank* total do canal e colocado na tabela *ChannelList*. São considerados como canais candidatos todos os canais com *rank* menores que 0.5. Ainda na Figura 20, na tabela *ChannelList*, no tempo 3, o módulo M1 escolhe o canal com menor *rank* onde o único canal é o 18. Desta forma, o módulo M1 decide saltar para o canal 18.

Para o módulo M1 fazer a mudança de canal, ainda é necessário a anúncio para os seus vizinhos. Desta forma, o M1 continua anunciando no seu antigo canal até a completa anúncio do novo canal para todos seus vizinhos. A Figura 21 mostra este cenário.

Figura 21 – Cenário 3 dos serviços na comunicação do RITMC

De acordo com a Figura 21, ainda no serviço de escolha do melhor canal, o módulo M1 espera a anúncio de M2 no seu melhor canal (Canal 19) para notificar a mudança de canal. Então M1 envia a mensagem CBC para M2 e espera a confirmação. No instante seguinte, ele espera a anúncio de M3 no seu melhor canal (Canal 17). Então M1 envia a mensagem CBC para M3 e espera a confirmação. O serviço de escolha do melhor canal tem um campo de pendente na tabela da *LiveList* para saber se foi enviado ou não. Caso o envio do comando tenha sucesso, o campo de pendente é limpo. Caso o módulo não responda ele faz retransmissões. No caso de o vizinho não responder pela quantidade de retransmissões ele retira o módulo da *LiveList* e considera o comando enviado. Somente depois de ter enviado o comando CBC para todos os elementos da *LiveList* é que M1 passa a trabalhar no seu novo canal 18.

5 MÉTODOS E MATERIAIS

Este capítulo destina-se a descrever as características de implementação e de validação do sistema IoT que serão utilizadas nesta tese. Conforme apresentado no capítulo 2, um sistema IoT é composto de vários níveis e para cada um destes níveis existem vários protocolos envolvidos. Esta tese utiliza o ambiente de desenvolvimento de IoT do OpenWSN.

O OpenWSN é uma plataforma de ambiente aberto que segue os principais padrões atuais para uma solução completa de Internet das Coisas. O objetivo principal do OpenWSN é mostrar a viabilidade de soluções de IoT em diferentes aplicações industriais e acadêmicas (WATTEYNE et al., 2012).

5.1 Arquitetura do OpenWSN

A arquitetura da rede do OpenWSN possui três níveis principais que são o nível do cliente remoto, o nível de *gateway* ou roteador de borda (LBR), e o nível da rede de sensores. Um exemplo destes diferentes níveis é mostrado na Figura 22.

Figura 22 – Arquitetura OpenWSN de IoT

Fonte adaptada: Watteyne et al., (2012)

No **nível do cliente remoto**, a aplicação remota acessa a rede sem fio através de uma aplicação web. No ambiente OpenWSN o papel da aplicação é realizado através do protocolo HTTP/CoAP, através de aplicações simples de sensoriamento de dados via Internet. Este acesso é realizado através do IP do módulo sensor e passado pelo LBR para acessar a rede.

Segundo (WATTEYNE et al.,2012) as aplicações típicas de IoT podem ser mapeadas em três principais cenários:

- **Leitura das características do sensor:** Este cenário geralmente feito no início da comunicação é um pedido para o nó de rede enviar a lista de recursos disponíveis pelo nó sensor. Por exemplo, em um caso de uma rede urbana a aplicação pode pedir para um determinado módulo sensor qual são os recursos disponíveis. O nó pode retornar uma lista de parâmetros que será possível ser operado pela aplicação remota.
- **Coleta de dados da rede:** Uma aplicação no nível do cliente remoto periodicamente coleta informações de uma determinada área. Isto é feito pela aplicação através do protocolo CoAP através da porta UDP de um servidor de dados pela internet. Um exemplo deste caso de uso é o pedido de diagnóstico de uma lâmpada em um poste em uma rede urbana. O sistema requisita a informação se um determinado ponto de luz está aceso ou apagado. Neste caso a aplicação envia um comando GET para ler a informação do nó sensor: `coap://ipv6::addr/light/`.
- **Atuação em um determinado ponto da rede:** Neste caso a aplicação remota deve enviar sob demanda um determinado dado para um módulo específico ou para uma área da rede. O cliente remoto na internet envia através do protocolo CoAP um comando PUT na porta UDP.

O cenário de aplicação do sistema IoT do OpenWSN é baseado na comunicação P2MP. Ou seja, as informações da aplicação sempre seguem o caminho divergente, onde a aplicação faz uma requisição para a WSN. Na versão utilizada nesta tese não é possível testar o cenário convergente ou MP2P, onde a rede envia informações diretamente para a aplicação sem necessidade de uma requisição da aplicação.

No **nível do gateway**, o módulo LBR é o módulo responsável pela interface entre a rede sem fio e a rede ethernet. Conforme mostrado no capítulo 2, a base de toda a comunicação IoT é o IPv6. O protocolo 6LoWPAN permite a WSN ter um endereço ao protocolo IPV6 de forma reduzida, preservando somente as informações mais importantes. O LBR é o elemento responsável por fazer a conversão entre estes dois protocolos de rede IPV6 e 6LoWPAN.

Ainda no contexto do LBR, o OpenWSN implementa um software chamado OpenVisualizer que interage com o DAGROOT através da comunicação serial RS232 e que também completa o conjunto de camadas de interface entre a ethernet e a rede de sensores sem fio. Em adição, o OpenVisualizer traz informações importantes de diagnóstico da rede.

O LBR foi desenvolvido em linguagem Python e tem o código multiplataforma. (OpenWSN, 2018). O OpenVisualizer é o responsável de conversão dos protocolos web da aplicação no protocolo de comunicação da rede IEEE802.15.4.

O LBR necessita de um módulo de rádio para comunicação na WSN no padrão IEEE802.15.4. Desta forma, o LBR é acoplado a um módulo de rádio que faz o papel da DAGROOT, ou raiz da rede RPL.

O nível da rede de sensores é formado por elementos de baixo consumo e de baixa potência. Estes possuem a informação a ser coletada e a comunicação sem fio. O módulo sensor que coleta a informação sistema (sensor ou atuador) pode estar na mesma placa de comunicação da rede sem fio ou em placas distintas, onde neste último caso ele utiliza comunicação serial para troca de informações dentro da rede. Com relação à pilha de protocolos do módulo sensor, o OpenWSN segue os protocolos da Tabela 1, na qual na camada MAC é utilizado como padrão o protocolo síncrono TSCH, conforme mostrado na Figura 22. Os detalhes de implementação dos protocolos serão mostrados na próxima seção.

Quanto ao hardware da WSN, o código OpenWSN já foi testado em diferentes plataformas como GINA e TelosB utilizando processador MSP430 da Texas Instruments, LPC, CC2538, STM32W utilizando cpu ARM das empresas NXP, Texas Instruments e ST, K20 utilizando CPU K20Dx da Freescale, entre outras (OpenWSN, 2018).

5.2 Pilha de Protocolos do OpenWSN

A plataforma OpenWSN é uma implementação de código aberto de uma pilha de protocolos baseado nos padrões mais atuais sobre IoT. As camadas inferiores são baseadas na especificação IEEE802.5.4e (2012). As camadas superiores seguem os padrões de IoT especificados pelo IETF, tais como 6LoWPAN, RPL e CoAP de acordo com a Tabela 1.

É importante ressaltar que esta tese é baseada em uma versão da plataforma do OpenWSN de 2015, onde foi a base de todo o desenvolvimento. Muitas alterações foram realizadas pelo OpenWSN em paralelo com os trabalhos desta tese. A versão mais recente da

plataforma inclui a implementação do 6TiSCH, além de aspectos de segurança da rede, que foram apresentados no capítulo 2, porém como não é objetivo desta tese a análise do protocolo 6TiSCH e da segurança da rede, não foi necessário a integração desta versão.

O código fonte do módulo sensor na WSN do OpenWSN é formado por sete camadas, onde cada camada se interliga com as outras através de uma camada de interligação (“*cross-layer*”). A Figura 23 mostra a implementação do OpenWSN com a adaptação realizada nesta tese.

Figura 23 – Arquitetura do Firmware do OpenWSN com o RITMC

Fonte adaptada: OpenWSN, (2018)

De acordo com a Figura 23, em verde estão representados os módulos que foram substituídos no código original do OpenWSN e em amarelo os módulos que somente precisaram uma pequena adaptação no código. Os demais módulos foram mantidos conforme o código original. A seguir será detalhada cada uma das camadas do OpenWSN.

- PHY - A camada física de acesso ao meio para as redes de sensores sem fio já está bem consolidada na literatura, e hoje existe uma gama de equipamentos comerciais que pode ser utilizada nas aplicações de redes sem fio. As plataformas de hardware mais comuns já possuem soluções do protocolo IEEE 802.15.4 (2012) implementadas e também já trazem diversas melhorias disponíveis no hardware.

Em relação ao código do protocolo RITMC proposto nesta tese, dentro do módulo do rádio foi necessária a implementação do mecanismo de CSMA/CA para a plataforma de hardware, que não é utilizado pelo protocolo TSCH. Também foi necessária a adequação dos temporizadores para inclusão do CSMA/CA.

- MAC_LOW - A camada de enlace na plataforma OpenWSN foi dividida em parte alta e parte baixa. Especificamente na camada inferior é implementado o protocolo TSCH (IEEE802.15.4e, 2012) para o acesso ao meio. A base do mecanismo do TSCH é o tempo de slot que permite a comunicação entre os módulos sensores. O sincronismo acontece entre os vizinhos que se ajustam de acordo com a mensagem recebida e um coordenador central gerencia o sincronismo.

Em relação ao código do protocolo RITMC, na camada MAC_LOW, todo o mecanismo do protocolo TSCH foi substituído pelo protocolo RITMC responsável pelo mecanismo de transmissão e recepção da mensagem. Da mesma forma que o RITMC, também foram implementados os outros protocolos assíncronos analisados neste trabalho: AMCA, A-MAC e ARM.

Ainda no contexto da camada MAC_LOW, o mecanismo de funcionamento do OpenWSN é baseado no tempo de slot. Primeiramente, o algoritmo deve processar as mensagens pendentes das camadas superiores, empilhadas através do módulo *OpenQueue* e enviá-las pelo rádio para a WSN. Quando uma mensagem é recebida pelo rádio e o destino é deste módulo, ela segue o sentido oposto. Desta forma, uma mensagem pendente a ser enviada na rede deve ser processada no mesmo tempo de slot. Caso não for possível processar neste tempo, ela deve ser empilhada e enviada de volta para as camadas de origem. As camadas superiores são responsáveis por controlar o reenvio desta mensagem de acordo com critérios internos.

- **MAC_HIGH** - A camada de escalonamento ou 6TiSCH é responsável por fazer funcionar a comunicação através do estabelecimento dos tempos dos links dentro do slot. O *SixTop* é o módulo que conecta o TSCH com o 6LoWPAN e fornecer informações mais determinísticas para as camadas superiores e para a rede como um todo. Ele é responsável ainda pela coleta de informações de diagnóstico para ser utilizado pelo roteamento. Em um cenário com alta escalabilidade, ele é responsável por prover o sincronismo entre os vários coordenadores (roteadores de borda) através de coordenador de mais alto nível (PCE). A camada de escalonamento é responsável, adicionalmente, por manter a tabela de vizinhos e suas estatísticas que fornecem subsídios para a decisão do roteamento RPL.
- **IPHC** - A camada de adaptação é realizada através do protocolo 6LoWPAN que serve para compactar a mensagem do protocolo IPV6, adaptando-o para a mensagem IEEE802.15.4. A especificação 6LoWPAN define um conjunto de regras para ser analisadas no IPV6, e que remove os campos que não são necessários e comprime outros campos importantes. Como a aplicação somente conhece o protocolo IPV6, é necessário o LBR que faz o papel de converter a mensagem de um IPV6 para 6LoWPAN e vice-versa. Através do uso do 6LoWPAN cada nó da rede de sensores pode ser assinalado com um único endereço IPV6 e aparece na internet com um equipamento qualquer no protocolo IPV6. Isto possibilita que, do lado da cliente, o módulo sensor seja visto como qualquer outro ponto da internet. Neste nível do IPHC também existe a interface com o módulo *OpenBridge*, que faz o papel de ponte entre o módulo sensor e o OpenVisualizer, quando o módulo é o DAGROOT.
- **IPv6** – Esta camada é responsável pelo protocolo gerenciamento do RPL. Na pilha do OpenWSN, o RPL é usado no topo do 6LoWPAN para prover o roteamento da rede. O RPL tem um mecanismo de coleta de informação baseado no gradiente da rede. A métrica do OpenWSN é o inverso da taxa de entrega. Outro mecanismo do RPL é o roteamento do destino que é mantido pelo OpenVisualizer. O OpenVisualizer possui uma tabela com a rota de cada um dos possíveis destinos na rede. Esta tabela é atualizada periodicamente pelos comandos do RPL enviados na rede (WATTEYNE et al., 2012).
- **TRANS e APP** - Nos últimos níveis da pilha de protocolos do OpenWSN estão a camada de transporte, que utiliza o protocolo UDP e na camada de aplicação o CoAP. O CoAP possibilita a interação *RESTful* com os módulos sensores, através do HTTP. Um módulo sensor através do CoAP pode atuar como *Web Server*.

Em relação a implementação dos protocolos assíncronos, como o objetivo era obter uma mudança mínima para minimizar os impactos no código, optou-se por não utilizar fila de mensagens pendentes na camada MAC. O código dos protocolos assíncronos se adaptou ao mecanismo de tratamento de mensagem por interrupção e dentro de um tempo de slot, como foi implementado no protocolo TSCH. Para isso, utilizou-se apenas duas interrupções importantes, de temporização e de fim de mensagem. Desta forma, toda mensagem que não fosse enviada em um determinado período era sinalizada para as camadas superiores e, assim realizadas as retransmissões. No caso das implementações de algoritmos específicos do protocolo RITMC, como os mecanismos de *LiveList* e *DeadList*, o tratamento foi realizado no próprio nível da camada MAC.

Com relação aos módulos de escalonamento de tarefas (*Schedule*), de tratamento de mensagens das camadas superiores (*Sixtop*) e de vizinhança (*Neighbors*), eles precisaram sofrer pequenas alterações para adequação do protocolo assíncrono. O mecanismo de vizinhança da *LiveList* e *DeadList* possui mecanismos próprios de vizinhança independente do mecanismo *Neighbors* referente ao protocolo RPL. Ou seja, primeiramente o módulo deve ser reconhecido como vizinho da camada MAC para posteriormente ser reconhecido como vizinho do RPL.

Nas Figuras 23 e 24 são mostrados os fluxogramas de estados de um período do protocolo RITMC para transmissão e recepção de mensagens. De acordo com as figuras, uma mudança de estado pode ocorrer quando houver um evento de Timer (T), que representa o estouro do tempo que foi programado anteriormente. A mudança de estado pode ocorrer também devido a um evento de final de mensagem (F). Neste caso, significa que ocorreu uma interrupção devido a uma mensagem que acabou de ser transmitida com sucesso, ou de uma mensagem que foi recebida. Quando não existir (T) ou (F) dentro de um determinado estado, significa que foi utilizada chamada direta da função.

A Figura 24 mostra o diagrama de estados do serviço de transmissão de mensagens. O serviço de transmissão tem prioridade sobre o serviço de recepção. O serviço se inicia quando ocorre a interrupção de início do *slot* de tempo, e existe uma mensagem pendente em *OpenQueue* que é encaminhado para (Tx00 – *TxInit*). Então, começa a preparação para o estado de espera de uma mensagem de anúncio do vizinho (Tx02 - *RxHelloPrepare*). Em (Tx03- *RxHello*) o rádio é ligado, e um temporizador é programado para o tempo de TxIL. Caso ocorra um estouro de tempo (TxIL) sem receber mensagens em (Tx03- *RxHelloTimeout*), então o rádio notifica as camadas superiores e retorna para o estado de repouso (*EndSlot*). Quando o rádio receber uma nova mensagem em (Tx04- *TxHAPPrepare*), ele analisa a mensagem. Se a mensagem é uma anúncio do endereço que o transmissor possui uma mensagem pendente,

então será enviada em (Tx05- *TxHA*) a mensagem HA sem o mecanismo de CSMA/CA. Em seguida, em (Tx07- *TxData*) após o tempo T_d , o transmissor envia o dado utilizando CSMA/CA. Em (Tx08- *RxCwAckPrepare*) é o início da preparação para receber uma mensagem ECW ou DA. Em (Tx09- *RxCwAck*) o algoritmo habilita o rádio a espera uma nova mensagem. Se for recebido uma mensagem em (Tx0a- *TxCworDA*) é verificado que tipo de mensagem se trata. Caso for uma mensagem de erro (ECW), ele se prepara para novamente enviar o dado, retornando para (Tx07- *TxData*). Se for uma mensagem de DA então o algoritmo notifica as camadas superiores e termina o slot (*EndSlot*) retornando o controle para o sistema operacional e aguardando o início de um novo slot de tempo.

