INSTITUTO FEDERAL DE SANTA CATARINA

ANDRÉ FELIPPE WEBER

Provimento de QoS para Aplicações em Redes de Sensores sem Fios Baseadas em Redes Definidas por Software

São José - SC

agosto/2017

PROVIMENTO DE QOS PARA APLICAÇÕES EM REDES DE SENSORES SEM FIOS BASEADAS EM REDES DEFINIDAS POR SOFTWARE

Trabalho de conclusão de curso apresentado à Coordenadoria do Curso de Engenharia de Telecomunicações do campus São José do Instituto Federal de Santa Catarina para a obtenção do diploma de Engenheiro de Telecomunicações.

Orientador: Prof. Tiago Semprebom, Dr. Eng.

Coorientador: Prof. Eraldo Silveira e Silva, Dr. Eng.

São José - SC

agosto/2017

RESUMO

As Redes de Sensores sem Fios (RSSFs) tornaram-se um recurso valioso no contexto da Internet das Coisas (IoT). Diversos domínios de aplicação como militar, hospitalar, de segurança e de sistemas de controle utilizam dados escalares e multimídia, produzidos por inúmeros módulos espalhados em uma região. O gerenciamento destes módulos é essencial para o correto funcionamento da RSSF. Porém as limitações de bateria, poder de processamento e memória, em conjunto com a heterogeneidade dos módulos torna este processo complexo. Como consequência, garantir níveis de QoS (*Quality of Service*), aliados ao baixo consumo energético e roteamento eficiente nas RSSFs torna-se um desafio. O conceito de Redes Definidas por Software (SDN) permite que a complexidade do roteamento seja separada, fisicamente, dos módulos e seja repassada para um controlador em um servidor remoto. Deste modo, este trabalho propõe implementar uma RSSF, que utiliza o protocolo SDN-WISE e algoritmos de roteamento executados na nuvem, para garantir largura de banda e eficiência energética aos módulos e, portanto, prover QoS às aplicações que executam sobre a RSSF.

Palavras-chave: Redes de Sensores Multimídia sem Fios, roteamento, QoS, IoT, SDN.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

LR-WPAN *Low-Rate Wireless Personal Area Network* 12

MAC Controle de Acesso ao Meio 12

MANET Rede *Ad hoc* móvel 16

PAN *Personal Area Network* 13

PHY física 12

QoS Quality of Service 3

RAM *Random Access Memory* 11

RFD Dispositivo de Função Reduzida 13

ROM *Read Only Memory* 11

RSMSF Rede de Sensores Multimídia sem Fio 12

RSSF Rede de Sensores Sem Fio 3

RSSI *Received signal strength indication* 22

SDN Redes Definidas por Software 3

SDN-WISE *Software Defined Networking for WIreless SEnsor networks* 3

TCP *Transmission Control Protocol* 23

TD *Topology Discovery* 22

TTL *Time To Live* 21

SUMÁRIO

1 INTRODUÇÃO

O conceito de Internet das Coisas (IoT) tem sido amplamente investigado pela comunidade acadêmica nos últimos anos e atualmente ganha espaço nos setores de desenvolvimento das empresas de tecnologia. De acordo com o conceito de IoT objetos comuns do cotidiano, como lâmpadas e sensores são conectados à Internet, permitindo, por exemplo, o controle de eletrodomésticos e a disponibilização de dados como umidade e temperatura de ambientes na Internet.

As aplicações de sensoriamento na IoT possuem o desafio de organizar uma rede com centenas de dispositivos. Por isso, as RSSF são um recurso valioso da IoT (CAPELLA et al., 2016). As RSSFs tendem a executar uma função colaborativa onde os sensores proveem dados, que são processados (ou consumidos) por módulos especiais chamados de sorvedouros (*sink nodes*). Estes módulos sensores geram dados escalares (pressão, temperatura, umidade, etc) e multimídia através de suas câmeras, microfones e sensores. Transmitem essas grandezas utilizando transceptores acoplados aos módulos, e são limitados em memória, processamento e bateria.

Com a popularização da IoT, a dificuldade de gerenciamento das RSSF ficou ainda mais latente, principalmente, devido às limitações de *hardware* dos módulos (i.e. processamento, memória e bateria). Dessa forma, existe a demanda no desenvolvimento de técnicas de gerenciamento que melhorem a escalabilidade das RSSFs. O conceito de Redes Definidas por Software (SDN) surge como uma das técnicas utilizadas nas RSSFs para gerenciamento da topologia, consumo energético e roteamento da rede para satisfazer às exigências de QoS das aplicações (NDIAYE; HANCKE; ABU-MAHFOUZ, 2017).

O principio básico da SDN é a separação da rede em plano de controle e plano de dados. Onde o primeiro é responsável pelas decisões de roteamento e o segundo pelo roteamento dos dados. Dessa forma, um controlador centralizado, em um servidor remoto, controla e gerencia o roteamento realizado pelos módulos no plano de dados. Por exemplo, um controlador de uma RSSF em uma *Smart Home* pode utilizar algoritmos complexos de roteamento para garantir, em tempo real, a banda necessária para que um sensor com capacidade multimídia transmita seus dados, de acordo com as demandas de QoS (*Quality of Service*) da aplicação.

O protocolo SDN-WISE utiliza a arquitetura SDN para reduzir a complexidade de configuração e gerenciamento das RSSFs. Desta forma, um controlador separado fisicamente do restante da rede executa algoritmos de roteamento complexos em um servidor remoto, enquanto módulos da rede encaminham os dados de um nó *source* até o *sink*, de acordo com a tabela de roteamento recebida do controlador.

A utilização da abordagem SDN economiza energia dos módulos e permite o uso de algoritmos complexos de roteamento que visam a garantia de QoS para as aplicações da rede. Deste modo, este trabalho propõe um sistema que utiliza o protocolo SDN-WISE para capturar informações do estado dos enlaces da RSSF e enviar para um controlador remoto (na nuvem), que executa um algoritmo de roteamento. A Figura 1 ilustra uma RSSF que, pode usufruir do sistema proposto. Os módulos da RSSF enviam, através do protocolo SDN-WISE, informações como qualidade do enlace e, nível de bateria para um controlador da rede, hospedado na nuvem. O controlador, por sua vez, executa um algoritmo de roteamento que, considera métricas de QoS da aplicação em sua execução. Por exemplo, maximização do tempo de vida da rede e largura de banda. Por fim, após a execução do algoritmo, o controlador envia uma mensagem de atualização para os módulos da rede que, podem alterar ou adicionar novas rotas de encaminhamento de mensagens.

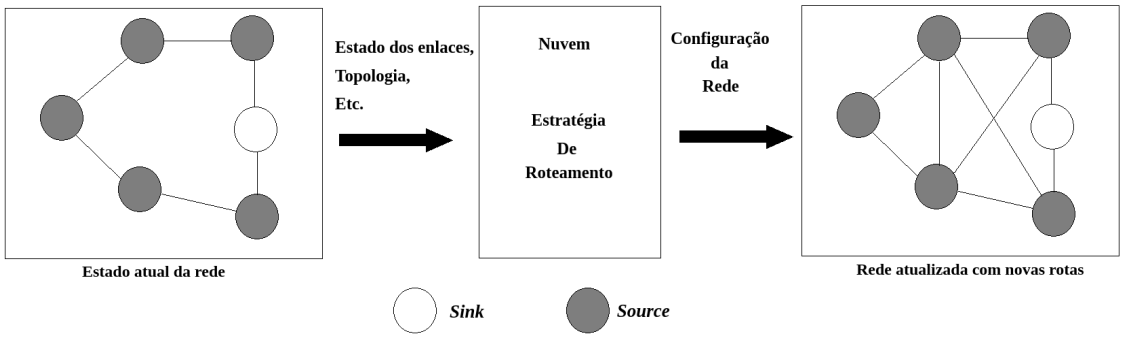


Figura 1 – Modelo do sistema.

1.1 Objetivos Gerais

O objetivo geral deste trabalho é implementar uma rede de sensores sem fio, baseada no conceito de Redes Definidas por Software (SDN) com fins de provimento de QoS para aplicações que se executam sobre a RSSF.

1.2 Objetivos específicos

- Implantar e analisar o comportamento de uma aplicação sobre a plataforma SDN-WISE no simulador Cooja e nos sensores Micaz;
- Conceber e implementar um controlador SDN que permita construir regras de roteamento a partir de uma configuração estática dos módulos e, dos fluxos que possuem requisitos de QoS associados à largura de banda e ao nível de bateria dos módulos;
- Implantar uma rede real com a aplicação definida anteriormente e avaliar o comportamento da mesma.
- Analisar algoritmos de roteamento para RSSF e, adequá-los para o cumprimento de requisitos de QoS referentes à vazão e consumo energético das aplicações que executam sobre o SDN-WISE;
- Implementar as regras de roteamento em uma aplicação piloto, utilizando o simulador Cooja.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão apresentados temas essenciais para a compreensão e desenvolvimento deste trabalho. Na Seção 2.1 é explicado como as RSSFs são utilizadas para dar forma à muitas aplicações de sensoriamento da IoT. Na Seção 2.2 é apresentado o conceito de SDN e como tais redes são empregadas. Na sequência, Seção 2.3, o conceito de SDN é explorado no contexto de RSSFs. Para tanto, a plataforma SDN-WISE será usada como referência, através de um exemplo. Finalmente, a aplicabilidade desta plataforma em aplicações com requisitos de QoS é discutida.

2.1 Redes de Sensores para a Internet das Coisas

A IoT considera que os objetos comuns do cotidiano se comuniquem através da Internet. Esta ideia já pode ser observada na prática nos dispositivos inteligentes, como televisores, celulares e relógios além de outros objetos, como lâmpadas e sensores. Porém, o aumento do número de dispositivos conectados à Internet impõe o desafio de rotear, de maneira eficaz, o volume de dados gerados que cresce proporcionalmente.

