UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
COORDENAÇÃO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

AVALIAÇÃO DA CONFIABILIDADE DE SIMULADOR EM REDES DE SENSORES SEM FIO COM BASE EM PLATAFORMA REAL DE SENSORIAMENTO

GUSTAVO NÓBREGA MARTINS

CAMPINA GRANDE
2016

Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

Avaliação da Confiabilidade de Simulador em Redes de Sensores Sem Fio com Base em Plataforma Real de Sensoriamento

Gustavo Nóbrega Martins

Campina Grande, Paraíba, Brasil
Agosto - 2016

Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Coordenação de Pós-Graduação em Ciência da Computação

Avaliação da Confiabilidade de Simulador em Redes de Sensores Sem Fio com Base em Plataforma Real de Sensoriamento

Gustavo Nóbrega Martins

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal de Campina Grande - Campus I como parte dos requisitos necessários para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Ciência da Computação
Linha de Pesquisa: Redes de Computadores

Prof. Reinaldo Cézar de Morais Gomes, D.Sc.

Campina Grande, Paraíba, Brasil

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL DA UFCG

M386a Martins, Gustavo Nóbrega.

Avaliação da confiabilidade simulador em redes de sensores sem fio com base em plataforma real de sensoriamento / Gustavo Nóbrega Martins. – Campina Grande, 2016.

83 f. : il. color.

Dissertação (Mestrado em Ciências da Computação) – Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática, 2016.

"Orientação: Prof. Dr. Reinaldo Cézar de Morais Gomes".

Referências.

1. Redes de Computadores. 2. Redes de Sensores sem Fio. 3. Avaliação de Confiabilidade – Redes de Sensores sem Fio. I. Gomes, Reinaldo Cézar de Morais. II. Universidade Federal de Campina Grande, Campina Grande (PB). III. Título.

CDU 004.7(043)

'ALIANDO CONFIABILIDADE DE SIMULADOR EM REDES DE SENSORES SEM FIO COM BASE EM PLATAFORMA REAL DE SENSORIAMENTO"

GUSTAVO NÓBREGA MARTINS

DISSERTAÇÃO APROVADA EM 26/08/2016

REINALDO CÉZAR DE MORAIS GOMES, Dr., UFCG
Orientador(a)

JOSEANA MACÊDO FECHINE RÉGIS DE ARAÚJO, D.Sc, UFCG
Examinador(a)

IGUATEMI EDUARDO DA FONSECA, Dr., UFPB
Examinador(a)

MARCELO PORTELA SOUSA, Dr., IFPB
Examinador(a)

CAMPINA GRANDE - PB

Resumo

As Redes de Sensores Sem Fio (RSSF) têm ganhado notoriedade em vários contextos de aplicações (*e.g.* aplicações militares, médicas e ambientais) em razão dos avanços obtidos no âmbito das tecnologias de sistemas eletromecânicos em escala micrométrica (*Microscale Electro-Mechanical Systems* - MEMS) e nanométrica (*Nanoscale Electro-Mechanical Systems* - NEMS). Esses progressos implicaram no desenvolvimento de nós sensores com menor custo, tamanho reduzido e com maior eficiência no consumo de energia.

Mesmo diante dos avanços tecnológicos obtidos, os sensores sem fio ainda apresentam algumas restrições de processamento e memória, além do fornecimento de energia limitado devido ao uso de baterias como fonte de energia. Parte das pesquisas realizadas no âmbito das RSSF tem utilizado ferramentas de simulação para validar os achados de pesquisa devido ao menor custo e tempo exigidos por essa abordagem. Entretanto, algumas investigações apontaram que ferramentas de simulação em RSSF não contemplam fatores importantes que podem influenciar o desempenho das redes, como também questiona a confiabilidade dos resultados dessas ferramentas.

Este trabalho de dissertação visou avaliar a confiabilidade do simulador de RSSF, Castalia, com base em *testbed* no tocante a dois protocolos de comunicação: LEACH e Protocolo de Comunicação Direta. Resultados apontaram que o Castalia, referente ao protocolo LEACH, superestimou o tempo de vida da rede com carga de tráfego estatisticamente equivalente. Em relação ao protocolo de Comunicação Direta, não houve equivalência de dados trafegados, bem como o tempo de vida também foi superestimado, além de não apresentar diferença nos resultados do tempo de vida entre os nós da rede, indicando um comportamento de descarga determinístico.

Os achados deste trabalho demonstram que ferramentas de simulação trouxe significativos avanços no âmbito das RSSF, entretanto as saídas obtidas por meio dessas ferramentas ainda não podem representar com exatidão alguns cenários modelados em RSSF.

Palavras-Chave: Redes de Sensores Sem Fio, Avaliação, Ferramentas de Simulação, Castalia, LEACH, Confiabilidade

Abstract

The Sensors Wireless Networks (WSN) have gained notoriety in various application contexts (*e.g.* military applications, medical and environmental) due to the advances achieved in Microscale Electro Mechanical Systems – MEMS and Nanoscale Electro Mechanical Systems – NEMS. These growth resulted in sensors development with lower cost, reduced size and higher efficiency in energy consumption. Even with these technological advances, wireless sensors still have some constrains on processing and memory, in addition, the limited power source from batteries also is a challenge.

Many research in WSN field have adopted simulation tools to validate their findings due to lower cost and time required for this approach. However, some research highlights that simulation tools in WSN do not comprise important factors that can influence the network performance, as well as the reliability of the results from these tools are questioned.

This research aimed to evaluate the reliability of WSN simulator, Castalia, based on testbed concerning two communication protocols: LEACH and Direct Communication. Results showed that LEACH protocol running over Castalia overestimates the network lifetime with traffic load statistically similar. Regarding Direct communication protocol, there was not similarity in data traffic load as well as the lifetime was also overestimated. Moreover, all nodes were depleted in the same time on simulation scenarios, indicating a deterministic discharge behavior. From testbed, the sensors' batteries had different lifetime as expected.

The research findings demonstrate that simulation tools brought significant advances in the WSN, however the outputs obtained through these tools still not able to accurately represent some modeled scenarios in WSN.

Keywords: Wireless Sensors Network, Evaluation, Simulation Tools, Castalia, LEACH, Reliability

Agradecimentos

- Ao criador, por todas as minhas conquistas alcançadas.
- Aos meus pais (Severino e Lúcia), por sempre terem me apoiado, com o melhor de si;
- À minha família querida, em especial à Elisa Nóbrega (*In Memorian*), por acreditar em meus projetos, dar-me o incentivo para seguir em frente e compreender os momentos de ausência em detrimento desta jornada.
- Aos meus amigos Magno Jefferson, Paola Francinete e Lívia Brito, por sempre terem me incentivado a trilhar este caminho da pesquisa e docência.
- Aos professores, mestres e mentores, Prof. Reinaldo Cézar de Morais Gomes, Prof. Anderson Fabiano Batista Ferreira da Costa e Prof. Marcelo Sampaio de Alencar, por todo apoio e ensinamento oferecido. Sou grato pela amizade conquistada.
- À família LATEC, em especial ao Petrônio, Fellype, Lucas, César e Ana Cristina, pelo companheirismo do dia-a-dia, o que tornou o ambiente de trabalho em um espaço de aprendizado e colaboração.
- Aos colegas do IECOM e/ou DEE, em especial ao Prof. Marcelo Portela, Prof. Ruan Delgado e Prof. Gutemberg Junior, pelos *insights* prestados que contribuíram para a realização deste trabalho.
- Ao meu amigo da graduação para vida, Adson Diego, que sempre esteve solícito para contribuir com este trabalho por meio de críticas e sugestões.
- Aos colegas e amigos da pós-graduação, pelas discussões geradoras de aprendizado e conhecimento em todos os aspectos: Adriano Santos, Danilo Abreu, Zé Gildo, Caio Nóbrega, Nailson Boaz e Yuri Lacerda.
- A Marcos Procópio e sua equipe, por ter subsidiado a instrumentação necessária para realização deste trabalho, como também aos meus amigos Brayner, Israel e Rodrigo que integram a referida equipe.

- À CAPES pelo suporte na realização desta pesquisa.

Conteúdo

Glossário

C4ISRT Command, Control, Communications, Computers, Intelligence, Surveillance, Reconnaissance, and Targeting. 12

CDMA Code Division Multiple Access. 30

CDMA Carrier Sense Multiple Access. 29

CDMA Time Division Multiple Access. 29

CIA Construindo Cidades Inteligente da Instrumentação dos Ambientes ao Desenvolvimento de Aplicações. 8

DSSS Direct Sequence Spread Spectrum. 18

EBI External Bus Interface. 17

FND First Node Die. 6

GPRS General Packet Radio Service. 7

GPS Global Positioning System. 9

GQM Goal-Question-Metric. 37

IEEE Institute of Electrical and Electronics Engineers. 7

IoT Internet of Things. 7

ISM Industrial, Scientific and Medical. 18

LEACH Low Energy-Aware Clustering Protocol. 3–6

LND Last Node Die. 6

MANET Mobile Ad-hoc Network. 10

MEMS Microscale Electro-Mechanical Systems. 1

MTE Minimum Transmission Energy. 32

NEMS Nanoscale Electro-Mechanical Systems. 1

NTP Network Time Protocol. 61

RAM Random Access Memory. 40

RF Radio Frequence. 19

RSSF Redes de Sensores Sem Fio. 1, 3, 6, 10

RSSI Received Signal Strengh Indicator. 29

SDRAM Synchronous Dynamic Random Access Memory. 18

SNR Signal Noise Ratio. 40

SoC System on a Chip. 17

SunSPOT Sun Small Programmable Object Technology. 16

WBAN Wireless Body Area Network. 14

Lista de Figuras

Lista de Tabelas

Lista de Códigos Fonte

Capítulo 1

Introdução

As Redes de Sensores Sem Fio (RSSF) têm ganhado notoriedade em vários contextos de aplicação (*e.g.* aplicações militares, médicas e ambientais) em razão dos avanços obtidos no âmbito das tecnologias de sistemas eletromecânicos em escala micrométrica (*Microscale Electro-Mechanical Systems* – MEMS) e nanométrica (*Nanoscale Electro-Mechanical Systems* – NEMS) [18, 25]. Esses progressos implicaram no desenvolvimento de nós sensores com menor custo, maior capacidade computacional e de sensoriamento e maior eficiência no consumo de energia. Uma RSSF, normalmente, é composta de vários nós sensores, distribuídos sobre uma área geográfica, com finalidade de monitorar grandezas e fenômenos físicos, tais como: temperatura, umidade, luminosidade, atividades sísmica, etc. [46]. Geralmente, esses sensores operam de forma autônoma, com restrições de memória e processamento, e um dos nós que compõe a RSSF atua como estação base (*Basestation*), ou nó sorvedouro, que é responsável por receber todos os dados sensoreados pela rede e realizar algum tratamento/processamento para disponibiliza-los por meio de alguma aplicação, seja localmente ou via internet [22].

No contexto das RSSF, alguns desafios são bem conhecidos, como por exemplo o desenvolvimento de padrões de comunicação, tamanho da rede, faixa de frequência de operação aberta, interoperabilidade entre equipamentos de diferentes fabricantes, etc [33]. Adicionalmente, uma outra forte restrição em RSSF está relacionada ao fornecimento de energia limitado oriundo das baterias, pois em alguns tipos de aplicação os nós da RSSF são depositados aleatoriamente no ambiente e em locais de difícil acesso (e.g. aplicações militares e ambientais), não sendo possível assim realizar a substituição da bateria quando necessá-

rio. Dessa forma, o tempo de vida da rede se torna um requisito crucial para o sucesso do funcionamento da aplicação.

1.1 Problema

Para minimizar os impactos da restrição de consumo de energia existente na maior parte das aplicações utilizadas em RSSF, diversas iniciativas surgiram visando elaborar estratégias para minimizar o consumo de energia dos sensores, alcançando assim o prolongamento do tempo de vida da rede. Como exemplo dessas iniciativas, uma quantidade considerável de protocolos de roteamento específicos para RSSF têm como objetivo maximizar o tempo de vida da rede por meio da redução do consumo de energia [19]. Esses protocolos são denominados de *Energy-Aware Routing Protocols*.

Outra abordagem que visa resolver o mesmo problema baseia-se em esquemas de melhoria da organização da comunicação entre os nós presentes na rede. Uma das estratégias bem difundidas com essa finalidade é a de agrupamento de nós, que compreende dividir a rede em grupos para equilibrar o consumo de energia da rede entre os sensores. Dessa forma, os nós que compõem a rede podem atuar em diferentes tarefas e tipicamente são organizados dentro de grupos, de acordo com requisitos específicos. É possível encontrar na literatura algumas pesquisas que utilizaram algoritmos tradicionais para o processo de formação de grupos em RSSF, entretanto o problema de agrupamento é considerado NP-Difícil, ou seja, nenhum algoritmo determinístico é capaz de encontrar uma solução em tempo polinominal [11,22].

Um exemplo de utilização de algoritmo tradicional de agrupamento, aplicado à formação de grupos em RSSF, é o *k-means* [49], que utiliza centróides[1] como líderes de grupo. Para realizar a escolha do centróide, o algoritmo busca a minimização da soma dos quadrados das distâncias (distância euclidiana) entre os nós e o centróide. Essa abordagem que utiliza algoritmos tradicionais para formação de grupos se torna pouco eficiente devido ao alto custo de processamento[2] e às restrições impostas pelos sensores [21].

Outra abordagem no âmbito dos protocolos de agrupamento, com estrutura da rede si-

[1]Elemento que atua como coordenador do grupo.

[2]RSSF com quantidade de sensores razoável.

milar à supracitada, entende que cada grupo compreende um líder chamado de coordenador (*Cluster Head*), e uma parcela de nós subordinados ao coordenador que são chamados de outros nós membros (*Other Member Nodes*) ou nós ordinários (*Ordinary Node*) [38]. Os dados sensoreados pelos nós ordinários são enviados para o coordenador, para serem processados e enviados ao nó sorvedouro. Essa comunicação pode ocorrer de forma direta, ou por múltiplos saltos *(multi-hop)*, entre os coordenadores e/ou nós membros, até alcançar o destino. A Figura 1.1 ilustra a organização da rede em grupos e como se dá a comunicação entre os participantes. Pode-se observar que a comunicação entre os nós ordinários se dá apenas com o seu respectivo nó coordenador, e apenas os nós coordenadores podem se comunicar com a estação base. Ademais, no exemplo da Figura 1.1, os nós ordinários e coordenadores se comunicam apenas com um único salto, ou seja, não há participação de nó intermediário entre nó ordinário e coordenador ou coordenador e nó sorvedouro.

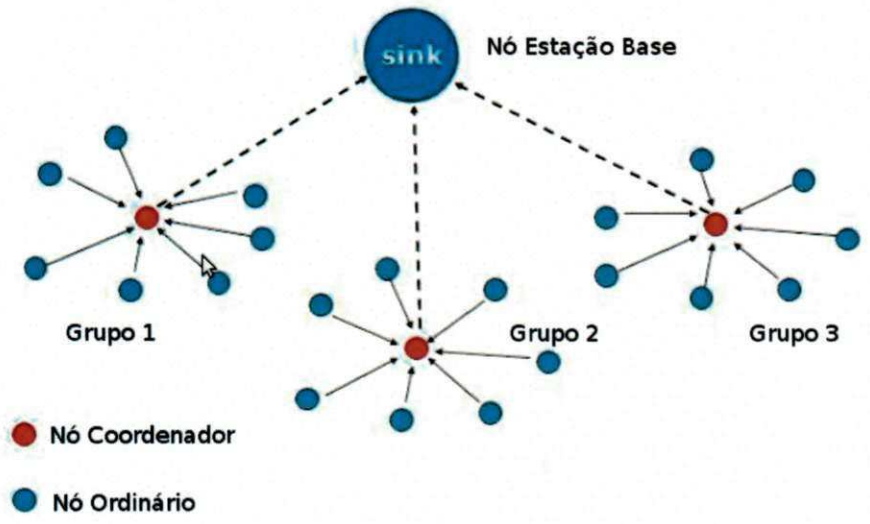

Figura 1.1: Esquema de organização de uma RSSF com LEACH [40].

Uma das iniciativas mais conhecidas no tocante à organização de RSSF em grupos foi o desenvolvimento do protocolo *Low-Energy Adaptative Clustering Hierarchy*. O LEACH é um protocolo de roteamento e agrupamento que realiza a escolha dos coordenadores de grupo por meio de um esquema de rodízio entre os nós da rede. Esse esquema adota uma abordagem distribuída em que cada nó é capaz de se autoeleger como coordenador, com base em um cálculo de probabilidade. Uma vez eleito coordenador, o mesmo não poderá ser reeleito até que todos os nós já tenham se tornado coordenadores. Assim, o rodízio

propicia um balanceamento no consumo da energia da rede. A partir do advento do LEACH, outras variações do protocolo surgiram na literatura a fim de buscar resolver algumas das suas limitações, como também otimizar o seu processo de eleição dos coordenadores. Por meio de uma revisão sistemática realizada em [54], e discutida na Seção 3 deste trabalho, é possível obter uma visão geral sobre outras abordagens oriundas do LEACH e suas proposituras. As versões do protocolo apresentadas no trabalho supracitado foram classificadas conforme seus atributos, com base em uma taxonomia. Um dos aspectos relevantes a ser ressaltado a partir dessa revisão de literatura é que todas as versões do LEACH citadas no trabalho foram validadas por meio do uso de simuladores, prática comum em pesquisas no âmbito das RSSF e tema de pesquisa desta dissertação.

Com o surgimento de protocolos com proposituras acerca da eficiência energética, frequentemente validados por meio de simuladores, algumas investigações começaram a elencar questões relativas aos métodos de validação. Como apontado por Grobler et al., uma quantidade considerável de pesquisas desenvolvidas em RSSF, relacionadas à avaliação de eficiência energética de protocolos, são realizadas por meio de simulação, por motivos como custo e tempo. Todavia, haja vista que a simulação não é capaz de englobar todos os fatores (e.g. interferência eletromagnética e os diversos ruídos presentes no ambiente) que têm influência sob os experimentos, a acurácia dessa abordagem em muitos cenários não é 100% [27]. Pham et al. em [45], indagam que pesquisadores têm se sustentado em simuladores para validar suas ideias e métodos devido à complexidade envolvida para implantar sistemas reais (*i.e.* implantar centenas ou milhares de sensores em um espaço físico, programá-los, excitá-los e monitorar o comportamento/estado dos sensores). Outros trabalhos, tais como [16,20,34,42], apontaram a necessidade de avaliar a acurácia de simuladores em RSSF, bem como ressaltaram a importância dessa temática, viabilizando o surgimento de questionamentos que possam despontar em investigação sobre o tema.

Logo, a partir desse panorama, fica notório a importância de iniciativas que visem avaliar protocolos de RSSF, concernentes à eficiência energética, em relação à acurácia das ferramentas de simulação em RSSF com base em redes de sensores reais.

1.2 Avaliação

A avaliação realizada neste trabalho buscou analisar a acurácia entre ferramenta de simulação em RSSF (comumente adotada em trabalhos científicos) e uma plataforma real de sensoreamento. Na revisão de literatura realizada, foi possível encontrar alguns trabalhos, tais como [20, 34, 44, 50], que buscaram verificar a acurácia de simuladores em RSSF ou investigar o comportamento de protocolos de roteamento implementados em sensores reais.

Em [50], o protocolo LEACH foi implementado na plataforma SunSPOT e algumas modificações foram propostas a fim de otimizar o funcionamento do protocolo sob os sensores. A investigação realizada por Perez et al. [44] efetuou uma comparação entre os resultados obtidos a partir de um experimento realizado com uma pequena RSSF e o seu respectivo modelo implementado na ferramenta de simulação OPNET, com o intuito de analisar a acurácia do modelo. Numa perspectiva de analisar a acurácia entre ferramentas de simulação, Khan et al.[34] investigaram algumas das ferramentas de simulação mais adotadas em trabalhos científicos, segundo levantamento bibliográfico apresentado em seu trabalho. Em [20], Colesanti et al. buscaram avaliar a acurácia da ferramenta de simulação OMNET++ em relação a um *testbed* composto por alguns nós TMote Sky executando um algoritmo de *flooding*.

Como pode ser visto, entre os trabalhos supracitados, uma investigação [50] englobou a implementação do protocolo de agrupamento LEACH na plataforma SunSPOT. Ou seja, uma das contribuições dessa dissertação foi publicado por Sheta et al.. Contudo, a implementação do protocolo não foi disponibilizada à comunidade para possíveis replicações ou continuidade da pesquisa. É importante ressaltar que nenhum desses trabalhos contempla a análise da acurácia entre a ferramenta de simulação em RSSF e uma plataforma real de sensoriamento, com base nos protocolos de comunicação direta e de agrupamento - LEACH. Neste trabalho de dissertação foi realizado uma avaliação a fim de analisar a acurácia do simulador OMNET++/Castalia tomando como base a plataforma experimental de sensoriamento, Oracle SunSPOT. Para que essa comparação fosse exequível, o simulador foi ajustado para compreender as especificações técnicas dos sensores SunSPOT no tocante a vários fatores (*e.g.* especificação da bateria, interface de rádio, modelo do canal sem fio, tráfego de dados, etc.).

Dessa forma, com ajustes atinente ao ambiente de simulação, adequando-o às especifici-

dades do sensor, avaliou-se a acurácia da ferramenta de simulação em relação a um *Testbed* composto de vinte nós e uma estação base, com os protocolos anteriormente citados implantados nos sensores. Com base em resultados preliminares, um ajuste fino foi efetuado com a finalidade de buscar minimizar divergência entre os resultados oriundo de especificação errônea do sensor no simulador. A comparação foi realizada com base nas seguintes métricas: tempo de vida da rede e tráfego de dados. Contudo, o primeiro nó morto (FND) e último nó morto (LND) também foram utilizados como métricas em uma análise complementar.

1.3 Resumo de Contribuições

As principais contribuições desta dissertação são detalhadas a seguir.