Figura 24 – Diagrama de Estados do serviço de transmissão do RITMC

A Figura 25 mostra o diagrama de estados do serviço de recepção. A recepção se inicia quando ocorre o tempo do início de um período, e não existe uma mensagem pendente. Neste caso, o algoritmo é encaminhado para (Rx00-*TxHelloPrepare*). Em seguida, é enviada uma mensagem de anúncio H, indicando o início da janela de RIT em (Rx01-*TxHello*). No estado (Rx02-RxHAPrepare) é o início da preparação para a recepção do HA. Em (Rx03-RxHA) é habilitado o rádio e um temporizador é programado para aguardar uma mensagem HA de algum vizinho que deseja comunicar. Caso ocorrer o estouro do tempo de espera, o algoritmo vai para (Rxe03- *RxHATimeout*) e retorna ao repouso (*EndSlot*). Este tempo é chamado de RxIL e é comum ocorrer em protocolos assíncronos baseados em RIT. Se chegar uma mensagem, ela é tratada em (Rx04- *RxDataOffset*). Quando ocorre uma mensagem HA em (Rx04) e não ocorreu colisão, então é aguardado o dado em (Rx05- *RxDataPrepare*). Caso ocorra colisão, é sinalizado o ECW1 em (Rx44- ECW1), e também é aguardado o dado em (Rx05). Se chegar a mensagem de dado em (Rx06-*RxData*), é verificada a mensagem e se tudo estiver ok é enviado o DA em (Rx08-*TxAckPrepare*) e (Rx09-*TxAck*). Desta forma, o algoritmo notifica as camadas superiores em (Rx0A-*RxProc*) e volta ao repouso (*EndSlot*), retornando o controle para o sistema operacional e aguardando o início de um novo slot de tempo. Caso ocorra algum erro, de ECW1 quando enviando o HA, ou ECW2 quando ocorreu erro no dado recebido, é enviada a mensagem de erro (ECW) em (Rx75-TxCW). Após este estado, é incrementado o número de retransmissões do ECW e o código retorna para (Rx06-*RxData*) para aguardar novamente o dado. Este retorno somente é realizado pelo número de retransmissões do ECW no algoritmo.

Figura 25 – Diagrama de Estados do serviço de recepção do RITMC

5.3 Implementação do mecanismo do CSMA/CA

Em relação ao mecanismo de acesso ao canal, o mecanismo CSMA/CA é fundamental na comunicação nos protocolos assíncronos. Para os protocolos síncronos como o TSCH, apesar da especificação prever a utilização algoritmo slotted CSMA/CA, ele não é fundamental para o funcionamento do mecanismo. No caso na plataforma OpenWSN ele não foi implementado.

Quanto ao algoritmo do CSMA/CA ele pode ser *slotted* ou *unslotted* dependendo do tipo de necessidade do protocolo. O algoritmo *slotted* é utilizado por algoritmos que possuem *beacon* ou uma mensagem de sincronismo, como TSCH. Para algoritmos como os assíncronos que não possuem um *beacon*, utiliza-se o algoritmo *unslotted*.

No caso do algoritmo *unslotted*, ele utiliza a unidade de tempo chamada período de atraso (BP- *backoff*), onde um BP é igual a unidade $UnitBackoffPeriod = 80\text{bits} = 320\mu\text{s}$. O algoritmo depende de dois parâmetros principais: expoente de *backoff* (BE) e o número de *backoff* (NB). O parâmetro BE representa o número de BP que o canal deve ser mantido inativo antes de acessar o canal. O parâmetro NB representa o número de vezes que o algoritmo CSMA/CA foi requisitado, enquanto tentava acessar o canal (IEEE802.15.4, 2015).

Alguns microcontroladores já possuem toda a infraestrutura para tratamento do CSMA, como é o caso do micro controlador CC2538 utilizado nesta tese. Neste caso, o algoritmo a ser implementado do CSMA é aberto e pode ser alterado. Neste projeto utilizou-se o algoritmo *unslotted* proposto em IEEE802.15.4, (2015) mostrado na Figura 26.

Em relação à implementação, uma característica do microcontrolador CC2538 é a necessidade de utilização de um temporizador do rádio especificamente para o CSMA/CA. Este mesmo temporizador foi utilizado pelo OpenWSN, e houve a necessidade de reformulação de todo o código para trabalhar com outro temporizador, não tão precisos como o do rádio. Outra característica do CC2538 é não ter disponibilidade de mudar o nível de gatilho do CCA. E algumas vezes, o CCA se apresentava ocupado em condição que somente ele estava comunicando, sem interferência de parede ou aparentemente de outra rede que pudesse interferir no sinal, como o IEEE802.11.

Figura 26 – Algoritmo unslotted CSMA/CA

Fonte adaptada:IEEE802.15.4e (2012)

A implementação do código do rádio foi realizada de forma flexível para permitir a escolha de transmitir com ou sem o uso do algoritmo CSMA/CA. O uso do algoritmo do CSMA faz com que aumente a latência da comunicação. Na Figura 27 é mostrada a variação do CSMA durante a transmissão da mensagem H. Neste teste, foram utilizados somente dois módulos sensores. A captura foi habilitada desde o início do envio do H (Rx01) até a indicação de fim de transmissão (Rx02) de acordo com o algoritmo de transmissão mostrado na Figura 25. O tempo de transmissão da mensagem H de 12 bytes variou de 1,4ms até 7,2 ms. A Figura 27 mostra a imagem da captura do osciloscópio com retenção dos dados.

Figura 27 – Captura do envio do H utilizando CSMA

Em um outro teste, com somente dois módulos sensores sem influência de outras redes, foi enviado 127 bytes de dados com e sem CSMA/CA, mostrado na Figura 28. Neste caso, foi mostrada a captura da transmissão do HA sem utilização do CSMA e o envio do dado com e sem CSMA/CA. Nas duas imagens da Figura 28, no canal 2 (acima) representa o sinal TxHA e no canal 1 (abaixo) o *TxData*. Na Figura 28.a é mostrada a captura com o *TxData* sem CSMA/CA, enquanto que Figura 28.b é mostrado a captura com o *TxData* com CSMA/CA.

Figura 28 – Envio de dados com e sem CSMA no protocolo RITMC.

De acordo com a Figura 28, o tempo para envio da mensagem HA (12 bytes) foi de 2ms. Após o HA, o algoritmo espera o tempo TD com um tempo fixo de 1,10ms em média. Por fim, para transmissão do dado (127 bytes) o tempo variou de 4,30ms até 8,64ms quando usando o CSMA/CA.

A análise das Figuras 27 e 28 mostrou que o uso do algoritmo CSMA/CA traz um atraso na comunicação grande. No primeiro cenário, o uso do CSMA/CA fez o tempo de envio variar de 5,8ms para uma mensagem de 12 bytes, ou seja, aproximadamente 4 vezes maior que o

menor valor. No segundo cenário, o CSMA/CA fez o tempo de envio variar de 4,34 ms para uma mensagem de 127bytes, que foi aproximadamente 1 vez maior que o menor valor. Resumindo, o uso do CSMA/CA deve ser feito realmente quando necessário.

5.4 Limitação da camada de Rede do OpenWSN

Uma das principais características de WSN em relação a outras tecnologias é a flexibilidade das topologias, podendo variar de topologia estrela a *mesh*. Redes do tipo mesh possuem a vantagem de serem redes de baixo custo, fácil implantação e bastante tolerantes a falhas.

Uma rede mesh é composta de vários módulos espalhados, que se comunicam entre si e que permite a comunicação de qualquer módulo da rede. Os módulos têm a função de repetidores e cada módulo está conectado a um ou mais módulos da rede. Desta maneira é possível transmitir mensagens de um módulo a outro distantes a vários saltos por diferentes caminhos. A Figura 35 mostra um exemplo de rede mesh onde para a aplicação web enviar uma mensagem para o módulo M10 são necessários 5 saltos por diferentes caminhos.

Em relação à implementação do OpenWSN utilizada neste trabalho, a implementação do protocolo de roteamento RPL utiliza o modo *NoStoreMode*. Como mencionado anteriormente no capítulo 2, o modo *NoStoreMode* é um mecanismo de roteamento onde os módulos sensores não armazenam a rota. Desta forma, o roteamento é realizado somente no LBR. Portanto, na plataforma do OpenWSN, o OpenVisualizer é responsável por descobrir a rota até o módulo destino e então passar esta rota na mensagem de transmissão.

Como a mensagem do IEEE802.15.4 é limitada em 127 bytes, o número máximo de saltos que é possível atingir com o OpenWSN em uma rede mesh é de 5 saltos. Este limite pode ser obtido utilizando a equação (4) para determinação do número de bytes da mensagem de requisição da aplicação CoAP.

$$AppMsglen = MsgH + CoapIPV6 + Rota * n + CoapD \quad (4)$$

Sendo *AppMsglen* é o tamanho da mensagem de aplicação em bytes; *MsgH* é o total de bytes de uma mensagem do tipo DATA dentro do protocolo IEEE802.15.4 e na implementação do OpenWSN possui 45 bytes; *CoapIPV6* é o cabeçalho de uma mensagem do protocolo CoAP e possui 14 bytes; *CoapD* representa o tamanho dos dados (*payload*) de uma mensagem da aplicação padrão “get coap://[bbbb::DST.ADDR]/.well-known” que contém 25 bytes. Por fim,

a rota é o endereçamento da mensagem, que para a implementação do OpenWSN do protocolo RPL *NoStoreMode* depende do número de saltos (n). Caso a mensagem de destino estiver à distância de somente um salto ($n=1$) do DAGROOT o tamanho da rota será zero. Caso o módulo destino estiver com o número de saltos maior que um ($n > 1$), então a rota será múltipla do endereço da rota. Para uma rede com endereçamento de 128 bits seria múltiplo de 8 bytes. Desta forma, para 1 salto na rede a mensagem terá o tamanho de 84 bytes. Para dois saltos 94 bytes, e assim por diante. Então, o limite da rede seria 5 saltos, atingindo um total de 124 bytes.

Em relação à resposta desta requisição, ela segue sempre uma mensagem fixa com o endereço do DAGROOT e tem 81 bytes para esta mensagem CoAP para qualquer número de saltos.

Como o objetivo desta tese é realizar a análise da rede utilizando uma escalabilidade de módulos sensores através do uso de protocolos assíncronos, seria desejável analisar a rede quando tivesse mais do que 5 saltos.

A solução para conseguir testar os protocolos assíncronos em mais saltos seria implementar o algoritmo *StoreMode* dentro do RPL. Como o foco desta tese são os protocolos da camada MAC, e o RPL é o protocolo de roteamento, este trabalho foi atribuído a um segundo desenvolvimento sendo realizado no grupo, relacionado a melhorias no protocolo de roteamento RPL e está sendo desenvolvido em paralelo a esta tese.

Para continuar com os testes propostos nesta tese, a solução foi utilizar na camada MAC um endereçamento IPV6 com menor número de bytes. A mudança realizada foi, devido aos módulos disponíveis ter endereço IPv6 próximos, alterando somente os dois bytes menos significativos do endereço. Desta forma, foi possível alterar a rota do endereçamento IPV6 de um endereço de 128 bits original para um endereço somente de 16 bits. Neste caso, conseguiria atingir até 18 saltos com um total de 126 bytes na mensagem. Esta mudança foi necessária somente para as mensagens CoAP, onde se implementou mudanças no OpenVisualizer e no firmware dos equipamentos. Esta alteração foi realizada somente no protocolo RITMC proposto.

5.5 Plataforma de Hardware

Em relação ao desenvolvimento e testes de validação desta tese, foram realizados em ambiente real. O sistema IoT montado era composto por módulos sensores utilizando o módulo

CC2538EM da empresa Texas Instrument, (2018) e um módulo desenvolvido pelos autores, baseado no mesmo microprocessador CC2538.

O CC2538EM é um módulo que utiliza a CPU CC2538F512RKU, um microprocessador ARM Cortex-M3 de 32 bits com *clock* de 32Mhz com 512 Kbytes de Flash, 32 Kbytes de RAM e com transceptor de rádio 2.4GHz IEEE 802.15.4 com taxa de comunicação interna de 250 Kbps e com antena PCB do tipo Inverted-F Antenna. O rádio tem uma sensibilidade de recepção de -97 dBm e transmissão programável até 7 dBm de potência. Isto possibilita aplicações de até 30 metros em ambiente interno (indoor) e de mais de 100 metros em ambiente externo (outdoor).

O módulo desenvolvido nesta tese teve como principais requisitos uma forma mais flexível de alimentação (pilha ou energia), possibilidade de incorporar outros sensores ou outras placas de comunicação, possuir *leds* indicativos, e ter um custo reduzido comparando com as outras plataformas existentes. O projeto chamado de MBA, utiliza a CPU CC2538NS256 com memória flash de 256 Kbytes, antena PCB do tipo diferencial e algumas características de IO melhoradas em relação ao módulo CC2538EM. Todos os pinos de IO estão disponíveis, desta forma é possível configurar qualquer sensor em qualquer tipo de comunicação serial ou acesso direto ao GPIO. A opção por uma antena diferencial é devido a facilidade de implementação e também custo menor. A Figura 29 mostra a imagem dos dois módulos sensores CC2538EM e MBA utilizados nesta tese.

Para programação o código foi feito em linguagem C, utilizado a plataforma eclipse IDE Neon com os devidos pacotes do compilador GNU ARM GCC para a plataforma Cortex M3.

Em relação ao tamanho do código fonte desenvolvido, o algoritmo assíncrono implementado tem um tamanho pouco superior ao OpenWSN original. O RITMC proposto teve um código total de aproximadamente 48 Kbytes enquanto que com TSCH tem um código total de aproximadamente 39 Kbytes. Quanto a área de dados ocupada o RITMC ocupa 11K enquanto o TSCH utiliza 9K. Neste caso, deve ser considerado que o código não está otimizado, contendo variáveis de depuração de código, enquanto o código do TSCH atual é otimizado, sem variáveis adicionais.

Figura 29 – Hardware dos módulos sensores utilizados no projeto

5.6 Plataforma de Software

Em relação ao ambiente de aplicação IoT, a plataforma OpenWSN disponibiliza a interface *web browser*, ou através de uma aplicação desenvolvida em Python. Para a validação desta tese, foram utilizadas as duas interfaces. A aplicação web através do browser é realizada através de um *plugin* CoAP instalado no browser conforme mostrado na Figura 30.

Figura 30 – Ambiente Web de aplicação do OpenWSN

Como o objetivo do trabalho é um sistema IoT, foram feitos testes de validação em alguns cenários reais de aplicação como leitura e escrita em variáveis reais. Para isso, foram implementados alguns métodos dependendo do tipo de sensor. O exemplo da Figura 30 mostra uma aplicação realizada na tela do browser para leitura do estado de uma lâmpada. A requisição CoAP do tipo de sensor é mostrado no comando GET abaixo:

```
GET coap://[bbbb::12:4b00:3a6:4cbe]:5683/d
```

onde entre colchetes está o IP do nó sensor, na porta 5683 e o “/d” significa que é pedido todas as características do módulo.

A resposta do módulo sensor é o seguinte comando:

```
res: { "v":1, "m":"CC2538EMTI", "id":1, "npts":1 }
```

A resposta do módulo sensor indica a descrição do módulo “CC2538EMTI”, o identificador (“1”) significa que há somente 1 parâmetro (“npts”, “1”) correspondente ao estado da lâmpada.