O desafio de organizar uma rede com centenas de dispositivos é uma realidade, principalmente quando se trata de aplicações de sensoriamento. Por isso, as RSSFs são um recurso valioso da IoT (CAPELLA et al., 2016). Criadas na década de 1950, as RSSFs foram, no princípio, utilizadas pelo exército norte americano para detectar a movimentação de submarinos soviéticos (NASIR; KU-MAHAMUD, 2006). Com o avanço tecnológico elas passaram a ser implementadas em zonas terrestres e subterrâneas, além das subaquáticas. Dentre as aplicações mais citadas em publicações atuais estão aquelas que abrangem a área da saúde, de controle de tráfego de veículos, monitoramento e militar.

As RSSF atuais são compostas por diversos módulos equipados com processador, memórias *Read Only Memory* (ROM) e *Random Access Memory* (RAM), rádio transceptor, um ou mais sensores e uma fonte de energia. Algumas redes utilizam estações base com melhor poder de processamento e capacidade energética para receber e tratar os dados gerados pelos módulos. Dessa forma, é necessário garantir que os módulos comuniquem-se entre si e com a estação base, mesmo que indiretamente, a fim de realizar a entrega de pacotes a qualquer destinatário da rede e garantir que as exigências de QoS sejam cumpridas (GUY, 2006).

O conjunto de métricas de performance utilizadas para definir a qualidade do serviço nas RSSFs variam de acordo com a necessidade de cada aplicação. Porém, segundo Nasir e Ku-Mahamud (2006), existem oito métricas que são comumente utilizadas: (I) vazão, (II) atraso fim-a-fim, (III) taxa de sucesso na entrega de pacotes, (IV) perda de pacotes, (V) distância medida através da quantidade de saltos (*hops*) entre a fonte e o destinatário, (VI) consumo e eficiência energética, (VII) carga da rede e (VIII) longevidade.

A topologia de rede comumente utilizada para organizar as RSSFs é a Ponto-a-Ponto. Como mostrado na Figura 2, a comunicação das redes ponto-a-ponto é, normalmente, realizada entre os vizinhos mais próximos. Os integrantes da rede podem atuar como fonte, receptor ou roteador dos dados já que para realizar a comunicação com um destino fora de alcance é necessário rotear os dados através dos vizinhos. Devido às limitações de processamento e de consumo de energia dos módulos a comunicação entre vizinhos é a principal razão para o uso desta topologia nas RSSFs.

Atualmente, verifica-se que parte da atenção destinada ao estudo das RSSFs é dirigida ao

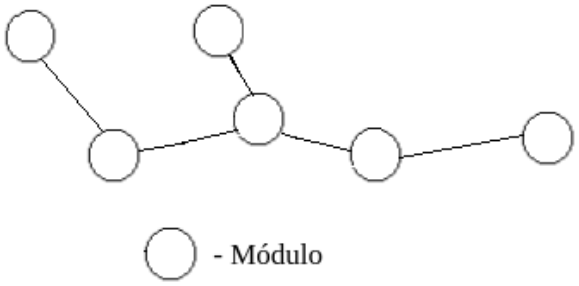


Figura 2 – Topologia ponto-a-ponto.

estudo de redes de sensores com capacidade de processar dados multimídia. Isto se deve ao avanço tecnológico e à redução do custos de aquisição dos módulos e outros componentes utilizados nas Rede de Sensores Multimídia sem Fio (RSMSF)s. As RSMSFs são compostas por módulos equipados com câmeras, microfones e outros sensores produtores de conteúdo multimídia (SHAKSHUKI; XING; MALIK, 2009), além de módulos equipados somente com transceptores utilizados no roteamento de dados na rede.

As RSMSFs atraíram atenção dos pesquisadores da área devido ao seu potencial científico e aplicações atrativas (SHEN; BAI, 2016). Por exemplo, a distribuição de câmeras ao longo de uma rodovia assegura a redundância do sistema de videomonitoramento, disponibiliza múltiplos pontos de vista e, através da escolha de câmeras mais próxima ou com qualidade superior, amplia o ângulo de visão de um evento. A adição de microfones ao sistema permite detectar e determinar a localização aproximada de sons incomuns ao ambiente como barulhos de tiros (ANG et al., 2013) ou freadas bruscas.

Devido ao maior fluxo de dados gerado pelas aplicações implementadas nas RSMSFs, em comparação com as RSSFs, as métricas de QoS que atuam diretamente na garantia de entrega e integridade dos dados multimídia ganham maior relevância em detrimento do consumo de energia que, nas redes RSSFs, ocupa o posto de destaque (HAMID; HUSSAIN, 2014; AKYILDIZ; MELODIA; CHOWDURY, 2007). As exigências de QoS nas RSMSFs podem variar de uma aplicação para outra. Por exemplo, uma aplicação de controle de fluxo de tráfego é mais tolerante à falhas na entrega de pacotes que uma aplicação militar. Porém, de forma geral, as métricas de QoS utilizadas nas redes RSMSFs visam a garantia de largura de banda, tempo de vida da rede, controle do *jitter*[1], taxas de atraso toleráveis, controle de erro através de técnicas de codificação de fonte de baixa complexidade e estratégias de melhoria da eficiência energética (HAMID; HUSSAIN, 2014).

2.1.1 Tecnologias para Redes de Sensores sem Fios

Dentre as tecnologias desenvolvidas e implementadas nas RSSFs está a *Low-Rate Wireless Personal Area Network* (LR-WPAN) que define redes com baixas taxas de dados, baixo consumo energético e baixo custo. O padrão IEEE 802.15.4 especifica as camadas física (PHY) e de Controle de Acesso ao Meio (MAC) para redes LR-WPAN e portanto se caracteriza pela comunicação sem fio com baixa taxa de transmissão de dados, baixo consumo de bateria e custo de implementação. Segundo Semprebom (2012), apesar de não ter sido projetada especificamente para as RSSFs, o IEEE 802.15.4 vem sendo amplamente adotado e possibilitou o surgimento de novos produtos baseados nesta tecnologia.

Na implementação de uma RSSF utilizando a tecnologia IEEE 802.15.4 os módulos podem ser categorizados, de acordo com sua capacidade de processamento ou função na rede, em dois tipos:

- **Dispositivo de Função Completa (FFD)**: Obrigatório em uma LR-WPAN, este dispositivo

[1] Variação no atraso

suporta até três modos de operação:

1. **Coordenador PAN**: O dispositivo fica responsável por controlar a *Personal Area Network* (PAN) e identificar a rede para que outros dispositivos se associem (SILVA, 2017).
2. **Coordenador**: Dispositivo que deve, obrigatoriamente, estar associado à um Coordenador PAN e oferece o serviço de sincronização através da transmissão de *beacons*.
3. **Dispositivo simples**: Dispositivo que atua apenas como um nó na rede e, portanto não implementa as funções de coordenador PAN ou coordenador.

- **Dispositivo de Função Reduzida (RFD)**: Dispositivo associado a um único FFD e que opera com implementação miníma do IEEE 802.15.4.

No padrão IEEE 802.15.4 é possível utilizar as topologias estrela e ponto-a-ponto. Na topologia estrela existe um único nó central ao qual todos os módulos se conectam, como mostrado na Figura 3. Este dispositivo central deve ser um FFD operando como Coordenador PAN que fica responsável por iniciar uma nova rede, com um identificador único, e rotear todos os pacotes sendo transmitidos. Os outros integrantes são RFDs ou FFDs operando como Coordenador ou Dispositivo simples (SEMPREBOM, 2012).

O nó central da topologia estrela pode ser visto como um ponto crítico de falha, uma vez que toda a comunicação da rede é realizada através dele. Por isso, devido às limitações energéticas dos FFDs, o padrão IEEE 802.15.4 recomenda que o Coordenador PAN esteja conectado em alguma fonte de alimentação, e que esta topologia seja usada em aplicações que apresentem facilidade na substituição da fonte de energia do Coordenador PAN, como automação residencial e industrial (SILVA, 2017).

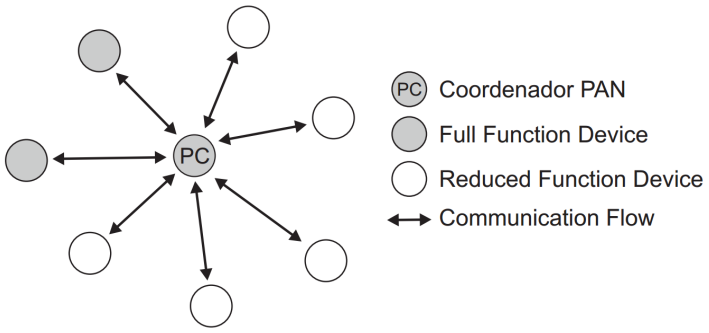


Figura 3 – Topologia estrela. (SEMPREBOM, 2012)

A topologia ponto-a-ponto (Figura 4), assim como a estrela, possui um único coordenador PAN. Porém é mais confiável já que, por ser descentralizada, possui múltiplos caminhos para o roteamento dos pacotes. Por não possuir um nó central, os módulos se comunicam diretamente com os vizinhos em seu raio de cobertura, e através de múltiplos saltos com os vizinhos mais distantes.

As funcionalidade de roteamento por múltiplos saltos não são definidas pelo padrão IEEE 802.15.4, pois devem ser definidas na camada de rede (SEMPREBOM, 2012). As funções de roteamento utilizadas para realizar os múltiplos saltos possuem a desvantagem de introduzir complexidade e reduzir a longevidade da rede, uma vez que os módulos precisam reencaminhar dados recebidos dos vizinhos. Porém, tem como vantagem o aumento do alcance da rede.