- Implementação de protocolos de agrupamento em plataforma real.
- Elucidação da acurácia da ferramenta de simulação em relação aos sensores reais.
- Identificação dos parâmetros ajustados para obtenção de uma melhor acurácia da ferramenta de simulação e a respectiva pertinência dos ajustes.
- Análise do comportamento da plataforma experimental SunSPOT enquanto às expectativas na implementação do protocolo.
- Análise da aplicabilidade do protocolo em um contexto de experimentação real.

1.4 Organização

Este documento está organizado da seguinte forma: a Seção 2 trata das RSSF. O protocolo LEACH é explicado em detalhes na Seção 3. Na seção 4 discutidos os simuladores e apresentados alguns comumente adotados em investigações sobre RSSF. Conseguinte, na Seção 5, a metodologia do trabalho e a definição do problema são apresentadas. O experimento é abordado na Seção 6. Conclusões deste trabalho discorrem na seção 7. Por fim, na Seção 8, a pretensão de trabalho futuro é discutida.

Capítulo 2

Rede de Sensores Sem Fio

2.1 Introdução

As RSSF têm ganhado notoriedade em diversos tipos de aplicações em razão dos avanços obtidos no âmbito das tecnologias de sistemas eletromecânicos em escala micrométrica (*Microscale Electro-Mechanical Systems – MEMS*) e nanométrica (*Nanoscale Electro-Mechanical Systems – NEMS*) [18, 25, 60]. Além de ser comumente utilizada em aplicações militares, ambientais e industriais para fins de monitoramento e rastreamento, as RSSF vêm sendo adotadas como um importante pilar em novos conceitos emergentes, tais como Cidades Inteligentes e Internet das Coisas (IoT).

Uma das iniciativas de destaque da aplicabilidade das RSSF no âmbito das Cidades Inteligentes é o projeto *Smart Santander*. O projeto tem como proposta oferecer um centro de pesquisa experimental singular no mundo, para desenvolvimento de aplicações e serviço voltado a Cidades Inteligentes, em escala de cidade. Essa proposta prevê a implantação de 20.000 nós sensores em Belgrado, Guildford, Lübeck e Santander, explorando uma grande variedade de tecnologias. Atualmente, o *Testbed* do *Smart Santander* integra uma estrutura com aproximadamente 3000 nós sensores IEEE 802.15.4, 200 módulos GPRS e 2000 etiquetas de identificação por rádio frequência implantados em pontos estáticos (*e.g.* semáforos, ponto de ônibus), bem como em alguns pontos móveis, exemplo dos nós sensores embarcados em veículos (*e.g.* ônibus e táxis) [7].

No Brasil, o projeto Construindo Cidades Inteligente da Instrumentação dos Ambientes ao Desenvolvimento de Aplicações (CIA^2), similar ao *Smart Santander*, também vem ga-

nhando destaque. O CIA tem como proposta a construção de uma infraestrutura que suporte computação e comunicação para viabilização de Cidades Inteligentes, abrangendo desde a infraestrutura (*e.g.* aquisição dos dados urbanos brutos por meio de tecnologias de redes de sensores e IoT, as tecnologias de comunicação e informação, o armazenamento, e o acesso a esses dados por meio de diferentes tecnologias e protocolos de RSSF) até o desenvolvimento de aplicações que tomem proveito de todo o "arcabouço", suportando uma melhor gestão pública e provendo uma melhor qualidade de vida aos cidadãos [2]. Para tanto, as RSSF desempenham um papel fundamental nesse cenário, como pode ser visto na Figura 2.1, que ilustra conceitualmente as principais vertentes que alicerçam o projeto. Em outras palavras, a vertente "Comunicação entre Redes Sem Fio"assume um elo importante entre as solução/aplicações propostas e a captação dos diversos tipos de informações necessárias para subsidiar os serviços de Cidades Inteligentes.

Figura 2.1: Arquitetura conceitual do projeto CIA^2 adaptado de [2].

Os diversos cenários de aplicação das RSSF trazem consigo vários desafios conhecidos pelo seu mercado, tais como: envolvimento de padrões, tamanho de rede, largura de banda aberta, segurança industrial, interoperabilidade entre equipamentos de diferentes fabricantes, etc. Dependendo do ambiente onde os sensores sem fio sejam implantados, pode-se encontrar situações inóspitas para o seu funcionamento [33]. Problemas de interferência eletromagnética e as disposições de estruturas dinâmicas, compreendida em vários tipos de ambiente (*e.g.* perímetros urbanos, industrial, etc.), o que pode acarretar em problemas de desvanecimento, por exemplo, podem ser considerados um desses cenários hostis [26]. Outro exemplo dos desafios em RSSF está relacionado ao fornecimento de energia limitado oriundo das baterias, pois em alguns tipos de aplicações a RSSF é implantada aleatoriamente e em locais de difícil acesso (*e.g.* aplicações militares e ambientais), assim não sendo possível realizar a substituição da bateria quando necessário [46]. Dessa forma, o tempo de vida da rede se torna um requisito crucial para o sucesso da aplicação.

A partir desse panorama, o problema abordado por este trabalho de dissertação se apresenta como sendo relevante, uma vez que o estudo referente à avaliação da acurácia dos simuladores - com base no funcionamento real de sensores - pode acarretar no desenvolvimento de simuladores capazes de reproduzir resultados mais acurados. Logo, os resultados desta investigação podem contribuir na elaboração e validação de projetos ou pesquisa em RSSF por meio de simuladores, com foco nos protocolos fundamentados em restrição de energia.

2.2 Arquitetura de um Nó Sensor Sem Fio

Nós sensores sem fio são o elemento central em uma RSSF. É por meio de um nó que é possível sensorear, processar e transmitir os dados pela rede até o destino. Um nó sensor consiste de três subsistemas: um responsável pela aquisição do dado sensoreado, um subsistema responsável pelo processamento e armazenamento local dos dados e um subsistema de comunicação sem fio responsável por transmitir e receber os dados. Em alguns casos, outros subsistemas podem compor o nó sensor, como exemplo do subsistema de localização, que geralmente engloba sistemas de localização via satélite (GPS), e os atuadores responsáveis por executar ações de forma programada [23]. A Figura 2.2 a seguir ilustra os subsistemas descritos e suas relações.

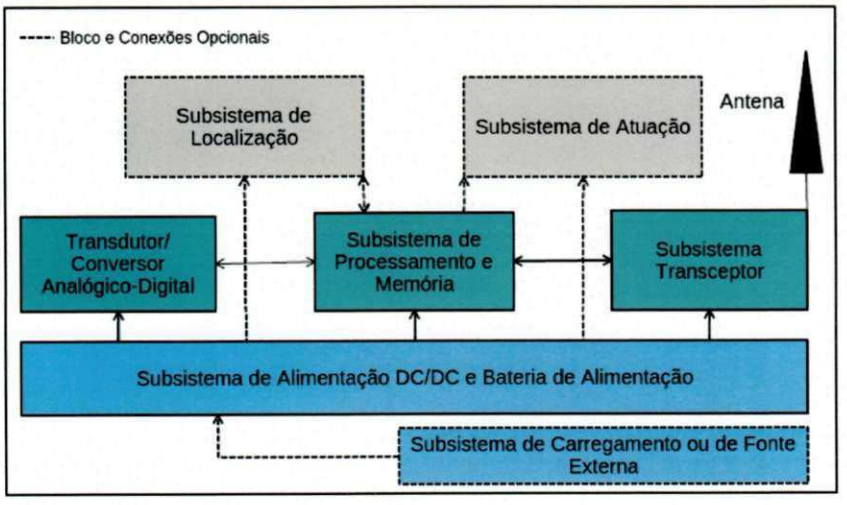

Figura 2.2: Arquitetura simplificada de um sensor sem fio.

2.3 Característica de um Nó Sensor Sem Fio

Como mencionado anteriormente, um conjunto de nós sensores depositados em um espaço geograficamente delimitado formam uma RSSF. A depender do tipo de aplicação, diferentes tipos de protocolos de comunicação podem ser adotados. O tipo de protocolo adotado na solução rege a forma como os nós da RSSF se comunicam. Um exemplo claro dessa distinção da forma dos nós se comunicarem pode ser identificada comparando os protocolos planos e hierárquicos.

Em redes com protocolo de roteamento plano, todos os nós desempenham a mesma função e colaboram entre si para realizar a tarefa. A comunicação para enviar o dado sensoreado pode ocorrer diretamente ou por meio de múltiplos saltos para alcançar o nó destino - estação base. Além do mais, esses protocolos não necessitam de *layout* pré-definido nem perímetro estabelecido, como também possibilitam a entrega de pacotes por meio de qualquer rota disponível sem considerar hierarquia e distribuição.

Numa outra abordagem, redes que operam com protocolos hierárquicos atribuem diferentes papéis aos nós sensores que constituem a RSSF. Como já visto em seções anteriores, protocolos hierárquicos compreende a constituição de grupos na RSSF, em que cada grupo é composto por um nó coordenador e um ou vários outros nós denominados de nós ordinários. O nó coordenador é incumbido de receber os dados sensoreados pelos seus demais nós ordinários, subordinados ao coordenador, processá-los e enviá-los à estação base.

Essa diferença de papéis na estrutura hierárquica exige um maior consumo de energia por parte do nó coordenador. Além dessa distinção da forma de operar e se comunicar na RSSF, os nós que compõem a rede podem ter suas características diferentes dos demais. A diferença entre a especificação técnica dos nós caracteriza a RSSF como sendo heterogênea. Nos casos em que todos os nós são idênticos, a rede é denominada homogênea. Redes heterogêneas geralmente contemplam diferentes tipos de sensores (*e.g.* captura de vídeo, infravermelho, umidade, temperatura, etc.) em que esses contemplam diferentes especificações de processamento e memória devido aos requisitos de sua aplicação fim.

Regularmente, as aplicações acerca das RSSF têm características estacionárias, em que os nós apresentam pouca ou nenhuma mobilidade espacial. Entretanto, existem aplicações similares às RSSF, como redes móveis *ad hoc* (*Mobile Ad Hoc Networks* - MANETs), que

tem a mobilidade como um requisito fundamental da aplicação, o que faz esses dois tipos de redes divergirem em alguns aspectos.

2.4 Cenários de Aplicação das RSSF

Com os avanços obtidos na fabricação de sensores sem fio, os quais possibilitaram a redução do custo de fabricação e sua miniaturização, a aplicabilidade desse equipamento cresceu de forma vertiginosa. É possível identificar uma vasta gama de aplicações em nosso cotidiano, desde aplicações de automação residencial até aplicações médicas. Adiante, um breve resumo de algumas aplicações em RSSF [9]:

2.4.1 Aplicações Residenciais

As redes de sensores sem fio vêm sendo cada vez mais utilizadas nas casas, com funções diversas, sendo as principais apresentadas a seguir.

- **Automação de Tarefas Domésticas:** Por meio da instalação de sensores em equipamentos utilizados no dia-a-dia, é possível automatizar tarefas comuns, como iluminação e controle de equipamentos como geladeiras, fornos de micro-ondas e aparelhos de ar condicionado.

- **Segurança:** Uma das aplicações domésticas mais procuradas é voltada para a área de segurança. A distribuição de sensores de temperatura e de movimento pela casa permite a detecção de incêndios e invasões, além do controle de movimentos de crianças e idosos pela casa.

- **Desenvolvimento de Ambientes Inteligentes:** Um dos principais atrativos das redes de sensores sem fio é a aplicação em ambiente doméstico inteligente. A rede permite a integração dos diversos equipamentos nos quais são instalados sensores, podendo-se, inclusive, controlá-los por voz ou por telefone. Esta rede permite a comunicação entre estes equipamentos. A integração de câmeras nesta rede permite que os moradores visualizem o que está acontecendo em um determinado local da casa, assim que algum sensor detectar qualquer irregularidade. A integração da linha telefônica permite que a

polícia, os bombeiros ou um hospital sejam avisados automaticamente em caso de invasões ou acidentes. Sensores de intensidade luminosa podem apagar as luzes durante o dia e acendê-las à noite.

2.4.2 Aplicações Militares

As aplicações militares de sensores são bastante comuns atualmente, principalmente pelo fato de não ser possível definir uma infraestrutura de comunicação em meio a uma operação de guerra. Além de isto exigir tempo, a instalação de uma central tornaria a rede vulnerável (a destruição da central acabaria com a rede). As RSSF são utilizadas em programas C4ISRT [1] (*Command, Control, Communications, Computers, Intelligence, Surveillance, Reconnaissance, and Targeting*) [9].

As principais aplicações nesta área são apresentadas a seguir.

- **Localização de Soldados:** Sensores instalados nos uniformes dos soldados permitem à central monitorar a posição e os movimentos de cada soldado. Este monitoramento pode até mesmo ser visualizado por um comandante em campo de batalha, por meio de um visor.

- **Controle de Equipamentos e Munição:** A instalação de sensores nos equipamentos e armas que os soldados carregam tornam possível o controle da munição ou outros equipamentos disponíveis.

- **Reconhecimento de Inimigos:** Por meio do espalhamento de sensores num campo de batalha, é possível monitorar e mapear os movimentos das tropas inimigas.

- **Sistemas de Mira:** Auxiliam os soldados no controle da mira, principalmente em casos de alvos móveis ou em situações de pouca visibilidade, como ambientes escuros, nebulosos ou com obstáculos no meio.

- **Detecção de Ataques Nucleares, Biológicos ou Químicos:** O sensoriamento do solo permite a detecção de gases tóxicos, substâncias radioativas, químicas e biológicas, como também a detecção de minas instaladas em determinado perímetro.

2.4.3 Aplicações Ambientais

Na monitoração de condições ambientais, as RSSF também são bastante úteis, pois em determinados casos, condições de relevo irregulares e vegetação densa são desfavoráveis para a instalação de estruturas de monitoramento destes ambientes.

A seguir, alguns exemplos dessas aplicações [9].

- **Detecção de Incêndios:** A distribuição de sensores em uma floresta permite que incêndios sejam detectados em pouco tempo e sejam localizados imediatamente e com precisão, o que permite o controle dos incêndios rapidamente, antes que se espalhem por uma área muito grande.

- **Detecção de Enchentes:** Utilizando o mesmo princípio descrito na detecção de incêndios, é possível controlar enchentes, mesmo em locais de difícil acesso.

- **Monitoração de Movimentos de Animais:** Utilizado para caça e para controle de movimentos de animais perigosos.

- **Controle de Condições Ambientais:** É possível monitorar diversos fatores ambientais, como o nível de poluição no ar ou na água, a concentração de pesticidas na água e as condições de temperatura e umidade.

- **Mapeamento da Biodiversidade do Ambiente:** Uma rede de sensores espalhada em determinado ambiente torna possível a detecção de animais e plantas localizados na região.

- **Agricultura de Precisão:** RSSF tem desempenhado um papel fundamental no âmbito da agricultura de precisão. Os recursos naturais para manutenção da agricultura (*e.g.* água) estão cada vez mais escassos impondo ao agricultor um regime de racionamento na administração desses recursos [37]. Com o advento das RSSF na agricultura, a utilização de recursos pode ser realizada de forma otimizada, por meio de aplicações de automação que viabilizem alguns processos existentes na agricultura. Um exemplo da aplicabilidade dos sistemas de monitoramento usando RSSF na agricultura de precisão, parte da necessidade de atender os diferentes requisitos durante as fases de crescimento de vegetais, por exemplo. Controle de umidade, temperatura e água são

fatores que devem ser controlados durante essa fase. Vinícolas é um dos cenários que demandam esses tipos de requisitos [14, 37].

2.4.4 Aplicações Médicas

Avanços tecnológicos em circuitos integrados, comunicação sem fio, e miniaturização de transdutores de sensoreamento fisiológico, esses com baixíssimo consumo de energia, possibilitou a criação de dispositivos de monitoramento inteligente. Parte desses sensores podem ser integrados em *Wireless Body Area Network* (WBAN), que visa suportar novas tecnologias para monitoramento de saúde. Alguns exemplos de aplicações são dadas a seguir [9].

- **Monitoração de Pacientes:** Sensores podem ser utilizados em hospitais para monitorar os movimentos dos pacientes ou controlar determinadas funções do corpo, como os batimentos cardíacos ou a pressão arterial.
- **Administração de Medicamentos:** Torna-se possível o controle da quantidade de medicamentos utilizados por cada paciente.
- **Rastreamento de Médicos:** Permite a localização precisa e imediata de médicos em um hospital, em casos emergenciais.

2.5 Desafios em RSSF

Apesar da grande aplicabilidade das RSSF, alguns dos cenários de aplicação trazem consigo desafios bem conhecidos pela comunidade científica como também pelo mercado, o que tem motivado as várias iniciativas de pesquisas com objetivo de solucionar ou mitigar alguns desses problemas. Adiante, alguns dos problemas mais comuns encontrados em diferentes ambientes de aplicação acerca das RSSF.

- **Gerenciamento de Energia:** O baixo custo de fabricação de nós sensores sem fio é uma das principais características que possibilitaram a popularização relativo às suas aplicações. Contudo, é sabido que esse tipo de dispositivo ainda impõe restrições de processamento e memória na aplicação. Essas restrições têm sido minimizadas com

os avanços tecnológicos ao longo do tempo. Todavia, o progresso "tímido"relativo ao fornecimento de energia, oriundo de baterias, ainda persiste e não demonstra evolução significativa. Dependendo do tipo de aplicação, os nós sensores podem ser implantados em locais de difícil acesso inviabilizando a sua substituição, assim tornando o fornecimento de energia uma forte restrição sob a aplicação. Em alguns casos, mesmo com a possibilidade de substituição das baterias, devido ao fácil acesso decorrente da área em que os sensores foram implantados, aplicações como vigilância requer o tempo de funcionamento maximizado, não admitindo trocas frequentes do suprimento de energia [53].

- **Heterogeneidade entre nós:** A maioria dos trabalhos sobre RSSF considera homogeneidade entre os nós sensores (*i.e.* mesmo poder de processamento, comunicação e fornecimento de energia). Entretanto, dependendo do tipo de aplicação, pode-se encontrar sensores com diferentes capacidade e reserva de energia assumindo diferentes papéis na RSSF. Algumas aplicações podem requerer diferentes tipos de nós sensores para sensorear temperatura, umidade, eventos em vídeos, assinaturas acústicas, etc. Os diferentes tipos de sensores podem possuir maior ou menor capacidade de processamento, quantidade de memória e largura de banda, o que pode impactar no desempenho de protocolos hierárquicos, devido à alternância do papel de coordenador da rede [12].

- **Qualidade de Serviço:** A larga variedade de aplicações em RSSF pode demandar diferentes requisitos, restrições e especificações. Ambientes industriais promovem um ambiente de aplicação extremamente desafiador devido aos diversos fatores, como exemplo interferência eletromagnética oriunda dos motores e máquinas elétricas, ambiente com estruturas metálicas dinâmicas que podem provocar desvanecimento de canal, etc. Requisitos que envolvam baixa latência, acurácia do dado reportado ao nó sorvedouro, perda de pacotes, por exemplo, são afetados devido ao ambiente e devem ser tratados a fim de cumprir os requisitos da aplicação [28].

2.6 Plataforma Experimental de Sensoriamento - Oracle SunSPOT

Na busca de criar uma plataforma de sensoriamento experimental, a Oracle desenvolveu os sensores SunSPOT, que integra um *kit* de desenvolvimento de software flexível para RSSF. Cada kit de desenvolvimento SunSPOT integra dois nós sensores e uma estação base. Os sensores motes compreendem um hardware com processador ARM de 400 MHz 32 bits, memória RAM de 1 MB, Rádio 802.15.4 (*Zigbee*), bateria recarregável de 3.7 V e 770 mAH (720 mAh em alguns modelos), 8 MB de memória *flash* e uma placa de extensão chamada *EDemoboard*, a qual oferece uma interface de comunicação externa que contempla sensores *on-board*, portas de entrada e saída, analógicas e digitais. A estação base contempla o mesmo hardware descrito, mas não contém bateria e a *EDemoboard*. A Figura 3.2 ilustra as dimensões e formato físico do mote.

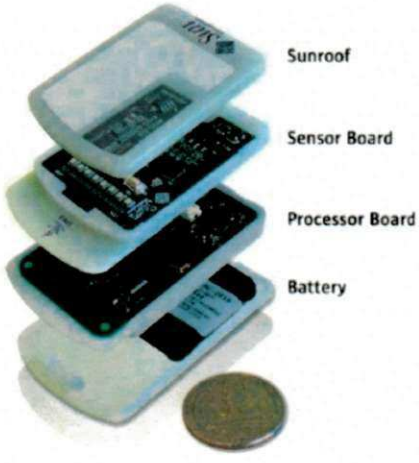

Figura 2.3: Mote SunSPOT.

A arquitetura simplificada da plataforma é ilustrada na Figura 2.4, a partir da qual é possível verificar as relações entre os subsistemas presentes no mote. A placa eDemoboard[1] é um exemplo de classe de placas de extensão compatível com a placa principal eSPOT. Essa placa de extensão contém um acelerômetro de três eixos, um sensor de luz, um sensor de temperatura, oito diodos emissores de luz de três cores, dois botões de contato, seis entradas

[1] Versão de hardware v8.0

analógicas, quatro saídas de alta-corrente/alta-tensão e cinco portas I/O de propósito geral.

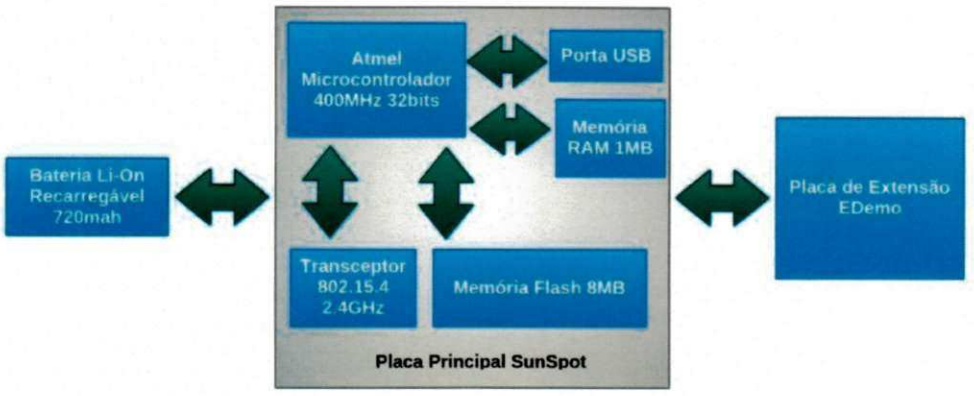

Figura 2.4: Arquitetura simplificada do mote SunSPOT.