. Uma outra ferramenta de aplicação foi desenvolvida em linguagem Python para automatizar os testes e a análise dos resultados. Para executar os testes de validação deste trabalho, foi utilizado o mesmo mecanismo de métodos GET e PUT para o protocolo CoAP mostrados anteriormente no capítulo 2. A Figura 31 mostra uma captura da aplicação realizada, onde são mostrados tanto a informação do OpenVisualizer quanto do software IoT. É possível ver na figura o dado da pergunta e da resposta dentro do OpenVisualizer. Desta forma, são obtidos tanto o tempo da aplicação quanto da comunicação com o OpenVisualizer, baseados na

base de tempo do relógio do Windows. Os dados também são salvos em arquivos texto para armazenamento e posterior análise.

Figura 31 – Exemplo de aplicação e logs utilizados nos testes

A Figura 32 mostra um diagrama de blocos de comunicação entre a aplicação e WSN. Todas as soluções de aplicação se comunicam com o sistema através do OpenVisualizer.

De acordo com a Figura 32, uma aplicação IoT basicamente utiliza o módulo CoAP que é fornecido pelo OpenWSN e é responsável pela comunicação entre a aplicação e o LBR (OpenVisualizer). Os comandos básicos de uma aplicação são comandos GET para leitura de parâmetros e PUT para escrita de alguma informação.

Figura 32 – Diagrama de blocos da comunicação entre aplicação e a rede IEEE802.15.4

5.7 Aspectos de configuração dos cenários de teste

Para analisar os mecanismos de acesso ao meio e o comportamento da rede quanto aos múltiplos saltos, era necessário montar uma topologia de rede que conseguisse explorar estes requisitos e também que fosse viável ser realizada dentro do ambiente de teste. Devido à limitação de espaço físico para montagem da rede e pelo fato do desenvolvimento e os testes sempre serem realizados em ambiente real, foi necessário utilizar topologia fixa.

Houve esta necessidade para conseguir analisar exatamente a estratégia de teste proposto. Se a topologia fosse construída livremente, seria dinamicamente montada pelo protocolo de rede de acordo com características da comunicação, como distância entre nós, potência do rádio e barreiras na transmissão do sinal.

Na topologia fixa, o código fonte é projetado para somente responder aos vizinhos que ele foi previamente designado a comunicar. A rejeição das mensagens que não é da vizinhança fixa do módulo, é realizada na camada MAC do código fonte. O objetivo é que estas mensagens não influenciassem os testes. Isto porque se um módulo sensor receber uma mensagem não desejável, e ele estiver aguardando uma mensagem de outro vizinho, ele precisa desligar o rádio, verificar se a mensagem é para ele, rejeitar uma mensagem indesejável, e então voltar a abrir o rádio novamente para novas mensagens. Durante este tempo pode ter ocorrido a mensagem que o módulo estava realmente esperando. Este mecanismo de topologia fixa foi utilizado para a grande maioria dos testes desta tese, de forma comum a todos os protocolos

comparados. Porém o código do OpenWSN está preparado para trabalhar com uma topologia variável. A montagem da topologia está relacionada às camadas de rede RPL que não foram alteradas.

5.8 Metodologia de validação

O objetivo desta tese é analisar as tecnologias assíncronas da camada MAC de WSN em aplicações de IoT para atender cenários com consumo reduzido, onde as características rígidas de determinismo são ausentes. Para isso, foi proposto o protocolo RITMC trabalhando no ambiente IoT do OpenWSN.

Para validação do sistema IoT proposto, foram feitos testes em diferentes topologias com até 15 módulos sensores nos cenários descritos a seguir. Com o objetivo de realizar a comparação de desempenho do protocolo RITMC com os protocolos assíncronos de camada MAC no cenário de canal único e multicanal. Adicionalmente, houveram outros testes comparando o protocolo assíncrono proposto com o protocolo síncrono da plataforma IoT do OpenWSN. As métricas utilizadas na comparação foram: medição de consumo, análise da latência e do tráfego da rede, e aplicação IoT utilizando ambiente OpenWSN.

Como critério de comparação dos protocolos assíncronos, foram implementados três protocolos assíncronos multicanais mostrados no capítulo 3 desta tese: A-MAC, ARM e AMCA. Esta implementação baseou-se nos algoritmos descritos nas publicações dos correspondentes protocolos. Para esta implementação, somente foram realizados os serviços básicos de comunicação para análise dos protocolos.

Para a avaliação do consumo da rede foi obtido um modelo de energia, de forma a conseguir estimar o consumo na comunicação entre os módulos sensores e como critério de comparação entre os protocolos em diferentes cenários de uso.

Nos testes de consumo, sempre foi utilizado alimentação por duas pilhas alcalinas (3V). Os testes foram feitos através da captura da queda de tensão através de um resistor de 10 ohms em série com a alimentação de energia do módulo sensor. Para medição do consumo, empregou-se um osciloscópio digital SIGLENT SDS 1204CFL. Para maior acurácia dos valores medidos, as capturas foram salvas e a corrente foi integrada no software Matlab, permitindo a determinação da carga fornecida.

Para os experimentos de medição do consumo do protocolo assíncrono, manteve-se o cenário de testes para todos os protocolos estudados. A rede era formada por dois módulos

sensores, transmitindo pacotes de dados de 127 bytes com reconhecimento e com potência do rádio de 0dBm.

De forma similar, foi realizada a comparação do consumo do protocolo síncrono e o assíncrono proposto. Para isso, foram feitos testes medindo o consumo do protocolo síncrono de dois módulos sensores enviando os comandos básicos do RPL dos serviços de transmissão, recepção e de IL.

A partir dos dados coletados dos protocolos assíncronos e síncrono, foi obtido um modelo de energia que pudesse comparar os protocolos em diferentes configurações.

Para avaliação da taxa de entrega e da latência da rede foram montados quatro diferentes cenários de teste. Os dois primeiros cenários foram utilizados para validação dos protocolos assíncronos. O terceiro cenário foi utilizado para comparação do protocolo TSCH com o RITMC proposto. Por fim, um quarto cenário foi montado para validação do RITMC em um cenário de múltiplos saltos.

Em um primeiro cenário, cada nó sensor deveria enviar mensagens de dados gerados na própria camada MAC para um outro vizinho pré-determinado. A mensagem de dados precisava de confirmação (Data.Ack). Os testes consistiram de todos os módulos enviando mensagens continuamente para um de seus vizinhos em uma determinada taxa de pacotes variando de um período de 0,5s a 5s. Uma pequena rede foi montada com seis módulos sensores dispostos, como mostrado na Figura 33.

Figura 33 – Cenário de teste dos protocolos assíncronos com 6 módulos

Para uma avaliação de um cenário mais realístico um segundo cenário foi montado utilizando o ambiente do OpenWSN em uma aplicação de Internet das coisas. Neste cenário foram utilizados até 16 módulos sensores espalhados em uma topologia mesh conforme Figura 34. A rede WSN foi montada com os módulos sensores separados a uma distância de 2 metros

um do outro e com vizinhança (lógica) conforme as linhas de ligação entre os módulos da figura. Porém, como os módulos sensores estavam próximos um dos outros, eles se enxergavam. Cada grupo de 3 módulos sensores estavam dispostos em um canal específico.

De acordo com a Figura 34, o Ambiente do OpenWSN é formado basicamente por aplicação web (CoAP), um roteador de borda formado pelo *OpenVisualiser*, pelo módulo *sink*, que corresponde ao primeiro nó de entrada na rede IEEE802.15.4 e o restante dos módulos da WSN.

Figura 34 – Cenário de teste dos protocolos assíncronos com até 16 módulos e somente 1 caminho para cada módulo

Para gerar a aplicação IoT, foi utilizado o software para requisição CoAP, mostrado na Figura 31 para cada nó da rede, a cada 10 segundos. Tanto a aplicação quanto o OpenVisualizer foram executados na mesma estação. A métrica, neste caso, foi a avaliação da latência da rede e a taxa de entrega de pacotes para alcançar cada um dos módulos da rede. A medição sempre ocorreu usando a ferramenta de escuta de linha Wireshark com um módulo de escuta (*dongle*) para captura de pacotes na rede, e do próprio software da aplicação. Houve, além disso, o software OpenVisualizer capturando a amostra de tempo de envio e resposta das mensagens, e a própria mensagem de resposta do módulo que continha informação de diagnóstico.

Um terceiro cenário foi montado com 11 módulos e dois caminhos para cada módulo conforme mostrado na Figura 35. A WSN foi montada com os módulos separados a uma distância de 6 metros um do outro e com vizinhança (lógica) conforme as linhas de ligação entre os módulos com potência do rádio de -5dBm. Cada grupo de 2 módulos sensores em cada salto estavam dispostos em um canal específico. Os testes foram realizados em ambiente multicanal na comparação do protocolo RITMC proposto com o protocolo síncrono do TSCH.

Figura 35 – Cenário de teste da rede com até 11 módulos e dois caminhos para cada módulo

Um quarto cenário foi montado com 13 módulos e somente 1 caminho para cada nó. A WSN contava com os módulos sensores separados a uma distância de 6 metros um do outro e com vizinhança (lógica) sequencial de acordo com o a Figura 36. A potência do rádio foi configurada em -5dBm para todos os rádios. Os módulos estavam dispostos em canais específicos com até dois módulos no mesmo canal variando os canais de 17 a 25. Neste cenário foi montado um ambiente heterogêneo de comunicação, com diferentes obstáculos, variando entre ambiente interno e externo. Neste caso, foram realizados testes de validação do protocolo RITMC em ambiente multicanal com um maior número de saltos.

Com relação ao mecanismo de descoberta da rede e de escolha do melhor canal, o desenvolvimento do algoritmo e testes de validação foram realizados somente no software Matlab. Os resultados obtidos foram simulados com o objetivo de verificar o comportamento da rede alterando a quantidade de canais e a quantidade de módulos sensores. O objetivo do experimento era de analisar o comportamento da rede quando ocorre uma mudança da qualidade do sinal devido à variação do número de módulos por canal ou a mudança das condições da rede, alterando ou diminuindo a taxa de erros.

Figura 36 – Cenário com somente um caminho para cada módulo e diferentes obstáculos

6 RESULTADOS

Neste capítulo serão apresentados os resultados obtidos para validação dos algoritmos propostos nesta tese, bem como a comparação do desempenho com os diferentes protocolos estudados. Para os estudos e validação do protocolo proposto foram analisados taxa de entrega, latência e consumo da rede em diferentes cenários.

6.1 Análise do consumo

A análise do consumo foi realizada primeiramente comparando o protocolo RITMC com outros protocolos assíncronos multicanais com período de RIT fixo. Posteriormente, comparando o protocolo RITMC com o TSCH com diferentes períodos de RIT.

Consumo protocolos assíncronos

Os experimentos descritos nesta seção referem-se à avaliação do consumo da rede dos protocolos assíncronos. Além do RITMC proposto, foram analisados os protocolos AMCA, A-MAC, e ARM descritos no capítulo 3. Os algoritmos aqui estudados foram escolhidos por apresentarem características peculiares relacionadas à economia de energia e ambiente multicanal.

Com relação à implementação dos protocolos assíncronos baseados em RIT, dentro dos serviços de transmissão e recepção, os estados TxHello, TxIL, TxData e RxData são comuns a todos os protocolos. Além destes, cada protocolo tem outros estados específicos.

O estado TxHello consiste no envio de uma mensagem de anúncio de 12 bytes. O estado de espera na recepção (RxIL) ocorre quando não há vizinho para comunicar. Como mencionado anteriormente, o RxIL é um dos principais gastos de energia na comunicação assíncrona, pois ele ocorre periodicamente dentro da rede (KIM, et.al.,2015). A Figura 37 mostra o consumo para o estado TxHello com RxIL para o protocolo RITMC. O canal 1 mostra o consumo do rádio, enquanto que o canal 2 mostra os estados do serviço.

No estado (1) da Figura 37 é o momento do envio da mensagem de (H), enquanto que em (2) é mostrado o estado do rádio à espera de uma mensagem de um transmissor (RxIL). O sinal obtido pelo canal 1 é referente a tensão em cima do resistor shunt de 10 ohms, de onde se obtém a corrente absorvida na entrada do circuito de alimentação da placa. A análise do sinal de corrente do rádio mostra que existe uma corrente absorvida mínima de 11mA. Como a corrente foi medida na alimentação da placa e não somente no rádio, esta corrente está

relacionada com o consumo total do sistema em operação normal sem operação do rádio. O microcontrolador CC2538 possui um modo de repouso profundo (*deep sleep*) que não estava ligado na versão de firmware do OpenWSN. Apesar de este valor ser significativo, ele é comum a todos os protocolos implementados dentro da plataforma OpenWSN. Para efeitos de comparação de consumo, ele será desconsiderado do cálculo.

Figura 37 – TxHello com RxIL. CH1 - tensão em mV e CH2 - estados do rádio

Com relação à implementação do código dentro da plataforma OpenWSN, toda transmissão pode ser dividida em três etapas: (a) TxEnable, (b) TxNow, (c) TxSendOK. Esses três estágios são vistos na Figura 37 para o estado TxHello. No entanto, para facilitar a compreensão deste trabalho, a transmissão da mensagem será resumida como apenas uma etapa. Da mesma forma, o RxIL será considerado um serviço diferente ao serviço de transmissão e recepção quando for considerado o cálculo do consumo de energia.

Ainda na anúncio, mostrado na Figura 37, o estado (2), que representa o RxIL, tem o valor máximo determinado pelo parâmetro *RxWaitTimeout*. Este parâmetro é importante para otimização do RxIL. O critério adotado neste trabalho para determinar o parâmetro *RxWaitTimeout* foi obter o tempo da primeira comunicação do transmissor com o receptor. Neste caso foi escolhido o maior tempo entre 1000 amostras. Em seguida, foi estabelecida uma margem de segurança de espera de 3 vezes o maior tempo medido na recepção da primeira mensagem. No experimento da Figura 37, foi considerado o *RxWaitTimeout* de 6ms.

Quanto ao estado de TxIL, relacionado com o início da transmissão de dados, o tempo de espera é aleatório e vai até o limite do *TxWaitTimeout*. Na implementação, foi adotado para este parâmetro em todos os protocolos assíncronos estudados, o tempo de *RxWaitTimeout* adicionado 50ms.

Com relação às partes específicas dos serviços de transmissão e recepção de cada protocolo assíncrono analisado, para o protocolo RITMC proposto, a transmissão é realizada em quatro etapas principais como mostrado anteriormente no capítulo 4. A Figura 38 mostra o consumo do rádio e os estados de cada uma das etapas para transmissão de 127 bytes de dados. As etapas de transmissão conforme o canal 2 da Figura 38 são: (1) TxIL, (2) TxHA, (3) TxData e (4) RxDataAckouECW.

Figura 38 – Transmissão de 127 bytes no protocolo RITMC. CH1(acima) - consumo e CH2 (abaixo) - estados do rádio

De acordo com a Figura 38, o custo de energia total medido foi de 2602.1 uJ. Se for desconsiderado o TxIL, o custo de energia foi de 410.71 uJ. A média de consumo de energia para transmissão foi de 463.44 uJ.

Na Figura 39 é apresentada a corrente consumida pelo rádio para o serviço de recepção de 127 bytes de dados. Para o serviço de recepção do protocolo RITMC proposto também são necessárias quatro etapas: (1) TxHello, (2) RxHA, (3) RxData e (4) TxAck. O custo de energia medido foi de 464,81 uJ. A média de consumo de energia para recepção foi de 482,38 uJ.

Figura 39 – Recepção de 127 bytes de dados no protocolo RITMC.
CH1(acima) - consumo e CH2 (abaixo) - estados do rádio

A respeito de outros protocolos assíncronos estudados neste trabalho, adotou-se o mesmo procedimento para determinação dos estados e consumo do rádio. Na Tabela 7 são apresentados os tempos médios medidos de cada uma das etapas da transmissão e recepção de 127 bytes de dados com confirmação dos quatro protocolos pesquisados. Para a construção da tabela foram obtidos os tempos médios (t), o desvio padrão e os tempos máximos ($t_{\max}$) de 1000 amostras após a realização de 10 testes para cada um dos protocolos. Os valores de corrente (I) foram obtidos da medição do osciloscópio de 20 amostras por protocolo com tensão fixa de 3V. Por fim, foram calculados os valores de potência (P) e energia (E).