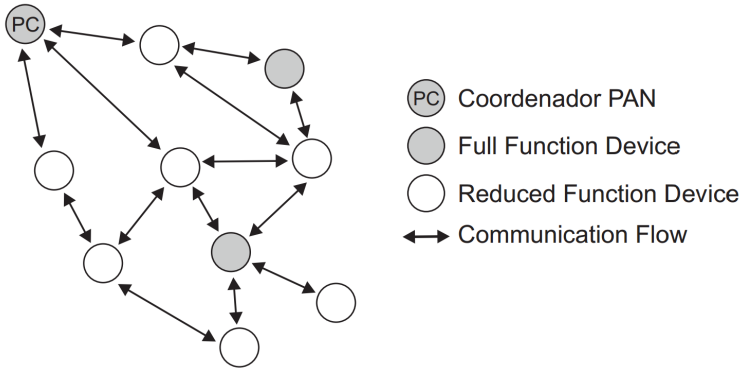


Figura 4 – Topologia ponto-a-ponto (SEMPREBOM, 2012)

Outra tecnologia que se destaca na criação de RSSFs é o LoRa. Esta é uma tecnologia de camada física e uma modulação para comunicação sem fio que possibilita enlaces de longa distância, baixo consumo energético e taxa de transmissão. Segundo Saraiva (2017), quanto menor a taxa de transmissão, maior a duração da mensagem (tempo do sinal no ar) e, consequentemente maior a energia de bit. Como resultado, há o aumento da qualidade de recepção do sinal (cerca de -150 dBm), proporcionando uma maior cobertura, estimada na ordem de 10 a 15 quilômetros em áreas rurais e 2 a 5 quilômetros em áreas urbanas. A frequência de operação do LoRa varia por região porém, assim como o IEEE 802.15.4 operam na faixa Industrial, Cientifico e Médico (ISM), mais especificamente em 2.4 GHz, 868/915 MHz, 433 MHz e 169 MHz. A taxa de transmissão está na ordem de centenas de bits ou dezenas de kilobits por segundo (CENTENARO et al., 2016).

O LoRaWAN define o protocolo de comunicação e a arquitetura do sistema para a rede, enquanto a camada física LoRa possibilita o uso de enlaces de longa distância. Apesar do longo alcance, as redes ponto-a-ponto apresentam a desvantagem do consumo desnecessário de bateria no módulos que atuam como roteadores da rede. Por isso, no LoRaWAN adotou-se a topologia estrela de longa distância. O que permite uma maior economia energética para conexões de longa distância.

As redes LoRaWAN são implementadas com o objetivo de otimizar redes *Low Power Wide Area Network* (LPWAN) para aplicações IoT. As LPWANs oferecem excelente eficiência energética com expectativa de vida útil maior que dez anos para as baterias, e são pensadas para aplicações que transmitem pequenas quantidades de dados, por grandes distâncias e poucas vezes por hora (ALLIANCE, 2015).

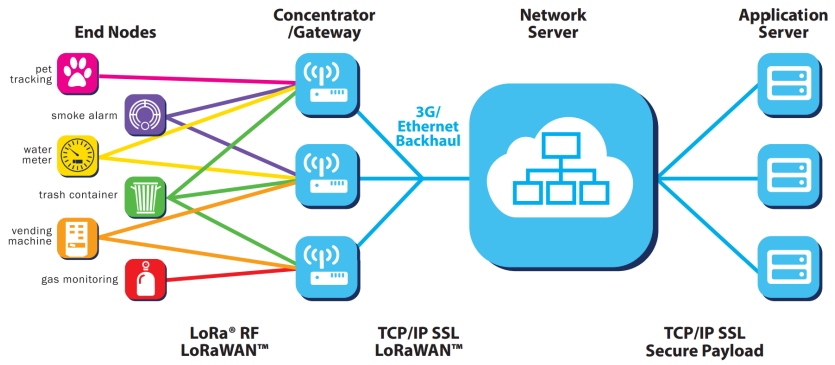


Figura 5 – Visão geral de uma rede LoRaWAN com topologia em estrela (ALLIANCE, 2015)

De acordo com alliance (2015), as redes LoRaWAN são compostas por *gateways* que atuam como nó central da rede e módulos (*end-devices*) que compõem a rede. São classificados de acordo com a relação entre disponibilidade de recepção de dados e vida útil de sua bateria:

- **Classe A:** São sensores, energizados via bateria, com a melhor eficiência energética da rede LoRaWAN. Os *end-devices* de classe A transmitem somente quando necessário e recebem dados, apenas, após realizar uma transmissão. Dessa forma, ao realizar uma transmissão o *end-device* espera por duas janelas curtas de tempo e, volta a desabilitar seu rádio transceptor.
- **Classe B:** São atuadores, energizados via bateria, que além de implementar as janelas de recepção do *end-devices* classe A, também agendam janelas de tempo para recepção de dados. Para isso, é necessário a sincronização entre *gateway* e *end-device* classe B que, é realizada através de *beacons* enviados pelo *gateway*.
- **Classe C:** São atuadores conectados à uma fonte externa de energia que, aceitam recepção o tempo todo com exceção para quando há uma transmissão a ser realizada. Esse *end-device* tem a vantagem de eliminar latência para recepção de dados, porém necessita de uma fonte contínua de energia.

A topologia em estrela é a ideal para as rede LoRaWAN por causa do longo alcance proporcionado pela camada física dos dispositivos LoRa (OLIVEIRA, 2017). Esta topologia permite uma melhor eficiência energética quando comparada à topologia ponto-a-ponto, adotada nas redes de curto alcance como as redes IEEE 802.15.4, visto que nenhum módulo atua como roteador. E para contornar o ponto de falha único, característico da topologia em estrela, usa-se redundância de *gateways* (ALLIANCE, 2015).

Diferentemente do que ocorre na topologia estrela tradicional, em que há apenas um nó central em que todos os módulos de conectam, na LoRaWAN existem diversos nós, implementados como *gateways*. Assim sendo, segundo alliance (2015) as mensagens produzidas por um módulo não são recebidas por apenas um, mas sim por múltiplos *gateways*. Cada *gateway* encaminha os dados gerados pelo *end-device* até um servidor responsável por tratar dos dados, eliminar dados redundantes, administrar a taxa de dados, etc. Dessa forma, através da redundância criada por múltiplos *gateways* elimina-se o ponto crítico da rede, que é dispor de um único nó central.

Em alliance (2015), atribuiu-se às redes LoRaWAN a característica de gerenciar a longevidade da bateria, capacidade e segurança da rede, onde:

- **Longevidade da bateria:** Os módulos na LoRaWAN são assíncronos, ou seja, não há troca de mensagem entre *end-devices* ou com o *gateway* para manter sincronia. Logo, ganha-se em eficiência energética, já que o processo de sincronização é o fator número um de redução da vida das baterias (ALLIANCE, 2015).
- **Capacidade da rede:** O LoRaWAN utiliza a sua capacidade de adaptação da taxa de dados (*Adaptive Data Rate* (ADR)) para aumentar a capacidade da rede e que, um *gateway* possui de receber mensagens de um grande volume de *end-devices* (ALLIANCE, 2015). Isto ocorre pois o uso da *chirp spread spectrum* (CSS) na camada física permite que um *gateway* receba dados com diferentes taxas de transmissão no mesmo canal e, ao mesmo tempo. Assim, é possível que dispositivos mais próximos do *gateway* operem com uma taxa de transmissão mais alta e, com potência menor que dispositivos mais distantes (SARAIVA, 2017).

  O ADR auxilia, também, no aumento da capacidade da rede já que ele possibilita que seja aumentado a taxa de dados, de forma a reduzir o tempo de transmissão de um módulo e abrindo espaço de tempo para que outros transmitam. Além do mais, é possível reduzir o consumo energético de um módulo ao reduzir sua potência e taxa de dados.
- **Segurança da rede:** Na LoRaWAN é utilizado criptografia *Advanced Encryption Standard* (AES) e garante-se segurança nas camadas de aplicação e rede. Na camada de aplicação os dados do usuário final são protegidos enquanto na de rede garante-se a autenticidade do módulo (ALLIANCE, 2015).

2.1.2 Técnicas de Roteamento em RSSFs

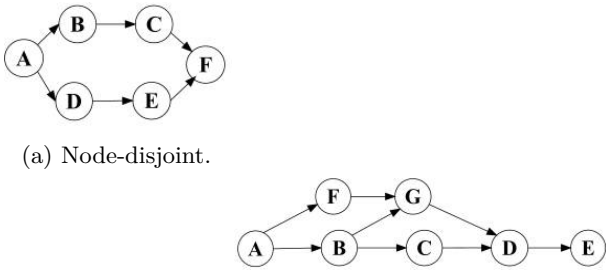
Devido às limitações de recursos dos módulos utilizados nas RSSFs, é importante projetar protocolos de roteamento que possibilitem a transmissão de pacotes por longas distâncias na rede. Para isso, alguns protocolos utilizam técnicas de múltiplos caminhos e saltos para criar e manter rotas de encaminhamento dos pacotes. Como resultado, o protocolo melhora a eficácia da comunicação entre módulos e em alguns casos, com a Internet.

Muitos protocolos de roteamento para RSSFs foram propostos nas últimas décadas considerando as limitações energéticas, de processamento e memória dos módulos, assim como as exigências das aplicações e arquitetura das redes de sensores sem fio (ALAZZAWI; ELKATEEB, 2008). Em muitas aplicações de RSSFs espera-se que a rede tenha capacidade de organização automática, uma vez que os módulos são distribuídos aleatoriamente e se comportam como componentes de uma rede *ad hoc*. Porém, o roteamento dos dados nas RSSFs não pode ser feito como em redes sem fio tradicional. Segundo Singh, Singh e Singh (2010), as RSSFs se diferenciam das redes celulares e da Rede *Ad hoc* móvel (MANET) por possuírem algumas características especiais:

- Grandes quantidades de dispositivos na rede. Algumas redes podem ser significantemente mais densamente povoadas que as redes MANET;
- Limitação energética e computacional dos módulos contrastante com o que acontece nas redes sem fio tradicionais;
- Auto-organização dos módulos implementada por algumas tecnologias utilizadas nas RSSFs, como o Zigbee, que utiliza o padrão IEEE 802.15.4 nas camadas inferiores (PHY e MAC) e, implementa as camadas superiores introduzindo a capacidade de auto configuração e auto recuperação de falhas;
- Os módulos não são confiáveis, uma vez que podem apresentar falha devido às condições ambientais em que são utilizados e por esgotamento da bateria;
- Possível redundância na transmissão de dados correlatos devido à proximidade entre módulos equipados com sensores;
- Uma RSSF é, normalmente, criada para uma aplicação específica visto que as métricas de QoS e exigências da rede variam entre aplicações;
- A topologia da rede muda constantemente devido à falta de confiabilidade dos módulos, interferências e à possível movimentação dos módulos.