A programação do SunSPOT se dá por meio da linguagem Java *Micro Edition* (ME). Squawk, a máquina virtual usada pelo SunSPOT, implementa o Java ME. Dessa forma, toda aplicação SunSPOT estende a classe MIDlet. Diferentemente das aplicações Java ME padrão, que suporta a execução de apenas uma aplicação por vez, o Squawk consegue proporcionar a execução de várias aplicações ao mesmo tempo, sendo possível mensagens serem trocadas durante a execução de ambas. Isso é possível graças a uma classe especial de isolamento que permite a execução das aplicações sem interferência entre elas, mesmo compartilhando os mesmos recursos subjacentes do SunSPOT. Além do mais, por meio dessa máquina virtual, é possível criar aplicações com códigos extensos ocupando pouco espaço em memória.

2.6.1 Especificação da Plataforma

2.6.1.1 Processador

O processador principal da plataforma de sensoriamento é o Atmel AT91SAM9G20, circuito integrado tipo *System on a Chip* (SoC). Essa unidade incorpora um processador AT91 ARM Thumb, com base na arquitetura ARM v4T ARM9TDMI. O processador tem um *clock* interno máximo de 400 MHz e 32 bit. O SoC contém um cache de dados com 16 KB associativa, com 64 blocos de cache e um cache de instrução de 16 KB. A memória externa é controlada e acessada por meio de um módulo *External Bus Interface* (EBI). O EBI

contém um controlador de memória estático, um controlador *Synchronous Dinamic Random Access Memory*(SDRAM) e um controlador de memória flash [6, 8].

2.6.1.2 Memória

Memória é uma unidade Spansion S71PL032J40, um pacote *multichip* que compreende 8 MB de memória flash e a 512 KB de memória pSRAM (pseudo-Static Random Access Memory). O tempo de acesso à pSRAM é de 70 nseg e à memória flash é de 65 nseg. A pSRAM é autoatualizável e automaticamente entra no modo de economia de energia quando não acessada. O conteúdo da memória é mantido enquanto houver energia da fonte de força externa ou da bateria conectada, mesmo em períodos de desligamento ou de *stand-by* profundo. A memória flash utiliza uma interface comum com operações de modo de leitura por página, escrita aleatória e por setores e função de apagar [6].

2.6.1.3 Transceptor

A interface de comunicação sem fio utiliza um transceptor integrado, TI CC2420. O transceptor é compatível com a norma IEEE 802.15.4 e opera com frequência a partir de 2,4 GHz até 2,4835 GHz, banda ISM (*Industry, Scientific and Medical*), não licenciada. Junto ao processo de modulação e demodulação adotado no transceptor, a técnica DSSS (*Direct Sequence Spread Spectrum*) é incorporada oferecendo maior robustez à comunicação. A seguir, a Tabela 2.1 com a especificação dos canais e suas respectivas frequências é apresentada:

Tabela 2.1: Canais e frequências IEEE802.15.4.

Canal	Frequência	Canal	Frequência
11	2405 MHz	19	2445 MHz
12	2410 MHz	20	2450 MHz
13	2415 MHz	21	2455 MHz
14	2420 MHz	22	2460 MHz
15	2425 MHz	23	2465 MHz
16	2430 MHz	24	2470 MHz
17	2435 MHz	25	2475 MHz
18	2440 MHz	26	2480 MHz

As potências de saída RF do transceptor também pode ser checada na Tabela 2.2 abaixo:

Tabela 2.2: Potência de saída RF - Transceptor CC2420.

Potência (dBm)
0
-1
-3
-5
-7
-10
-15
-25

2.7 Simulação em RSSF

Simulações são consideradas imitações do *Modus Operandi* de procedimentos ou sistemas do mundo real, executado ao longo do tempo. Em outras palavras, simuladores envolvem a criação de uma história artificial do sistema, em que, por meio de observações, é possível retirar inferências sobre as características do sistema real. Esses modelos são elaborados tomando como base um conjunto de suposições em relação à operação do sistema. Tais suposições são representadas utilizando expressões matemáticas, lógicas e um relacionamento simbólico entre as entidades ou objetos de interesse compreendidos no sistema. Uma vez desenvolvido e validado, um modelo pode ser utilizado para investigar uma vasta diversidade de questionamentos sobre o sistema no mundo real [32].

Em projetos de sistemas que se encontram em fases iniciais, como concepção e *design*, é comum a utilização de simuladores para averiguar requisitos antes da sua materialização. Modelos de simulação também podem ser utilizados como ferramenta de análise para predizer efeitos das mudanças existentes no sistema, como também ferramenta de projeto para predizer o desempenho de novos sistemas sob variação de um conjunto de circunstâncias [32]. Simulação tem se demonstrado como uma solução viabilizadora em vários domínios, pois a observação com base empírica, apenas no mundo real, pode se tornar inviável devido

ao alto custo e investimento de tempo necessário para tal. Entretanto, em algumas situações, simulações também podem se apresentar inviáveis quando o sistema a ser modelado envolve certa complexidade em termos de expressar/representar algo matematicamente [32].

No campo da comunicação e redes de computadores, simuladores englobam um conjunto de modelos que são capazes de representar, com certo grau de aproximação, o comportamento dos elementos contidos na rede se prevalendo de cálculos matemáticos para descrever a interação entre as entidades e seus respectivos eventos. Usando como exemplo as ferramentas de simulação aplicadas à RSSF, essas são capazes de simular desde o comportamento do próprio nó sensor (*i.e.* camadas de abstração e seu funcionamento, bateria e sua forma de descarga) até as condições ambientais que interferem no envio e recebimento dos pacotes transmitidos (*i.e.* modelo do canal sem fio) [4, 17].

Ferramentas de simulação têm sido um recurso frequentemente adotado em pesquisas no campo dos sistemas de comunicação. Em [34], Khan et al. realizaram uma pesquisa com objetivo de elicitar a quantidade de trabalhos suportados por simuladores nesse âmbito. O resultado dessa investigação traz uma quantidade significativa de trabalhos que utilizaram simuladores como forma de validação. Cerca de 11% dos 36103 trabalhos publicados, durante o período de janeiro de 2006 à dezembro de 2010, que atenderam aos critérios da pesquisa, utilizaram simuladores para efetuar testes e validar seus achados. As conferências adotadas como fonte de pesquisa foram as seguintes: IEEE ICC, VTC, INFOCOM, GLOBECOM, PIMRC, WiCOM, WCNC e MILCOM. Os trabalhos resultantes desse levantamento compreendem uma larga variedade de tópicos na área de redes computadores e comunicação, e RSSF faz parte desse quadro. Também é importante realçar que parte das ferramentas de simulação citadas em [34], tais como OMNET++ e NS, são utilizadas para simular diversos tipos de redes (*e.g.* BAN, PAN, WLAN, MANETS, VANETS), assim destacando a concordância das ferramentas de simulação para realização de pesquisa nesse âmbito. Entretanto, de forma discordante a esse panorama, alguns trabalhos como [27] têm indagado que a predominância do uso de simuladores em RSSF, principalmente acerca das investigações de eficiência energética dos *energy-aware protocols* e consumo de energia por parte dos transceptores, podem negligenciar fatores importantes no desempenhos das RSSF, assim criando "viés"sob os resultados. Ademais, outros trabalhos como [16, 20] também destacam a necessidade de realizar pesquisas a fim de avaliar a acurácia das ferramentas de simulação no

âmbito das RSSF.

Além do aspecto da existência de fatores não abrangidos em RSSF por meio de ferramentas de simulação, existem outros tipos de preocupação inerentes ao uso desse tipo de solução que desperta questionamentos acerca da sua adoção em alguns cenários. Como pode-se analisar a seguir, é possível encontrar várias iniciativas de investigação que buscaram analisar desde comparar o desempenho entre as plataformas até a acurácia dos modelos englobados nos simuladores [16, 20, 34, 42, 44, 45, 56]. Na perspectiva de análise de desempenho, não muito aderente ao tema desta dissertação, porém relevante ao contexto de análise comparativa entre simuladores, em [56, 58], pôde-se evidenciar a existência de atenuações referente ao desempenho das ferramentas de simulação, em que essas podem exibir variabilidade no desempenho para um mesmo *set-up* experimental (*e.g.* tempo de simulação, consumo de memória), bem como mudança de performance devido à capacidade de escalabilidade. Além das questões relacionadas ao desempenho das ferramentas, outras investigações como [16, 20, 34, 42, 44] buscaram averiguar a confiabilidade dos resultados gerados pelas ferramentas em contextos variados.

Em [20], Colesanti et al. realizaram uma das pesquisas precursoras sobre avaliação da acurácia da ferramenta OMNET++ e uma das sua extensões, a MAC *Simulator framework*. Para tanto, um *testbed* composto de seis nós TMoteSky foi montado e quatro cenários de avaliação experimental definidos. O mesmo *set-up* foi configurado na ferramenta de simulação. A avaliação consistiu em observar um algoritmo de *flooding* (protocolo B-MAC) em ambos os ambientes de implantação, com base em um conjunto de métricas e nos diferentes cenários estabelecidos. Os achados desse trabalho apontaram para uma superestimação das métricas por parte do simulador em todos os cenários. Entretanto, após as modificações realizadas no simulador com o subsídio de informações coletadas durante o experimento, a acurácia da ferramenta foi superior a 90% em todos os cenários.

Com o mesmo objetivo de avaliar a acurácia de ferramentas de simulação em RSSF, Bergamini et al. utilizaram como referência, em [16], dados reais obtidos pelo *Motelab testbed* [57]. Para tal, o autor reproduziu o mesmo *set-up* utilizado no *testbed* em duas ferramentas de simulação, Castalia e NS2. Os achados da pesquisa elicitaram questionamentos relevantes acerca do tema. A acurácia[2] da conectividade e topologia, como também a taxa de entrega

[2] A acurácia foi medida utilizando o coeficiente de Jaccard na comparação entre as amostras

de pacotes, foram as métricas usadas na avaliação. Como parte do produto dessa investigação, pode-se constatar que os simuladores podem atingir um nível de acurácia maior se o "ajuste fino", com base na especificação do sensor, for efetuado no simulador, o que é sabido. Entretanto, a maior contribuição dessa pesquisa foi a elicitação de resultados controversos entre as ferramentas de simulação. A ferramenta NS2 teve o pior desempenho nas métricas de similaridade da topologia e conectividade, todavia superou o Castalia na taxa de entrega de pacotes. Como justificativa aos resultados controversos, os autores recomendaram melhorias no modelo de interferência do Castalia, os mesmos também ressaltam que alcançar uma boa taxa de entrega de pacotes em uma topologia distante da real torna-se algo sem valia para o usuário final. Nesse trabalho não foi especificada a versão do Castalia adotada no experimento.

Numa outra perspectiva, uma investigação recentemente conduzida por Perez e Kostanic [44] também buscou avaliar a acurácia de simuladores em RSSF, contudo a ferramenta utilizada nesse estudo é proprietária[3], diferentemente das demais supracitadas que são *open-source*. Nessa pesquisa, os autores coletaram informações de tráfego e características da rede a partir de um pequeno *testbed*, composto de três sensores (*Crossbow Starter Kit*), por meio de um *sniffer*, a fim de ajustar os parâmetros de simulação da ferramenta. O *sniffer* foi utilizado para processar todos os dados relativos ao tráfego e a taxa de transmissão por meio de um *script* em MATLAB, que calcula a média e o desvio padrão. Essas informações estatísticas são usadas na construção do modelo de tráfego da ferramenta, como também na comparação pelo modelo de validação. Resultados demonstraram um comportamento muito semelhante da ferramenta de simulação comparado ao *testbed* montado. Com 1% de significância, não foi possível refutar a hipótese nula do estudo que afirma igualdade de variância entre dos dados coletados da plataforma de sensoriamento e da ferramenta de simulação.

Durante o levantamento dos trabalhos relacionados desta pesquisa, não foi encontrada nenhuma iniciativa no tocante à avaliação de ferramentas de simulação utilizando protocolos com diferentes abordagens (*i.e.* comunicação direta, agrupamento) utilizando o Castalia como simulador, como também utilizando sensores com maior capacidade de processamento, tais como SunSPOT. Sendo assim, esta dissertação traz uma investigação que pode contribuir significativamente com a comunidade no âmbito das RSSF.

[3]Versão acadêmica disponível por meio do *OPNET University Program*.

2.7.1 OMNET++

O *Objective Modular Network Testbed in C++* (OMNET++) é um arcabouço de simulação em redes, modular, orientado à objetos e baseado em eventos. O OMNET++ em si não é um simulador com fins específicos, mas um arcabouço que provê ferramentas e uma infraestrutura para escrever simulações [4]. O OMNET++ foi desenvolvido com propósito de atender o público acadêmico e educacional no âmbito das redes de computadores ou sistemas paralelos e distribuídos. Outro motivo na percepção da ferramenta foi a resolução da lacuna existente entre plataformas *open-source*, softwares de simulação voltados à pesquisa e ferramentas de simulação proprietárias de alto custo, como exemplo OPNET [55].

A arquitetura do OMNET++ é genérica, sendo capaz de encorporar vários módulos de forma hierarquizada por meio de interfaces, denominada de portas (*Gates*), o que viabiliza a troca de mensagens entre os módulos. Dois tipos de portas são utilizadas para troca de mensagens, entrada e saída, as quais podem ser interconectadas por meio de conexões. Algumas propriedades podem ser atribuídas a essas conexões, como exemplo de atraso de propagação, taxa de transferência e erro de *bit*. A Figura 2.5 mostra a relação entre módulos e sua composição.

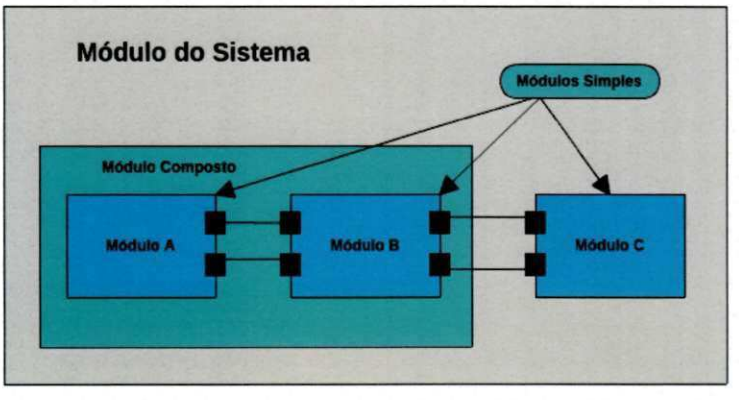

Figura 2.5: Organização dos Módulos - OMNET++. Adaptado de [4].

Como pode-se observar na Figura 2.5, um conjunto de módulos pode formar um módulo composto, que também podem interagir com outros módulos simples. Não existe quantidade limite para níveis de hierarquização entre os módulos. A composição dos módulos no OMNET++ é declarada por meio da linguagem de definição de estrutura do modelo de simulação, linguagem NED (*Network Description*). A simplicidade do mecanismo de adicionar e

remover módulos no OMNET demonstra a extensibilidade do arcabouço de simulação para encorporar novos modelos de simulação [55]. No intuito de demonstrar a extensibilidade da ferramenta e sua vasta aplicabilidade, em [3] pode-se encontrar diversos modelos de simulação com propósitos específicos, tais como: INET *framework*, MAC *Framework Simulator*, Castalia, etc. O modelo de simulação adotado para realização deste trabalho foi o Castalia, que é descrito na próxima seção.

2.7.2 Castalia

Castalia é uma ferramenta de simulação voltada à RSSF, *Body Area Network* e redes com dispositivos embarcados de baixo consumo. A ferramenta de simulação funciona com base na plataforma OMNET++. O modelo de simulação Castalia tem como público alvo pesquisadores e desenvolvedores que busquem soluções para testar algoritmos e protocolos em um canal de comunicação sem fio realístico e modelos de rádio, como também comportamento realístico por parte do sensor relativo ao acesso do rádio. A ferramenta também pode ser utilizada para avaliar diferentes características da plataforma para aplicações específicas devido ao Castalia ser altamente parametrizado, além de simular uma vasta variedade de plataformas de sensoriamento.

Castalia não é simulador de plataforma específica. A ferramenta provê um arcabouço genérico confiável e realístico para realizar validação de primeira ordem de um algoritmo, antes de ser implementado em uma plataforma de sensoriamento específica. O Castalia não é recomendável nos casos de teste de código compilado a partir de uma plataforma específica.

Devido à alta aceitabilidade do Castalia pela comunidade de pesquisa em RSSF, e as novas tendências no contexto de sensoriamento, a lacuna de algumas funcionalidades para atender às demandas emergentes foram notadas. À exemplo, o suprimento de energia com base em recursos renováveis (*i.e.* energia solar). Para resolver essa lacuna, um *framework* extensível para simular RSSF com suporte à diferentes fontes de energia (*e.g.* renováveis, baterias recarregáveis) foi desenvolvido e proposto por [15]. Essa extensão possibilita a utilização de recursos não contemplados pelo Castalia, tais como: suporte à arquitetura de múltiplas fontes de energia heterogêneas (*e.g.* super capacitores, painel de energia solar, bateria recarregável), suporte à modelos de descarga de bateria (não ideal) com base em modelo empírico, utilização de fontes heterogêneas em paralelo, etc [15]. Essa extensão é

uma demonstração da extensibilidade suportada pelo OMNET++/Castalia.

2.7.3 Resumo

Nesse capítulo foi abordado tópicos de fundamental importância sobre RSSF, tais como cenários de aplicações, arquitetura e características de um nó sensor, desafios e a especificação da plataforma de sensoriamento adotada neste trabalho. Um panorama geral sobre a relevância de simuladores de RSSF em trabalhos científicos também foi apresentado.

Capítulo 3

O protocolo LEACH

3.1 Introdução

Uma das principais restrições das RSSF é a quantidade de energia limitada disponível para manter o seu funcionamento. Encontrar meios eficientes de gestão de energia em RSSF tem sido um desafio recorrente enfrentado pela comunidade científica nos últimos anos. Parte do esforço tem se dividido em elaborar meios eficientes de consumo de energia em diversos aspectos no âmbito das RSSF, tais como: melhorias no design de rede e na camada MAC, bem como o desenvolvimento de novas abordagens relativos aos protocolos de roteamento, por exemplo. Entre as várias categorias de protocolos de roteamento (*e.g.* protocolo baseado em localização, centrado a dados, baseado em mobilidade, hierárquico, baseado em QoS)[51], o hierárquico tem se destacado como uma solução promissora em gestão de energia nesse domínio. O desenvolvimento de protocolos capazes de organizar a RSSF em grupos, visando otimizar a comunicação entre os nós e reduzir o consumo de energia, tem se tornado um problema vastamente explorado. Um dos exemplos mais difundidos pela comunidade científica nesse sentido é o protocolo LEACH.

3.2 O Protocolo LEACH e seu funcionamento

O LEACH é um exemplo de protocolo hierárquico que visa maximar a utilização dos recursos energéticos da RSSF. O seu funcionamento consiste em dividir a rede em grupos que são compostos por um nó coordenador e por um ou vários nós denominados de nós

ordinários.

Os nós coordenadores desempenham um papel diferenciado entre os grupos formandos. Esses são responsáveis por receber os dados oriundos dos sensores que fazem parte do respectivo grupo coordenado, dentro de um agendamento estabelecido pelo próprio coordenador via *Time Division Multiple Access* (TDMA). Além disso, sintetizar os dados sensoreados e repassá-los ao destino, o nó sorvedouro (*Sink Node*), também é uma das atribuições do coordenador. Enquanto os coordenadores da rede ficam ativos durante todo o período da formação do grupo estabelecido, os demais nós entram em estado de espera após o envio do dado sensoreado, conforme o agendamento criado pelo seu coordenador concernente. Dessa forma, os nós coordenadores têm um consumo de energia superior aos demais. O engenho do LEACH visa equilibrar o consumo de energia, já que todos os nós da rede desempenham ambas as funções em um esquema de rodízio entre os nós. Para assegurar essa alternância da função de coordenador, um cálculo de probabilidade é executado por cada um dos sensores, em que os nós estabelecidos como coordenador em um dado ciclo, não poderão ser reeleitos como coordenador até que todos os nós já tenham assumido esse papel.

A Figura 3.1 ilustra a formação dos grupos conforme a evolução dos ciclos [30]. Os pontos na figura representam os nós sensores enquadrados em diferentes grupos. A cor dos pontos faz menção a um respectivo grupo estabelecido. Os nós com círculo representam os coordenadores e os demais representam os nós ordinários. É possível observar que, em um instante de tempo t, a rede está agrupada de forma diferente ao instante de tempo $t + 1$.

Cada rodada ou ciclo do funcionamento do LEACH compreende dois momentos: *SET-UP phase* e *Steady phase*. A primeira fase é responsável pela formação de grupos, ou seja, a eleição dos nós coordenadores e a adesão dos nós ordinários ao coordenador; a segunda contempla a fase de recepção dos dados sensoreados dos nós ordinários pelos coordenadores, manipulação/processamento e envio ao nó sorvedouro. Ambas as fases são detalhadas a seguir.