Tabela 7 – Custo de Energia para transmissão e recepção dos protocolos assíncronos

Protocolo	Serviço	Estados	t [ms]	Desvpad t [ms]	$t_{\max}$ [ms]	I [mA]	P [mW]	E [uJ]
RITMC	TX	1-TxIdle	41.06	28.45	104.64	17.74	53.23	2185.70
		2-TxHA	0.77	0.16	0.96	12.04	36.13	27.94
		3-TxData	5.73	0.80	12.16	18.51	55.54	318.23
		4-RxDataAck	1.42	0.16	1.60	17.83	53.48	75.72
	RX	1-TxHello	2.05	0.75	7.36	11.75	35.26	72.29
		2-RxHA	1.76	0.06	1.92	13.45	40.34	71.10
		3-RxData	7.40	2.64	12.28	15.32	45.97	340.17
		4-TxDataAck	0.80	0.13	0.96	12.89	38.68	30.94
	RxIL	1-TxHello	1.80	0.87		11.65	34.94	62.89
		2-RxWaitTimeout	10.00	0.32		17.12	51.37	513.69
AMCA	TX	1-TxIdle	63.38	25.42	168.96	17.35	52.05	3298.96
		2-TxData	5.74	0.82	10.24	19.50	58.50	335.87
		3-RxDataAck	2.24	0.03	2.24	18.10	54.30	121.49
	RX	1-TxHello	3.16	0.74	4.16	12.50	37.50	118.66
		2-RxData	7.96	0.76	9.28	16.10	48.30	384.31
		3-TxDataAck	0.73	0.15	0.96	11.45	34.35	25.22
	RxIL	1-TxHello	3.11	0.78	12.60		37.80	117.56

		2-RxWaitTimeout	28.00	0.34	17.00		51.00	1428.00
AMAC	TX	1-TxIdle	46.52	29.16	106.24	17.54	52.61	2447.38
		2-Tx_HelloMask	2.13	0.79	6.72	13.78	41.35	87.91
		3-RxHello2	2.43	0.79	7.68	14.08	42.25	102.67
		4-Tx_HHA	0.20	0.00	0.20	13.65	40.95	8.19
		5-TxData-Tx0b	5.88	0.83	11.52	19.28	57.85	340.01
		6-RxDataAck	1.18	0.15	1.28	12.73	38.20	45.22
	RX	1-TxHello	3.17	0.75	6.72	11.71	35.13	111.36
		2-RxHelloMask	2.50	0.82	8.30	12.71	38.13	95.28
		3-Tx_Hello2	2.19	0.76	6.08	13.71	41.13	89.97
		4-Rx_HHA	0.61	0.10	0.64	12.76	38.29	23.28
		5-RxData	6.68	0.80	15.68	14.51	43.53	290.91
		6-TxDataAck	0.70	0.13	0.96	10.80	32.40	22.76
	RxIL	1-TxHello	3.17	0.80		12.10	36.29	114.89
		2-RxWaitTimeout	24.00	0.31		14.52	43.56	1045.41
ARM	TX	1-TxIdle	87.97	40.72	126.72	17.79	53.36	4694.20
		2-TxRTS	2.09	0.79	7.36	14.53	43.59	91.09
		3-RxCTS	2.28	0.78	6.72	13.06	39.18	89.22
		4-TxData	5.74	0.75	9.28	18.45	55.34	317.45
		5-RxDataAck	2.31	0.50	10.56	17.60	52.80	122.04
	RX	1-TxHello	2.25	0.85	6.72	11.34	34.03	76.62
		2-RxRts	2.23	0.80	8.32	12.83	38.50	85.67
		3-TxCts	1.98	0.78	7.36	12.52	37.57	74.41
		4-RxData	5.74	0.81	11.20	13.00	39.00	224.05
		5-TxDataAck	0.92	0.12	0.96	12.13	36.40	33.35
	RxIL	1-TxHello	2.20	0.83		12.10	36.29	79.87
		2-RxWaitTimeout	25.00	0.63		14.52	43.56	1088.97

De acordo com a Tabela 7, o valor de $t_{\max}$ indica o maior tempo obtido entre 1000 amostras entre o envio do H e a recepção de uma primeira mensagem. Por exemplo, para o protocolo RITMC proposto, a recepção da primeira mensagem do transmissor acontece no estado (2-RxHA) como mostrado anteriormente na Figura 14, e o maior tempo obtido ($t_{\max}$) do receptor após abrir a janela de recepção até a recepção da primeira mensagem foi de 1.92ms. Quanto aos outros protocolos, o protocolo AMCA recebe a primeira comunicação em (2-RxData) mostrada na Figura 4. O tempo mais alto obtido do receptor após a abertura da janela de recepção foi de 9,28 ms. O A-MAC recebe a primeira mensagem em 2-RxHelloMask mostrada na Figura 13. O protocolo ARM recebe a primeira mensagem em 2-RxRts mostrada na Figura 12. Aplicando o critério de 3 vezes o tempo máximo, foram obtidos os valores da Tabela 8 para o parâmetro *RxWaitTimeout*.

Tabela 8 – Cálculo Tempo RxWaitTimeout

Protocolo	Estado	tmax [ms]	RxWaitTimeout [ms]
RITMC	2-RxHA	4.95	15
AMCA	2-RxData	9.28	28
A-MAC	2- RxHelloMask	8.30	24
ARM	2-RxRts	8.32	25

Para o protocolo RITMC proposto, foi criado um mecanismo de leitura periódica do tempo RxHA no serviço de LiveList. Desta forma, os testes ao longo do desenvolvimento do projeto mostraram que, dependendo das condições de rede, este valor apresentava variação em relação ao obtido na Tabela 9. O maior valor obtido de RxHA e, então o valor de RxWaitTimeout, ficou em 15ms.

Modelo de energia

Para a avaliação do consumo entre os módulos sensores foi obtido um modelo de energia. Com o modelo é possível estimar o consumo da rede dos diferentes protocolos, e compará-los em diferentes condições.

A energia gasta pelo nó sensor pode ser estimada baseada na energia gasta pelos serviços principais de anúncio, transmissão (Tx) e recepção (Rx) (DUTTA, et.al., 2012). No modelo de energia para os protocolos assíncronos baseados no receptor, a energia total consumida (E_{Total}) por um módulo sensor pode ser determinada pela Equação (5).

$$E_{Total} = ET_{RxIL} + ET_{Tx} + ET_{Rx} \quad (5)$$

De acordo com (1), ET_{RxIL} é o custo de energia total para anúncio com timeout ou (RxIL), ET_{Tx} é o custo total para transmissão e ET_{Rx} é o custo total para recepção.

A anúncio (H) é enviada periodicamente na rede por todos os módulos com uma frequência f_H . Desta forma, o custo de energia depende da frequência de (H) multiplicado pela energia gasta para o envio do H e o tempo de espera pela comunicação de algum vizinho. Ela pode ser equacionada conforme (6).

$$ET_{RxIL} = f_H * E'_{RxIL} \quad (6)$$

Onde E'_{RxIL} corresponde ao custo de energia médio de uma anúncio com IL. O tempo da Equação (6) considera que o módulo receptor se anuncia e nenhum vizinho deseja comunicar, estourando o tempo de espera por uma mensagem como mostrado na Figura 37.

Este tempo é equivalente a todos os protocolos assíncronos baseado em RIT e ajustado pelo parâmetro *RxWaitTimeout*, cujo objetivo desta tese é de otimização e será específico para cada um dos protocolos.

Com relação à frequência f_H devido ao não sincronismo entre os relógios dos módulos, existe um atraso de *clock* para cada módulo. Desta forma, não se tem exatamente n pacotes dentro do período de dados medidos. Para obter uma melhor precisão do modelo de energia, foi utilizado um parâmetro de ajuste α equivalente à média de pacotes medido. Foram realizados testes nos protocolos assíncronos para diferentes taxas de dados e diferentes janelas de RIT, e o valor médio obtido de ajuste foi de aproximadamente meio pacote ou $\alpha=0.47$.

O custo de energia da transmissão e da recepção para os protocolos assíncronos baseados no receptor são estimados de acordo com o protocolo em questão, considerando que existem somente dois módulos na rede. Quando houver uma transmissão, não vai existir uma anúncio do transmissor. Supondo que o receptor irá receber a mensagem quando o vizinho transmitir. Desta forma, pode-se considerar que, tanto para transmissão quanto para a recepção, tem-se:

$$ET_{Tx} = f_{Tx} * E'_{Tx} \quad (7)$$

$$ET_{Rx} = f_{Rx} * E'_{Rx} \quad (8)$$

Onde E'_{Tx} é o custo de energia médio da transmissão e E'_{Rx} é o custo de energia médio da recepção. Nesse caso, os valores exatos de E'_{Tx} e E'_{Rx} dependem de diferentes fatores e são complicados de obter com precisão em protocolos assíncronos baseados no receptor (DUTTA, et.al., 2012). Desta forma, nos estudos dos protocolos assíncronos, o valor de E'_{Tx} será dividido em dois elementos:

$$E'_{Tx} = E'_{TxIL} + E'_{TxD} \quad (9)$$

Na equação (9), E'_{TxIL} representa a energia consumida dentro do estado TxIL. Neste estado, como mostrado anteriormente é aleatório para o protocolo baseado no receptor. O E'_{TxD} representa a energia consumida pelo restante do serviço de transmissão após terminado o TxIL. Portanto, como critérios de comparação entre os protocolos assíncronos, a parcela de consumo de E'_{TxIL} não será considerada. Assim, a resposta será mais precisa e isso não gera qualquer influência na comparação entre os protocolos assíncronos, pois todos seguirão o mesmo critério. Na comparação com o protocolo síncrono, é necessário considerar o consumo total. Neste caso, o valor de E'_{TxIL} terá considerado um valor médio.

Para verificar o modelo de energia obtido, será considerado um exemplo de cálculo de consumo de energia na comunicação entre dois módulos sensores no protocolo RITMC para transmitir uma mensagem à uma taxa de 1s. Considerando um período de RIT (*macRITPeriod*) de 100ms, ou $f_H = 10\text{Hz}$, e que somente ocorre uma mensagem de transmissão e uma mensagem de recepção, ou seja, $f_{Tx} = f_{Rx} = 1$. De (6), tem-se:

$$ET_{RxIL} = (10 - f_{Tx} - f_{Rx} - \alpha) * E'_{RxIL} \quad (10)$$

Para o protocolo RITMC proposto, E'_{RxIL} representa a média da energia total de recepção com IL, que foi de 848.04 uJ. O parâmetro de ajuste médio α é 0.47. Então:

$$E_{Total} = ET_{RxIL} + E'_{TxD} + E'_{Rx} \quad (11)$$

O parâmetro E'_{TxD} corresponde à soma de energia de TxHA, TxData e RxDataAck. O valor de E'_{TxD} desconsiderando TxIL foi de 463,44 uJ. E'_{Rx} corresponde à soma total de energia de TxHello, RxHA, RxData e DataAck, e foi de 482,38 uJ. Portanto, a energia total obtida pelo protocolo RITMC de acordo com a equação (11) foi de 7.31 mJ.

Para validação do método, foram feitos testes de medição da comunicação de dois módulos usando o protocolo RITMC proposto. Na Figura 40 é apresentada a medição de corrente de um período de dados de 1s na comunicação entre dois módulos sensores.

No experimento da Figura 40, foram utilizados dois módulos sensores com um período de RIT de 100 ms enviando dados de 127 bytes a cada 1s. O consumo medido entre os instantes 2 e 3s compreendido entre um período completo foi de 11,52 mJ considerando o TxIL. Quando TxIL não é mais considerado, o valor medido de energia foi de 7,01mJ. O valor médio medido foi de 7.11 mJ.

Figura 40 – Comunicação dois módulos a cada 1s no protocolo RITMC

A Figura 41 mostra uma comparação entre a energia estimada e medida para o protocolo RITMC com tempos de medição de 1s a 3s. O experimento foi novamente com dois módulos sensores e com período de RIT de 100ms. Foram feitos 10 testes para cada um dos períodos analisados. O valor estimado foi calculado utilizando a equação (11). O gráfico mostra a comparação da energia total considerando o valor médio para TxIL (E_{tot}) e a energia parcial ($E_{semTxIL}$) desconsiderando o parâmetro TxIL da equação (11).

Figura 41 – Comparação entre energia medida e estimada para protocolo RITMC

A análise da Figura 41 mostra que os valores estimados de energia (EST) estão dentro do desvio padrão do valor medido (MED). Desta forma, pode-se comprovar que o valor estimado pode ser uma boa aproximação do consumo para o RITMC para um período fixo de 100ms.

Também foram realizados testes alterando o tamanho da janela do RIT como forma de ajuste do algoritmo para aumento da latência e consequentemente diminuição do consumo de energia. Para este experimento foi alterado o parâmetro *RITTimePeriod* variando entre 100ms a 1000 ms para dois módulos sensores igualmente configurados. Para a transmissão foi utilizado o envio de pacote de dados de 100 bytes variando o período de transmissão de dados de 1 a 3 segundos. Foram feitas 5 coletas de dados e obtida a média entre eles. A Tabela 9 resume os resultados.

Tabela 9 – Energia média em diferentes períodos de RIT para protocolo RITMC

Período Dado[s]	RIT Period	100 [ms]	200 [ms]	300 [ms]	400 [ms]	500 [ms]	1000 [ms]
1	Med	(9.65;1.64)	(5.27;0.31)	(3.19;0.22)	(3.02;0.35)	(2.21;0.015)	-
	Est	9.87	5.63	4.19	3.51	3.09	-
2	Med	(18.56;1.11)	(8.77, 0.91)	(4.75, 0.08)	(4.55, 0.10)	(3.36, 0.19)	(1.85, 0.21)
	Est	18.35	9.87	6.99	5.63	4.78	3.09
3	Med	(29.92;0.54)	(13.09;0.27)	(9.00;0.21)	(13.93;0.21)	(5.27;0.47)	(1.83.0.08)
	Est	26.83	14.11	9.87	7.75	6.48	3.93

A Tabela 9 mostra a média de consumo obtida para diferentes taxas de dados em diferentes janelas de RIT sempre considerando o consumo total, ou seja, incluindo o TxIL. Como critério de validação do modelo estimado de energia com diferentes taxas de RIT, foi comparado os valores medidos da Tabela 7 com o estimado através da equação (11). O gráfico da Figura 42 mostra a comparação entre o valor medido e o estimado para um período de transmissão de dados de 2 segundos para o protocolo RITMC.

Figura 42 –Consumo medido e estimado para período de dados de 2s no protocolo RITMC

A análise da Figura 42 mostra que o consumo estimado (E_{Est}) apresenta um resultado estimado de consumo de energia próximo ao valor medido (E_{Mes}). Uma análise estatística de variância foi realizada para o experimento de variação do período de RIT, para verificar se os valores medidos e estimados são condizentes. A energia média estimada foi de 8.034 mJ com desvio padrão de 5.093, enquanto que para a energia média medida (MED) foi de 6.964 com desvio de 5.706. Também foi feita uma análise de variância usando o método Tukey de confiança de 95%. O resultado da comparação mostrou um PValue de 0.312 que é menor que 0.5. Desta forma, a análise da confiança mostrou que os valores medidos e estimados são estatisticamente equivalentes.

Por fim, neste experimento é realizado uma comparação entre o consumo de todos os protocolos assíncronos com um período fixo de RIT (*macRITPeriod*) de 100ms. Para obter o consumo de energia para cada protocolo, os tempos médios foram medidos a partir de cada estado para transmitir e receber uma informação de 127 bytes de dados. Os valores médios obtidos para cada protocolo analisado são apresentados na Tabela 7.

Seguindo o modelo de energia da equação (11) foi feita a estimativa de consumo de cada um dos protocolos com as diferentes taxas de dados. As taxas variaram de 1 pkt/250ms (0.5 Kbps) até 1 pkt/5000ms (0.025 Kbps). Os resultados estão apresentados na Figura 43.

Figura 43 – Gráfico do custo de Energia para transmissão de dados para os diferentes protocolos

A análise do gráfico da Figura 43 mostra que o protocolo RITMC tem uma economia de energia significativa com relação aos outros protocolos principalmente quando a taxa de comunicação é menor. Isto é explicado pelo fato de a janela do RxIL ser menor para o protocolo. Os protocolos A-MAC e ARM também apresentam resultados melhores que o AMCA.