Tradicionalmente, protocolos de roteamento utilizam a abordagem *single-path* para escolha de caminhos. Ou seja, baseado em uma métrica de performance ou na menor distância, um único caminho é escolhido. Porém, através da abordagem de roteamento por múltiplos caminhos é possível melhorar a resiliência da rede, ou seja, a capacidade que a rede possui de se recuperar de falhas, visto que protocolos que utilizam a abordagem *single-path* interrompem o roteamento por algum tempo ou por completo em um evento de falha. Segundo Al-karaki e Kamal (2004), o preço que se paga ao melhorar a resiliência da rede é o aumento de tráfego e consumo energético na rede, visto que mensagens periódicas serão transmitidas para manter os caminhos ativos e atualizados.

Diferentes abordagens podem ser utilizadas para projetar um algoritmo que faz uso de roteamento por múltiplos caminhos. Isso ocorre, pois pode-se priorizar a eficiência energética, ou a confiabilidade da rede, por exemplo. Assim, é possível optar por evitar rotear pacotes por módulos que estão com baixo nível de bateria, dividir um pacote por múltiplos caminhos ou enviar o mesmo pacote por diversos caminhos.



(a) Node-disjoint.

(b) Link-disjoint.

(c) Partially-disjoint.

Figura 6 – Categorias de protocolos de múltiplos caminhos, de acordo com o grau de separação dos caminhos. (RADI et al., 2012)

Porém, todos os protocolos de roteamento por múltiplos caminhos compartilham três etapas básicas: Descoberta, seleção e manutenção dos caminhos (RADI et al., 2012).

Quanto à descoberta dos caminhos, os protocolos podem ser divididos em três categorias de acordo com o grau de separação dos caminhos (*disjointedness*).

- ***Node-disjoint*:** Os caminhos escolhidos não possuem nenhum módulo em comum, além do *source* e *sink*. Ou seja, são completamente independentes um dos outros. Esse tipo de caminhos possuem a desvantagem de apresentar pontos de falha em caso de perda de comunicação em um dos enlaces entre módulos. Por exemplo, na Figura 6a, é possível observar que em caso de falha do módulo C, módulo B também não consegue mais transmitir seus pacotes para o *sink* F.
- ***Link-disjoint*:** Os caminhos possuem módulos em comum, porém não compartilham os enlaces, como mostrados na Figura 6b.
- ***partially-disjoint*:** Os caminhos compartilham seus módulos e enlaces. Possui a principal desvantagem de ser vulnerável às falhas desses componentes da rede. Porém, é possível criar diversos caminhos, redundantes, que melhoram a resiliência da rede (RADI et al., 2012).

A seleção dos caminhos ocorre de acordo com a necessidade da aplicação. No caso das RSSFs é interessante que os caminhos sejam escolhidos levando em conta o estado da bateria dos módulos que rotearão os pacotes. Porém, pode-se optar por melhorar a confiabilidade da rede. Para isso, os pacotes podem ser transmitidos, de forma redundante, por múltiplos caminhos. Outra possibilidade seria transmitir os dados por apenas um dos caminhos, deixando o restante dos caminhos como *backup* para eventos de falha de módulos ou enlaces.

A manutenção dos caminhos pode ser feita, segundo Radi et al. (2012), em um dos seguintes três eventos: (1) Quando um dos caminhos falha, (2) Quando todos os caminhos falham ou, (3) Quando um certo número de caminhos falham. Alguns protocolos classificados como *on-demand* permitem que a manutenção da rede seja realizada em tempo real, através da descoberta e o constante monitoramento dos caminhos. Nessa classificação, existem os *proactive on-demand* que descobrem e mantêm os caminhos ao longo da vida útil da rede, e os *reactive on-demand* que descobrem um novo caminho somente quando necessário, isto é, antes de uma transmissão de dados (Mbarushimana e Shahrabi (2007)).

2.2 *Redes Definidas por Software (SDN)*

As infraestruturas de redes tradicionais são compostas por dispositivos, como roteadores e *switches*, que executam algoritmos complexos e, muitas vezes com interfaces de controle limitadas e específicas do

fabricante do equipamento. Consequentemente, a manutenção e atualização dos protocolos da rede (e.g. IPV6), torna-se um desafio (NUNES et al., 2014). O SDN foi projetado para facilitar o gerenciamento das redes de forma simples e programática. Isto é feito ao retirar a complexidade de controle da rede dos equipamentos (*hardware*) e, repassá-la para uma aplicação controladora, em *software* (KLAUBERG, 2016).

O principio básico do SDN é a separação da rede em planos de controle e dados (Figura 7). Onde o primeiro é responsável pelas decisões de roteamento e o segundo pelo roteamento dos dados:

- ***Camada de controle*:** Esta camada é composta por um controlador que possui uma visão geral da rede (Camada de Dados) e capacidade para centralizar toda a inteligência da rede, isto é, gerenciar o roteamento, topologia, segurança, QoS, e controlar o consumo energético da rede (NDIAYE; HANCKE; ABU-MAHFOUZ, 2017). Desta forma, os módulos da camada inferior enviam requisições, através de uma *application programming interface* (API), para o controlador sempre que uma nova rota é necessária. O controlador, que pode estar em um servidor remoto, responde com a informação necessária para completar a tabela de roteamento do módulo
- ***Camada de dados*:** Os equipamento (*hardware*) que executam uma mesma ou diferentes aplicações são atribuídos à camada de dados. Estes módulos são responsáveis por realizar o sensoriamento da rede e rotear dados de acordo as informações recebidas pelo controlador.

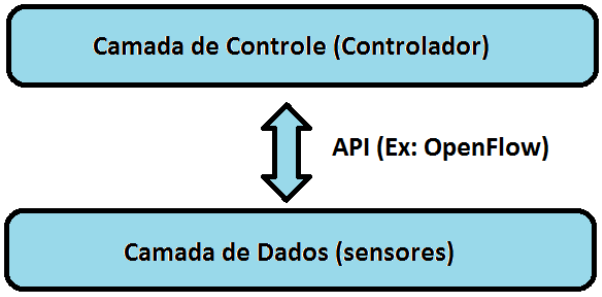


Figura 7 – Arquitetura do *Software Defined Networking*

Ao facilitar a manutenção e entregar a inteligência da rede para o controlador, o SDN permite que as exigências de QoS sejam melhor administradas. Isso ocorre pois, o administrador pode gerenciar, configurar e otimizar os recursos da rede, em tempo real, utilizando *softwares* automatizados na implementação do controlador (KARAKUS; DURRESI, 2017). Além de que, o controlador pode possuir múltiplos algoritmos que atendam às exigências de QoS especificas de cada fluxo da rede. Por fim o uso de protocolos como o*OpenFlow*[2] permite o enfileiramento de pacotes para garantia da taxa de pacotes e garantia de QoS.

O *OpenFlow* é a principal API, utilizada nas redes SDN, para padronizar a comunicação entre as camadas de controle e dados. Os dispositivos da camada de dados, chamados de *OpenFlow switch*, possuem uma ou mais tabelas de roteamento (*flow table*) e um canal de comunicação segura com o controlador (KLAUBERG, 2016).

As tabelas de roteamentos são compostas por regras de correspondência, estatísticas e ações. Quando um pacote é recebido, as regras de correspondência são comparadas com informações presentes no cabeçalho, porta de ingresso ou metadado do pacote. Se alguma das regras é validada, então a ação correspondente da *flow table* é disparada. No caso de não haver correspondências, uma regra conhecida como *table-miss* é utilizada para definir o destino do pacote, que pode ser descartado, encaminhado para

[2] <https://www.opennetworking.org/technical-communities/areas/specification/open-datapath/>

outra *flow table* ou transmitido para o controlador. Por fim, as informações de estatística incluem dados como número de pacotes e *bytes* recebidos (NUNES et al., 2014).

O controlador da rede utiliza o canal de comunicação segura, através do protocolo *OpenFlow*, para criar, remover ou modificar regras da *flow table*. O que pode ser feito de maneira reativa (em resposta ao pacote recebido) ou proativa. Já, os *OpenFlow switches* utilizam o canal, no caminho inverso, para requisitar instruções de processamento de um pacote que não possui um regra de correspondência.

O *Forwarding and Control Element Separation* (ForCES) é outra API que, assim como o *OpenFlow*, prevê a criação das camadas de controle e dados, e padroniza a comunicação entre elas. Criado pela *Internet Engineering Task Force* (IETF), o padrão instancia uma entidade (*Forwarding Element* (FE)) para a camada de dados e outra (*Control Element* (CE)) para a de controle. No ForCES, o FE possui um bloco (*Logical Function Block* (LFB)) que o controlador utiliza para configurar o *hardware* e definir como os pacotes serão processados (NUNES et al., 2014). Porém, diferentemente do que ocorre no *OpenFlow*, as camadas de dados e controle continuam coexistindo no mesmo *hardware*. Por esse motivo, este padrão é menos utilizado na criação de redes SDN, se comparado ao *OpenFlow*.

2.3 SDN em redes de sensores: SDN-WISE

Apesar de atenderem às necessidades de comunicação dos módulos, as RSSFs se tornam complexas de gerenciar e configurar a medida que o número de componentes aumenta. Dessa forma, a utilização da arquitetura do SDN, é interessante devido à possibilidade de retirar a complexidade do processo de roteamento dos módulos limitados em bateria, processamento e memória e entregá-la para a camada de controle. Além disso, segundo Ndiaye, Hancke e Abu-Mahfouz (2017), o potencial visto no SDN se deve à simplificação do gerenciamento e, da reconfiguração da rede devido à existência de um controlador centralizado.