3.3 Formação de Grupos

A fase de formação de grupos no LEACH funciona de forma autônoma e descentralizada. Isso significa que cada nó, a partir de um cálculo de probabilidade, é capaz de se autoeleger

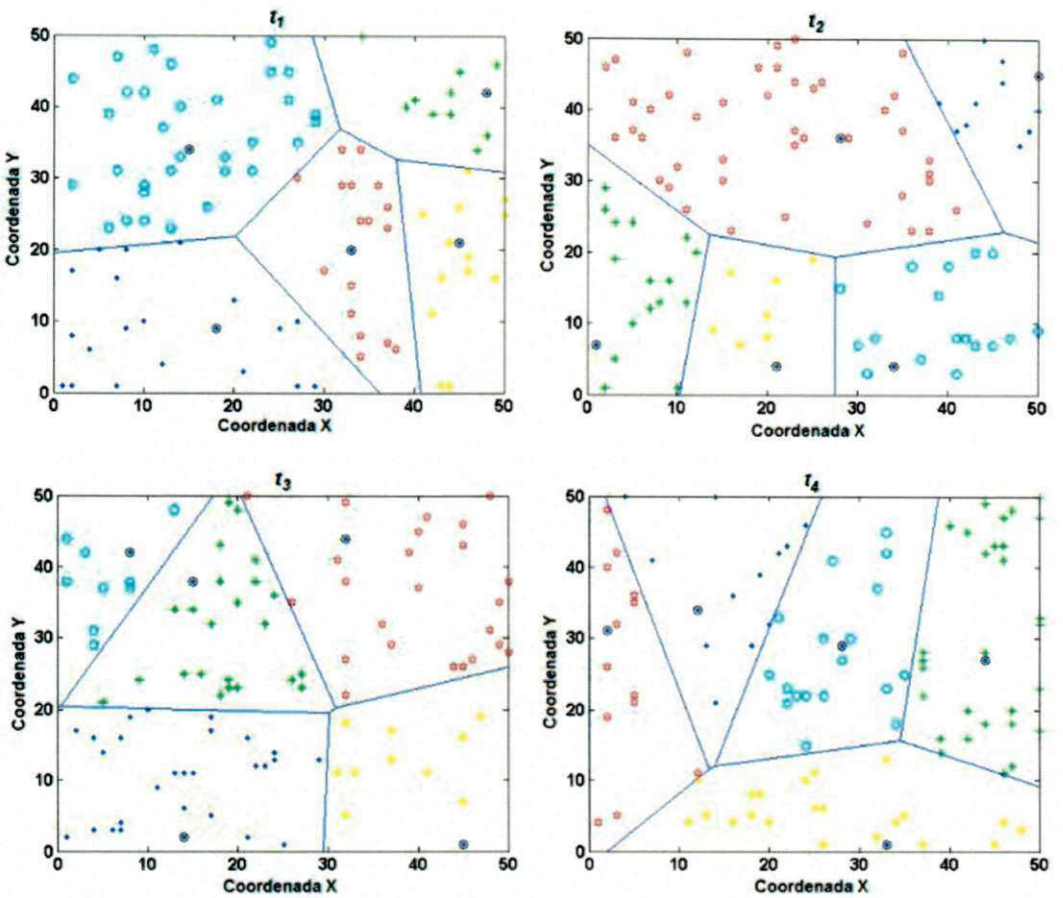

Figura 3.1: Rodadas de formação de grupos [46].

como coordenador. Inicialmente, é necessário definir previamente um percentual relativo à quantidade de coordenadores por ciclo de formação de grupos. No processo de eleição, cada nó calcula um número n, aleatório, entre 0 e 1. A partir do resultado obtido pela Função T(n), caso o valor aleatório seja inferior ao valor obtido pela função, o nó se autoelege como coordenador. A função que calcula o limiar é dada pela Equação 3.1.

$$T(n) = \begin{cases} \frac{P}{1-P(rmod\frac{1}{P})}, & n \in G, \\ 0, & \text{caso contrário}, \end{cases} \tag{3.1}$$

Em que P representa o valor em percentual desejado da quantidade de coordenadores por rodada na RSSF (*e.g.* P = 0,10), r representa o ciclo corrente e G é o conjunto de nós que ainda não se elegeram como coordenadores no ciclo $\frac{1}{P}$. Na rodada inicial (r = 0),

todos os nós têm a mesma probabilidade P de se tornarem coordenadores. Os nós já eleitos coordenadores têm o retorno da função $T = 0$ enquanto houver nó não eleito. Conforme o incremento dos ciclos na função, a probabilidade do nó ainda não eleito aumenta chegando à $T = 1$ no ciclo $\frac{1}{P} - 1$.

Após eleito coordenador, uma mensagem de anúncio é enviada em difusão, para todos os nós dentro do perímetro de alcance, por meio do protocolo *Carrier Sense Multiple Access* (CDMA), a fim de verificar se o canal de comunicação já está sendo ocupado antes de iniciar a transmissão da mensagem. Após o envio da mensagem de anúncio, os nós ordinários podem receber mais de uma mensagem de anúncio oriunda de diferentes coordenadores. Para saber qual das mensagens será respondida no próximo passo, o nó ordinário tomará como válida a mensagem com maior intensidade de sinal, o *Received Signal Strengh Indicator* (RSSI).

Para responder a mensagem de anúncio e ingressar ao grupo, o qual decidiu pertencer, o nó ordinário envia uma mensagem de participação diretamente ao coordenador. No envio dessa mensagem, o protocolo CSMA também é adotado para evitar problemas de congestionamento do canal. Com o recebimento das mensagens de participação, o coordenador elabora um esquema de agendamento utilizando *Time Division Multiple Access* (CDMA) e o envia aos nós participantes do grupo. A partir desse momento, todos os nós ordinários entram em estado de espera, com objetivo de diminuir o consumo energético, aguardando o momento alocado (fatia de tempo) via agendamento para coleta e envio do dado sensoreado ao nó coordenador [30].

3.4 Fusão de Dados

A partir da criação dos grupos e instanciação do agendamento TDMA, a transmissão de dados pode ser iniciada. Todos os nós ordinários, ao assumir a fatia de tempo a si alocada, conforme o agendamento, transmite o dado sensoreado ao coordenador. De posse dos dados recebidos, o coordenador inicia uma etapa local de pré-processamento. Essa etapa visa sintetizar os dados com a finalidade de gerar um dado composto, dependendo da natureza do dado capturado pelo transdutor [30]. Caso o sinal captura seja de natureza sísmica ou de áudio, o coordenador pode empregar uma conformação de feixes dos sinais individuais

para gerar um composto [52]. Essa síntese dos dados tem por consequência uma diminuição considerável do tráfego entre os coordenadores e o nó sorvedouro.

3.5 Múltiplos Grupos

Com a criação de múltiplos grupos na rede, a transmissão de um grupo pode afetar a comunicação dos demais. Como podemos verificar na Figura 3.2, a transmissão entre dois sensores pode ser afetada devido à interferência de nós periféricos. Para mitigar esse problema, os nós durante a transmissão se valem de códigos para se comunicar entre os participantes do grupo usando o método *Code Division Multiple Access* (CDMA).

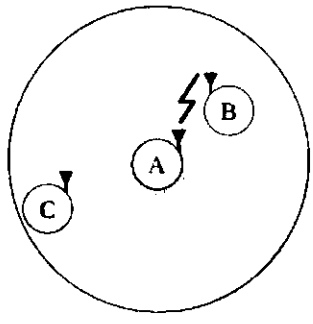

Figura 3.2: Interferência de transmissão entre nós sensores [30].

Utilizar códigos CDMA, apesar de não ser necessariamente a opção mais eficaz no aproveitamento da largura de banda, soluciona o problema de múltiplo acesso de uma maneira distribuída, além de prover maior segurança à transmissão das informações [46].

3.6 Outras Versões do LEACH

Como mencionado em seções anteriores, o LEACH sofreu várias modificações com a finalidade de melhorar o esquema de rodízio, consequentemente a utilização dos recursos de energia da rede. A primeira versão do LEACH em seu engenho para formação de grupos não leva em consideração fatores relevantes que influenciam no funcionamento da RSSF (*e.g.* consumo energético do nó em rodadas anteriores, energia residual), o que motivou a comunidade científica a propor novas abordagens. Entre essas versões propostas na literatura,

é possível encontrar alternativas do protocolo em que muda completamente o engenho originalmente proposto, como exemplo do LEACH-C, desenvolvido pela proponente da primeira versão - o protocolo deixa de adotar um esquema distribuído de eleição do nó coordenador para uma abordagem centralizada. Com o objetivo de sumarizar algumas das versões do LEACH existentes na literatura, Kumar et al. apresentam uma revisão sistemática em que é possível verificar as variações do protocolo em relação à versão original [36]. A seguir, um resumo das principais versões do LEACH é apresentada.

- **LEACH-A (Advanced Low Energy Adaptive Clustering Hierarchy):** A primeira versão do LEACH apresenta ineficiência no consumo de energia, por parte do coordenador de grupo, em que o seu consumo se torna superior aos demais devido ao papel desempenhado. O protocolo LEACH-A é um protocolo heterogêneo utilizado para diminuir a probabilidade de falha de nós e que se prolonga para o intervalo de tempo antes da morte do primeiro nó (chamado período de estabilidade). Cada sensor conhece a partida de cada rodada usando relógio sincronizado. Seja n o número total de nós, e m a fração de n que contém mais energia que outros nós, os chamados nós CGA (nós selecionados como *gateways* ou chefes de fragmentação), então o resto de $(1 - m) * n$ nós funcionam como nós normais [10].

- **LEACH-C (Centralized Low Energy Adaptive Clustering Hierarchy):** LEACH-C tem o mesmo comportamento da fase estabilizada em relação a primeira versão do protocolo, apenas mudando a abordagem na fase de formação de grupos. O processo de eleição do nó coordenador deixa de ser distribuído e passa ser centralizado, dessa forma, a responsabilidade do processo de eleição fica por conta da estação base ou nó sorvedouro. Relativo ao cálculo de probabilidade, a função engloba dois novos parâmetros para formação de grupos, a energia residual do nó e a localização obtida via GPS ou por outra técnica de localização [29].

- **LEACH-M (Mobile Low Energy Adaptive Clustering Hierarchy):** O protocolo LEACH-M foi proposto para contemplar aspectos de mobilidade. Este protocolo proporciona mobilidade para os dois tipos de nós, coordenador e ordinário. Nesse caso, admite-se que os nós são homogêneos e localização de cada nó é coletada via GPS. Os nós com mobilidade mínima e o menor atenuação estão mais propensos a se tor-

narem coordenadores de grupo [35]. A primeira versão do protocolo não compreende nenhum aspecto de mobilidade.

- **LEACH-E (Energy Low Energy Adaptive Clustering Hierarchy):** Na versão LEACH-E subtende-se que inicialmente todos os nós têm a mesma energia e mesma probabilidade de se tornar líder de grupo. Após a primeira rodada, o nível de energia de cada nó muda. Em seguida, a quantidade de energia residual de cada nó é usado no cálculo de probabilidade para eleger os nós coordenadores. Os nós com maior energia residual são os nós com maior probabilidade de se tornarem coordenadores. O engenho do LEACH-E visa melhorar o tempo de vida de rede, aumentando o balanceamento da utilização dos recursos energéticos entre os nós na rede [59].

3.7 Outros Protocolos

No âmbito das RSSF, vários tipos de protocolos de roteamento foram propostos na literatura, entre esses protocolos os de comunicação direta e rota mínima de energia, os quais foram utilizados em [30] a fim de realizar uma análise comparativa em relação ao protocolo proposto. Ambos os protocolos visam atender requisitos de restrição de energia em RSSF, como o LEACH. Um resumo do funcionamento desses protocolos de roteamento é realizado a seguir.

- **Comunicação Direta:** O protocolo de comunicação direta permite que todos os nós possam enviar os dados sensoreados diretamente ao nó sorvedouro. Ou seja, caso o nó sorvedouro esteja distante de um dado nó ordinário, prestes a estabelecer transmissão, o protocolo de comunicação direta irá requerer uma alta potência de transmissão do nó sensor. Isso acarreta numa descarga rápida da bateria e num menor tempo de vida do sistema. Dessa forma, caso os nós sensores estejam próximos do nó destino, ou a energia requerida para recepção dos dados seja expressiva, comunicação direta pode ser um método aceitável [30].

- **Rota Mínima de Energia:** No protocolo de rota mínima de energia, do inglês *Minimum Transmission Energy* (MTE), a comunicação entre os nós admite múltiplos saltos até alcançar o nó destino, o que não é possível com comunicação direta. O princípio

de funcionamento desse protocolo se baseia em encontrar a rota mínima em relação ao consumo de energia. Ou seja, para que um nó intermedie a comunicação para alcançar o nó destino, o custo relativo à potência de transmissão necessária deve ser inferior ao custo da comunicação direta. Para exemplificar, considere um nó fonte **A**, um nó intermediário **B** e um nó destino **C**. Considere que a distância entre **A** e **B** é representada por d_{AB}, a distância entre **B** e **C** é dada por d_{BC} e, por fim, a distância entre **A** e **C** é representada por d_{AC}. Então, o nó **A** transmite um pacote para o nó **C** por meio do nó **B**, se e somente se $d_{AB}^2 + d_{BC}^2 < d_{AC}^2$ [30].

Ambos protocolos de roteamento, comunicação direta e MTE, apresentam problemas de balanceamento de carga na rede. O protocolo MTE, devido às requisições de roteamento aos nós com menor distância entre o nó de origem e sorvedouro, tem por consequência exaurir a fonte de energia do nó rapidamente, assim parcela da RSSF torna-se inativa forçando a comunicação direta com alta potência de transmissão por parte dos nós mais afastados do sorvedouro. Dessa forma, o efeito cascata transcorre ocasionando a baixa do sistema precocemente.

Conforme experimento realizado por [30], pode-se verificar os resultados relativos ao balanceamento de carga da RSSF, observando as Figuras 3.3 e 3.4. Para ambas as figuras, os pontos escuros representam os nós inativos e os círculos os nós ativos. Esses resultados compreendem carga inicial em cada nó de 0,5J para 180 ciclos. O nó sorvedouro está localizado nas coordenadas ($x = 0$, $y = -100$).

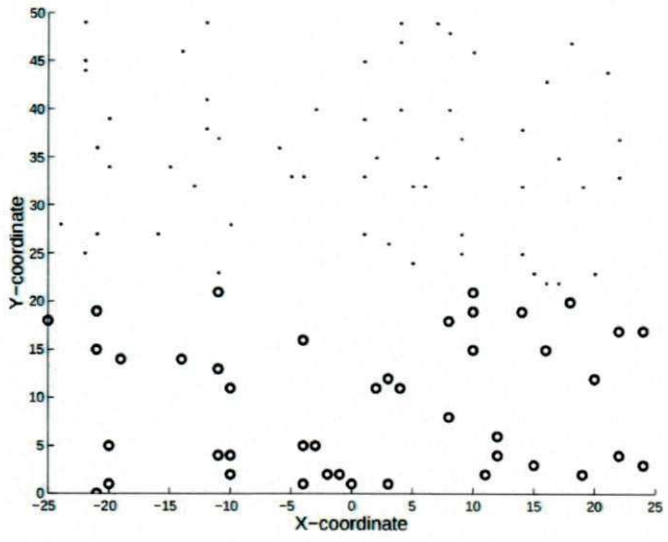

Figura 3.3: Resultado simulação protocolo de roteamento comunicação direta [30].

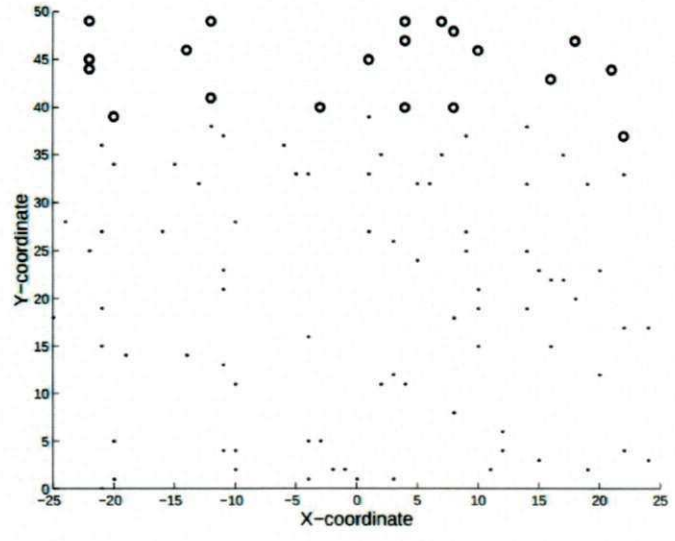

Figura 3.4: Resultado simulação protocolo MTE [30].

3.8 LEACH em Prática

A seguir, para exemplificar o funcionamento da versão predecessora do protocolo LEACH, um instância do seu funcionamento é explanada:

Inicialmente, subentende-se que todos os nós são homogêneos e contêm a mesma carga de energia inicial. Para esse exemplo, adotou-se uma RSSF composta de 10 nós sensores e uma estação base. O percentual de agrupamento da rede foi definido em 20%, ou seja,

dois nós devem ser eleitos em cada rodada. No início do engenho do LEACH, todos os nós da RSSF, com exceção da estação base, geram um número aleatório entre 0 e 1. A Tabela 3.1 demonstra os números aleatórios gerados hipoteticamente com finalidade de realizar um teste de mesa.

Tabela 3.1: Números aleatórios gerados pelos nós.

ID do Nó	**Número Aleatório**
1	0,498
2	0,311
3	0,123
4	0,192
5	0,567
6	0,453
7	0,317
8	0,231
9	0,909
10	0,766

Após geração dos números aleatórios de cada sensor, o cálculo de limiar deve ser realizado por meio da Equação 3.2, com taxa de eleição de coordenador ou formação de grupos igual a 20%, como mencionado. Com esse percentual especificado, o exemplo compreende $\frac{1}{0,2}$ ciclos, sendo que no ciclo $\frac{1}{0,2} - 1$ todos os nós que ainda não foram eleitos, tornam-se coordenadores automaticamente. A Equação 3.2 demonstra o cálculo em seu ciclo inicial, em que $r = 0$. Note que no ciclo atual, todos os nós n pertencem ao conjunto de nós elegíveis G, pois nenhum elemento n assumiu o papel de coordenador no ciclo atual.

$$T(n) = \begin{cases} \frac{0.2}{1-0.2(0 mod \frac{1}{0.2})}, & n \in G, \\ 0, & \text{caso contrário}, \end{cases} \tag{3.2}$$

Os valores de limiar obtidos nesse exemplo por meio da Equação 3.2 são apresentados na Tabela 3.2. Com base nos números aleatórios gerados e os valores de limiar correspondentes, pode-se observar a eleição dos nós coordenadores por ciclo, como exposto na Tabela 3.3. No ciclo $r = 2$, não foi possível manter o percentual de 20%, o que acarretou em um percentual

Tabela 3.2: Limiares obtidos com taxa de agrupamento de 20%.

# Rodada	Limiar
0	0,200
1	0,250
2	0,330
3	0,500
4	1,000

Tabela 3.3: Eleição dos nós coordenadores.

# Rodada	Limiar	Nós Eleitos
0	0,20	{3;4}
1	0,25	{2;8}
2	0,33	{7}
3	0,50	{1;6}
4	1,00	{5;9;10}

de 30% no último ciclo. Isso ocorre mais frequentemente em redes com baixa quantidade de sensores.

3.8.1 Resumo

O protocolo LEACH e seu engenho operacional foi apresentado e discutido neste capítulo. Foi apresentado também algumas versões variantes do LEACH, bem como as respectivas diferença quando comparado à versão genuína. Ao final, um teste de mesa foi realizado a fim de demonstrar as etapas de funcionamento.

Capítulo 4

Preparação e Execução dos Experimentos

4.1 Definição do Problema

Tendo em vista a necessidade de iniciativas de pesquisa com abordagens práticas no âmbito das RSSF, como apontado por [27, 61], este trabalho de dissertação compreende a avaliação do funcionamento de protocolos de comunicação direta e agrupamento, ambos implementados em plataforma real de sensoriamento e em ferramenta de simulação, a fim de analisar a confiabilidade do simulador em relação a um cenário composto de sensores reais.

Para tanto, o experimento engloba o protocolo de comunicação direta (*i.e.* sem roteamento, via *broadcast* e *non-beacon*) e o protocolo de agrupamento LEACH, ambos implementados na plataforma de sensoriamento SunSPOT quanto na ferramenta de simulação Castalia. A comparação dos resultados coletados tem com base as métricas detalhadas a seguir. Visando delimitar o objetivo da investigação de forma sucinta, utilizou-se a definição do objetivo no formato *Goal* oriundo do *Goal-Question-Metric* (GQM), utilizado no contexto da engenharia de software . A seguir, o *Goal* é descrito em detalhes.

Será analisado o protocolo de comunicação direta e de agrupamento LEACH, **com a intenção de** identificar a confiabilidade da ferramenta de simulação Castalia em relação à plataforma real, **com respeito a sua** eficácia no tocante às métricas de tempo de vida da rede e tráfego de dados, **do ponto de vista de** desenvolvedores de simuladores de RSSF,

pesquisadores e especialista em implantação de RSSF, **no contexto de** experimentação com RSSF, *indoor*, controlada.

A partir da definição do escopo de pesquisa utilizando o *Goal*-GQM, esta pesquisa visa responder as seguintes perguntas no tocante ao problema técnico e de negócio:

- **Problema de Negócio:** Avaliações de protocolos de comunicação em RSSF por meio de simulação apresentam resultados confiáveis em relação aos obtidos em plataformas reais de sensoriamento?
- **Problema Técnico:** Há diferença significativa entre os resultados obtidos a partir de simuladores e sensores reais, relativo à comparação entre protocolos de comunicação direta e agrupamento em RSSF no tocante às seguintes métricas: Tempo de vida da rede e quantidade de dados trafegados?

Com base nas questões de problema técnico e de negócio, as hipóteses a seguir foram estabelecidas:

Hipótese Nula 0-0: O protocolo LEACH avaliado em ambiente de simulação e sensoriamento em plataforma real apresentou desempenho equivalente tendo como base as seguintes métricas: Tempo de vida da rede e quantidade de dados transmitidos durante seu funcionamento.

Hipótese Alternativa 0-1: O protocolo LEACH avaliado em ambiente de simulação e sensoriamento em plataforma real não apresentou desempenho equivalente tendo como base as seguintes métricas: Tempo de vida da rede e quantidade de dados transmitidos durante seu funcionamento.

Hipótese Nula 1-0: O protocolo de comunicação direta avaliado em ambiente de simulação e sensoriamento em plataforma real apresentou desempenho equivalente tendo como base as seguintes métricas: Tempo de vida da rede e quantidade de dados transmitidos durante seu funcionamento.