Consumo protocolo síncrono

Os experimentos descritos nesta seção referem-se à comparação do consumo do protocolo assíncrono proposto e do síncrono TSCH. Os testes foram feitos de forma análoga aos protocolos assíncronos mostrados anteriormente, porém com alteração do período do RIT. Quanto ao protocolo síncrono, os testes foram realizados utilizando as mensagens dos protocolos das camadas MAC e de RPL de forma a não alterar o sincronismo do algoritmo.

Em relação ao protocolo TSCH dentro da plataforma OpenWSN, ele trabalha com janelas (slots) de tempo de 15ms. Para uma rede com poucos módulos, inicialmente, o tempo de scan ou macrociclo é formado por 6 slots de tempo com os 16 canais de frequência. Cada slot de tempo pode ser alocado como de recepção ou transmissão, onde o slot 0 é fixo para recepção do *beacon* de sincronismo. Caso a rede tiver um aumento da quantidade de módulos sensores, será alocado mais slots de tempo. Na Figura 44 é mostrado o consumo de três ciclos de trabalho do protocolo TSCH.

Figura 44 – Consumo de transmissão e recepção de um RPL.DIO no protocolo TSCH

De acordo com a Figura 44 cada pico de consumo significa a atividade do rádio na execução de um slot de tempo. Cada slot quando ligado habilita o rádio para transmissão ou recepção. O rádio não tem atividade quando está no estado de repouso. Na figura, o intervalo de tempo (a) representa um ciclo de trabalho e o intervalo de tempo (b) representa um período de scan ou macrociclo. O intervalo de tempo restante (b-a) é o tempo de repouso.

Modelo de energia

Para obter um modelo de energia para o TSCH foram feitos testes variando o período da mensagem RPL.DIO de 1 a 3s e medido o macrociclo para cada um dos serviços de transmissão, recepção e de IL. Para envio de dados, foi utilizado, na análise do consumo, a transmissão e recepção do RPL.DIO. A mensagem RPL.DIO possui 81 bytes e sem confirmação. O experimento consistiu em 10 amostras de macrociclo para cada um dos serviços e, em seguida, feita a média de cada um deles. Foram feitos testes para medição do tempo para cada uma das mensagens. A Tabela 10 mostra os valores obtidos.

Tabela 10 – Medição do consumo de cada serviço dentro do slot de tempo no protocolo TSCH

	Tempo [s]	Energia [mJ]	Desvio Padrão [mJ]
EnergiaSlotIL (E_{SIL})	0.0153	0.1245	0.0232
EnergiaSlotRxDIO ($E_{SRx\text{dio}}$)	0.0147	0.1207	0.0146
EnergiaSlotTxDIO ($E_{STx\text{dio}}$)	0.0143	0.1710	0.0104

A Figura 45 mostra uma captura do consumo do rádio na comunicação entre dois módulos sensores. A captura representa o momento entre as transmissões de duas mensagens

RPL.DIO (TXDIO) cujo duração foi de 1.98s. Dentro da captura ainda existe a recepção (RXDIO) do protocolo TSCH. A energia total medida neste intervalo foi de 8.77 [mJ].

Figura 45 – Consumo do rádio no protocolo TSCH em um período de RPL.DIO de 2s

A média de tempo medido entre os slots foi de 15.1ms. O ciclo de trabalho medido dentro de um período foi em média de 87.5ms. O tempo de repouso foi de 78.2ms. Portanto, o macrociclo do TSCH com 6 slots alocados foi em média de 166ms.

De acordo com os tempos obtidos anteriormente foi estimado o custo de energia do protocolo E_TSCH com a equação (12).

$$E_{TSCH} = f_{SIL} * E_{SIL} + f_{SRxDIO} * E_{SRxDIO} + f_{STxDIO} * E_{STxDIO} \quad (12)$$

Onde E_{SIL} é o custo de energia médio de um slot em IL. O E_{STxDIO} é o custo de energia médio de um slot transmitindo uma mensagem RPL.DIO. O E_{SRxDIO} é o custo de energia médio de um slot recebendo uma mensagem RPL.DIO. Os parâmetros f_{SIL} , f_{SRxDIO} , f_{STxDIO} correspondem às respectivas frequências de ocorrência das mensagens.

O cálculo da frequência de slot é baseado no período de transmissão de um pacote. Na comunicação entre dois módulos sensores, para uma taxa de transmissão de dados fixa, tem-se:

$$f_{SIL} = \frac{P_{tx}}{P_M} * NRSlots - (f_{SRxDIO} + f_{STxDIO}) \quad (13)$$

Onde, P_{tx} é o período da transmissão, P_M é o período do macrociclo, e $f_{STxDIO} = f_{SRxDIO} = 1$, admitindo uma transmissão e uma recepção na comunicação entre dois módulos sensores.

Considere que, o macrociclo para o OpenWSN possui slot de 15ms, o número de slots por ciclo de trabalho é seis, e o tempo de repouso de 80ms. Então $P_M = 0.15 * 6 + 0.80 = 0.17s$. E para um período de transmissão de RPL.DIO igual a um segundo, tem-se:

$$f_{IL} = 1/0.17 * 6 - (2) = 34 \quad (14)$$

e desta forma, de acordo com (8) tem-se:

$$E_{TSCH} = 34 * 0.1245 + 1 * 0.1207 + 1 * 0.1710 = 4.5247mJ \quad (15)$$

Então, a energia para transmissão e recepção de um pacote de dados é de 4.52mJ por segundo. Para comprovar o valor calculado foram feitos testes variando o período da transmissão da mensagem RPL.DIO de 1 a 3s, e comparado o resultado medido e calculado. Foram realizados 10 experimentos para cada período de dados analisado. A comparação entre os dados é mostrada na Figura 46.

Figura 46 – Comparação consumo medido e calculado no protocolo TSCH

De acordo com a Figura 46, o consumo calculado (E_{Est}) de acordo com a equação (12) apresenta um resultado estimado de consumo de energia para dois módulos sensores similar ao valor medido (E_{Mes}). Uma análise estatística de variância foi realizada para verificar se os valores medidos e estimados são estatisticamente equivalentes. A energia média estimada (EST) foi de 8.607 mJ com desvio padrão de 2.768 enquanto que para a energia média medida (MED) foi de 8.445 com desvio de 2.657.

Por fim, houve uma comparação entre o consumo do protocolo síncrono e assíncrono baseado no modelo estimado dos dois protocolos. Apesar do teste do RITMC não ser exatamente o mesmo do TSCH, onde as mensagens não são iguais, e com o TxIL ser aproximado, os resultados anteriores mostraram que é possível ter uma boa aproximação do estimado.

Para estes testes foi alterado o parâmetro *RITTimePeriod* variando entre 100 a 2000 ms enquanto que o TSCH foi utilizado o padrão com tempo entre slots de 15ms e macrociclo de 166ms. A Figura 47 mostra os resultados obtidos.

Figura 47 – Comparação consumo entre RITMC e TSCH

De acordo com a Figura 47, o protocolo TSCH tem um consumo muito similar ao RITMC com período de RIT de 200ms, o que é esperado já que o macrociclo dos dois é muito semelhante. As condições do teste, apesar de não serem exatamente iguais, podem ser consideradas válidas na comparação. Outra análise pode ser feita relacionada ao consumo dos dois protocolos. A corrente do protocolo RITMC é maior que a do TSCH. Isto pode ser explicado pelo fato do algoritmo deixar mais tempo o rádio ligado esperando a comunicação. Porém a vantagem dos protocolos assíncronos é mostrada quando a janela do RIT é maior. Neste caso, o consumo se torna muito pequeno, comparado com o protocolo síncrono.

6.2 Taxa de entrega dos protocolos assíncronos

Nesta seção serão mostrados os resultados dos testes de taxa de entrega para os protocolos assíncronos, cujo objetivo é analisar como estes se adaptam à variação de tráfego na rede.

Para avaliação da taxa de entrega, o experimento consistiu de cada módulo sensor enviar mensagens de dados gerado na própria camada MAC para um outro vizinho pré-determinado em uma topologia fixa. Neste caso, utilizou-se do serviço de *LiveList* para execução destes testes. A mensagem de dados era de 127 bytes com confirmação. A retransmissão na implementação está relacionada a camadas superiores, e neste teste não houve retransmissão de pacote. Desta forma o teste não alterou o mecanismo da camada MAC e foi possível testar realmente a taxa de entrega dos pacotes.

O experimento consistiu de todos os módulos enviando mensagens continuamente para um de seus vizinhos em uma determinada taxa de pacotes. Foi medida a performance dos protocolos RITMC proposto com os protocolos A-MAC, AMCA e ARM variando o intervalo de geração de dados de 1pacote/250ms (0.5 Kbps) até 1pacote/5000ms (0.025 Kbps) em cenários utilizando multicanal e canal único.

Primeiramente foram realizados testes no cenário da Figura 33, com canal único. Os testes consistiram de cada módulo sensor enviando 500 pacotes de dados com reconhecimento do receptor. Foram realizados dez testes para cada período de dados variando de 250ms a 5000ms para cada protocolo. O período do RIT (*macRITPeriod*) foi de 100ms para todos os protocolos. Cada equipamento era responsável por verificar os números de erros e acertos e reportar na rede através da própria mensagem de dados. Estas informações foram obtidas através da captura do software Wireshark. Os resultados obtidos são apresentados na Figura 48.

Figura 48 – Avaliação da taxa de entrega da rede com canal único

A Figura 48 mostra o gráfico da taxa de entrega de pacotes em porcentagem e o intervalo de confiança para cada protocolo em cada período analisado. Pode-se notar que, para períodos menores de dados tem-se uma maior perda de pacotes. Este resultado é esperado já que para taxa de comunicação próxima ao período (*macRITPeriod*) de 100ms, causa muita colisão. Porém, com o aumento do período, o erro diminui. Os melhores resultados foram obtidos pelo AMCA e RITMC com maiores valores e menor variação. Esta afirmação é comprovada pela análise da variância realizada em cada período. A Figura 49 mostra um exemplo da análise da variância realizado para período de 3000ms.

Figura 49 – Análise da variância com canal único para período de 3000ms

Uma análise estatística de variância foi realizada para o experimento de variação do período de RIT, para verificar a similaridade dos protocolos. A taxa de entrega média foi de

0.82 com desvio padrão de 0.12 para o protocolo A-MAC. O protocolo ARM apresentou taxa de entrega média de 0.85 com desvio de 0.09. O protocolo AMCA teve média de 0.91 com desvio de 0.10, enquanto que o RITMC proposto teve média de 0.93 com desvio de 0.09. A Figura 49.a mostra estes valores. Também foi feito uma análise de variância usando o método Tukey de confiança de 95%. O resultado da comparação mostrou um PValue 0.01. De acordo com Figura 49.b, estatisticamente o protocolo RITMC é similar ao AMCA, enquanto o AMAC se assemelha ao ARM. Esta mesma análise se repetiu em quase todos os períodos. Isto ocorre pelo reduzido número de passos no tratamento da mensagem.

Um segundo teste foi realizado para avaliação da taxa de entrega em ambiente multicanal. Foi montada a mesma topologia da rede da Figura 33 onde variou-se o número de módulos em cada canal. Neste caso, foi mantida uma taxa fixa de 1pkt/1000ms ou 0,1270 Kbps. O número de módulos variou de 1 a 6 módulos por canal. O objetivo aqui foi verificar a performance de cada protocolo quanto a ocupação de cada canal.

Os testes consistiram de cada módulo sensor enviando 500 pacotes de dados com reconhecimento do receptor. Foram realizados 10 testes para cada intervalo considerado, com resultado apresentado na Figura 50.

Figura 50 – Avaliação da taxa de entrega da rede com multicanal

A Figura 50 mostra a porcentagem de entrega e o intervalo de confiança para cada protocolo em cada grupo de módulos por canal analisado. Com a análise do gráfico, pode-se observar que os protocolos têm características semelhantes quando com 5 ou seis módulos sensores no mesmo canal e o protocolo AMCA tem uma ligeira melhora com menos sensores

no canal. Uma análise da variância foi realizada para cada intervalo de teste. A Figura 51 mostra a análise para o teste de 6 módulos por canal.

Figura 51 – Análise da Variância dos protocolos assíncronos multicanal com seis módulos por canal

Uma análise estatística de variância foi realizada para o experimento de variação do período de RIT, para verificar se a similaridade com os protocolos. A taxa de entrega média foi de 0.83 com desvio padrão de 0.09 para o protocolo A-MAC. O protocolo ARM apresentou taxa de entrega média de 0.86 com desvio de 0.09. O protocolo AMCA teve média de 0.93 com desvio de 0.08 enquanto que o RITMC proposto teve média de 0.91 com desvio de 0.08. A Figura 51.a mostra estes valores. Também foi feita uma análise de variância usando o método Tukey de confiança de 95%. O resultado da comparação mostrou um PValue 0.01. De acordo com Figura 51.b, estatisticamente o protocolo RITMC é similar ao AMCA para a maioria dos intervalos.

Os testes com e sem multicanal mostraram que o protocolo AMCA apresentou melhores resultados de taxa de entrega, porém o RITMC proposto possui resultados compatíveis. Vale ressaltar que o protocolo RITMC possui vantagem em relação ao AMCA pelo mecanismo de tratamento de erro ainda dentro do mesmo ciclo da mensagem.

6.3 Diagnóstico do protocolo RITMC

Nesta seção serão mostrados os testes realizados para validação da função de diagnóstico do receptor e verificar o quanto ela influencia nos resultados do protocolo RITMC.

Para o protocolo RITMC, o diagnóstico ocorre somente no receptor da mensagem que pode diagnosticar tanto erro no início da comunicação (ECW1) quando ocorre uma colisão ou

algum erro na mensagem de reconhecimento da anúncio (HA), ou no momento da recepção do dado (ECW2). Então o receptor envia uma mensagem ao final do período notificando o transmissor que houve erro na transmissão.

O experimento consistiu de todos os módulos enviando mensagens continuamente para um de seus vizinhos em uma determinada taxa de pacotes conforme cenário da Figura 33. Para este experimento também foi utilizado o serviço de *LiveList*, que não realiza retransmissão de mensagens. Neste caso, o transmissor enviava uma mensagem de 100 bytes de dados e esperava o reconhecimento pelo receptor, em um total de 500 mensagens. Foi medida a performance do protocolo RITMC variando o intervalo de geração de dados de 1pacote/250ms (0.5Kbps) até 1pacote/5000ms (0.025 Kbps) em ambiente com canal único. Foram realizados 10 testes com e sem diagnóstico para verificar a influência do diagnóstico na recepção das mensagens. Na Figura 52 é mostrado o gráfico de taxa de entrega com 6 módulos sensores variando o período de 250ms a 5s.

Figura 52 –Taxa de entrega de pacotes com e sem diagnóstico no protocolo RITMC.

A análise da Figura 52 mostra que sempre teve melhorias do diagnóstico para todas os períodos de transmissão de dados analisados. A análise estatística de variância foi realizada para o experimento. A taxa de entrega com diagnóstico teve média geral de 91% com desvio de 7.4% e sem diagnóstico teve média geral de 88% com desvio de 7.2%. Uma análise estatística da variância mostrou um P.Value de 0.02. A Figura 53 mostra um gráfico deste resultado geral.

Figura 53 – Análise da variância da taxa de entrega com e sem diagnóstico

Na Figura 54 é mostrada a porcentagem de mensagem de erro enviadas pelo receptor para cada período analisado.

Figura 54 – Taxa de erro das mensagens enviadas pelo receptor

De acordo com a Figura 54, o gráfico mostra que em média de 2 a 5% das mensagens recebidas pelo receptor encontraram algum problema que poderiam ser considerados para recuperação. Então, foi enviada uma mensagem de erro, onde foram somadas as mensagens de ECW1 e ECW2. Destas mensagens de erro 0,5% foram mensagens do tipo ECW1 e 99,5% foram do tipo ECW2. É possível notar também que com períodos de dados maiores de 1s ocorrem mais mensagens de erro do que em períodos menores. A conclusão que é feita neste caso é que para baixos períodos, próximos da janela do RIT, ocorre, muito mais falhas de

pacotes devido a outros fatores, às colisões ou reprogramação do rádio, e isso influencia na recepção do pacote.