Aplicações de sensoriamento ambiental, por exemplo, que detectam queimadas, poluição e temperatura podem exigir o uso de centenas ou até milhares de sensores espalhados em ambientes críticos, sem acesso à fontes de energia e com instabilidade climática. Portanto, a viabilidade da aplicação depende da facilidade de configuração, roteamento e manutenção dos dispositivos. Por isso, é evidente que o uso de um controlador centralizado, que possui visão completa da topologia da rede e, capacidade de tomada de decisões para garantir o funcionamento da rede facilita e, viabiliza a criação de amplas redes de sensores sem fio.

2.3.1 Visão Geral

O SDN se popularizou graças ao *OpenFlow* que consolidou-se como um dos primeiros protocolos SDN à definir uma API de comunicação entre Camadas de Controle e Dados. Porém, o *OpenFlow* foi pensado para redes cabeadas e, portanto, não pode ser utilizado para amenizar as dificuldades encontradas no gerenciamento das RSSFs em virtude das limitações de *hardware* (memória, bateria e processamento), e da heterogeneidade de dispositivos.

Assim sendo, em Galluccio et al. (2015), os autores se basearam no *OpenFlow* para criar o SDN-WISE, que define uma arquitetura SDN para RSSFs que, permite criar e comunicar controladores da rede, módulos sorvedouros (*Sink*) e fontes (*Source*). O SDN-WISE, diferentemente do *OpenFlow*, implementa funcionalidades que melhoram a eficiência energética da rede. Por exemplo, é possível desligar o rádio dos módulos periodicamente (*duty-cycle*); permite ainda que os módulos agreguem suas mensagens à pacotes oriundos dos vizinhos, que são roteados através deles. Além do mais, o protocolo SDN-WISE prevê que os módulos a capacidade de decisão nos casos em que não seja absolutamente necessária a interação

com o controlador. Isto contribui com a eficiência energética pois diminui o número de comunicação e troca de mensagens na rede.

O comportamento esperado dos módulos que compõe uma rede que utiliza o SDN-WISE foi descrito em Galluccio et al. (2015). Os módulos possuem três estruturas de dados: (i) *WISE States Array*, (ii) *Accepted IDs Array*, e (iii) *WISE Flow Table*, utilizadas para armazenar as informações de roteamento recebidas do controlador da rede.

O SDN-WISE foi projetado para operar em modo *stateful*. O que significa que os módulos podem ser projetados como máquinas de estados finitas, de modo à reduzir a quantidade de interações com o controlador, já que o módulo ganha capacidade de reagir à um dado estado com base em informações locais. o *WISE States Array* é utilizado para armazenar os estados de cada módulo da rede.

Todos os pacotes transmitidos na rede devem possuir uma identificação (ID) que é utilizada nos módulos receptores para verificar se o pacote deve ser processado ou descartado por eles. Esta verificação é possível pois existe uma lista de IDs, chamada de *Accepted IDs Array*, que são comparados ao ID recebido no pacote. Se o ID recebido combina com um dos IDs armazenados, então o receptor processa o pacote de acordo com a instrução repassada pelo controlador. Caso contrário, o pacote é descartado.

Caso haja a necessidade de processar o pacote, o receptor recorre à *WISE Flow Table* para encontrar uma regra que indique uma ação correspondente, que deve ser tomada para realizar o processamento. Se nenhuma regra for encontrada, então o receptor deve requisitar uma nova regra ao controlador. Portanto, o roteamento no SDN-WISE ocorre de acordo com regras de correspondência definidas na *WISE Flow Table*. Quando uma condição definida nas regras de correspondência é atendida, uma ação correspondente é executada e as estatísticas em relação à regra são atualizadas.

Matching Rule					Matching Rule					Matching Rule					Action					Statistics	
Op.	Size	S	Addr.	Value	Op.	Size	S	Addr.	Value	Op.	Size	S	Addr.	Value	Type	M	S	Addr.	Value	TTL	Counter
=	2	0	2	B	>	2	0	10	x_{Thr}	=	1	1	0	0	Modify	1	1	0	1	122	23
=	2	0	2	B	≤	2	0	10	x_{Thr}	=	1	1	0	1	Modify	1	1	0	0	122	120
=	2	0	2	B	-	0	-	-	-	-	0	-	-	-	Forward	0	0	0	D	122	143
=	2	0	2	A	=	1	1	0	0	-	0	-	-	-	Drop	0	0	-	-	100	42
=	2	0	2	A	=	1	1	0	1	-	0	-	-	-	Forward	0	0	0	D	100	32

Figura 8 – *WiseFlowTable* (GALLUCCIO et al., 2015)

Dessa forma, são definidas três seções na *WISE Flow Table*, como mostrado na Figura 8. A primeira seção é a de Regras de Correspondência (*Matching Rules*), que é composta por até três condições (colunas de *Matching Rule*) , que se atendidas, disparam uma ação. O campo S da Regra de Correspondência especifica se a regra é aplicável ao pacote atual ou, ao estado do módulo. Por exemplo, um módulo A que recebe um pacote de B pode executar a ação de acordo com uma informação contida neste pacote ou, com base no estado em que ele se encontra devido à um dado recebido, anteriormente, de um terceiro módulo C. Os campos *Address* e *Size*, indicam respectivamente, o primeiro *byte* e o tamanho da cadeia de caracteres, do pacote ou do estado que devem ser consideradas na regra. O campo *Operator* define o operador relacional a ser utilizado em conjunto com o campo *Value* para verificar se a regra é valida (GALLUCCIO et al., 2015).

A segunda seção da *WISE Flow Table* define a ação a ser executada quando as condições da seção Regra de Correspondência são atendidas. O campo *Type* da seção *Action* pode assumir o valor *"Forward to"* para encaminhar, *"Drop"* para descartar, *"Modify"* para modificar um valor do pacote, *"Send to INPP"* para encaminhar o pacote para a camada *In-Network Packet Processing* (INPP) ou *"Turn off"* para

desligar o rádio do módulo. Os campos *Address* e *Size* auxiliam a execução da ação, isto é, complementam a ação indicando, de acordo com o tipo dela, o endereço do módulo para o próximo salto, probabilidade de descarte, onde modificar o pacote ou o estado, o tipo de processamento a ser realizado na INPP e o tempo restante até que o rádio seja desligado. O campo S, assim como na seção Regra de Correspondência, indica se a ação é referente ao pacote ou ao estado. E o campo M indica se ação é exclusiva ou não. Ou seja, se outra ação deve ser executada após a atual ou se o processamento deve ser finalizado.

A terceira seção armazena as estatísticas referentes ao uso das regras de correspondência, que são utilizadas para verificar a validade das regras. O campo *Time To Live* (TTL) armazena o tempo restante da validade da regra. E o campo *Counter* indica a quantidade de vezes que a regra foi utilizada.

2.3.2 Estrutura dos Pacotes

Os pacotes (*WISE Packets*) projetados para uso nas RSSFs com SDN-WISE possuem no mínimo dez *bytes* de comprimento. Os campos do cabeçalho (Figura 9), segundo Galluccio et al. (2015), são organizados da seguinte maneira:

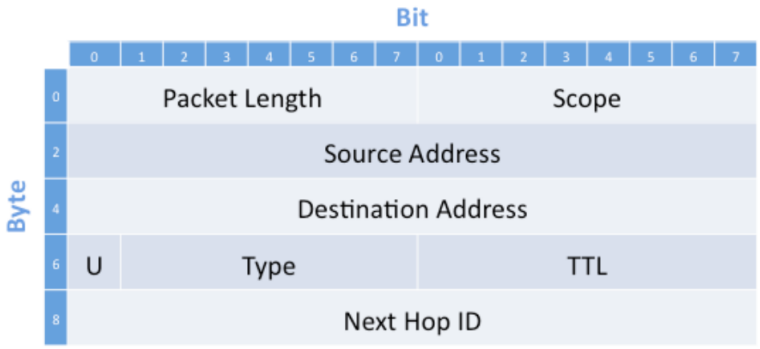


Figura 9 – Cabeçalho do *Wise Packet* (GALLUCCIO et al., 2015)

- ***Packet length*:** Indica o tamanho total do pacote (incluindo o *payload*).
- ***Scope*:** Indica um ou mais controladores da rede que têm interesse no pacote.
- ***Source Address* e *Destination Address*:** Indicam o endereço de origem e destino, respectivamente.
- ***Flag U*:** Indica que o pacote deve ser entregue para o *sink* mais próximo.
- ***Type of packet*:** Indica o tipo do pacote e determina como ele será processado pelo receptor. Existem oito tipos de pacotes. Dessa forma, além de dados, o pacote pode carregar informações da topologia da rede, configuração entre módulos e controlador, configuração da *WISE Flow Table* e instruções para desligamento do rádio por um determinado intervalo de tempo.
- ***TTL*:** Indica o tempo de vida do pacote e, é decrementado de um a cada salto.
- ***Next Hop ID*:** É o ID utilizado pelo receptor para verificar, no *Accepted IDs Array*, se o pacote deve ou não ser processado por ele.

Todos os pacotes transmitidos, internamente, na rede ou para o controlador possuem o mesmo formato do cabeçalho, porém informações distintas no *payload*. A diferença entre eles é destacado no campo *Type* que, como mostrado em SDN-WISE (2017b), pode assumir um dos seguintes valores:

0 ***Data*:** Pacote composto apenas pelo cabeçalho e *payload*.

1 ***Beacon*:** Transmitido através de um *broadcast*, o pacote é utilizado para que os módulos compartilhem informações de bateria e distância do *sink*.

2 ***Report*:** Utilizado para manter o controlador atualizado sobre o estado dos enlaces na rede. O pacote, transmitido pelos módulos da rede, é composto por informações da vizinhança (endereço e indicador *Received signal strength indication* (RSSI)), além da sua distância do *sink* e nível de bateria.