Hipótese Alternativa 1-1: O protocolo de comunicação direta avaliado em ambiente de simulação e sensoriamento em plataforma real não apresentou desempenho equivalente tendo como base as seguintes métricas: Tempo de vida da rede e quantidade de dados transmitidos durante seu funcionamento.

4.2 Experimento

Selecionar a técnica de avaliação e um conjunto de métricas são dois passos fundamentais para realizar avaliação de sistemas computacionais. Segundo Jain [31], técnicas para avaliação de desempenho de sistemas computacionais compreendem: modelagem analítica, simulação e mensuração. Descrevendo de forma sucinta cada uma das técnicas, a modelagem analítica se apoia em equações matemáticas para descrever o desempenho do sistema, simulação utiliza softwares que "imitam"o comportamento de um sistema na tentativa de aproximar-se do desempenho real do sistema, e mensuração visa avaliar o desempenho de sistemas reais. No âmbito das RSSF, é comum a adoção das técnicas supracitadas. Entretanto, a simulação é recorrentemente adotada pela comunidade, em relação à mensuração, devido às questões como custo e tempo, e em relação à modelagem analítica, devido à maior precisão da avaliação [31].

A investigação realizada neste trabalho de dissertação é de caráter experimental, que compreende duas das técnicas de avaliação devido ao objetivo de pesquisa estabelecido, avaliar a confiabilidade da ferramenta de simulação em comparação ao seu *testbed* (*i.e.* Simulação e Mensuração). Para isso, a metodologia do trabalho contempla uma análise comparativa de dois protocolos em RSSF, um de comunicação direta e outro de agrupamento - LEACH, implementados em *testbed* e ferramenta de simulação. O objetivo dessa comparação é avaliar a confiabilidade da ferramenta de simulação em RSSF, Castalia, a partir do funcionamento de uma plataforma experimental de sensoriamento, SunSPOT.

Com base nas métricas de avaliação definidas, um *set-up* experimental foi montado com a finalidade de estabelecer um cenário replicável em ambos ambientes de implantação. Para isolar algum viés dos protocolos utilizados no experimento em relação ao desempenho obtido nos diferentes ambientes de implantação, duas abordagens distintas de protocolos de comunicação em RSSF foram adotadas na avaliação.

Relativo à escolha da ferramenta de simulação, algumas das motivações na adoção do Castalia diz respeito ao alto índice de aceitabilidade em trabalhos científicos, a existência de modelos pré-definidos no simulador que compreendem o *hardware* do sensor SunSPOT, como exemplo do transceptor CC2420, o modelo de descarga de bateria recarregável não ideal, provido pela extensão Green Castalia [15], e a implementação do LEACH para o Cas-

talia disponibilizada pelo grupo de pesquisa GERCOM [1]. Essa implementação do LEACH já foi utilizada em outros trabalhos científicos como pode ser verificado em [24, 43, 48].

No tocante à adoção da plataforma, o sensor SunSPOT além de ter uma boa especificação de *hardware* (*e.g.* processamento, memória RAM e memória de armazenamento), a linguagem Java, contemplada pelo *kit* de desenvolvimento, é bem utilizada em diferentes tipos de aplicação, assim facilitando a portabilidade do código gerado para outras tecnologias.

4.3 Topologia do Experimento

A fim de minimizar a complexidade envolvida na montagem do cenário experimental, como também minimizar alguns efeitos conhecidos e indesejáveis (*e.g.* multipercurso, desvanecimento e alta razão SNR (*Signal Noise Ratio*)), a topologia do *testbed* foi montada de forma simples, em ambiente de dimensões reduzidas, e que possibilitasse mitigar as nuances desses efeitos sob o experimento. Para tanto, a topologia adotada foi a estrela. Os protocolos utilizados no experimento não foram implementados para lidar com múltiplos saltos para simplificar o experimento. É importante ressaltar que caso essas variáveis fossem desprezadas, o tráfego de dados da rede nas diferentes replicações do experimento poderiam mudar, assim afetando o experimento. A Figura 4.1 ilustra a topologia da rede:

No simulador Castalia, topologia é considerada parâmetro de nível superior[2], assim como tempo de simulação, quantidade de nós e área do campo de implantação. Para criar a topologia da rede manualmente no Castalia, faz-se necessário inserir a coordenada cartesiana tridimensional de cada nó. Caso o valor do eixo Z seja omitido, isso significa que todos os sensores estão em um mesmo nível em relação ao eixo. O valores encontrados na Figura 4.1 representam os parâmetros da topologia na ferramenta de simulação e no *testbed*.

[1] http://www.gercom.ufpa.br/

[2] Parâmetros nativos da ferramenta OMNET++.

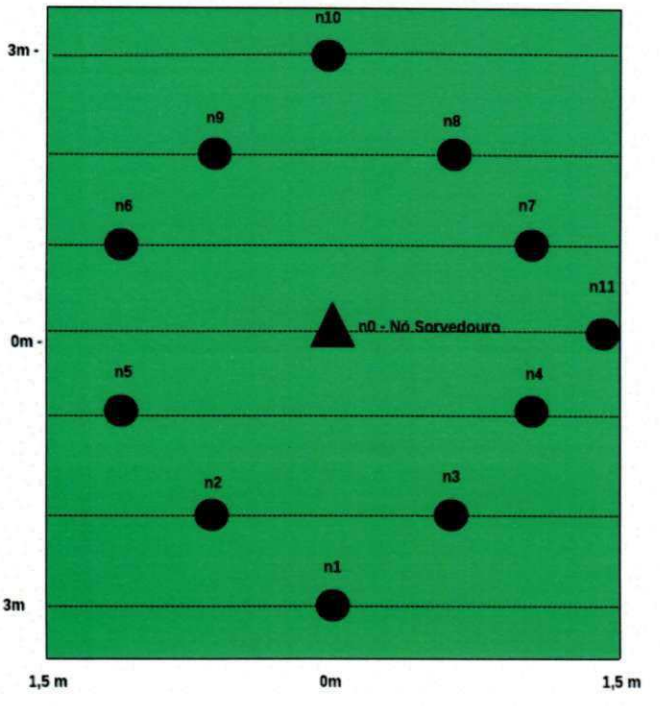

Figura 4.1: Topologia da rede adotada no experimento.

4.4 Parâmetros Ajustados no Simulador com base no *Testbed*

Para melhor endereçar os ajustes das variáveis da ferramenta de simulação, relativo aos parâmetros dos sensores, foi necessário agrupar as variáveis compreendidas nos respectivos modelos implementados no Castalia/GreenCastalia, como ilustrado na Figura 4.2.

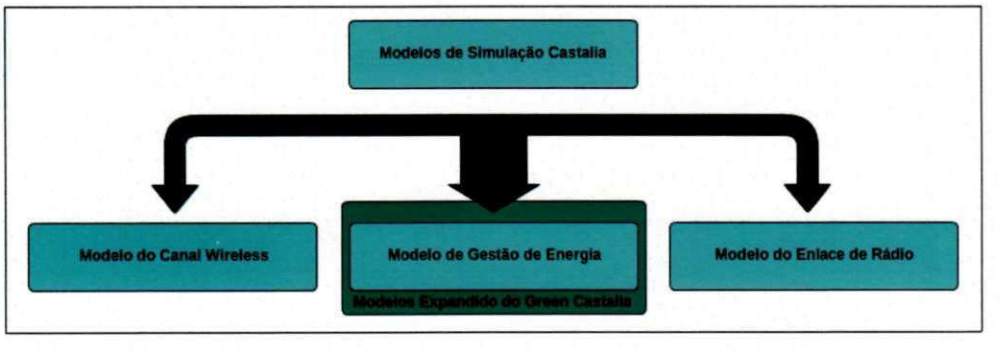

Figura 4.2: Visualização do modelos em blocos no ajuste da experimentação - Castalia.

4.4.1 Parâmetros do Canal *Wireless*

Iniciando pelo Canal *Wireless* do Castalia, as variáveis presentes na Tabela 4.1 foram ajustadas a fim de aproximar as nuances relativas ao ambiente de realização do *set-up* experimental do *testbed*:

Tabela 4.1: Parâmetros do canal sem fio - Castalia.

Parâmetros do Canal Sem Fio - Castalia	
Sigma σ	0
Sigma Bidirecional σ_b	0
Valor do Expoente de Propagação do Canal γ	2,4
Modelo de Propagação do Canal	*Log-Distance*

Os valores de σ e σ_b são parâmetros de qualidade do link. No caso em que $\sigma = 0$, assume-se que a qualidade do canal é fixa dado uma distância d_n invariável. Em relação ao valor de $\sigma_b = 0$, implica em equivalência da qualidade do canal entre dois nós. Assim, a qualidade do link entre A e B é a mesma entre B e A.

O modelo de propagação *Log-Distance* é o modelo adotado no Castalia para prever as perdas de percurso do canal. Em outras palavras, esse modelo de propagação representa as perdas do canal comum em ambientes densamente povoados e estruturas internas de edifícios. Logo, com o valor de $\gamma = 2,4$, referenciado com base na Tabela 4.2 de valores empíricos para propagação interna fornecida por [47], a configuração da propagação prevê um modelo probabilístico de perda equivalente a um escritório.

4.4.2 Parâmetros do Modelo de Rádio

O simulador Castalia traz três transceptores modelados em seu pacote de simulação, o BANRadio, CC2420 e CC1000. Cada um dos transceptores é representado por um arquivo próprio de configuração que compreende parâmetros como taxa de transmissão, modulação, sensibilidade, consumo de energia, etc. Os parâmetros do transceptor utilizado pelo SunSPOT, CC2420, foram consultados no *datasheet* fornecido pelo fabricante com a finalidade de checar os valores atribuídos ao arquivo de configuração. A Tabela 4.3 exibe os valores do arquivo de configuração:

Tabela 4.2: Valores dos coeficientes empíricos para propagação interna.

Edifício Tipo	Frequência de Transmissão	γ	σ [DB]
Vácuo, espaço infinito		2,0	0
Loja de Varejo	914 MHz	2,2	8,7
Bomboneria	914 MHz	1,8	5,2
Escritório com partição do disco	1,5 GHz	3,0	7
Escritório com partição suave	900 MHz/2.4 GHz	2,4	9,6
Escritório com partição suave	1,9 GHz	2,6	14,1
Têxteis ou Química	1,3 GHz	2,0	3,0
Têxteis ou Química	4 GHz	2,1	7,0;9,7
Escritório	60 GHz	2,2	3,92
Comercial	60 GHz	1,7	7,9

Tabela 4.3: Parâmetros de recepção do transceptor CC2420.

Parâmetros	Taxa de Dados (Kbps)	Modulação	Bit por Símbolo	Potência de TX (dBm)	Sensibilidade (dBm)	Potência Consum. (mW)
Valores	250	PSK	4	0	-95	62

4.4.3 Parâmetros do Modelo de Bateria

O modelo de descarga adotado no experimento é proveniente de uma extensão do Castalia denominada GreenCastalia. Essa extensão viabiliza a possibilidade do uso de diferentes fontes de energia (*e.g.* baterias recarregáveis, capacitores, painéis solares), paralelamente ou não, com base em um modelo de descarga não linear (*i.e.* não ideal). A adoção da extensão nesse experimento foi fundamental, pois como a avaliação visa verificar a confiabilidade da plataforma de simulação em relação ao funcionamento de um protocolo, pautado em eficiência energética, um modelo de descarga mais refinado se torna imprescindível.

Os parâmetros ajustados de acordo com a especificação da plataforma estão contidos na Tabela 4.4.

Tabela 4.4: Parâmetros de tipo de fonte e descarga.

Parâmetros	Fonte de Energia	Qtd. de Fontes	Tensão Nominal (V)	Capacidade (mAh)	Consumo Base (mW)	Tensão Mínima de Corte (V)
Valores	Bateria Recarregável	1	3,7	770	140	3,2

Os valores referentes ao consumo base dos sensores e a tensão mínima de corte foram

obtidos no manual de especificação e teoria de funcionamento do SunSPOT [6]. Essas informações são fundamentais para equalizar o consumo dos sensores, como também o ponto de parada em relação à descarga da bateria.

4.5 Execução dos Experimentos

4.5.1 Configuração do Simulador e Protocolos

Para garantir a replicabilidade do experimento desta investigação, toda a instrumentação em relação ao software de simulação adotado, parâmetros de funcionamento dos protocolos, sistema operacional e a especificação do PC (*Personal Computer*) são descritos a seguir.

PC

- **Processador:** Intel(R) Core(TM) i7-4790 CPU @ 3.60 GHz
- **Memória:** 16 GB
- **Sistema Operacional:** Ubuntu 12.04 x64

Simulador

- **Nome:** OMNET++
- **Versão:** 4.6
- **Extensão para RSSF:** Castalia
- **Versão da Extensão RSSF:** 3.3
- **Extensão do Módulo de Energia:** GreenCastalia
- **Versão da Extensão do Módulo:** 0.1

4.5.1.1 Protocolo de Comunicação Direta

A plataforma de simulação Castalia foi configurada baseada na implementação do protocolo de comunicação direta no SunSPOT - *broadcast non-beacon* com CSMA/CA simples. A implementação do protocolo TunableMAC já existente no Castalia é a correspondente,

com a inclusão do CSMA-CA no arquivo de configuração. Nos apêndices deste documento, é possível encontrar os *scripts* de configuração usados no Castalia, bem como a implementação do protocolo. A disposição dos nós contempla a topologia supracitada no documento e a comunicação entre os nós ordinários e o nó sorvedouro não permitem múltiplos saltos. A Tabela 4.5 apresenta os parâmetros de configuração do protocolo:

Tabela 4.5: Parâmetros protocolo.

Parâmetros	**Protocolo MAC**	**Protocolo de Roteamento**	**Taxa de Entrega (pkt/sec)**	**Data Payload**
Valores	TunableMAC	ByPassRouting	1	100

4.5.1.2 Protocolo LEACH

A implementação do protocolo LEACH para o simulador Castalia engloba alguns parâmetros de configuração como tamanho dos pacotes, taxa de envio, tamanho do *slot* de tempo, percentual de coordenadores por rodada, etc. A seguir esses parâmetros são descritos na Tabelas 4.6 e 4.7.

Tabela 4.6: Tamanho dos pacotes definidos no experimento.

Tipos de Pacotes	**Pacote ADV**	**Pacote JOIN**	**Pacote TDMA**	**Pacote Dados**	**Pacote Broadcast**
Tamanho em byte	100 Bytes	100 Bytes	100 Bytes	100 Bytes	100 Bytes

Tabela 4.7: Parâmetros do LEACH.

Parâmetros	**% de Coordenadores**	**Tempo por Slot**	**Tempo da Rodada**	**NetBufferSize**	**Nome da Aplicação**	**Taxa de Entrega (pkt/sec)**
Valores	10%	6s	67s	1000	ThroughputTest	1

A implementação do LEACH para simulador OMNET++/Castalia, empregada nesta investigação, foi disponibilizada, como discutido em seções anteriores, pelo grupo de pesquisa GERCOM [3]. Como pode-se verificar em [24, 43, 48], essa mesma implementação fornecida pelo grupo também tem subsidiado outras investigações no âmbito das RSSF. Parte do código implementado pelo GERCOM foi portado para a implementação do LEACH no SunSPOT, com os devidos ajustes no acesso ao meio de transmissão para manter o mesmo funcionamento do protocolo na plataforma. Alguns artefatos de código importantes para o experi-

[3] http://gercom.ufpa.br

mento, diagrama UML e *scripts* de configuração do simulador encontram-se no apêndice, bem como todo o empacotamento do experimento está disponível em repositório GitHub [4].

4.5.2 Análise Espectral de Controle

Com a finalidade de verificar a interferência do ruído oriundo do ambiente sob o experimento, uma análise espectral foi conduzida antes da realização do experimento. Para essa atividade, o equipamento *AirView2* foi utilizado para varrer o espectro operacional dos sensores. Por meio dos gráficos gerados pela ferramenta, pode-se visualizar, dentro do espectro de frequência analisado, a potência em dBm dissipada no meio. Assim, foi possível, ao decorrer do tempo da análise, obter a assinatura da energia RF presente no ambiente antes do experimento.

Com base na análise do gráfico representado pela Figura 4.3, em torno da frequência de 2475 MHz (Segmento mais a direita da figura), que corresponde ao canal 25 do padrão IEEE 802.15.4, é possível verificar que não há alocação do canal para transmissão por outros dispositivos. Outra razão por utilizar o canal 25, além do espectro limpo, está relacionada à restrição de potência estabelecida sob o canal 26 do SunSPOT, em que a potência máxima permitida é de -3 dBm conforme regulação da *Federal Communication Comission* (FCC) [5]. A mesma potência também foi ajustada no ambiente de simulação.

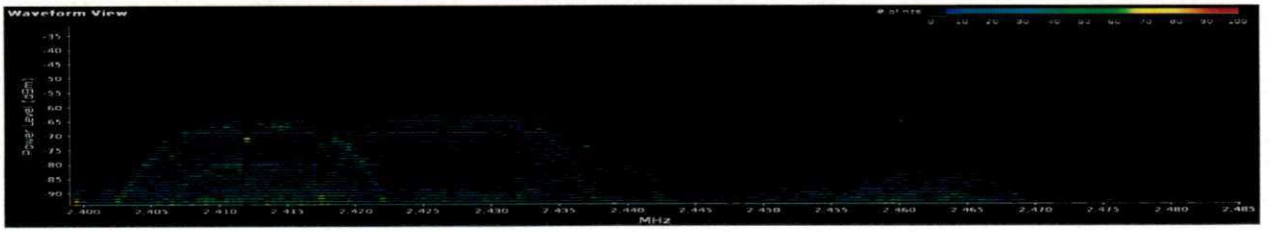

Figura 4.3: Análise espectral antes do experimento.

4.5.3 Adaptação no SunSPOT

Para adequação do SunSPOT em relação ao modelo de sensor compreendido pelo simulador Castalia, uma alteração nos nós foi realizada: a remoção da placa *eDemoboard*.

[4] https://github.com/gugs/leach_masterthesis

Como visto em seções anteriores, a placa *eDemoboard* é responsável pelas interfaces GPIO, como também contém alguns transdutores, por exemplo: sensor de temperatura, sensor de luminosidade, acelerômetro, etc. Os recursos da placa podem gerar um consumo energético adicional diminuindo a autonomia da bateria. Como visto na especificação da plataforma, o consumo da *eDemoboard* é expressivo e pode alcançar valores próximos ao consumo da placa principal a depender dos recursos utilizados. Assim, todos os parâmetros do consumo energético gerado pela placa principal foi ajustado no simulador - consumo base da placa principal do nó sensor.

4.6 Métricas

Nessa seção, as métricas previstas no experimento foram elicitadas com base no suporte oferecido pelos sensores. Em outras palavras, algumas métricas passíveis de serem coletadas neste experimento, como atraso, precisam de implementação adicional demandando tempo extra para realização do trabalho (*e.g.* NTP). Seguem abaixo as métricas compreendidas no experimento e a forma para obtê-las.

- **Tempo de vida da rede:** O tempo de vida da rede será avaliado conforme o último pacote enviado por um nó válido;
- **Tráfego de Rede:** Contador para cada tipo de mensagem deve ser averiguado. Por exemplo, cada tipo de pacote trocado deve incrementar um valor na persistência, na sua respectiva variável, por meio da identificação do tipo de cabeçalho dos pacotes trafegados (*e.g.* pacote de advertência, aceitação e TDMA).

Salientando-se que métricas como atraso não foram adotadas no experimento devido a necessidade de implementação de protocolos de sincronização da rede, causando uma sobrecarga adicional na análise entre os protocolos.

4.6.1 Resumo

Neste capítulo, os problemas norteadores desta pesquisa foram expostos e organizados como problema técnico e de negócio, os quais sustentam as hipóteses estabelecidas no trabalho. Em seguida, todo o experimento é detalhado em diferentes segmentos: topologia,

parâmetros de configuração da ferramenta de simulação, execução do experimento e métricas.

Capítulo 5

Resultados

Os resultados do experimento realizado são apresentados por meio de gráficos e tabelas. Para melhor entendimento dos resultados, a discussão está dividida em duas seções: LEACH e Comunicação Direta. Cada seção discorre acerca das abordagens de implantação dos protocolos contemplados na investigação, *Testbed* e Simulador, na perspectiva das métricas usadas no experimento. Ao final deste capítulo, uma sumarização dos resultados é apresentada.

5.1 LEACH

Inicialmente, a discussão dos resultados é realizada com base no *testbed*, pois os valores obtidos na análise a partir dos sensores reais são tidos como valores de referência para as demais avaliações. A análise do protocolo LEACH, implementado no *testbed*, não considerou os pacotes do tipo "Dados" no experimento, pois a versão do LEACH para o Castalia não trata a fusão dos dados pelo coordenador. Sendo assim, a análise do protocolo se dá acerca da sobrecarga dos pacotes de controle do LEACH na fase de formação de grupos. Os valores exibidos na Tabela 5.1 representam a média das métricas de tempo de vida e tráfego de dados de cada sensor que compõe a RSSF, em três replicações do experimento.

A partir dos dados apresentados na Tabela 5.1, os valores da média, desvio padrão e o intervalo de confiança da RSSF foram calculados e são apresentados na Tabela 5.2. Conforme Tabela 5.2, o intervalo de confiança do tráfego da rede, calculado com 5% de significância, apresentou sobreposição, indicando que nada pode-se afirmar na comparação das amostras.

Tabela 5.1: Média do tempo de vida e tráfego dos pacotes de controle do LEACH no *Testbed* e Simulador.

ID	Testbed		Simulador	
	Duração (min.)	**Tráfego de Dados (KB)**	**Duração (min.)**	**Tráfego de Dados (KB)**
1	827	169,66	941	174,51
2	800	179,26	870	154,62
3	811	171,90	941	193,88
4	806	166,70	934	184,47
5	819	178,51	902	169,53
6	816	167,25	944	184,67
7	801	153,39	940	187,30
8	791	185,25	941	188,31
9	801	166,08	929	184,89
10	817	174,41	941	182,55
11	781	168,55	944	166,80

Essa sobreposição é ilustrada por meio da Figura 5.1.