Um outro teste foi realizado para verificar a influência do diagnóstico com o aumento do número de módulos sensores. Para isso, variou-se o número de módulos de dois a seis, mantendo o período de dados fixo de 1s. Foram feitos 10 testes onde o transmissor enviava 500 mensagens e esperava a resposta, e obtidas a taxa de entrega e a porcentagem de erros enviados pelo receptor. A Figura 55 mostra os resultados obtidos.

Figura 55 – Taxa de entrega e taxa de erro com período fixo de 1000ms

A Figura 55.a mostra a taxa de entrega de pacotes visto pelo transmissor, enquanto que a Figura 55.b mostra a taxa de transmissão da mensagem ECW vista pelo receptor. Analisando as figuras, é possível comprovar os resultados obtidos nos testes anteriores, onde para 2 módulos sensores não existe colisão e já com 3 módulos sensores começam a aparecer as colisões que em média ficou em torno de 1 a 2% de pacotes recebidos. Com 6 módulos sensores teve a maior taxa e o maior desvio padrão.

6.4 Aplicação de IoT

Nesta seção são apresentados os testes com uma aplicação utilizando o ambiente de IoT do OpenWSN. A aplicação IoT basicamente consiste na requisição de uma informação através de uma plataforma *web* para qualquer módulo da rede, ou seja, somente explorando o fluxo divergente. Nos experimentos foram utilizados até 16 módulos sensores espalhados em uma topologia mesh conforme topologia mostrada na Figura 34 e Figura 35.

Para gerar a aplicação IoT, foi utilizado o software de requisição CoAP mostrado na Figura 31. A métrica neste caso foi a avaliação da taxa de entrega e a latência para alcançar cada um dos módulos da rede. Foram feitos testes do RITMC proposto comparando com os protocolos assíncronos estudados e também testes com o protocolo síncrono do OpenWSN sempre utilizando multicanal. Os resultados são mostrados a seguir.

Aplicação protocolos assíncronos

Nesta seção serão mostrados os resultados dos testes de aplicação para os protocolos assíncronos. O objetivo deste experimento é analisar a resposta dos protocolos assíncronos no ambiente IoT do OpenWSN.

No primeiro experimento, foram feitos testes multicanais, com um módulo por canal, e somente um ramo da Figura 34 com até 6 módulos. Para cada experimento foram realizadas requisições da aplicação web de 100 bytes de dados a cada 10 segundos para um determinado módulo da rede. Foram realizadas em média 100 requisições da aplicação para um determinado módulo e repetido o teste 10 vezes por salto em um total de 5 saltos. Desta forma, foi possível medir a latência bem como a taxa de entrega de pacotes da aplicação. Os testes foram repetidos igualmente para todos os protocolos assíncronos estudados. Na Figura 56 é apresentado o resultado da latência para cada salto na rede.

Figura 56 – Latência da rede em aplicação IoT no cenário com 6 módulos.

A partir do gráfico da latência da Figura 56 é possível obter os tempos entre saltos de cada protocolo. O valor médio de cada salto de todos os protocolos foi de 0.94 segundos no primeiro salto, 1.37 segundos no segundo, 1.80 segundos no terceiro, 2.31 segundos no quarto e 3.15 segundos no quinto salto. A análise do gráfico mostrou que os protocolos tiveram comportamentos muito próximos da média de 0.55 segundos entre saltos (após o primeiro salto) e um desvio padrão de 0.15 segundos. Os protocolos RITMC e AMCA tiveram comportamento muito semelhantes. Os piores resultados foram obtidos pelo protocolo ARM. A Figura 57 mostra o gráfico da taxa de entrega de pacotes pelo número de saltos.

Figura 57 – Taxa de entrega em Aplicação IoT no cenário com 6 módulos

Quanto à taxa de entrega mostrada na Figura 57, os valores são próximos quando o número de saltos é pequeno. Porém, quando se tem muitos saltos, os protocolos com algoritmos maiores (A-MAC e ARM) apresentam pioras em seus resultados, quando comparados aos protocolos com algoritmos menores (RITMC e AMCA).

Um segundo teste foi realizado montando exatamente o cenário da Figura 34 em ambiente multicanal, formando uma rede com 16 módulos sensores. Neste caso, foram colocados três módulos por canal a cada salto, e analisado o desempenho a cada salto. Foram realizadas 100 aquisições para cada um dos módulos, e 10 testes iguais totalizando 1000 amostras. Os resultados são resumidos nas Figuras 61 e 62.

Na Figura 58 é possível visualizar o resultado da latência da aplicação para atingir cada um dos módulos a cada salto. Analisando a latência da rede, o protocolo AMCA apresentou os menores tempos para atingir cada módulo em uma rede com 3 módulos por canal. O RITMC proposto teve desempenho semelhante ao AMCA com uma diferença média de 15% com o AMCA. O A-MAC teve diferença média de 48% enquanto que o ARM teve uma diferença de 74%.

Figura 58 – Latência da rede em Aplicação IoT no cenário com 16 módulos

Na Figura 59 é apresentado o resultado da latência da rede para a aplicação atingir cada um dos módulos a cada salto.

Figura 59– Taxa de entrega de aplicação IoT no cenário com 16 módulos

Quanto à taxa de entrega mostrada na Figura 59, os valores são próximos quando o número de saltos é pequeno. À medida que o número de saltos aumenta os protocolos RITMC e AMCA são muito parecidos em desempenho com 80% de entrega de pacotes com 4 e 5 saltos. A ligeira melhora do protocolo RITMC em relação ao AMCA é devido ao fato do diagnóstico que proporciona uma retransmissão do pacote quando houve colisão, ocorrer ainda dentro do mesmo ciclo. Quanto aos protocolos A-MAC e ARM, apresentaram desempenho similares, porém com taxas de entrega inferiores, na ordem de 50% de entrega com 4 e 5 saltos.

Uma conclusão que pode ser desenhada é que os protocolos com algoritmos maiores com maior número de mensagens trocadas dentro de um slot time (A-MAC e ARM) tem desempenho piores que os AMCA e RITMC. O melhor tratamento do erro, neste caso, faz com que o RITMC tenha uma melhor performance de taxa de entrega do que o AMCA. Porém isso leva a um pequeno atraso na comunicação como um todo. As redes começam a degradar quando se tem muitos saltos para executar, e isto é piorado com uma rede com uma quantidade grande de mensagens no mesmo canal, como é o caso dos protocolos A-MAC e ARM.

Aplicação protocolo síncrono

Nesta seção serão mostrados os resultados dos testes realizados comparando uma aplicação IoT usando o protocolo RITMC proposto com o protocolo síncrono TSCH. O objetivo

deste experimento é analisar a resposta dos protocolos assíncronos comparado com o síncrono no ambiente IoT do OpenWSN.

Para este experimento, foram realizadas requisições da aplicação *web* de 100 bytes de dados a cada 10 segundos para um determinado módulo da rede segundo o cenário da Figura 35. A estratégia utilizada na obtenção das medições foi a utilização do próprio pacote de dados para trazer as informações de diagnóstico de erros, bem como a medição pela aplicação do tempo decorrido e do número de erros na entrega fim-a-fim. A partir de comandos enviados pela aplicação foi possível medir a latência bem como a taxa de entrega de pacotes da aplicação. Foram realizados em média 100 requisições da aplicação, e repetido o teste 3 vezes por salto em um total de 5 saltos.

Para o protocolo RITMC, variou-se o período do RIT de 100 a 300ms. Para o protocolo síncrono TSCH sempre se utilizou a mesma janela de 15ms de tempo de slot com macrociclo de 160ms. A Figura 60 mostra o resultado da latência da rede para a aplicação atingir cada um dos módulos a cada salto.

Figura 60 – Latência da aplicação IoT em um cenário com 11 módulos

De acordo com a Figura 60 é possível analisar o resultado da latência da aplicação para atingir cada um dos módulos a cada salto em um cenário multicanal. Para o caso do TSCH o mecanismo do multicanal é utilizado o salto em frequência dinâmico, ou seja, a cada ciclo, os módulos buscam outra frequência para se comunicar. Já no protocolo RITMC, a frequência é fixa onde se tem 2 módulos por canal em cada salto. Analisando a latência da rede, o protocolo TSCH tem um melhor resultado devido à sua estabilidade e seu tempo de slot pequeno. O tempo em cada salto é de 0.12s em média com desvio de 0.14s. O protocolo RITMC considerando uma janela de RIT de 100ms, tem como média, em cada salto, 0.33s com desvio de 0.17s. O

RITMC com janela de RIT de 200ms, tem média de 0,80s com desvio de 0.37s. E, por fim, o RITMC com janela de RIT de 300ms, possui média de 1.16s com desvio de 0.55s. O que se pode concluir é que o TSCH é muito mais estável, devido ao sincronismo. O protocolo RITMC com janela de RIT de 100ms é o que mais se aproximou ao TSCH. Com o aumento da janela de RIT a latência é maior, também ocorre um aumento na variação de tempo entre os saltos e também do desvio padrão.

Na Figura 61 é apresentado o resultado da taxa de entrega da aplicação para atingir cada um dos módulos a cada salto em um cenário multicanal.

Figura 61 – Taxa de entrega da aplicação IoT em um cenário com 11 módulos

Quanto à taxa de entrega mostrada na Figura 61, o protocolo TSCH é mais estável devido ao seu mecanismo de salto em frequência e sua janela de tempo, evitando assim as colisões. À medida que os saltos vão ocorrendo, o protocolo tem perdas de comunicação. No limite do número de saltos o TSCH teve uma queda acentuada. O protocolo RITMC com janela de RIT de 100ms mostrou valores próximos ao TSCH. Porém quando a janela de RIT é aumentada, aumenta-se a latência e também a degradação da comunicação. Isto pode ser explicado pelas possíveis sintonias internas do protocolo, onde tempos maiores não estão bem ajustados dentro do algoritmo do OpenWSN.

A Figura 62 mostra o roteamento obtido pelo OpenVisualizer nos testes do protocolo RITMC para o cenário da Figura 35. A topologia possuía dois caminhos para cada módulo sensor. Como a rota é dinâmica e alocada de acordo com critérios de taxa de pacotes recebidos e transmitidos ele encontra a melhor rota para formar o grafo da Figura 62.

Figura 62 – Roteamento do protocolo RITMC com 11 módulos sensores

Um último experimento foi realizado com o protocolo RITMC dentro do ambiente mostrado na Figura 36. O objetivo deste experimento é explorar a rede com mais do que 5 saltos e verificar o desempenho quando tivesse muitos saltos. Os testes foram realizados com potência de -5dBm com 13 módulos sensores em uma distância de 5 a 6 metros de cada um. Os testes consistiriam em enviar mensagens da aplicação web de 100 bytes para cada um dos módulos da rede. A partir de comandos enviados pela aplicação, foi possível medir a latência bem como a taxa de entrega de pacotes da aplicação. A Figura 63 mostra os resultados para a latência da rede, enquanto que a Figura 64 mostra os resultados para a taxa de entrega dos pacotes.

Figura 63 – Latência da rede com protocolo RITMC com até 12 saltos

Figura 64 – Taxa de entrega da rede com protocolo RITMC com até 12 saltos

A análise das Figuras 67 e 68 mostrou que primeiramente, foi possível enviar os pacotes dentro do sistema OpenWSN devido à mudança que foi feita no código para tratamento de mais de 5 saltos, onde se poderia atingir até o limite de 19 saltos devido à implementação realizada, mas o fator limitante, neste caso, foi o recurso de hardware. Outro ponto que se pode analisar neste experimento é que os resultados anteriores onde os módulos possuem visada direta apresentou uma latência ligeiramente menor da rede até o quinto salto comparado com este experimento onde os módulos estão em diferentes ambientes. Mas a latência da rede é incremental e quase linear com pequeno desvio padrão. Quanto à taxa de entrega, esta foi degradando à medida que foram sendo incrementados os saltos. Aqui ocorreram vários problemas devido à rede ter somente 1 caminho para alcançar o nó e também o fato de ter muitos obstáculos na comunicação, como eventuais perdas do roteamento devido à comunicação entre os nós, e o emparelhamento de mensagens onde às vezes chegava 2 ou 3 mensagens de resposta atrasadas. Com isso, chegou-se a ter com doze saltos uma taxa de erro de 50% em média.

6.5 Mecanismo multicanal

Nesta seção são apresentados os testes para validação do mecanismo de escolha de melhor canal para o protocolo RITMC. Para validação deste mecanismo, foram realizados testes simulados da proposta do algoritmo básico.

No contexto dos mecanismos multicanal, o serviço de escolha do melhor canal determina qual o melhor canal (BC) para o módulo sensor e coordena a notificação na mudança de canal para a vizinhança do módulo. O serviço pode ser dividido em três etapas distintas. A primeira é a verificação da necessidade da mudança de canal. A segunda etapa é a descoberta do melhor canal e a notificação da vizinhança da escolha realizada. Por fim, a última etapa está relacionada com a efetivação da mudança de canal.

Com relação ao desenvolvimento do mecanismo de escolha de melhor canal apresentado no capítulo 4, foi implementado somente no software Matlab. Os resultados obtidos foram simulados no Matlab com o objetivo de verificar o comportamento da rede alterando a quantidade de canais e a quantidade de módulos sensores. O objetivo do experimento foi de analisar o comportamento da rede quando ocorre uma mudança da qualidade do sinal devido à variação do número de módulos por canal ou a mudança das condições da rede, alterando ou diminuindo a taxa de erros.

Quanto à simulação, foi implementado o algoritmo no Matlab onde era possível alterar a quantidade de canais disponíveis e a quantidade de módulos sensores. A simulação era executada continuamente, dependendo do número de iterações configuradas, ou seja, não foi considerado um modelo de propagação do sinal mais realista, considerando distância entre os nós ou taxa de erro devido à potência do rádio, ou reflexão do sinal devido a obstáculos.

De acordo com a simulação, inicialmente, escolhe-se aleatoriamente os melhores canais. O parâmetro de mudança é realizado através da escolha aleatória de uma taxa de erro dependente da quantidade de módulos por canal. Desta forma, o erro é maior para um canal com densidade maior de módulos sensores. A taxa de erro varia de 0 a 10% para cada módulo. A mudança das características da rede é realizada a cada n iterações no algoritmo onde n varia de 10 a 100 iterações. A mudança de canal somente é realizada uma vez por iteração.

Foram realizados experimentos com uma densidade pequena até uma quantidade grande de nós sensores. Os gráficos obtidos mostram a quantidade de módulos sensores e o *rank* acumulado em cada canal.

Em um primeiro experimento, foi montada uma rede pequena com 10 módulos e 10 canais, e realizada 1000 iterações. Na primeira iteração, mostrada na Figura 65.a os módulos sorteiam os seus melhores canais de forma aleatória. Baseados nos melhores canais escolhidos, eles trabalham nestes canais e calculam o *rank* de acordo com a equação (3), à cada iteração. A Tabela 11 mostra os resultados do cálculo do rank para cada um dos módulos após 1000 iterações. A cada iteração cada nó avalia se ele deve ou não alterar de canal de acordo com o critério do *rank*. O rank limite para a determinação de mudança de canal foi o valor de 5, ou

seja, valores de rank maiores que 5 gerariam uma mudança de canal. Depois de um determinado período de tempo, são geradas modificações nas características da rede, através na mudança na taxa de erro, e consequentemente no rank de cada canal. Na Figura 65.b é mostrado os resultados da simulação após 500 iterações. Por fim, após 1000 iterações, a rede apresentou a distribuição mostrada na Figura 65.c, onde é possível notar que os canais apresentam uma distribuição melhor do que na primeira iteração. O sinal do *rank* do gráfico representa o rank total do canal. Neste caso, o *rank* do canal é somente um critério para a escolha dos possíveis melhores canais, onde deve ser escolhido o canal com este acúmulo de *rank* com menores valores.

Tabela 11 – Resulta da simulação de cada módulo na 1000a iteração

Mote	BC	NViz	ErrRate	Rank
M1	2	2	0.00	0.40
M2	8	1	0.09	1.09
M3	7	1	0.01	0.26
M4	3	1	0.03	0.52
M5	5	1	0.09	1.05
M6	4	1	0.01	0.32
M7	2	2	0.00	0.40
M8	9	1	0.05	0.67
M9	6	1	0.03	0.50
M10	1	1	0.06	0.80

Figura 65 – Ocupação de canais em uma rede com 10 módulos

Em um segundo experimento, foi montada uma rede de médio porte com 50 módulos e 10 canais, e realizada 1000 iterações. A Figura 66.a mostra a primeira iteração da rede, e a Figura 66.b mostra a iteração 500, e a iteração 1000 é mostrado na Figura 66.c. A comparação

entre a 1ª iteração e a última é possível notar que os canais apresentam uma distribuição mais homogênea de módulos por canal.