3 ***Request*:** Pacote utilizado para transmitir ao controlador uma requisição de regra de correspondência. Ou seja, a transmissão deste pacote é realizada se, não há uma regra para tratar um pacote recém recebido pelo módulo. Para isso, o módulo encapsula o pacote que não possui regra de correspondência no *payload* da *Request*, e o envia para o controlador.

4 ***Response*:** Pacote, transmitido pelo controlador, que contém a resposta à requisição de uma nova regra de correspondência.

5 ***OpenPath* :** Este tipo de pacote tem como finalidade diminuir a quantidade de pacotes transmitidos na rede. Para isso, o controlador cria um caminho de roteamento e transmite, neste pacote, as condições de uso e, os endereços de todos os módulos que compõem o caminho.

6 ***Config*:** Utilizado na configuração da rede.

7 ***RegProxy*:** Pacote transmitido pelo *sink*, para o controlador, para notificar a sua existência.

2.3.3 Camadas de Dados

Na camada de dados do SDN-WISE, os módulos são classificados como sorvedouros (*sink*) e fonte (*source*). A diferença na arquitetura dos dois, como mostrado na Figura 10, é que o *sink* possui um módulo de adaptação de dados que serve para adequar o formato dos pacotes ao exigido para comunicação com o controlador da rede. Outras três camadas foram projetadas para os módulos da camada de dados:

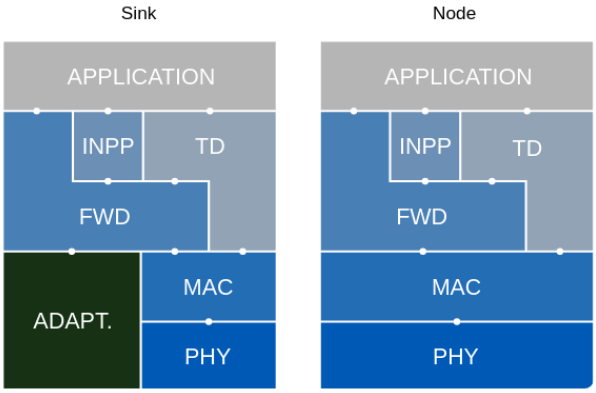


Figura 10 – Arquitetura dos módulos *sink* e *source*. (SDN-WISE, 2017a)

- ***In-Network Packet Processing* :** O INPP contribui para a eficiência energética do módulo, pois é nesta camada que é realizada a agregação de dados. O módulo pode agregar sua própria mensagem à pacotes que são roteados através dele e que serão entregues à um destino em comum.
- ***Topology Discovery* (TD):** A camada TD possui um protocolo que coleta informações, geradas pelos módulos, sobre a topologia da rede e às disponibiliza para o controlador. Além do mais, esta camada é responsável por descobrir e manter atualizado o próximo salto de roteamento em direção

à um dos *sinks* da rede. Assim, os *sinks* são responsáveis por enviar, periodicamente, pacotes de descoberta de topologia (*TD packet*), através de um *broadcast*, para os módulos *source* da RSSF. Neste pacote são enviados o ID do *sink*, informação de bateria e um contador, inicialmente zerado, que indica a distância, em saltos, que um módulo está do *sink*.

De acordo com Galluccio et al. (2015), o protocolo utilizado no TD pode ser descrito em 4 etapas. Supondo que um módulo A recebe um *TD packet* de um módulo B (que pode, ou não, ser um *sink*):

1. O módulo A cria, se já não existe, uma referência para B como um de seus vizinhos. E, armazena as informações de RSSI e nível de bateria de B.
2. O módulo A verifica se já não recebeu um *TD packet* que contenha informação de distância menor. Caso a informação de distância recebida no último pacote seja menor, então uma nova rota, mais rápida, para chegar à um *sink* foi encontrada. Caso contrário o módulo A ignora a informação de distância e apenas incrementa o contador de saltos do pacote.
3. O módulo A atualiza a informação de bateria do pacote com seu nível de bateria.
4. O módulo A retransmite o pacote recebido de B.

Os módulos *source* transmitem, periodicamente, informações sobre a sua vizinhança para o controlador da rede. A taxa de envio dessas informações deve ser escolhida de forma a encontrar um equilíbrio entre manter o controlador atualizado e evitar uma grande quantidade de pacotes na rede (*overhead*).

- **Forwarding (FWD):** O protocolo da camada FWD é o responsável por receber os pacotes e realizar o processamento deles de acordo com as informações recebidas no *header* do pacote e, com as disponíveis na *WISE flow table*.

2.3.4 Exemplo de um cenário simulado

O Cooja é uma ferramenta de simulação de rede de sensores sem fio, utilizada para criar cenários compostos por uma RSSF e um controlador na nuvem. Desenvolvido para simular e depurar códigos de programas escritos para o sistema operacional Contiki, o Cooja permite emular sensores à nível de *Hardware*. Dessa forma, é possível, através das ferramentas disponibilizadas, verificar a comunicação na rede, por exemplo. Além do mais, é possível acessar a RSSF simulada através de portas serial e *Transmission Control Protocol* (TCP).

Dessa maneira, é possível implementar um controlador que se conecte à rede, através de uma das portas disponibilizadas, e se comunique com um dos *sinks*. O controlador utilizado nas RSSFs, que implementam o SDN-WISE, pode ser desenvolvidos na linguagem de programação de preferência do administrador da rede. Bastando que o controlador implemente a API definida pelo protocolo. Por exemplo, os autores do SDN-WISE disponibilizam[3] um controlador, desenvolvido em Java, que se conecta, através de uma porta TCP, à um *sink* executado no Cooja. Ao conectar-se à RSSF, o controlador recebe pacotes de *RegProxy*, seguidos por mensagens de *Report* dos *sinks* da rede, que serão tratados e respondidos com pacotes de configuração ou dados.

Além do controlador, foram disponibilizados uma implementação dos módulos *source* e *sink*. O que possibilita a criação de cenários de testes do SDN-WISE, utilizando o Cooja. Por exemplo, a Figura 11 mostra um cenário em que uma RSSF composta por dez módulos *source* e, um *sink*, se comunica com um controlador sendo executado na nuvem.

[3] http://sdn-wise.dieei.unict.it/docs/guides/GetStarted.html

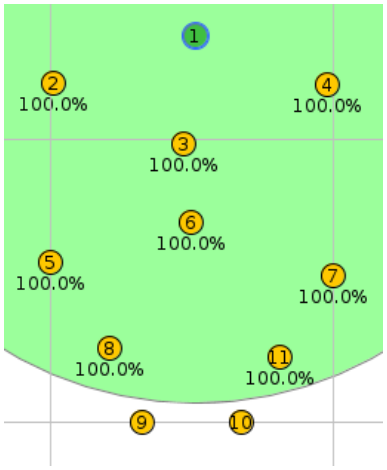


Figura 11 – Possível cenário de uma RSSF executando sobre o SDN-WISE.

Neste cenário, dez módulos *source* são espalhados em um espaço qualquer, sem interferências. Nota-se que, apenas os módulos identificados com o numeral 9 e 10, estão fora do raio de alcance (área em verde) do *sink*. O que obriga que eles transmitam os seus dados, através de múltiplos saltos até o *sink*. Para isso, o controlador deverá enviar regras de correspondência que serão adicionadas às *WISE Flow Table* de cada um dos módulos. Isto é realizado após a recepção, por parte do controlador, das mensagens de *Report* dos módulos da rede. A reação do controlador é iniciar o seu algoritmo de roteamento para encontrar os melhores caminhos na rede que, neste caso, serão aqueles com menor número de saltos e, portanto menor custo, de acordo com o algoritmo de Dijkstra. Ao fim da execução, o novo caminho de roteamento, é transmitido para a rede utilizando os pacotes de *Response* ou de *OpenPath*.

O Cooja oferece ferramentas que possibilitam depurar, em tempo real, as Regras de Correspondências enviadas do controlador para os módulos da rede. Por exemplo, na Figura 12, é possível observar que, o controlador conclui que a melhor forma de rotear pacotes do módulo 10 para *sink* e vice versa, é através do módulo 4. Logo, o módulo 1 insere uma regra em sua *WISE Flow Table* que define o roteamento para o módulo 10, através de 4 e, o módulo 10 recebe uma regra com o caminho inverso.

```
01:09.500  ID:1   Inserting rule IF (P.DST_HIGH == 10) { FORWARD_U 4; } (TTL: 254, U: 0) at position 9
01:09.501  ID:4   Inserting rule IF (P.DST_HIGH == 1) { FORWARD_U 1; } (TTL: 254, U: 0) at position 1
01:09.501  ID:4   Inserting rule IF (P.DST_HIGH == 10) { FORWARD_U 10; } (TTL: 254, U: 0) at position 2
01:09.502  ID:10  Inserting rule IF (P.DST_HIGH == 1) { FORWARD_U 4; } (TTL: 254, U: 0) at position 1
```

Figura 12 – Regra de roteamento através do módulo 4.

2.4 QoS em RSSFs com SDN-WISE

Uma das diferenças a ser destacada no SDN-WISE, em relação ao *OpenFlow*, é a sua capacidade de armazenar estados. Como já mencionado, reagir à recepção de um pacote de acordo com o estado do receptor, diminui o número necessário de comunicações entre módulo e controlador. O que afeta, positivamente, a eficiência energética da rede. Ademais, como demonstrado em Dio et al. (2016), os estados podem ser utilizados para representar o congestionamento percebido pelo módulo e, portanto auxiliar na garantia de QoS da rede.

A ideia proposta em Dio et al. (2016) é utilizar o tamanho da fila do *buffer* de recepção, no qual os pacotes são enfileirados até que sejam processados, para definir os estados do módulo. Por exemplo, um módulo que esteja com sua fila vazia, deve manter-se em um estado inicial *Green* (G). Já quando a fila está cheia, o estado deve ser modificado para *Red* (R) e, o roteamento deve ser alterado para evitar perda

de pacotes. Para isso, o controlador cria regras de roteamento com base no congestionamento percebido pelo módulo e, na categorização dos fluxos da rede, em k classes com prioridades distintas, onde C_1 possui a mais baixa prioridade e C_K, a mais alta. Desta maneira, é possível a diferenciar a forma como os fluxos são tratados e, oferecer garantias de entrega dos pacotes transmitidos.