Para prosseguir com a análise, é importante recapitular as perguntas de pesquisa deste trabalho, que buscam responder questões relacionadas a precisão do simulador no tocante à gestão de energia e ao modelo de descarga dos nós, com base em um *testbed* de sensores reais. Logo, faz-se necessário avaliar, inicialmente, se a quantidade de dados trafegados na

Tabela 5.2: Valores estatísticos do protocolo LEACH.

Variáveis	**Tempo de Vida**		**Tráfego de Dados**	
	Testbed	Simulador	Testebed	Simulador
Média:	806,36	929,73	171,00	179,23
Desvio Padrão:	13,38	23,18	8,46	11,57
Teste Duas Caldas - Tabela Z:	1.96	1,96	1,96	1,96
Lim. Inferior:	798,46	916,02	166,00	172,39
Lim. Superior:	814,27	943,43	176,00	186,07
Coeficiente de Variação:	0,01	0,02	0,04	0,06

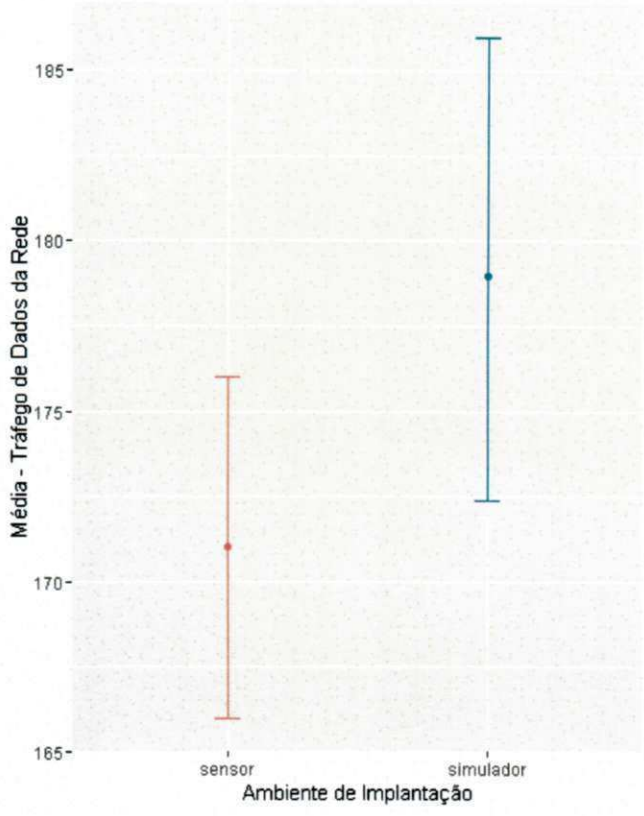

Figura 5.1: Intervalo de confiança do tráfego de dados da rede do protocolo LEACH.

RSSF tem equivalência em ambos os ambientes. A decisão de utilizar dois protocolos de abordagens diferentes teve o objetivo de isolar qualquer viés relativo à implementação do protocolo, por esse motivo duas abordagens distintas foram adotadas. Como os dados de tráfego apresentaram sobreposição no intervalo de confiança, o teste *t-student* foi conduzido a fim de validar a equivalência entre as médias de tráfego entre as amostras com 5% de significância. No entanto, para prosseguimento na condução do teste, as amostras devem tender à normalidade na distribuição dos dados. Com o objetivo de validar esse requisito, preliminarmente, o teste *Shapiro Wilk* foi efetuado para identificar o tipo de distribuição da amostra. Para ambas as amostras, a hipótese de que os dados tendem à normalidade não pôde ser refutada. Com os dados congruentes ao requisito do teste, o *t-student* foi conduzido e o valor-p retornado foi maior que a significância ajustada na comparação das médias (i.e. $valor - p = 0,07256$). Assim, não foi possível refutar a hipótese de que a diferença entre as médias é igual a zero. Em outras palavras, o tráfego da rede, dentro do intervalo de tempo definido em ambos os cenários, foi equivalente.

A partir dessa confirmação, pode-se observar que os valores relativos ao tempo de vida médio nos cenários estabelecidos são diferentes para um tráfego de dados equivalente. Com base na Figura 5.2, é possível verificar que os resultados alcançados pelo simulador são superiores aos do *testbed*. Após verificar o tipo de distribuição das amostras relativo ao tempo de vida, o mesmo procedimento para validar estatisticamente a comparação entre as médias foi conduzido com o teste *t-student* para endossar os resultados interpretados no gráfico. Como previsto, a hipótese nula da igualdade entre as médias foi refutada com $valor - p = 5,79^{-}11$.

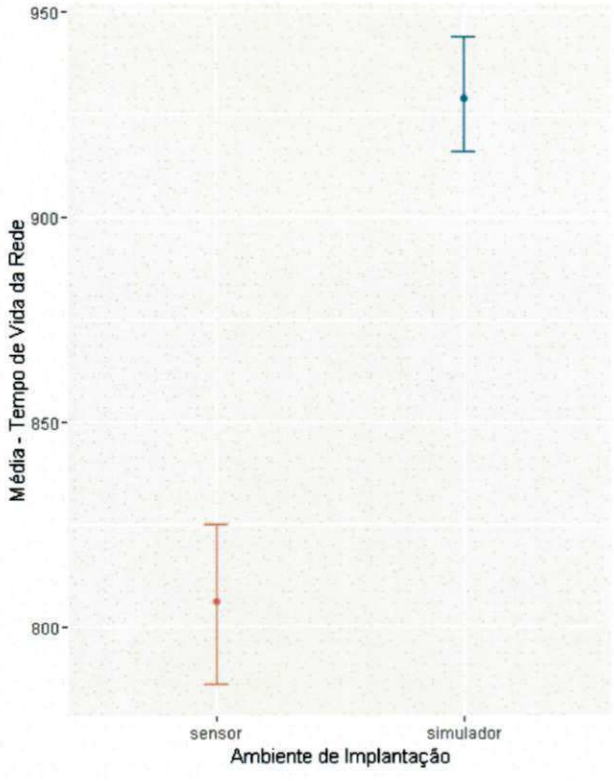

Figura 5.2: Intervalo de confiança do tempo de vida da rede do protocolo LEACH.

A razão do resultado superestimado pode ser justificada pela imprecisão do modelo de descarga da bateria na ferramenta de simulação. Um possível motivo para ocorrência da falha pode estar relacionado ao ajuste da variável *CutOffVoltage*, que define um limiar máximo de descarga da bateria. Durante o experimento foi observado que a mudança de valor dessa variável não impactava no tempo de vida dos nós sensores. Logo, pode-se subentender que o sensor continue funcionando mesmo ultrapassando o limiar de descarga, o que não acontece em sensores reais.

Tabela 5.3: Tempo de vida e tráfego de pacotes do protocolo de Comunicação Direta no *Testbed* e Simulador.

ID	Testbed		Simulador	
	Duração (min)	**Tráfego de Dados (KB)**	**Duração (min)**	**Tráfego de Dados (KB)**
1	795	4543,46	830	4862,50
2	790	4511,77	830	4862,30
3	795	4540,53	830	4862,60
4	812	4296,39	830	4862,50
5	812	4640,23	830	4862,40
6	809	4619,38	830	4862,69
7	783	4473,97	830	4862,69
8	623	4473,97	830	4862,69
9	782	4469,38	830	4862,50
10	783	4486,71	830	4862,50
11	738	4226,41	830	4862,69

5.2 Protocolo de Comunicação Direta

Como mencionado anteriormente, o protocolo de Comunicação direta implementado na plataforma de sensoriamento utiliza o CSMA/CA simples para realizar acesso ao meio. Diferentemente do LEACH, em que há tráfego de controle entre os nós na fase de formação de grupos, o protocolo de Comunicação Direta apenas envia diretamente para o nó sorvedouro, via *broadcast* os pacotes de dados com dados sensoreados. A partir desse panorama, apenas o tráfego de dados e o tempo de vida da rede foram observados. A Tabela 5.3, a seguir, apresenta a média dos dados coletados em três replicações para *Testbed* e Simulador.

Como pode-se observar, os valores do tempo de vida da rede e o tráfego de dados do protocolo, no simulador, são praticamente invariáveis ao longo das três replicações, mesmo com a presença do coeficiente de perda no modelo de propagação. Alguns prováveis motivos que justifiquem essa invariabilidade do tráfego de rede são a curta distância entre os nós, dentro da topologia estrela, e a potência de transmissão ($1mW$) - variáveis compreendidas no modelo de propagação do canal (*log-distance*). A estatística descritiva relativa aos dados

Tabela 5.4: Valores estatísticos do protocolo Comunicação Direta.

Variáveis	**Tempo de Vida**		**Tráfego de Dados**	
	Testbed	Simulador	Testbed	Simulador
Média:	791,45	830	4448,06	4862,55
SD:	20,90	0	126,16	0,13
Teste Duas Caldas - Tabela Z:	1,96	0	1,96	1,96
Lim. Inferior:	779,09	830	4414,33	4862,47
Lim. Superior:	803,81	830	4563,45	4862,63
Coeficiente de Variação	0,02	0	0,02	0

em questão é apresentada na Tabela 5.4, a seguir.

O resultado relacionado ao tempo de vida da rede no simulador apresentou um comportamento de descarga homogêneo e sem variação entre os sensores, diferentemente dos resultados oriundos do *testbed*. Da mesma maneira, a taxa de tráfego da RSSF, no simulador, também apresentou um desvio padrão muito abaixo quando comparado ao *testbed*. As Figuras 5.3 e 5.4 ilustram as diferenças entre os intervalos de confiança dos dados oriundos do *testbed* e simulador. Na perspectiva do *testbed*, o desvio padrão no tempo de vida da rede é esperado devido à vários fatores, mas, principalmente, pela diferença de ciclos de carga da bateria dos sensores.

A partir do panorama apresentado na análise de resultados, uma imprecisão emerge gerando questionamento entre os diferentes cenários encontrados. É notório a diferença entre o tráfego de dados gerado pelo os pacotes de controle do protocolo LEACH e o tráfego de dados ordinário do protocolo de comunicação direta, tendo em vista o tempo de vida da rede. Em ambos os ambientes, *testbed* e simulador, é possível notar que o LEACH foi ligeiramente superior ao protocolo de comunicação direta. Todavia, o tráfego de dados gerados pelo LEACH foi bem inferior. Esta questão pode estar relacionada ao consumo de processamento necessário para manter o engenho do LEACH em atividade. Outra possibilidade que pode ter afetado o tempo de vida da rede está associada ao relaxamento de carga sob as baterias [13], no período de inatividade do transceptor, e o período de *sleep* dos nós - procedimentos compreendidos a partir do TDMA.

Para avaliar a precisão do simulador no tocante ao tempo de vida da RSSF, com relação

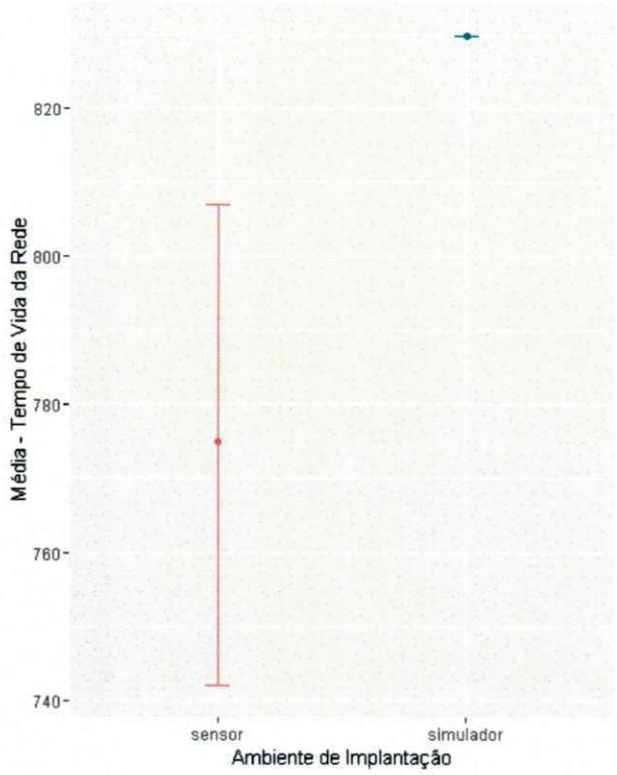

Figura 5.3: Intervalo de confiança do tempo de vida da rede do protocolo Comunicação Direta.

ao protocolo de Comunicação Direta, faz-se necessário comparar a carga de tráfego de rede gerada durante o experimento. Ambas as amostras foram submetidas ao teste *Shapiro Wilk*. Os resultados do teste apontaram que as amostras tendem a uma distribuição normal satisfazendo os requisitos do teste *t-student*. Na comparação das médias, com 5% de significância, o teste retornou um valor-p de $1,908^{-}06$, assim sendo foi possível refutar a hipótese nula de que há igualdade entre as médias ou equivalência do tráfego de rede. O mesmo cenário se repete na avaliação do tempo de vida da rede. Logo, não é possível afirmar que houve equivalência no tráfego no cenário que trata do protocolo de Comunicação Direta, bem como comparar o comportamento de descarga entre *testbed* e simulador.

5.2.1 Análise Complementar

Complementarmente à análise realizada, algumas observações na perspectiva de métricas como *First Node Die* (FND) e *Last Node Die* (LND) trazem consigo informações que

reforçam o que foi discutido previamente. Os dados apresentados nas Tabelas 5.5 e 5.6 foram provenientes das replicações mencionadas na seção anterior. Como pode-se observar, o desvio padrão do primeiro nó morto do protocolo LEACH, nos experimentos conduzidos, foi menor quando comparado ao desvio do outro protocolo. Essa informação reforça a tendência de recuperação da bateria no momento de relaxamento propiciado pelo engenho do LEACH - o tempo de *sleep* dos sensores. Todavia, é necessário uma quantidade maior de replicações do experimento para a realização de uma análise mais detalhada.

Tabela 5.5: Dados do FND e LND das unidades experimentais - LEACH.

Variáveis	**Testbed**			**Simulador**		
	FND	**LND**	**Diferença**	**FND**	**LND**	**Diferença**
Média	735	840	104,66	870	944	74
SD	0,57	1	0,57	0	0	0

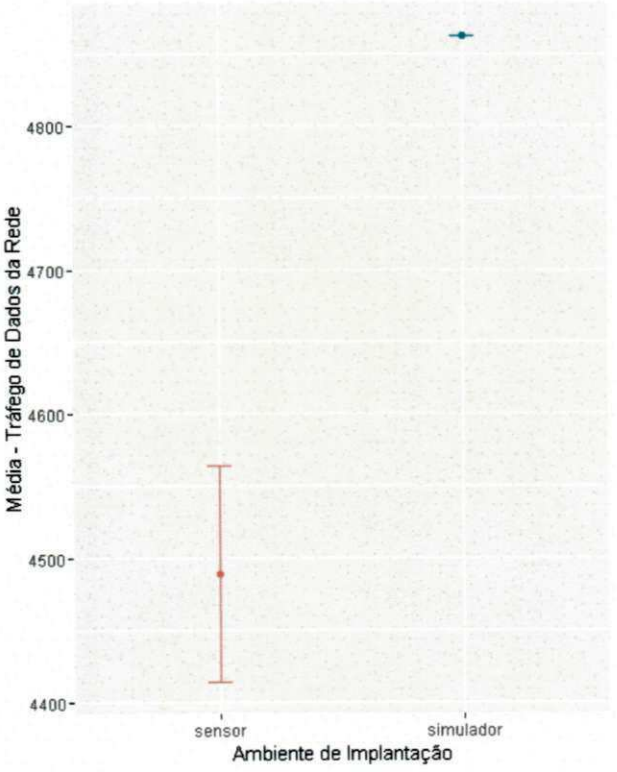

Figura 5.4: Intervalo de confiança do tráfego de dados da rede do protocolo Comunicação Direta.

Em contraste às falhas na estimativa de tempo de vida acerca do protocolo LEACH, pode-se notar que o FND no cenário de simulação está acima do LND oriundo do *testbed*. Esse mesmo comportamento ocorre no tocante ao protocolo Comunicação Direta, todavia não há diferença entre FND e LND do simulador.

Tabela 5.6: Dados do FND e LND das unidades experimentais - Comunicação Direta.

Variáveis	**Testbed**			**Simulador**		
	FND	**LND**	**Diferença**	**FND**	**LND**	**Diferença**
Média	725	817	92	830	830	0
SD	24	10	34	0	0	0

Um outro aspecto importante acerca dos achados deste trabalho faz menção ao consumo de energia do módulo de comunicação. Segundo [41], o módulo de comunicação do sensor é o componente que mais drena energia do nó. Ainda o referido trabalho destaca alguns fatores utilizados na avaliação da eficiência energética de protocolos em RSSF, tais como: distância entre os nós e quantidade de dados transmitidos. Esses fatores são apontados como elementos que influenciam sob o tempo de vida da rede. Ainda como análise complementar, por meio dos dados exibidos na Tabela 5.7, pode-se evidenciar que o tempo de vida do sensor e simulador, no tocante ao protocolo de comunicação direta, permanecem invariantes mesmo com carga de tráfego completamente distintas, o que contraria referências, conforme [41]. Um estudo abrangendo outros protocolos e/ou plataforma se faz necessário para maiores constatações.

Tabela 5.7: Comparação do tráfego e tempo de vida da rede - Comunicação direta.

Variáveis	**Testbed**		**Simulador**	
Tráfego	624840	43556	697966	49794
Tempo de Vida	761	745	830	830

5.3 Sumarização

Apoiado na análise e discussão dos resultados, pode-se verificar que:

- O simulador Castalia e sua extensão, com base no *testbed* de referência, não apresentou confiabilidade nos resultados do tempo de vida da rede referente ao protocolo LEACH. Sendo assim, a Hipótese Nula 0-0 pôde ser refutada;
- O simulador Castalia e sua extensão, com base no *testbed* de referência, não apresentou confiabilidade nos resultados do tempo de vida e tráfego da rede referente ao protocolo de Comunicação Direta. Sendo assim, a Hipótese Nula 0-0 pôde ser refutada;
- A variância do tempo de vida dos sensores, oriunda das baterias (i.e. diferentes ciclos de cargas) relativo ao protocolo de Comunicação Direta, pôde ser observada no *testbed*, entretanto não foi evidenciado o mesmo comportamento na ferramenta de simulação;
- A diferença do tempo de vida entre os resultados obtidos do protocolo LEACH e Comunicação Direta, entre os respectivos cenários, foi pouco expressiva, mesmo o LEACH realizando a desativação do rádio durante o período de *sleep* e a comunicação direta mantendo o rádio sempre ativo. Entretanto, o LEACH necessita de maior carga de processamento para manter o seu engenho funcionando, o que equiparou o consumo dos dois protocolos;
- Nos experimentos conduzidos, não é observada mudança significativa no tempo de vida dos nós, mesmo mudando consideravelmente a quantidade de dados transmitidos, o que demanda maior uso do rádio. Isso vai de encontro à maioria das pesquisas realizadas anteriormente, e demanda estudos mais aprofundados para a sua confirmação.

5.3.1 Resumo

Neste capítulo, os achados desta pesquisa foram apresentados e discutidos. Como previsto e detalhado no roteiro experimental, foi conduzida uma comparação entre os ambientes de implantação, ferramenta de simulação e plataforma de sensoriamento. Entretanto, alguns

resultados obtidos além dos cenários estabelecidos no experimento foram discorridos na seção análise complementar. Ao final, todos os achados da pesquisa foram sumarizados.

Capítulo 6

Considerações Finais

A partir da discussão dos resultados, pôde-se verificar que, com base na amostra coletada neste trabalho, o modelo de drenagem da bateria ainda continua apresentando resultados imprecisos, mesmo sendo considerado um modelo realístico de descarga. Entretanto, faz-se necessário avaliar os demais fatores que podem contribuir para uma drenagem de bateria imprecisa. Para elicitar resultados mais expressivos nesse âmbito, é necessário a realização de experimentos com maior diversidade de fatores, bem como uma quantidade de replicações superior para agregar maior valor estatístico.

Durante a realização deste trabalho, muitos desafios emergiram no decorrer da implementação do protocolo LEACH (e.g. tempo de experimentação, excitação dos sensores, armazenamento de informações, falha no estimador de bateria, sincronização da RSSF, etc.). Algumas das principais dificuldades são listadas a seguir.

- **Tempo de experimentação:** a longa duração de uma unidade experimental do *testbed* (i.e. tempo de descarga, carregamento das baterias e coleta de métricas) impossibilitou a coleta de uma quantidade de dados expressiva.

- **Falha do estimador de bateria:** a falha encontrada no estimador de bateria dos sensores, identificada em um trabalho preliminar por Martins G. [39], também limitou o estudo por restringir a adoção de protocolos que utilizam a leitura da bateria no engenho de eleição do coordenador.

- **Sincronização da rede:** entre todos os problemas supracitados, o obstáculo mais oneroso para esta investigação foi a sincronização da RSSF. Como sabido no âmbito das

RSSF, um dos maiores desafios acerca da implementação de qualquer protocolo com engenho distribuído é a sincronização dos nós da rede. Uma solução conhecida para resolver problemas de sincronização em RSSF é a adoção de protocolo de sincronização de tempo, tal como *Network Timer Protocol* (NTP). Outra abordagem para a resolução desse problema é a utilização de interrupção gerada por *hardware* baseado em temporizador de precisão, solução essa adotada e suportada pelo sensor SunSPOT.

O problema de sincronização do LEACH foi resolvido por meio do relógio de alta precisão disponibilizado pela interface IAT91_TC da *Application Program Interface* (API) do sensor. A placa eSPOT contém dois contadores de tempo AT91, que fazem parte do *chip* SoC ARM920T. Cada contador AT91 inclui três canais de contadores de tempo idênticos com 16-bit, totalizando seis contadores. Quatro contadores estão disponíveis para aplicações e dois estão reservados para uso do sistema. Logo, um dos contadores da plataforma foi usado para realizar o *reset* parcial da aplicação dentro de um período de tempo estimado.