Figura 66 – Ocupação de canais em uma rede com 50 módulos

Para um último experimento, foi montada uma rede grande com 200 módulos e 15 canais, e realizada 1000 iterações. A Figura 67.a mostra a primeira iteração da rede, a Figura 67.b mostra o resultado após 500 iterações, e a Figura 67.c mostra o resultado após 1000 iterações. Novamente, a comparação entre a 1ª iteração e a última nota-se que os canais apresentam uma distribuição mais homogênea de módulos por canal. O valor grande no valor do *rank* do canal está relacionado com a quantidade de nós no canal e o peso devido a esta quantidade de nó no critério adotado no algoritmo para simulação de erro.

Figura 67 – Ocupação de canais em uma rede com 200 nós e 15 canais

6.6 Comentários e discussões

Os resultados referentes à análise dos protocolos assíncronos baseados em RIT mostraram que o protocolo RITMC proposto minimiza o consumo de energia através do mecanismo do reconhecimento inicial que proporciona um RxIL menor. A economia foi de mais de 50% em relação ao protocolo AMCA e de 48% em relação aos protocolos A-MAC e ARM quando trabalhando com redes não carregadas.

A comparação do RITMC com o protocolo TSCH também mostrou resultados satisfatórios em relação ao consumo. Quando o RITMC tem um período de 100ms, o consumo é o dobro do TSCH que tem um macrociclo de 166ms. Para o RITMC com um período de 200ms, o consumo é praticamente o mesmo do TSCH. Para macrociclos maiores, o RITMC se torna mais econômico. No caso do RITMC com período de 300ms, a economia foi da ordem de 50% quando trabalhando com redes não carregadas.

Em relação à taxa de entrega dos protocolos assíncronos, como já é intrínseco do protocolo assíncrono, as colisões, os algoritmos mais otimizados apresentam melhor taxa de entrega de pacotes. Neste caso, os melhores resultados nos testes de comunicação com o vizinho em somente 1 salto e diferentes períodos de dados foram obtidos pelo AMCA e o RITMC. Neste caso, o AMCA apresentou 93% de taxa de entrega média, enquanto que o RITMC apresentou 90% em média. Os protocolos ARM e A-MAC obtiveram média de aproximadamente 86%.

Quanto à taxa de entrega da aplicação, onde é considerado todo o algoritmo de IoT e múltiplos saltos, faz-se necessário um mecanismo multicanal eficiente devido à quantidade de nós sensores na vizinhança. Novamente, os protocolos RITMC proposto e o AMCA apresentaram melhores resultados, devido à independência de canal de controle e também ao algoritmo com menos mensagens necessárias na comunicação. O RITMC teve uma taxa de entrega média de 89%, enquanto que o AMCA obteve média de 85%. Os protocolos A-MAC e o ARM tiveram piores resultados com valores médios de 61% e 60%, respectivamente. Neste caso, o motivo dos resultados obtidos pelos protocolos RITMC e AMCA se devem ao fato do algoritmo ser otimizado, com menos envio de pacotes na comunicação, como também influencia o mecanismo multicanal sem dependência do canal de controle. E apesar do resultado compatível do RITMC e AMCA, o fato do diagnóstico do RITMC proposto ser mais eficiente faz com que ele seja o mais indicado neste caso.

Com relação à comparação com o protocolo síncrono em uma aplicação IoT, o TSCH é mais estável, com pouca variação na taxa de entrega e latência da rede. No entanto, o RITMC

mostrou resultados compatíveis. Em média o TSCH tem uma taxa de entrega total de 93% considerando múltiplos saltos, enquanto que o RITMC com um período de RIT de 100ms obteve 90%. Quando o RITMC trabalhou com um período de 200ms, a taxa de entrega caiu para 86%, enquanto que, com o período de 300ms foi para 78% em média. É possível ver que uma rede com uma grande quantidade de módulos será limitada em links de comunicação e no número de saltos para o protocolo síncrono. Para uma rede com protocolo assíncrono, ela será independente do tempo de slot e poderá trabalhar em um número de saltos maior.

Em relação à latência da aplicação, os protocolos assíncronos apresentaram valores muito próximos uns dos outros. O protocolo AMCA teve a menor latência para 5 saltos, com tempo de 1,54s em média. O RITMC teve um tempo próximo de 1,87s em média. O A-MAC apresentou uma latência 2,23s, e o ARM apresentou 2,68s em média. Quando comparado com o TSCH, o RITMC também obteve um resultado inferior. Para atingir os 5 saltos a latência média do TSCH foi de 0,78s, enquanto que o RITMC com período de 100ms foi 1.56s, ou seja, quase o dobro do tempo. Para o RITMC com período de 200ms foi de 2,88s em média e para o período de 300ms foi de 4,26s em média.

Quanto à análise do algoritmo de busca de melhor canal das Figuras 65, 66 e 67 mostram resultados parecidos para as redes, onde após uma inicialização aleatória de canais, a tendência da rede, com o aumento da taxa de erro é de se normalizar com uma quantidade equivalente de módulos por canal.

7 CONCLUSÃO

Nesta tese foi proposto um protocolo de acesso ao meio assíncrono multicanal para WSN em aplicações de IoT com baixa taxa de transferência de dados e baixo consumo e que não possui características rígidas de determinismo. Para isso foi utilizada a plataforma OpenWSN, que mostrou resultados satisfatórios e confiáveis quanto à análise já que é uma plataforma atual, com os principais protocolos de IoT, sendo objeto de pesquisa de grandes universidades mundiais.

No contexto das WSN de baixo custo e baixo consumo, este trabalho propôs o desenvolvimento e a validação do RITMC, um protocolo assíncrono multicanal de camada MAC com transmissão iniciada pelo receptor. O RITMC é baseado no mecanismo de acesso ao meio com reconhecimento inicial proposto pelo protocolo A-MAC, com serviço de diagnóstico mais eficiente. O RITMC utiliza um mecanismo multicanal com mecanismo de melhor canal baseado no modelo proposto pelo protocolo AMCA, porém define de forma completa os serviços *Livelist*, *DeadList*, descoberta da vizinhança e busca do melhor canal, para protocolos assíncronos iniciado pelo receptor. Para o restante das camadas de uma rede IoT foram utilizados os mesmos padrões já utilizados no OpenWSN. Foi provado nesta tese que é possível a implementação de um protocolo assíncrono dentro do ambiente OpenWSN de IoT.

Em relação ao estudo de diferentes tecnologias de camada MAC baseado em RIT, foram implementados três outros protocolos assíncronos multicanais de relevância na literatura para efeito de comparação com o protocolo proposto.

Quanto aos resultados na comparação dos protocolos assíncronos, o protocolo proposto mostrou melhorias significativas em relação ao consumo de energia de mais de 48%. Quanto à taxa de entrega, o protocolo proposto mostrou resultados satisfatórios compatíveis com o protocolo AMCA, que para alguns cenários, mostrou-se melhor. E apesar do resultado compatível, o fato do diagnóstico do RITMC proposto ser mais eficiente faz com que ele seja o mais indicado neste caso.

A comparação do protocolo proposto com o protocolo síncrono, o protocolo síncrono é superior quando comparado a taxa de entrega e latência, devido a estabilidade, o sincronismo e a baixa colisão. Porém o protocolo proposto mostrou resultados compatíveis. Na comparação do consumo, quando o RITMC trabalha com ciclo de trabalho mais lento, ou seja, com uma maior janela de RIT, o resultado apresenta valores superiores ao síncrono.

Em relação aos serviços multicanais, foram propostos serviços de descoberta da vizinhança e escolha do melhor canal que somente foram simulados e não foram implementados na WSN. O motivo da não implementação foi a estratégia inicial de testar múltiplos saltos e, desta forma, optou-se sempre por trabalhar com topologia fixa. Porém, as simulações realizadas mostraram que o algoritmo é viável e atendeu as expectativas.

Outro trabalho realizado dentro desta tese foi o desenvolvimento de um módulo sensor baseado na tecnologia IEEE802.15.4. O módulo desenvolvido se mostrou eficiente, de acordo com os requisitos iniciais propostos de comunicação, formas de alimentação, possibilidade de incorporar outros sensores ou formas de comunicação e de ter um custo reduzido comparando com as outras plataformas existentes. Com relação ao tamanho do código implementado, houve uma pequena mudança no mecanismo do OpenWSN, com um código ligeiramente superior ao protocolo síncrono, porém este código pode ser otimizado com modularização e remoção de códigos de depuração de erros. A conclusão é que o código de um protocolo assíncrono se apresenta, às vezes, superior ao síncrono, devido à quantidade de mecanismos que se inclui no algoritmo para otimizar e melhorar a comunicação.

Durante o desenvolvimento e testes do sistema IoT, optou-se por sempre utilizar WSN com equipamentos reais. A simulação seria importante na validação de uma rede com escalabilidade de módulos sensores, porém, foi possível colher resultados satisfatórios somente com o ambiente real.

Por fim, os protocolos desenvolvidos nesta tese são adequados para aplicações onde se procura uma solução de economia de energia e não se há uma necessidade rígida de determinismo, como aplicações de *smart grid*, tais como monitoração de medidores de consumo de energia elétrica e água, medição de fatores climáticos, a fim de implementar sistemas de previsão de tempo, com sensores de umidade, temperatura, entre outros.

Trabalhos futuros

Nesta seção será apresentada algumas oportunidades relevantes para continuação deste trabalho.

Em relação às estratégias de melhoria de consumo, o mecanismo proposto pelo RITMC é baseado no problema do RxIL. Porém, alguns trabalhos apresentados no capítulo 3 desta tese, como o EH-MAC, propuseram mecanismo de melhoria no TxIL que também possui consumo considerável e aleatório. Foi feito um trabalho inicial, onde foi implementado somente uma parte da proposta. No entanto os resultados não foram satisfatórios, porém é algo que pode ser melhorado. Outro ponto que pode melhorar este trabalho específico com o RPL é trabalhar no roteamento levantado e trabalhar na variação nas características da rede, como foi feito no protocolo L-MAC.

Um outro aspecto seria a implementação do mecanismo proposto em um ambiente simulado, para validação de uma rede com uma escalabilidade maior de módulos. O OpenWSN possui uma plataforma de simulação chamada OpenSim. Porém, o OpenSim não possui mecanismo CSMA/CA e modelo de propagação do sinal, que seriam necessários para uma simulação mais precisa e criteriosa.

Neste trabalho foi proposto um algoritmo para levantamento da rede e do ambiente multicanal que somente foi simulado. Os testes simulados mostraram que a implementação é viável. Desta forma, seria interessante validar o algoritmo proposto em uma plataforma real.

Outro ponto interessante em se investigar é em relação a segurança da rede. A sugestão é estudar o protocolo AES e a compatibilidade do mesmo com os protocolos assíncronos na camada MAC.

REFERÊNCIAS

- ALSKAIF, TAREK, BORIS BELLALTA, MANEL GUERRERO ZAPATA, E JOSE M. BARCELO ORDINAS. “Energy efficiency of MAC protocols in low data rate wireless multimedia sensor networks: A comparative study.” *Ad Hoc Networks* 56 (2017): 141-157.
- ALVI, A. N., S. H. BOUK, S. H. AHMED, M. A. YAQUB, M. SARKAR, E H. SONG. “BEST-MAC: Bitmap-Assisted Efficient and Scalable TDMA-Based WSN MAC Protocol for Smart Cities.” *IEEE Access* 4 (2016): 312-322.
- BEAUDAUX, J., A. GALLAIS, J. MONTAVONT, T. NOEL, D. ROTH, E E. VALENTIN. “Thorough Empirical Analysis of X-MAC Over a Large Scale Internet of Things Testbed.” *IEEE Sensors Journal* 14 (2 2014): 383-392.
- BORGOHAIN, TUHIN, UDAY KUMAR, E SUGATA SANYAL. “Survey of Security and Privacy Issues of Internet of Things.” *CoRR* abs/1501.02211 (2015).
- BORMANN, C.;CASTELLANI, A.P.; SHELBY, Z. “CoAP: An Application Protocol for Billions of Tiny Internet Nodes.” *IEEE Internet Computing*, 16 2012: 62-67
- BOULFEKHAR, SAMRA, E MOHAMMED BENMOHAMMED. “Synchronous receiver initiated MAC protocol for long-lived sensor networks.” *Computers \& Electrical Engineering* 40 (2014): 504-516.
- CANO, C., B. BELLALTA, A. SFAIROPOULOU, E M. OLIVER. “Low energy operation in WSNs: A survey of preamble sampling MAC protocols.” *Computer Networks* 55 (2011): 3351-3363.
- CHANG, TENGFEI, THOMAS WATTEYNE, KRIS PISTER, E QIN WANG. “Adaptive synchronization in multi-hop TSCH networks.” *Computer Networks* 76 (2015): 165-176.
- DAIDONE, R., G. DINI, E M. TILOCA. “On experimentally evaluating the impact of security on IEEE 802.15.4 networks.” *In Proceedings: International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*. 2011. 1-6.
- DINH, NGUYEN QUOC, E DONG-SUNG KIM. “Performance evaluation of priority CSMA-CA mechanism on ISA100.11a wireless network.” *Computer Standards \& Interfaces* 34 (2012): 117-123.
- DINH, THANH, YOUNGHAN KIM, TAO GU, E ATHANASIOS V. VASILAKOS. “L-MAC: A wake-up time self-learning MAC protocol for wireless sensor networks.” *Computer Networks* 105 (2016): 33-46.
- DUJOVNE, D., T. WATTEYNE, X. VILAJOSANA, E P. THUBERT. “6TiSCH: deterministic IP-enabled industrial internet (of things).” *IEEE Communications Magazine* 52 (12 2014): 36-41.
- DUQUENNOY, SIMON, BESHAR AL NAHAS, OLAF LANDSIEDEL, E THOMAS WATTEYNE. “Orchestra: Robust Mesh Networks Through Autonomously Scheduled TSCH.” *Proceedings*

- of the 13th ACM Conference on Embedded Networked Sensor Systems. New York, NY, USA: ACM, 2015. 337-350.
- DUTTA, PRABAL, STEPHEN DAWSON-HAGGERTY, YIN CHEN, CHIEH-JAN MIKE LIANG, E ANDREAS TERZIS. "A-MAC: A Versatile and Efficient Receiver-initiated Link Layer for Low-power Wireless." *ACM Trans. Sen. Netw.* (ACM) 8 (9 2012): 30:1--30:29.
- ETSI_a. "Industry trends for M2M/IoT survey results from the ETSI M2M Workshop 2014.", 2014 , Disponível em:
<http://docbox.etsi.org/workshop/2014/201412_M2MWORKSHOP/INDUSTRY_TRENDSforM2M_IoT_SURVEY_RESULTS.pdf>, Acesso em: 2018-04-06.
- ETSI_b. "Machine-to-Machine Communications (M2M); M2M Service Requirements Release 1,ETSI Standard TS 102 689, V1.2.1.",2013, Disponível em:
<https://webapp.etsi.org/workprogram/Report_WorkItem.asp?WKI_ID=37695>, Acesso em: 2018-04-06.
- FAFOUTIS, XENOFON, ALESSIO DI MAURO, MADAVA D. VITHANAGE, E NICOLA DRAGONI. "Receiver-initiated medium access control protocols for wireless sensor networks." *Computer Networks* 76 (2015): 55-74.
- FAMBON, O., E. FLEURY, G. HARTER, R. P. GIBOLLET, e F. S. MARCEL. "FIT IOT-LAB tutorial: hands-on practice with a very large scale testbed tool for the Internet of Things." *10emes journées francophones Mobilité et Ubiquité*, 2014.
- FERNANDES, R.F., FONSECA, C.C., BRANDAO, D., FERRARI, P., FLAMMINI, A., VEZZOLI, A., "Flexible Wireless Sensor Network for smart lighting applications" In: 2014 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), 2014, Montevideo, IEEE, 2014, p434-439.
- GADDOUR, OLFA, E ANIS KOUBÂA. "RPL in a nutshell: A survey." *Computer Networks* 56 (2012): 3163-3178.
- GAZIS, V. "A Survey of Standards for Machine-to-Machine and the Internet of Things." *IEEE Communications Surveys Tutorials* 19 (2017): 482-511.
- GRANJAL, JORGE, EDMUNDO MONTEIRO, E JORGE S5n SILVA. "Security in the integration of low-power Wireless Sensor Networks with the Internet: A survey." *Ad Hoc Networks* 24 (2015): 264-287.
- HUANG, P., C. WANG, L. XIAO, E H. CHEN. "RC-MAC: A receiver-centric medium access control protocol for wireless sensor networks." In Proceedings: *IEEE 18th International Workshop on Quality of Service (IWQoS)*. 2010. 1-9.
- HUI, J. W., E D. E. CULLER. "IPv6 in Low-Power Wireless Networks." *Proceedings of the IEEE* 98 (11 2010): 1865-1878.