O mecanismo apresentado propõe o descarte de pacotes e balanceamento do tráfego da rede. No primeiro caso, é utilizada uma probabilidade menor de descarte dos pacotes oriundos de fluxos com prioridade alta, em detrimento daqueles com menor prioridade. De forma a garantir mais recursos da rede para os fluxos mais criticos, ou seja, com maior prioridade. E no segundo caso, o controlador deve sugerir caminhos alternativos ou múltiplos caminhos para evitar o ponto de congestionamento da rede.

	Bit 0-7	Bit 0-7
Byte 0-9	SDN-WISE Header	
10	No. Hop	Battery Level
12	Congestion Level	N
14	$Address_1$	
16	$RSSI_1$	...
18	...	
...	$Address_n$	
...	$RSSI_n$	

Figura 13 – novo pacote SDN-WISE report. (DIO et al., 2016)

Para integrar essa solução de QoS ao SDN-WISE, é proposto, ainda em Dio et al. (2016), que um campo *Congestion Level* seja adicionado ao pacote *Report*, como mostrado na Figura 13. E que, três estados, cujas transições ocorrem de acordo com o tamanho da fila no *buffer* de recepção, sejam utilizados para determinar como um pacote deve ser processado:

1. ***Green* (G):** Representa o estado inicial da fila, em que ela está vazia. Ou seja, não há congestionamento na rede e, portanto não há descarte de pacotes ou necessidade de caminhos alternativos. A transição para o próximo estado ocorrerá somente quando a fila alcançar o limite definido por T_{GY}.

2. ***Yellow* (Y):** Representa o estado em que há pacotes em espera na fila. Porém, o tamanho da fila ainda não atingiu o limite máximo, definido por T_{YR}. Neste estado, o módulo receptor passa a descartar pacotes com uma certa probabilidade, repassada pelo controlador, de acordo com a prioridade do fluxo. Isso é feito com objetivo de garantir a entrega dos pacotes oriundos de fluxos com alta prioridade.

3. ***Red* (R):** Representa o estado em que a fila está cheia. Neste caso, a probabilidade de descarte dos pacotes oriundos de fluxos de baixa prioridade é ainda maior que o repassado, pelo controlador, para o estado Y.

Um possível cenário de uso para a abordagem proposta em Dio et al. (2016), é mostrado na Figura 14. Onde em um primeiro momento T1, apenas os módulos do tipo A, transmitem seus pacotes para o *sink*. Estes pacotes têm prioridade C_3 e são roteados através do módulo 1, que está no estado G. Em um segundo momento T2, o módulo do tipo B inicia sua transmissão utilizando um fluxo com prioridade C_1, o que faz com que o limite T_{GY} do módulo 1 seja atingido, e o estado seja alterado para Y.

Como mostrado na Figura 15, a partir desse momento a probabilidade de descarte dos pacotes transmitidos dos módulos A passa de 0% para 5%. Já, os pacotes do módulo B que, têm a mais alta prioridade, são descartados com probabilidade de 1%. Em um terceiro momento T3, o módulo C inicia

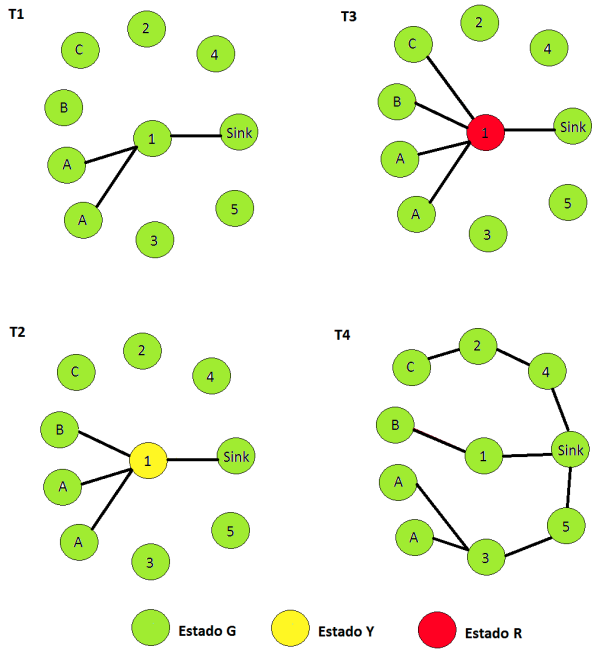


Figura 14 – Possível cenário de congestionamento e garantia de QoS utilizando SDN-WISE.

sua transmissão com prioridade C_2, o que faz com que o módulo 1 fique completamente congestionado e, transite seu estado para R. Com isso, as probabilidades de descarte aumentam para todas as classes de fluxos, sendo que na *Option 2*, o descarte dos pacotes pode chegar a 80% para os fluxos com baixa prioridade. Porém, mesmo neste cenário mais critico, os fluxos com alta prioridade possuem garantia de tráfego.

A transmissão, periódica, dos estados dos módulos para o controlador, permite que o roteamento seja realizado dinamicamente. Dessa forma, diante do cenário apresentado em T3, o controlador pode, por exemplo, definir que os fluxos com prioridade C_3 sejam desviados para um caminho alternativo. O mesmo pode ser feito para os fluxos com prioridade C_2. Resultando na reserva do melhor caminho, escolhido originalmente, para o fluxo do módulo B, que possui prioridade C_1, como mostrado na Figura 14 - T4.

			Option 1	Option 2
	Green State	**Yellow State**	**Red State**	**Red State**
C_1	NO DROP	1%	5%	10%
C_2	NO DROP	3%	20%	45%
C_3	NO DROP	5%	40%	80%

Figura 15 – Probabilidades de descarte de pacotes de acordo com a classe do fluxo. (DIO et al., 2016)

3 PROPOSTA

As limitações de *hardware* dos módulos das RSSFs tornam o gerenciamento e a consequente garantia de QoS, um desafio para os administradores da rede. Por isso, conforme referenciado no Capítulo 1, o objetivo geral deste trabalho é implementar uma rede de sensores sem fio, baseada no conceito de SDN com fins de prover QoS para aplicações que se executam sobre a RSSF.

Para isso, o sistema SDN-WISE será utilizado na comunicação entre as camadas da arquitetura SDN. Na camada de dados, módulos MICAz que serão, em um primeiro momento, emulados no Cooja e, em seguida utilizados em um cenário real, ficam responsáveis por gerar e rotear os dados da rede. E na camada de controle, um algoritmo de roteamento será utilizado para oferecer garantias de vazão e baixo consumo energético para os módulos da camada de dados. Para tanto, o controlador terá acesso às informações dos módulos, da topologia da rede e dos requisitos de QoS da aplicação, por conseguinte, o controlador aplicará as regras de roteamento, dinamicamente, na RSSF.

3.1 Detalhamento da proposta

Para cumprir com os objetivos propostos, o trabalho será dividido em etapas de implantação de uma RSSF baseada no conceito de SDN. A primeira etapa tem como objetivo ganhar experiência com o protocolo SDN-WISE e, implantar uma RSSF com módulos MICAz. Para isso, a ferramenta de simulação de redes Cooja e módulos MICAz reais, serão utilizados para executar uma aplicação concebida para o trabalho. Já na segunda etapa, pretende-se verificar o potencial do conceito de SDN em aplicações com requisitos de QoS.

O Cooja permite simular os módulos MICAz, e utilizar o mesmo código (em linguagem C) desenvolvido para os módulos físicos no ambiente de simulação. Além do mais, o Cooja disponibiliza várias ferramentas que auxiliam na depuração do código e na criação de cenários com grande quantidades de módulos. A Figura 16, mostra algumas das principais ferramentas. A *Network* permite a visualização da troca de pacotes e, o posicionamento dos módulos, de forma gráfica. Já as ferramentas *Mote Output* e *Radio messages*, registram, respectivamente, os *logs* publicados pelos módulos e as mensagens transmitidas através dos seus rádios. Na *Timeline*, é possível observar as transmissões realizadas. Por fim, o Cooja oferece uma interface TCP que, pode ser acessada, através de um terminal externo. O que, neste trabalho, permite a integração do controlador com os módulos simulados. Por estas razões, escolheu-se trabalhar com esta ferramenta para implantar uma RSSF, baseado no conceito de SDN.

Dessa forma, na primeira etapa, será utilizado um experimento básico, disponibilizado pelos autores do SDN-WISE para o Cooja. Neste exemplo, um controlador, desenvolvido em Java, utiliza o algoritmo de Dijkstra para encontrar os caminhos com menor custo. Os módulos (*sink* e *source*), também desenvolvidos em Java, utilizam os caminhos recebidos pelo controlador para rotear os pacotes, gerados de forma aleatória, até o único *sink* da rede. Em seguida, ainda na primeira etapa, pretende-se implantar e analisar o comportamento de uma aplicação sobre a plataforma SDN-WISE no Cooja, utilizando sensores MICAz emulados. Embora já exista o exemplo disponibilizado, a operacionalidade do mesmo sobre módulos MICAz deve ser avaliada, uma vez que os módulos (*sink* e *source*) disponibilizados, são genéricos, ou seja, não foram projetados de acordo com as características de *hardware* do MICAz.