Por fim, parte da sincronização dos ajustes do *timeout* das fases de formação de grupo e a consideração dos possíveis "atrasos", relativos à distância dos nós na topologia adotada, foram efetuados com base empírica.

6.1 Contribuições

As principais contribuições da pesquisa são detalhadas a seguir:

- Investigação inicial da comunicação dos sensores SunSPOT e do estimador de bateria publicada em conferência [39].
- Implementação do protocolo LEACH e comunicação direta, em plataforma real, disponibilizada *on-line* em `https://github.com/gugs/leach_masterthesis`.
- Empacotamento dos *scripts* de configuração, assegurando a replicabilidade do experimento.
- Análise do comportamento da plataforma experimental SunSPOT enquanto às expectativas na implementação do protocolo.

- Contribuição com a comunidade científica e com os desenvolvedores de Simuladores em RSSF sobre os detalhes do experimento e os resultados obtidos relativo à confiabilidade do simulador utilizado nesta pesquisa.

Além das contribuições apresentadas anteriormente, atualmente um artigo encontra-se em produção para submissão em periódicos e eventos especializados na área.

6.2 Ameaças à Validade

Todos os resultados obtidos nesta pesquisa se limitam aos cenários experimentais especificados, como também aos protocolos e plataforma adotada no experimento. Para realizar maiores inferências sobre os achados, faz-se necessário aumentar a quantidade de replicações do experimento, bem como utilizar outras ferramentas de simulação e plataformas de sensoriamento.

6.3 Sugestões Para Trabalhos Futuros

Algumas possíveis extensões deste trabalho podem ser caracterizadas como intenções de trabalhos futuros. A seguir, seguem algumas das intenções de continuidade desta investigação:

- Reprodução deste experimento com uma quantidade maior de replicações.
- Análise de um conjunto de fatores relativos à RSSF (i.e. protocolo, tráfego de dados, autonomia) em um *design* fatorial.
- Correção do componente estimador de bateria, ou desenvolvimento de uma nova interface para tal propósito.
- Implementação e avaliação de versões mais refinadas do LEACH que utilizam a estimação da bateria no processo de eleição do coordenador.

Bibliografia

[1] Cisrt in an operational context. http://www.jhuapl.edu/techdigest/TD/td2103/salamacha.pdf. Acessado em 2016-04-04.

[2] Construindo cidades inteligentes da instrumentação dos ambientes ao desenvolvimento das aplicações - (cia^2). http://www.nr2.ufpr.br/~cia2/. Acessado em 2015-07-17.

[3] Omnet++ - simulation models. https://omnetpp.org/models/. Acessado em 2016-07-30.

[4] Omnet++ documentation - user manual. https://omnetpp.org/doc/omnetpp/manual/usman.html. Acessado em 2016-04-04.

[5] Oracle sunspot programmer manual. http://www.sunspotdev.org/docs/Yellow/SunSPOT-Programmers-Manual.pdf. Acessado em 2015-08-12.

[6] Oracle sunspot theory of operation. http://www.sunspotdev.org/docs/Yellow/SunSPOT-TheoryOfOperation.pdf. Acessado em 2015-08-12.

[7] Smart santander testbed santander facility. http://www.smartsantander.eu/index.php. Acessado em 2015-10-12.

[8] Sunspot espots processor- atmel processor datasheet. https://www.embeddedarm.com/documentation/third-party/atmel_at91sam9g20_datasheet_may2010.pdf. Acessado em 2016-04-04.

[9] Teleco website. http://www.teleco.com.br/tutoriais/tutorialrssf/pagina_3.asp. Acessado em 2016-04-04.

[10] Ezzati Abdellah, SAID Benalla, A Beni Hssane, and Moulay Lahcen Hasnaoui. Advanced low energy adaptive clustering hierarchy. *IJCSE) International Journal on Computer Science and Engineering*, 2(07):2491–2497, 2010.

[11] Cecilia M. Agarwal, Pankaj K. and Procopiuc. Exact and approximation algorithms for clustering. In *Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '98, pages 658–667, Philadelphia, PA, USA, 1998. Society for Industrial and Applied Mathematics.

[12] Jamal N Al-Karaki and Ahmed E Kamal. Routing techniques in wireless sensor networks: a survey. *Wireless communications, IEEE*, 11(6):6–28, 2004.

[13] José Athayde. Análise de Protocolos de Roteamento Unicast em Redes Ad Hoc Móveis Baseada em um Modelo Realístico de Bateria. Master's thesis, Universidade Federal de Campina Grande, Campina Grande, Paraíba, Brasil, 2011.

[14] Richard Beckwith, Dan Teibel, and Pat Bowen. Report from the field: results from an agricultural wireless sensor network. In *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pages 471–478. IEEE, 2004.

[15] David Benedetti, Chiara Petrioli, and Dora Spenza. Greencastalia: an energy-harvesting-enabled framework for the castalia simulator. In *Proceedings of the 1st International Workshop on Energy Neutral Sensing Systems*, page 7. ACM, 2013.

[16] Lorenzo Bergamini, Carlo Crociani, Andrea Vitaletti, and Michele Nati. Validation of wsn simulators through a comparison with a real testbed. In *Proceedings of the 7th ACM workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, pages 103–104. ACM, 2010.

[17] Athanassios Boulis. Castalia user manual. *Online: http://castalia.npc.nicta.com.au/pdfs/Castalia-User Manual.pdf*, 2009.

[18] Anantha Chandrakasan, Rajeevain Amirtharajah, SeongHwan Cho, James Goodman, Gangadhar Konduri, Joanna Kulik, Wendi Rabiner, and Alice Wang. Design considerations for distributed microsensor systems. In *Custom Integrated Circuits, 1999. Proceedings of the IEEE 1999*, pages 279–286. IEEE, 1999.

[19] S.K. Chaurasiya, T. Pal, and S.D. Bit. An enhanced energy-efficient protocol with static clustering for wsn. In *Information Networking (ICOIN), 2011 International Conference on*, pages 58–63, Jan 2011.

[20] Ugo Maria Colesanti, Carlo Crociani, and Andrea Vitaletti. On the accuracy of omnet++ in the wireless sensornetworks domain: simulation vs. testbed. In *Proceedings of the 4th ACM workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*, pages 25–31. ACM, 2007.

[21] A. da Silva Rego, J. Celestino, A. dos Santos, E.C. Cerqueira, A. Patel, and M. Taghavi. Bee-c: A bio-inspired energy efficient cluster-based algorithm for data continuous dissemination in wireless sensor networks. In *Networks (ICON), 2012 18th IEEE International Conference on*, pages 405–410, Dec 2012.

[22] Antoniel da Silva Rêgo. BEE-C: UM ALGORITMO DE ROTEAMENTO BIO-INSPIRADO PARA ECONOMIA DE ENERGIA EM REDES DE SENSORES SEM FIO. Master's thesis, Universidade Estadual do Ceará, Fortaleza, Ceará, Brasil, 2011.

[23] Waltenegus W Dargie and Christian Poellabauer. *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons, 2010.

[24] Denis do Rosário, Rodrigo Costa, Helder Paraense, Kássio Machado, Eduardo Cerqueira, and Torsten Braun. A smart multi-hop hierarchical routing protocol for efficient video communication over wireless multimedia sensor networks. In *2012 IEEE International Conference on Communications (ICC)*, pages 6530–6534. IEEE, 2012.

[25] M.J. Dong, K.G. Yung, and W.J. Kaiser. Low power signal processing architectures for network microsensors. In *Low Power Electronics and Design, 1997. Proceedings., 1997 International Symposium on*, pages 173–177, Aug 1997.

[26] Ruan D Gomes, Marcelo S Alencar, Iguatemi E Fonseca, and Abel C Lima Filho. Desafios de redes de sensores sem fio industriais. *Revista de Tecnologia da Informação e Comunicação*, 4:1–12, 2014.

[27] Magdalena J. Grobler, Henri-Jean Marais, and Joubert G. J. Krige. A wireless sensor

network testbed for the evaluation of energy-aware routing schemes. *International Journal of Distributed Sensor Networks, 2014*, 2014.

[28] V. C. Gungor and G. P. Hancke. Industrial wireless sensor networks: Challenges, design principles, and technical approaches. *IEEE Transactions on Industrial Electronics*, 56(10):4258–4265, Oct 2009.

[29] W.B. Heinzelman, A.P. Chandrakasan, and H. Balakrishnan. An application-specific protocol architecture for wireless microsensor networks. *Wireless Communications, IEEE Transactions on*, 1(4):660–670, Oct 2002.

[30] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, pages 10 pp. vol.2–, Jan 2000.

[31] Raj Jain. *The art of computer systems performance analysis - techniques for experimental design, measurement, simulation, and modeling.* Wiley professional computing. Wiley, 1991.

[32] Banks Jerry. *Discrete-event system simulation.* Pearson Education India, 2009.

[33] William Kao. Sensor devices and sensor network applications for the smart grid/smart cities. Presented at SensorCon 2012, Santa Clara, CA, USA, 2012.

[34] Shahbaz Khan, Bilal Aziz, Sundas Najeeb, Aziz Ahmed, Muhammad Usman, and Sadiq Ullah. Reliability of network simulators and simulation based research. In *PIMRC*, pages 180–185, 2013.

[35] G Santhosh Kumar, Vinu Paul, and K Poulose Jacob. Mobility metric based leach-mobile protocol. In *Advanced Computing and Communications, 2008. ADCOM 2008. 16th International Conference on*, pages 248–253. IEEE, 2008.

[36] Vinay Kumar, Sanjeev Jain, Sudarshan Tiwari, et al. Energy efficient clustering algorithms in wireless sensor networks: A survey. *IJCSI International Journal of Computer Science Issues*, 8(5), 2011.

[37] Xuemei Li, Yuyan Deng, and Lixing Ding. Study on precision agriculture monitoring framework based on wsn. In *Anti-counterfeiting, Security and Identification, 2008. ASID 2008. 2nd International Conference on*, pages 182–185. IEEE, 2008.

[38] Xuxun Liu. A survey on clustering routing protocols in wireless sensor networks. *Sensors*, 12(8):11113–11153, 2012.

[39] Gomes R. Costa A. Rocha C. Martins, G. and P. Carlos. Ieee 802.15.4 (zigbee) standard under interference of ieee 802.11: Reliability in smart cities application. In *ICWI - Internation Conference WWW/Internet 2014*, pages 283–290, Oct 2014.

[40] M Mustafa, T Shah, Safdar H Bouk, Syed H Ahmed, and N Javaid. Distributed multiple criteria based clustering scheme for wireless sensor networks.

[41] Nikolaos A Pantazis, Stefanos A Nikolidakis, and Dimitrios D Vergados. Energy-efficient routing protocols in wireless sensor networks: A survey. *IEEE Communications Surveys & Tutorials*, 15(2):551–591, 2013.

[42] D Pediaditakis, SH Mohajerani, and A Boulis. Poster abstract: castalia: the difference of accurate simulation in wsn. In *4th European Conference on Wireless Sensor Networks,(Delft, The Netherlands, 29-31 January 2007)*, 2007.

[43] Jhoanna Rhodette I Pedrasa and Gregorio L Ortiz. Leach-based communication network with a modified sleep protocol. In *Smart Grid Technologies-Asia (ISGT ASIA), 2015 IEEE Innovative*, pages 1–6. IEEE, 2015.

[44] Gilbert E PEREZ and Ivica KOSTANIC. Comparing a real-life wsn platform small network and its opnet modeler model using hypothesis testing. *SYSTEMICS, CYBERNETICS AND INFORMATICS*, 12, 2014.

[45] Hai N Pham, Dimosthenis Pediaditakis, and Athanassios Boulis. From simulation to real deployments in wsn and back. In *World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE International Symposium on a*, pages 1–6. IEEE, 2007.

[46] Marcelo Portela. Diversidade Cooperativa Adaptativa Aplicada a Redes de Sensores sem Fio. Master's thesis, Universidade Federal de Campina Grande, Campina Grande, Paraíba, Brasil, 2009.

[47] Theodory S. Rappaport. *princípios e práticas de comunicação sem fio*. Prentice Hall, 2002.

[48] Denis Rosário, Rodrigo Costa, Helder Paraense, Kássio Machado, Eduardo Cerqueira, Torsten Braun, and Zhongliang Zhao. A hierarchical multi-hop multimedia routing protocol for wireless multimedia sensor networks. *Network Protocols and Algorithms*, 4(4):44–64, 2012.

[49] P. Sasikumar and S. Khara. K-means clustering in wireless sensor networks. In *Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on*, pages 140–144, Nov 2012.

[50] AA Sheta, SAS Abdelwahab, S Elaraby, and MI Mahmoud. Analysis and implementation of optimized energy-based ch selection for wsn using sun spot. In *2015 11th International Computer Engineering Conference (ICENCO)*, pages 44–51. IEEE, 2015.

[51] Shio Kumar Singh, MP Singh, DK Singh, et al. Routing protocols in wireless sensor networks–a survey. *International Journal of Computer Science & Engineering Survey (IJCSES) Vol*, 1:63–83, 2010.

[52] M.P. Sousa, M.S. de Alencar, A. Kumar, and W.T. Araujo Lopes. Scalability in an adaptive cooperative system for wireless sensor networks. In *Ultra Modern Telecommunications Workshops, 2009. ICUMT '09. International Conference on*, pages 1–6, Oct 2009.

[53] John A. Stankovic. Research challenges for wireless sensor networks. *SIGBED Rev.*, 1(2):9–12, July 2004.

[54] Sudhanshu Tyagi and Neeraj Kumar. A systematic review on clustering and routing techniques based upon {LEACH} protocol for wireless sensor networks. *Journal of Network and Computer Applications*, 36(2):623 – 645, 2013.

[55] András Varga et al. The omnet++ discrete event simulation system. In *Proceedings of the European simulation multiconference (ESM'2001)*, volume 9, page 65. sn, 2001.

[56] Elias Weingärtner, Hendrik Vom Lehn, and Klaus Wehrle. A performance comparison of recent network simulators. In *Communications, 2009. ICC'09. IEEE International Conference on*, pages 1–5. IEEE, 2009.

[57] Geoffrey Werner-Allen, Patrick Swieskowski, and Matt Welsh. Motelab: A wireless sensor network testbed. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, page 68. IEEE Press, 2005.

[58] Xiaodong Xian, Weiren Shi, and He Huang. Comparison of omnet++ and other simulator for wsn simulation. In *Industrial Electronics and Applications, 2008. ICIEA 2008. 3rd IEEE Conference on*, pages 1439–1443. IEEE, 2008.

[59] Fan Xiangning and Song Yulin. Improvement on leach protocol of wireless sensor network. In *Sensor Technologies and Applications, 2007. SensorComm 2007. International Conference on*, pages 260–264. IEEE, 2007.

[60] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Computer Networks*, 52(12):2292 – 2330, 2008.

[61] Seong-eun Yoo, Poh Kit Chong, Seong Hoon Kim, and Minh-Long Pham. Verification and validation of the performance of wsn. *International Journal of Distributed Sensor Networks & Hindawi Journal, 2015*, 2015.

Apêndice A

Diagramas UML

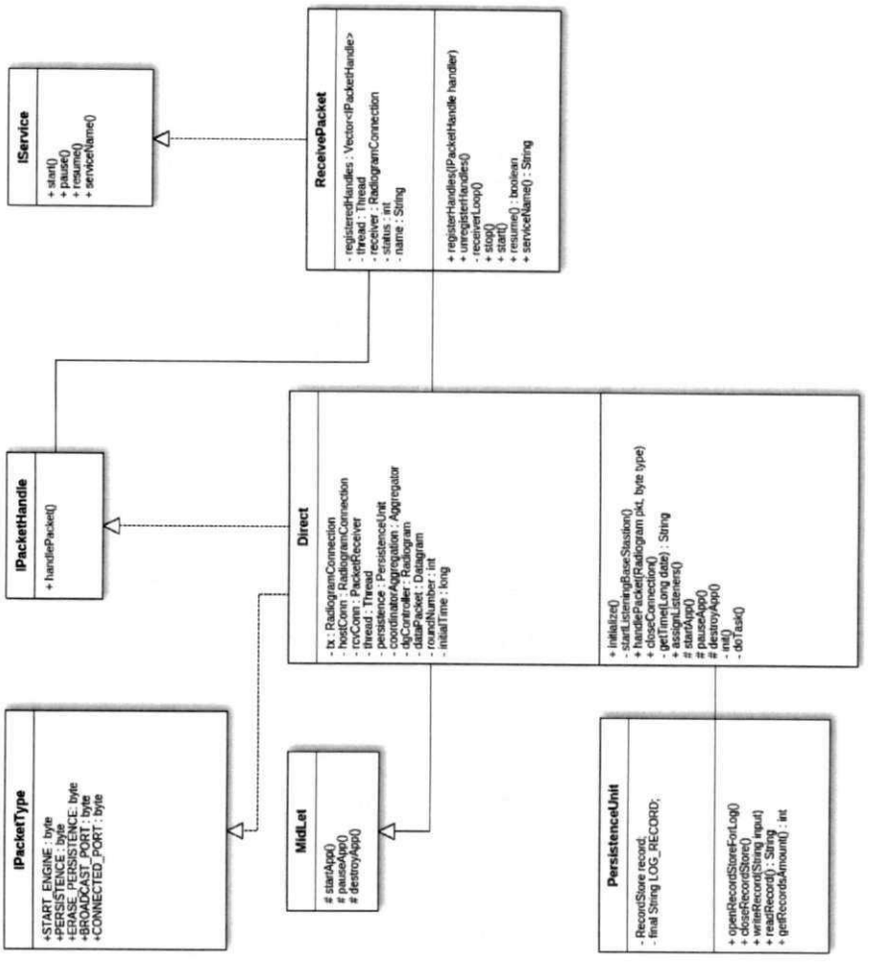

Figura A.1: Diagrama de classe UML - Implementação do Protocolo de Comunicação Direta.

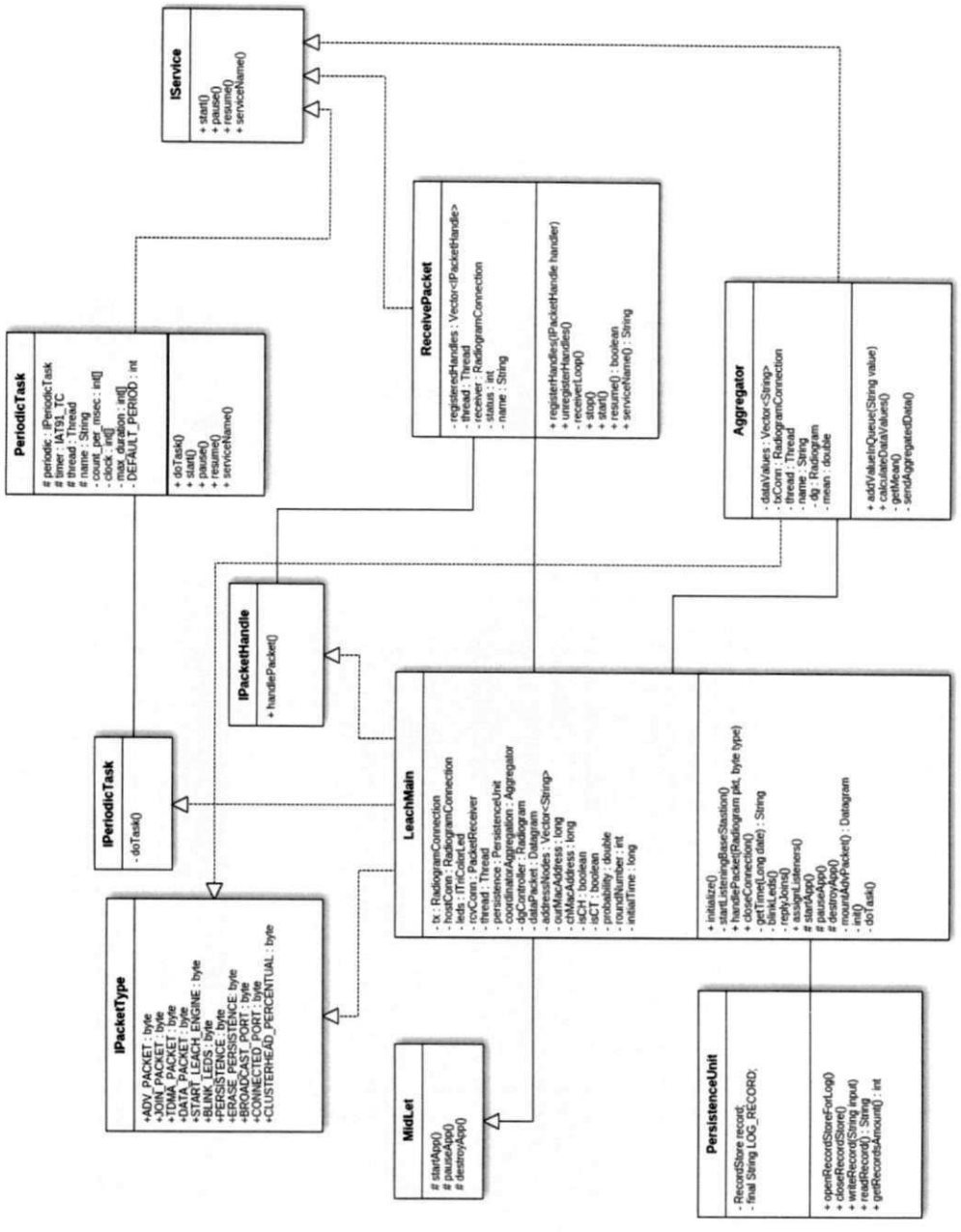

Figura A.2: Diagrama de classe UML - Implementação do LEACH.