- IEEE802.15.4. "IEEE Standard for Low-Rate Wireless Networks." *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, 4 2016: 1-709.
- IEEE802.15.4e. "IEEE Standard for Local and metropolitan area networks--Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer." *IEEE Std 802.15.4e-2012 (Amendment to IEEE Std 802.15.4-2011)*, 4 2012: 1-225.
- TEXAS INSTRUMENTS. "CC2538EM Reference Design.", 2018, Disponível em: <<http://www.ti.com/tool/CC2538EM-RD>>, Acesso em: 2018-03-09.
- ITU-T. "Recommendation ITU-T Y.2060, Overview of the Internet of Things,ITU-T Standard Y.2060." 2012, Disponível em:<<https://www.itu.int/rec/T-REC-Y.2060-201206-I/en>>, Acesso em: 2018-04-09.
- J. HUI, P. THUBERT. "RFC 6282 : Compression Format for IPv6 Datagrams over IEEE 802.15.4-Based Networks.",2012, Disponível em: <<https://tools.ietf.org/html/rfc6282>>, Acesso em: 2017-11-09.
- JAIN, PUNEET. "Study on Machine-Type Communications (MTC) and Other Mobile Data Applications Communications Enhancements, 3rd Generation Partnership Project (3GPP) TR 23.887." 2013.
- KEOH, S. L., S. S. KUMAR, E H. TSCHOFENIG. "Securing the Internet of Things: A Standardization Perspective." *IEEE Internet of Things Journal* 1 (6 2014): 265-275.
- KHALIL, M. I., M. A. HOSSAIN, E I. AHMED. "DURI-MAC: A dual channel receiver initiated MAC protocol for wireless sensor network (WSN)." *In Proceedings: International Conference on Electrical, Computer and Communication Engineering (ECCE)*. 2017. 577-582.
- KIM, H. S., H. IM, M. S. LEE, J. PAEK, E S. BAHK. "A measurement study of TCP over RPL in low-power and lossy networks." *Journal of Communications and Networks* 17 (12 2015): 647-655.
- KIM, H. S., J. KO, D. E. CULLER, E J. PAEK. "Challenging the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL): A Survey." *IEEE Communications Surveys Tutorials* 19 (2017): 2502-2525.
- KIM, T., I. H. KIM, Y. SUN, E Z. JIN. "Physical Layer and Medium Access Control Design in Energy Efficient Sensor Networks: An Overview." *IEEE Transactions on Industrial Informatics* 11 (2 2015): 2-15.
- LAMPIN, Q., D. BARTHEL, I. AUGÉ-BLUM, E F. VALOIS. "SARI-MAC: The Self Adapting Receiver Initiated MAC protocol for Wireless Sensor Networks." *In Proceedings: IEEE 8th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. 2012. 12-18.
- LEE, H., J. HONG, S. YANG, I. JANG, E H. YOON. "A pseudo-random asynchronous duty cycle MAC protocol in wireless sensor networks." *IEEE Communications Letters* 14 (2 2010): 136-138.

- LI, J., D. ZHANG, L. GUO, S. JI, E Y. LI. "ARM: An asynchronous receiver-initiated multichannel MAC protocol with duty cycling for WSNs." In *Proceedings: International Performance Computing and Communications Conference*. 2010. 114-121.
- LI, SHANCANG, LI DA XU, E SHANSHAN ZHAO. "The internet of things: a survey." *Information Systems Frontiers* 17 (4 2015): 243-259.
- LIAO, WEN-HWA, SSU-CHI KUAI, E YU-CHIEH LIN. "A Receiver-Initiated MAC Protocol with Packet Train Design for Underwater Acoustic Sensor Networks." *Wireless Personal Communications* 82 (6 2015): 2155-2170.
- LIN, E. Y. A., J. M. RABAEY, E A. WOLISZ. "Power-efficient rendez-vous schemes for dense wireless sensor networks." In *Proceedings: IEEE International Conference on Communications (IEEE Cat. No.04CH37577)*. 2004. 3769-3776 Vol.7.
- LORA. "LORAWAN.", 2018, Disponível em: <<https://loro-alliance.org/about-lorawan>>, Acesso em: 2018-04-09>
- MINH, N. N., E M. K. KIM. "Reducing idle listening time in pipeline-forwarding MAC protocols of wireless sensor networks." In *Proceedings: International Conference on Advanced Technologies for Communications (ATC)*. 2016. 186-190.
- MOHREHKESH, S., M. C. WEIGLE, E S. K. DAS. "DRIH-MAC: A Distributed Receiver-Initiated Harvesting-Aware MAC for Nanonetworks." *IEEE Transactions on Molecular, Biological and Multi-Scale Communications* 1 (3 2015): 97-110.
- MORELL, A., X. VILAJOSANA, J. L. VICARIO, E T. WATTEYNE. "Label switching over IEEE802.15.4e networks." *Transactions on Emerging Telecommunications Technologies* 24 (2013): 458-475.
- NAMBOOTHIRI, P.G., SIVALINGAM, K.M. (2013) Throughput analysis of multiple channel based wireless sensor network, *Wireless Network*, Springer, LLC, volume 19, Issue 4, pp 461-476, DOI 10.1007/s11276-012-0478-4, May 2013.
- NGUYEN, KIEN, VU-HOANG NGUYEN, DUY-DINH LE, YUSHENG JI, DUC ANH DUONG, E SHIGEKI YAMADA. "A Receiver-Initiated MAC Protocol for Energy Harvesting Sensor Networks." Em *Ubiquitous Information Technologies and Applications: CUTE 2013*, 603-610. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.
- OJO, M., D. ADAMI, E S. GIORDANO. "Performance evaluation of energy saving MAC protocols in WSN operating systems." In *Proceedings: International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)*. 2016. 1-7.
- OLSSON, JONAS. "6LoWPAN demystified.", 2014, Disponível em: <<https://www.ti.com/lit/wp/swry013/swry013.pdf>>, Acesso em: 2017-11-09.
- oneM2M. "oneM2M Functional Architecture Baseline Draft.", 2014, Disponível em: <<http://www.onem2m.org/images/files/deliverables/TS-0001-oneM2M-Functional-Architecture-V-2014-08.pdf>>, Acesso em: 2018-04-09.

- OPEN MOBILE ALLIANCE. "OMA Device Management V1.2, Open Mobile Alliance Stand., San Diego, CA, USA, 2013." 2013.
- OPEN GEOSPATIAL CONSORTIUM. "OGC Sensor Web Enablement: Overview And High Level Architecture.",2011, Disponível em: <https://portal.opengeospatial.org/files/?artifact_id=48492>, Acesso em: 2018-04-09.
- OpenWSN. "OpenWSN Protocol Stack.", 2018, Disponível em: < <https://openwsn.atlassian.net>>, Acesso em: 2018-04-06.
- PALATTELLA, MARIA RITA; ACCETTURA , NICOLA ; VILAJOSANA , XAVIER ; WATTEYNE , THOMAS AND GRIECO , LUIGI ALFREDO ; BOGGIA , GENNARO ; DOHLER , MISCHA. "Standardized Protocol Stack For The Internet Of (Important) Things." *IEEE Communications Surveys and Tutorials*, 12 2012.
- PATEL, AMOL, E RAKSHA UPADHYAY. "Performance analysis of Slotted CSMA/CA MAC protocol under different parameters for static IEEE 802.15.4 Wireless Sensor Networks." 2013.
- PENG, Y., Z. LI, D. QIAO, E W. ZHANG. "Delay-bounded MAC with minimal idle listening for sensor networks." In Proceedings: *IEEE INFOCOM*. 2011. 1314-1322.
- PÉREZ, M. RODRÍGUEZ, S. HERRERÍA ALONSO, M. FERNÁNDEZ VEIGA, E C. LÓPEZ GARCÍA. "A Self-Tuning Receiver-Initiated MAC Protocol for Wireless Sensor Networks." *IEEE Wireless Communications Letters* 4 (12 2015): 601-604.
- PETROLO, RICCARDO, VALERIA LOSCRÌ, E NATHALIE MITTON. "Towards a smart city based on cloud of things, a survey on the smart city vision and paradigms." *Transactions on Emerging Telecommunications Technologies* 28 (2017): e2931--n/a.
- PU, LINA, YU LUO, ZHENG PENG, HAINING MO, E JUN-HONG CUI. "Traffic Estimation Based Receiver Initiated MAC for Underwater Acoustic Networks." *Proceedings of the Eighth ACM International Conference on Underwater Networks and Systems*. New York, NY, USA: ACM, 2013. 7:1--7:5.
- RAJANDEKAR, A., E B. SIKDAR. "A Survey of MAC Layer Issues and Protocols for Machine-to-Machine Communications." *IEEE Internet of Things Journal* 2 (4 2015): 175-186.
- ROUSSELOT, J., A. EL-HOIYDI, E J. D. DECOTIGNIE. "WideMac: a low power and routing friendly MAC protocol for Ultra Wideband sensor networks.", In Proceedings: *IEEE International Conference on Ultra-Wideband*. 2008. 105-108.
- SCIANCELEPORE, SAVIO, MALIŠA VUČINIĆ, GIUSEPPE PIRO, GENNARO BOGGIA, E THOMAS WATTEYNE. "Link-layer security in TSCH networks: effect on slot duration." *Transactions on Emerging Telecommunications Technologies* 28 (2016): e3089.
- SHELBY, Z. "RFC 6690 : Constrained RESTful Environments (CoRE) Link Format." 8 de 2012. Disponível em = <<https://tools.ietf.org/html/rfc6690>>, Acesso em: 2017-11-09.

- SHENG, Z., S. YANG, Y. YU, A. V. VASILAKOS, J. A. MCCANN, E K. K. LEUNG. "A survey on the ietf protocol suite for the internet of things: standards, challenges, and opportunities." *IEEE Wireless Communications* 20 (12 2013): 91-98.
- SIGFOX. "Sigfox technology overview." 2018. Disponível em = <<https://www.sigfox.com/en/sigfox-iiot-technology-overview>>, Acesso em = 2018-04-06.
- SMART, G., N. DELIGIANNIS, R. SURACE, V. LOSCRI, G. FORTINO, E Y. ANDREOPOULOS. "Decentralized Time-Synchronized Channel Swapping for Ad Hoc Wireless Networks." *IEEE Transactions on Vehicular Technology* 65 (10 2016): 8538-8553.
- SUN, YANJUN, OMER GUREWITZ, E DAVID B. JOHNSON. "RI-MAC: A Receiver-initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Loads in Wireless Sensor Networks." In Proceedings: *6th ACM Conference on Embedded Network Sensor Systems*. New York, NY, USA: ACM, 2008. 1-14.
- SUN, YANJUN, OMER GUREWITZ, SHU DU, LEI TANG, E DAVID B. JOHNSON. "ADB: An Efficient Multihop Broadcast Protocol Based on Asynchronous Duty-cycling in Wireless Sensor Networks." In Proceedings: *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. New York, NY, USA: ACM, 2009. 43-56.
- TANG, L., Y. SUN, O. GUREWITZ, E D. B. JOHNSON. "PW-MAC: An energy-efficient predictive-wakeup MAC protocol for wireless sensor networks." In Proceedings: *IEEE INFOCOM*. 2011. 1305-1313.
- TANG, LEI, YANJUN SUN, OMER GUREWITZ, E DAVID B. JOHNSON. "EM-MAC: A Dynamic Multichannel Energy-efficient MAC Protocol for Wireless Sensor Networks." *Proceedings of the Twelfth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. New York, NY, USA: ACM, 2011. 23:1--23:11.
- TOKOIGNON, C. ARCADIUS, B. GAO, G. Y. TIAN, E Y. YAN. "Structural Health Monitoring Framework Based on Internet of Things: A Survey." *IEEE Internet of Things Journal* 4 (6 2017): 619-635.
- VILAJOSANA, X., Q. WANG, F. CHRAIM, T. WATTEYNE, T. CHANG, E K. S. J. PISTER. "A Realistic Energy Consumption Model for TSCH Networks." *IEEE Sensors Journal* 14 (2 2014): 482-489.
- VUCINIC, MALISA, BERNARD TOURANCHEAU, FRANCK ROUSSEAU, ANDRZEJ DUDA, E LAURENT DAMON. "OSCAR: Object Security Architecture for the Internet of Things." In Proceedings: *IEEE 15th International Symposium on Mobile and Multimedia Networks (WoWMoM)*. 2014.
- WANG, X., X. ZHANG, G. CHEN, E Q. ZHANG. "Opportunistic Cooperation in Low Duty Cycle Wireless Sensor Networks." In Proceedings: *IEEE International Conference on Communications*. 2010. 1-5.

- WATTEYNE, T.; VILAJOSANA, X. ; KERKEZ, B. ; CHRAIM, F. ; WEEKLY, K. ; WANG, Q. ; GLASER, S. ; PISTER, K. . “OpenWSN: a standards-based low - power wireless development environment.” *Transactions on Emerging Telecommunications Technologies* 23 (s.d.): 480-493.
- WINTER, T., P. THUBERT, A. BRANDT, J. HUI, R. KELSEY, P. LEVIS, K. PISTER, R. STRUIK, JP. VASSEUR, R. ALEXANDER “RFC 6550 : RPL - IPv6 Routing Protocol for Low-Power and Lossy Networks.” 3 de 2012. Disponível em: <<https://tools.ietf.org/html/rfc6550>>. Acesso em: 2017-11-09.
- WON, M., T. PARK, E S. H. SON. “Asym-MAC: A MAC Protocol for Low-Power Duty-Cycled Wireless Sensor Networks with Asymmetric Links.” *IEEE Communications Letters* 18 (5 2014): 809-812.
- YADAV, P., E J. A. MCCANN. “YA-MAC: Handling unified unicast and broadcast traffic in Multi-hop Wireless Sensor Networks.” *In Proceedings: International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*. 2011. 1-9.
- YANG, DONGYU, YING QIU, SHINING LI, E ZHIGANG LI. “RW-MAC: an asynchronous receiver-initiated ultra low power MAC protocol for wireless sensor networks.” *IET Conference Proceedings* (Institution of Engineering and Technology), s.d.: 393-398(5).
- YANG, O., E W. HEINZELMAN. “Modeling and Performance Analysis for Duty-Cycled MAC Protocols with Applications to S-MAC and X-MAC.” *IEEE Transactions on Mobile Computing* 11 (6 2012): 905-921.
- YAQOOB, I., AHMED, E. ; HASHEM, I. A. T.; AHMED, A. I. A. ; GANI, A. ; IMRAN, M. ; GUIZANI, M. “Internet of Things Architecture: Recent Advances, Taxonomy, Requirements, and Open Challenges.” *IEEE Wireless Communications* 24 (6 2017): 10-16.
- YONG, YUEH-TIAM, CHEE-ONN CHOW, JEEVAN KANESAN, E HIROSHI ISHII. “EE-RI-MAC: An energy-efficient receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks.” *International Journal of Physical Sciences* (Academic Journals) 6 (2011): 2633-2643.
- ZANELLA, A., N. BUI, A. CASTELLANI, L. VANGELISTA, E M. ZORZI. “Internet of Things for Smart Cities.” *IEEE Internet of Things Journal* 1 (2 2014): 22-32.
- ZHUO, S., Z. WANG, Y. Q. SONG, Z. WANG, E L. ALMEIDA. “A Traffic Adaptive Multi-Channel MAC Protocol with Dynamic Slot Allocation for WSNs.” *IEEE Transactions on Mobile Computing* 15 (7 2016): 1600-1613.