Apêndice B

Script de Configuração - Castalia

Código Fonte B.1: Script de configuração do Castalia - Protocolo LEACH.

```
1
2  [General]
3
4  ########################################
5  ## Network - Top-level parameters #####
6  ########################################
7
8  include ../Parameters/Castalia.ini
9  include ../Parameters/MAC/CSMA.ini
10 sim-time-limit = 57000s
11 SN.field_x = 12
12 SN.field_y = 12
13 SN.numNodes = 12
14
15 ########################################
16 ## Nodes Locations - Deployment  ######
17 ########################################
18
19 SN.node[0].xCoor = 3
20 SN.node[0].yCoor = 5
21 SN.node[1].xCoor = 3
22 SN.node[1].yCoor = 0
23 SN.node[2].xCoor = 2
24 SN.node[2].yCoor = 2
25 SN.node[3].xCoor = 4
26 SN.node[3].yCoor = 2
27 SN.node[4].xCoor = 5
28 SN.node[4].yCoor = 4
29 SN.node[5].xCoor = 1
30 SN.node[5].yCoor = 4
31 SN.node[6].xCoor = 1
32 SN.node[6].yCoor = 6
33 SN.node[7].xCoor = 5
34 SN.node[7].yCoor = 6
35 SN.node[8].xCoor = 4
36 SN.node[8].yCoor = 8
37 SN.node[9].xCoor = 2
38 SN.node[9].yCoor = 8
39 SN.node[10].xCoor = 3
40 SN.node[10].yCoor = 10
41 SN.node[11].xCoor = 6
```

```
42  SN.node[11].yCoor = 5
43
44  ####################################
45  ## Traces                          ######
46  ####################################
47
48  SN.wirelessChannel.collectTraceInfo = false
49  SN.node[*].Communication.Radio.collectTraceInfo = false
50  SN.node[*].Communication.MAC.collectTraceInfo = false
51  SN.node[*].Communication.Routing.collectTraceInfo = true
52  SN.node[*].Application.collectTraceInfo = false
53  SN.node[*].SensorManager.collectTraceInfo = false
54  SN.node[*].ResourceManager.collectTraceInfo = false
55
56  ####################################
57  ## EnergyStorage from GreenCastalia ##
58  ####################################
59
60  SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numBatteries = 0
61  SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numRechBatteries = 1
62  SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numSupercaps = 0
63
64  ####################################
65  ## [Rechargeable battery]          ###
66  ## 1 Prismatic Battery 3.7V 770mAh ###
67  ####################################
68
69  SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[0].maxVoltage = 3.7
70  SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[0].mAmpereHour = 720
71
72  ####################################
73  ## [Rechargeable battery] Sink     ###
74  ## 1 Prismatic Battery 3.7V 770mAh ###
75  ####################################
76
77  SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.Batteries[0].maxVoltage = 3.7
78  SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.Batteries[0].mAmpereHour = 1000
79
80  SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.numBatteries = 0
81  SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.numRechBatteries = 1
82  SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.numSupercaps = 0
83
84  ########################################
85  ## [Sensor Power Consumption Baseline] ##
86  ########################################
87
88  SN.node[*].ResourceManager.baselineNodePower = 104 #mW
89  SN.node[*].ResourceManager.cutOffVoltage = 3.3 #V
90
91
92
93  ########################
94  ## MAC                 #####
95  ########################
96
97  ##---CSMA-CA File ---####
98
99  ########################
100 ## Routing Parameters ###
101 ########################
102
103 SN.node[*].Communication.RoutingProtocolName = "LeachRouting"
104 SN.node[*].Communication.Routing.netBufferSize = 1000
105 SN.node[0].Communication.Routing.isSink = true
```

```
106 SN.node[*].Communication.Routing.slotLength = 6
107 SN.node[*].Communication.Routing.roundLength = 67s
108 SN.node[*].Communication.Routing.percentage = 0.1
109 SN.node[*].Communication.Routing.powersConfig = xmldoc("powersConfig.xml")
110
111 ##########################
112 ## Application       #####
113 ##########################
114
115 SN.node[*].ApplicationName = "ThroughputTest"
116 SN.node[*].Application.constantDataPayload = 100
117
118 ##########################
119 ## Wireless Channel #####
120 ##########################
121
122 SN.wirelessChannel.onlyStaticNodes = true
123 SN.wirelessChannel.sigma = 4
124 SN.wirelessChannel.bidirectionalSigma = 1
125 SN.wirelessChannel.pathLossExponent = 2.4
126
127 ##########################
128 ## Radio            ######
129 ##########################
130
131 SN.node[*].Communication.Radio.RadioParametersFile = "../Parameters/Radio/CC2420.txt"
132 SN.node[*].Communication.Radio.TxOutputPower = "0dBm"
```

Código Fonte B.2: Script de configuração do Castalia - Protocolo Comunicação Direta.

```
1
2  [General]
3
4  # ================================================
5  # Always include the main Castalia.ini file
6  # ================================================
7  include ../Parameters/Castalia.ini
8  include ../Parameters/MAC/CSMA.ini
9  sim-time-limit = 57000s
10 SN.field_x = 12
11 SN.field_y = 12
12
13 SN.numNodes = 12
14
15 ######################################
16 ## Nodes Locations - Deployment  #####
17 ######################################
18
19 SN.node[0].xCoor = 3
20 SN.node[0].yCoor = 5
21 SN.node[1].xCoor = 3
22 SN.node[1].yCoor = 0
23 SN.node[2].xCoor = 2
24 SN.node[2].yCoor = 2
25 SN.node[3].xCoor = 4
26 SN.node[3].yCoor = 2
27 SN.node[4].xCoor = 5
28 SN.node[4].yCoor = 4
29 SN.node[5].xCoor = 1
30 SN.node[5].yCoor = 4
31 SN.node[6].xCoor = 1
32 SN.node[6].yCoor = 6
33 SN.node[7].xCoor = 5
34 SN.node[7].yCoor = 6
35 SN.node[8].xCoor = 4
36 SN.node[8].yCoor = 8
37 SN.node[9].xCoor = 2
38 SN.node[9].yCoor = 8
39 SN.node[10].xCoor = 3
40 SN.node[10].yCoor = 10
41 SN.node[11].xCoor = 6
42 SN.node[11].yCoor = 5
43
44 ########################
45 ## Traces          #####
46 ########################
47 #SN.wirelessChannel.collectTraceInfo = true
48 #SN.node[*].Communication.Radio.collectTraceInfo = true
49 SN.node[*].Communication.MAC.collectTraceInfo = true
50 #SN.node[*].Communication.Routing.collectTraceInfo = true
51 #SN.node[*].Application.collectTraceInfo = true
52 #SN.node[*].SensorManager.collectTraceInfo = true
53 SN.node[*].ResourceManager.collectTraceInfo = true
54
55
56 ########################
57 ## EnergyStorage ########
58 ########################
59
60 SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numBatteries = 0
61 SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numRechBatteries = 1
62 SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.numSupercaps = 0
```

```
63
64  ############################################################
65  ##[Rechargeable battery] 1 Prismatic Battery 3.7V 770mAh###
66  ############################################################
67
68  SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[0].maxVoltage = 3.7
69  SN.node[*].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[0].mAmpereHour = 770
70
71  SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[0].maxVoltage = 3.7
72  SN.node[0].ResourceManager.EnergySubsystem.EnergyStorage.RechBatteries[0].mAmpereHour = 1300 #More energy allows Sink node
        die after all nodes.
73
74
75  ########################################
76  ## Energy Resource - Power Consumption ###
77  ########################################
78  SN.node[*].ResourceManager.baselineNodePower = 144
79  SN.node[*].ResourceManager.cutOffVoltage = 3.2
80
81  ########################
82  ## Routing          #####
83  ########################
84
85  SN.node[*].Communication.MACProtocolName = "TunableMAC"
86  SN.node[*].Communication.RoutingProtocolName = "BypassRouting"
87  SN.node[0].Communication.Routing.isSink = true
88
89  ########################
90  ## Application      #####
91  ########################
92  SN.node[*].ApplicationName = "ThroughputTest"
93  SN.node[*].Application.packet_rate = 1
94  SN.node[*].Application.constantDataPayload = 100
95
96  ########################
97  ## Wireless Channel #####
98  ########################
99  SN.wirelessChannel.onlyStaticNodes = true
100 SN.wirelessChannel.sigma = 4
101 SN.wirelessChannel.bidirectionalSigma = 1
102 SN.wirelessChannel.pathLossExponent = 2.4       # Free Space
103
104
105 ########################
106 ## Radio            #####
107 ########################
108 SN.node[*].Communication.Radio.RadioParametersFile = "../Parameters/Radio/CC2420.txt"
109 SN.node[*].Communication.Radio.TxOutputPower = "0dBm"
```

Apêndice C

Código Java (J2ME) - Protocolos

Código Fonte C.1: Manipulando pacote de controle do LEACH

```
1  /**
2       * Callback from PacketReceiver when a new command is received
3       * from the host.
4       * @param type the command
5       * @param pkt the radiogram with any other required information
6       */
7      public void handlePacket(byte type, Radiogram pkt)
8      {
9
10         try
11         {
12             switch (type)
13             {
14                 //Base station commands
15                 case START_APP:
16
17                         ...
18
19                     break;
20
21
22                 case ERASE_PERSISTENCE:
23
24                         ...
25
26                     break;
27
28                 case PERSISTENCE:
29
30                         ...
31
32                     break;
33
34                 //LEACH
35
36                 case START_LEACH_ENGINE:
37
38                     dgController = pkt;
39                     overAllCount++;
40
41                     try
```

```
42                  {
43                      if(persistenceControl)
44                      {
45                          persistence.updateTimeElapsed( (byte)1,
46                                  "Starting cycle: "
47                                  +getTime(System.currentTimeMillis()));
48                              persistenceControl = false;
49                      }
50                      else
51                      {
52                          persistence.updateTimeElapsed( (byte)2,
53                                  "Ending cycle: "
54                                  +getTime(System.currentTimeMillis()));
55                      }
56                  }
57                  catch(Exception e)
58                  {
59                      e.printStackTrace();
60                  }
61
62                  System.out.println("[System Information],"+
63                          IEEEAddress.toDottedHex(Spot.getInstance().
64                          getRadioPolicyManager().getIEEEAddress())
65                          +", LEACH STARTED in, "
66                          +getTime(System.currentTimeMillis()));
67
68                  initialTime = System.currentTimeMillis();
69                  joinStartTimeControl = false;
70                  controlADV = false;
71
72                  Utils.sleep(100);
73                  maxRssiCoordinator = 0;
74                  lastAddressAdv = 0L;
75
76                  if(roundNumber >= 1/CLUSTER_HEAD_QUANTITY)
77                  {
78                      roundNumber = 0;
79                      isCH = false;
80                      isCT = false;
81                      addressNodes.removeAllElements();
82                  }
83
84                  randomNumber = r.nextDouble();
85
86                  if(isCH)
87                  {
88                      isCH = false;
89                      isCT = true;
90                      addressNodes.removeAllElements();
91                  }
92                  if(isCT)
93                  {
94                      probability = 0;
95
96                  }
97                  else
98                  {
99                      if(roundNumber >= (1/ CLUSTER_HEAD_QUANTITY - 1))
100                         probability = 1;
101                     else
102                     {
103                         probability = CLUSTER_HEAD_QUANTITY
104                                 /(1 - CLUSTER_HEAD_QUANTITY*
105                                 (roundNumber %
```

```
106                         (int)(1/CLUSTER_HEAD_QUANTITY)));
107                     }
108                 }
109
110                 if(randomNumber<probability)
111                 {
112                     roundNumber++;
113                     rcvrBS.unregisterHandler(this,ADV_PACKET);
114                     init();
115                     isCH = true;
116
117                     System.out.println("[System Information],"+
118                         IEEEAddress.toDottedHex(Spot.getInstance().
119                         getRadioPolicyManager().getIEEEAddress())
120                         +", was elected CH in, "
121                         +getTime(System.currentTimeMillis()));
122
123                     clusterHeadLed = ledColor;
124
125                     if(txConn != null)
126                     {
127                         txConn.close();
128                         txConn = null;
129                     }
130
131                     now = 0L;
132
133                     Utils.sleep(20);
134
135                     try
136                     {
137
138                        txConn = (RadiogramConnection) Connector.
139                                open("radiogram://broadcast:" +
140                                BROADCAST_PORT);
141                        Datagram xdg = txConn.newDatagram(ADV_PACKET_SIZE);
142                        System.out.println("[System Information], "+
143                         IEEEAddress.toDottedHex(Spot.getInstance().
144                         getRadioPolicyManager().getIEEEAddress())
145                         +", seding ADV_PACKET in, "
146                         +getTime(System.currentTimeMillis()));
147
148
149                     xdg = mountAdvPacket(ADV_PACKET, xdg,
150                                ourMacAddress, ledColor);
151
152                     Utils.sleep(20);
153                     txConn.send(xdg);
154                     advCountCH++;
155
156                     rcvrBS.stop();
157
158                     hostConn.close();
159
160                     hostConn = (RadiogramConnection) Connector.
161                                 open("radiogram://:"+CONNECTED_PORT);
162
163                     rcvrBS = null;
164                     rcvrBS = new PacketReceiver(hostConn);
165                     rcvrBS.registerHandler(this, TDMA_PACKET);
166                     rcvrBS.registerHandler(this, DATA_PACKET);
167                     rcvrBS.registerHandler(this, RESET_LEACH_ENGINE);
168                     rcvrBS.registerHandler(this, JOIN_PACKET);
169                     rcvrBS.start();
```

```
170
171                 System.out.println("[System Information], "+
172                     IEEEAddress.toDottedHex(Spot.getInstance().
173                     getRadioPolicyManager().getIEEEAddress())
174                     +", checking is PacketReceiver is alive: "+
175                     rcvrBS.isRunning()+" in,"
176                     +getTime(System.currentTimeMillis()));
177
178                 }
179                 catch (IOException ex)
180                 {
181                     System.out.println(ex.getMessage());
182                 }
183                 finally
184                 {
185                    if(txConn != null)
186                    {
187                        txConn.close();
188                        txConn = null;
189                    }
190                 }
191             }
192             else
193             {
194                 isCH = false;
195                 roundNumber++;
196                 init();
197             }
198             break;
199
200         case ADV_PACKET:
201
202             advCountNode++;
203
204             flagTDMA = false;
205
206             if(!isCH)
207             {
208                 if(maxRssiCoordinator < pkt.getRssi())
209                 {
210                     maxRssiCoordinator = pkt.getRssi();
211                     clusterHeadAddress = pkt.getAddress();
212                     pkt.resetRead();
213                     pkt.readByte();
214                     pkt.readLong();
215                     clusterHeadLed = pkt.readInt();
216                 }
217
218                 if(!controlADV)
219                 {
220                     thread = new Thread(new Runnable()
221                     {
222                         public void run()
223                         {
224                             try
225                             {
226                                 waitForAdvPackets();
227                             }
228                             catch (IOException ex)
229                             {
230                                 ex.printStackTrace();
231                             }
232                         }
233                     });
```

```
234                     thread.setPriority(Thread.MAX_PRIORITY);
235                     thread.start();
236                     controlADV = true;
237                 }
238             }
239
240             break;
241
242         case JOIN_PACKET:
243
244             if(!joinStartTimeControl)
245             {
246                 joinStartTimeControl = true;
247                  threadJoin.setPriority(Thread.NORM_PRIORITY);
248                  threadJoin.start();
249              }
250
251              if(!addressNodes.contains(pkt.getAddress()))
252              {
253                  addressNodes.addElement(pkt.getAddress());
254                  System.out.println("[System Information], "+
255                      IEEEAddress.toDottedHex(Spot.getInstance().
256                      getRadioPolicyManager().getIEEEAddress())
257                      +", received Join Request from node "+
258                      pkt.getAddress()+" in,"
259                      +getTime(System.currentTimeMillis()));
260              }
261
262
263             break;
264
265         case TDMA_PACKET:
266
267             tdmaCountNode++;
268             System.out.println("[System Information], "+
269                     IEEEAddress.toDottedHex(Spot.getInstance().
270                     getRadioPolicyManager().getIEEEAddress())
271                     +", isCH: "+isCH+" received a TDMA_Packet from "+
272                     pkt.getAddress()+" in,"
273                     +getTime(System.currentTimeMillis()));
274
275             pkt.resetRead();
276             pkt.readByte();
277             double fator = 0;
278             int tempFix = 0;
279
280             txConn = null;
281
282             indexSlotTimmer = (pkt.readByte());
283             indexSlotTimmer++;
284             clusterLength = pkt.readByte();
285             fator = ((double)indexSlotTimmer/(double)clusterLength);
286
287             int time = (int)(tdmaSlotTime*fator);
288
289             tempFix = (int) 200-(200*(int)(Math.ceil(fator*100)/100));
290
291             Utils.sleep(tempFix);
292
293             System.out.println("[System Information], "+IEEEAddress.
294                             toDottedHex(Spot.getInstance().
295                             getRadioPolicyManager().getIEEEAddress())+
296                             ", TDMA Summary: \n"
297                             + "-IndexSlotTimmer: "
```

```
298                         +indexSlotTimmer+"\n -ClusterLength: "+
299                         clusterLength+"\nisCH: "+isCH+"\nFlagTMDA: "
300                         +flagTDMA+"\nTDMA slot time"+tdmaSlotTime
301                         +"\n"+ "Factor: "+Math.ceil(fator*100)/100
302                         +" in,"+
303                         getTime(System.currentTimeMillis()));
304
305                 txConn = (RadiogramConnection) Connector.
306                         open("radiogram://"+clusterHeadAddress+":"
307                         +CONNECTED_PORT);
308
309                 txConn.setTimeout(-1);
310
311                 dataPacket = txConn.newDatagram(DATA_PACKET_SIZE);
312
313                 now = System.currentTimeMillis();
314
315                 txConn.setRadioPolicy(RadioPolicy.OFF);
316                 rcvrBS.stop();
317                 thread = null;
318
319                 Utils.sleep(time+10);
320
321                 txConn.setRadioPolicy(RadioPolicy.ON);
322                 dataPacket.reset();
323                 dataPacket.writeByte(DATA_PACKET);
324                 dataPacket.writeInt(10); //supposed value collected from transductor
325
326                 for(int i = 0; i < QTDDATAPKT; i++)
327                 {
328                     txConn.send(dataPacket);
329                     Utils.sleep(10);
330                     dataCountNode++;
331                 }
332
333
334                 if (txConn != null)
335                 {
336                     txConn.close();
337                     txConn = null;
338                 }
339                 Utils.sleep((tdmaSlotTime - time));
340                 rcvrBS.start();
341
342
343             break;
344
345             case DATA_PACKET:
346
347                 dataCountCH++;
348                 if(dataPacketPhase)
349                 {
350                     thread = null;
351                     //nowDataPacket = System.currentTimeMillis();
352                     System.out.println("[System Information], "+
353                         IEEEAddress.toDottedHex(Spot.getInstance().
354                         getRadioPolicyManager().getIEEEAddress())
355                         +",isCH: "+isCH+
356                         " received first Data_Packet from CH: "+
357                         pkt.getAddress()+" in,"+
358                         getTime(System.currentTimeMillis()));
359                     //aggregator = new Aggregator(nowDataPacket);
360                     //aggregator.start();
361                     dataPacketPhase = false;
```

```
362                     }
363
364                     System.out.println("[System Information], "+
365                             IEEEAddress.toDottedHex(Spot.getInstance().
366                             getRadioPolicyManager().getIEEEAddress())
367                             +",isCH: "+isCH+
368                             " received first Data_Packet from CH: "+
369                             pkt.getAddress()+" in,"+
370                             getTime(System.currentTimeMillis()));
371
372                 break;
373
374                 //END LEACH
375
376                 case DISPLAY_SERVER_QUITTING:
377
378                         ...
379
380                 case BLINK_LEDS_REQ:
381
382                         ...
383
384                 break;
385
386             }
387         }
388         catch (IOException ex)
389         {
390             closeConnection();
391         }
392     }
```

Código Fonte C.2: Manipulando pacote de controle do LEACH

```
1
2       /**
3        * Callback from PacketReceiver when a new command is received
4        * from the host.
5        * @param type the command
6        * @param pkt the radiogram with any other required information
7        */
8       public void handlePacket(byte type, Radiogram pkt)
9       {
10
11          try
12          {
13              switch (type)
14              {
15
16                  case START_APP:
17
18                          ...
19
20                      break;
21
22
23                  case ERASE_PERSISTENCE:
24
25                          ...
26
27                      break;
28
29                  case PERSISTENCE:
30
31                          ...
32
33                      break;
34
35
36                  case START_ENGINE:
37
38                      initialTime = System.currentTimeMillis();
39
40                      txConn = (RadiogramConnection)Connector.
41                              open("radiogram://broadcast:"
42                              +BROADCAST_PORT);
43
44                      txConn.setMaxBroadcastHops(0);
45
46                      Datagram rdg = txConn.newDatagram(100);
47
48                      String address = IEEEAddress.toDottedHex(Spot.getInstance().
49                              getRadioPolicyManager().getIEEEAddress());
50
51                      rcvrBS.stop();
52
53                      try
54                      {
55                          persistence.updateTimeElapsed((byte) 1,
56                                  getTime(System.currentTimeMillis()));
57                      }
58                      catch (Exception ex)
59                      {
60                          ex.printStackTrace();
61                      }
62
```

```
63                  while(condition)
64                  {
65                      rdg.reset();
66                      rdg.writeByte(DATA);
67                      rdg.writeInt(overAllCount+1);
68                      rdg.writeUTF(address);
69                      Utils.sleep(100);
70                      txConn.send(rdg);
71                      overAllCount++;
72                      System.out.print("Packet has been sent successfully "
73                              + "by node: "+address+";\n");
74
75                      if(overAllCount%10 == 0)
76                      {
77                          try
78                          {
79                                  persistence.updateTimeElapsed((byte) 2,
80                                          getTime(System.currentTimeMillis()));
81                                  persistence.updatePacketsCounterNonCoordinator(
82                                          PersistenceUnit.OVERALL,
83                                          String.valueOf(overAllCount));
84                          }
85                          catch (Exception ex)
86                          {
87                                  ex.printStackTrace();
88                          }
89                      }
90
91                      Utils.sleep(SEND_RATE);
92                  }
93
94              break;
95
96
97              case RESET_ENGINE:
98
99                  ...
100
101             break;
102
103         }
104     }
105     catch (IOException ex)
106     {
107         closeConnection();
108     }
109 }
```