O foco da segunda etapa de desenvolvimento é explorar o conceito de SDN em aplicações com requisitos de QoS. Para isso, o controlador utilizado na primeira etapa deverá ser modificado. A ideia é que

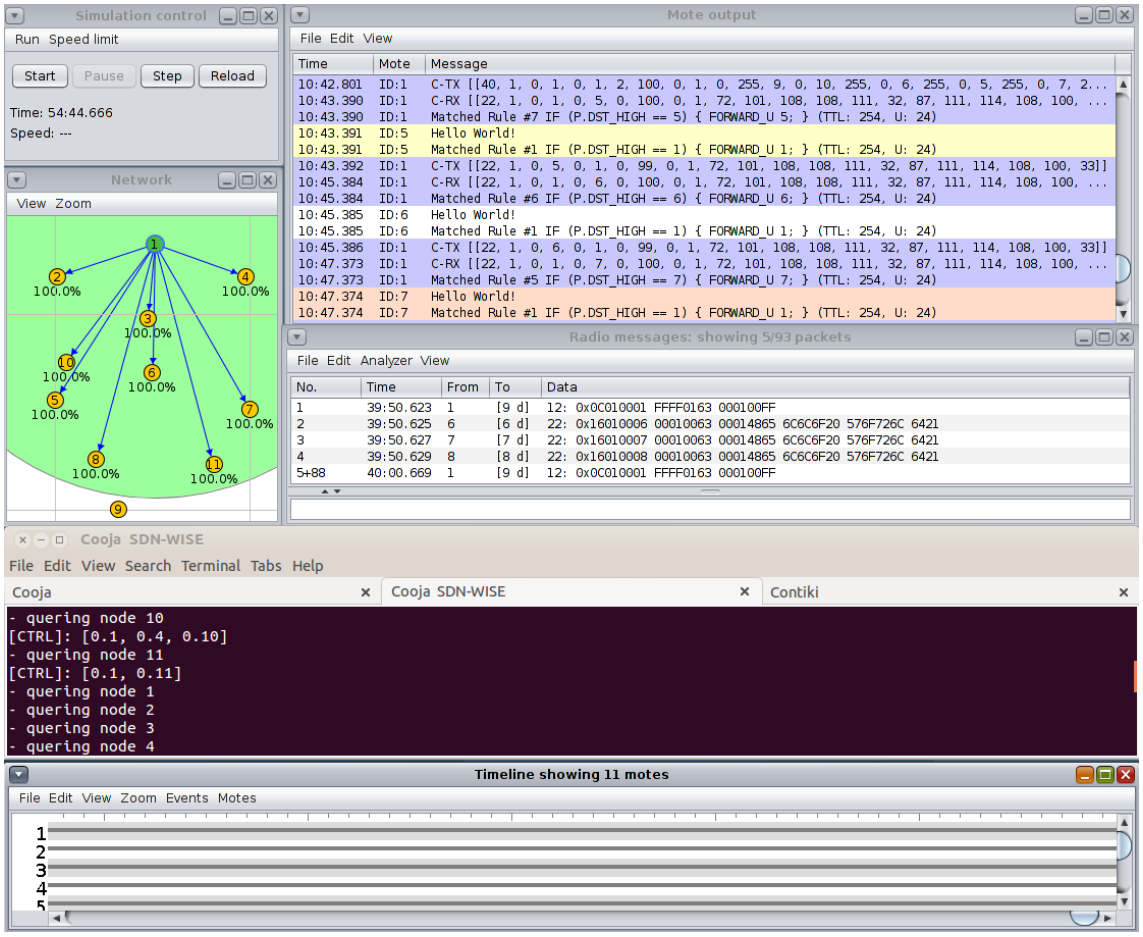


Figura 16 – Visão geral do simulador de redes Cooja.

ele possibilite a configuração estática dos módulos e, dos fluxos que possuem requisitos de QoS associados à vazão e, ao nível de bateria dos nodos, através de arquivos *Extensible Markup Language* (XML). Além disso, o algoritmo de Dijkstra, padrão do controlador, será substituído por um que faça a alocação de fluxos a caminhos da rede de forma a atender aos requisitos de vazão e consumo energético da aplicação.

Além das modificações no controlador, outra medidas podem ser implementadas com objetivo de garantir QoS para a aplicação da rede SDN. Por exemplo, pretende-se verificar a possibilidade do uso dos estados, propostos pelo SDN-WISE, para alterar o roteamento, sem a necessidade de intervenção constante do controlador e, garantir o tráfego de dados na rede, como mostrado na Seção 2.4. Pode-se, ainda, utilizar a capacidade, do SDN-WISE, de ajuste da potência de transmissão dos módulos, para modificar a topologia da rede, uma vez que, a alteração deste parâmetro impacta diretamente nas condições de conectividade da rede.

Para esta etapa, planeja-se também, verificar a melhor forma de avaliação da largura de banda disponível para cada enlace e, como alocar caminhos com banda disponível para fluxos com exigências de QoS. Por fim, pretende-se investigar a possibilidade do uso de algoritmos de roteamento por múltiplos caminhos em RSSFs baseadas no conceito de SDN. Neste sentido, a ideia inicial é dispersar os fluxos através de caminhos disjuntos, explorando o roteamento multicaminho e reduzindo o esforço dos nodos em tempos de trabalho.

Assim, pretende-se, ao fim deste trabalho, apresentar a estrutura mostrada na Figura 17. A partir dela observa-se que o controlador utiliza as configurações recebidas, através de um arquivo XML, em conjunto com informações dos módulos e da topologia da rede para criar regras de roteamento. É

importante mencionar que, o foco deste trabalho não é propor um algoritmo ótimo para garantia de QoS em aplicações da RSSF. Mas sim, analisar a viabilidade do uso do conceito de SDN, para atender aos requisitos de QoS das aplicações das RSSFs e, explorar regras de roteamento, que garantam a maximização da vida dos nodos, ao mesmo tempo que garantam a banda na transmissão de fluxos que, partem de nodos em direção aos sorvedouros da rede.

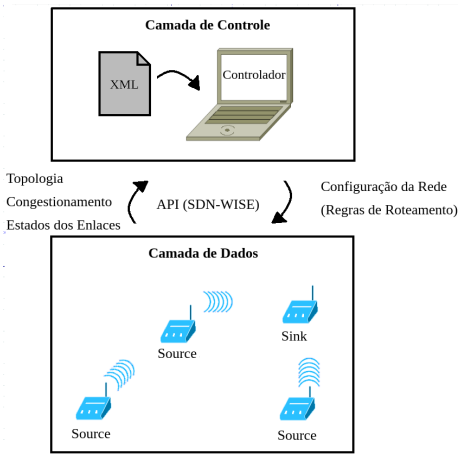


Figura 17 – Arquitetura SDN na RSSF para garantia de QoS

3.2 Recursos, Etapas e Cronograma

3.2.1 Recursos necessários

O controlador da arquitetura SDN, será executado em um *laptop*, e os módulos *sink* e *source* serão implementados com módulos MICAz (Figura 18a). Porém, o *sink* será conectado ao *laptop* através da sua interface de programação (Figura 18b)

3.2.2 Etapas de desenvolvimento

- Implantação, no Cooja, de uma RSSF com arquitetura SDN, utilizando o controlador e, módulos *sink* e *source* disponibilizados pelos autores do SDN-WISE.
- Migração do código dos módulos genéricos para a plataforma MICAz.
- Estudo e implementação de uma aplicação piloto.
- Análise e implementação de algoritmos de roteamento para RSSFs que, ofereçam garantia de vazão e/ou eficiência energética, para a aplicação piloto.
- Criar um cenário real com módulo MICAz físicos.
- Avaliação do comportamento da RSSF implantada no cenário real.

3.2.3 Cronograma

O cronograma mostrado na Tabela 1 será seguido, com o objetivo de implantar uma RSSF, baseada no SDN-WISE para provimento de QoS. Cada etapa de desenvolvimento descrita anteriormente será realizada, em pelo menos, duas semanas. Porém, será realizado o acompanhamento semanal, de acordo com abordagens de gestão, como o Kanban. Como visto na tabela, exceto pela escrita e correção do texto, todas as etapas devem ser feitas separadamente, e na sequência que garanta a implementação da etapa seguinte.

(a) Mote MICAz. (MICAZ, 2013)

(b) Interface de programação para MICAz. (MICAZ, 2013)

Figura 18 – MICAz e programadora necessária para o trabalho.

Etapa	Mês/Semanas											
	Jul 1-2	Jul 3-4	Ago 1-2	Ago 3-4	Set 1-2	Set 3-4	Out 1-2	Out 3-4	Nov 1-2	Nov 3-4	Dez 1-2	Dez 3-4
Implantação no Cooja	X											
Migração para o MICAz		X	X									
Aplicação Piloto				X								
Algoritmos					X	X						
Escrita do TCC2			X	X	X	X						
Cenário Real							X	X	X			
Avaliação										X		
Correções do Texto							X	X	X	X		
Entrega da Monografia											X	
Apresentação à banca											X	
Correções e Entrega Final												X

Tabela 1 – Cronograma das atividades previstas

REFERÊNCIAS

AKYILDIZ, I. F.; MELODIA, T.; CHOWDURY, K. R. Wireless multimedia sensor networks: A survey. *Wireless Personal Communications*, v. 14, n. 6, p. 32–39, 2007. Disponível em: <https://ieeexplore.ieee.org/document/4407225/>. Acesso em: 30 de abril de 2018. Citado na página 12.

AL-KARAKI, J. N.; KAMAL, A. E. Routing techniques in wireless sensor networks: a survey. *IEEE Wireless Communications*, v. 11, n. 6, p. 6 – 28, 2004. Disponível em: <http://ieeexplore.ieee.org/abstract/document/1368893/>. Acesso em: 26 de abril de 2018. Citado na página 16.

ALAZZAWI, L.; ELKATEEB, A. Performance evaluation of the wsn routing protocols scalability. *Journal of Computer Systems, Networks, and Communications*, v. 2008, 2008. Disponível em: <https://www.hindawi.com/journals/jcnc/2008/481046/>. Acesso em: 15 de maio de 2018. Citado na página 16.

ALLIANCE lora. *What is LoRaWAN^{TM}*. [S.l.], 2015. Disponível em: <https://lora-alliance.org/resource-hub/what-lorawantm>. Citado 3 vezes nas páginas 3, 14 e 15.

ANG, L. minn et al. Chapter 2 wireless multimedia sensor network technology. In: _____. *Wireless Multimedia Sensor Networks on Reconfigurable Hardware*: Information reduction techniques. New Delhi, India: Springer-Verlag Berlin Heidelberg, 2013. p. 5–38. Citado na página 12.

CAPELLA, J. C. et al. A reference model for monitoring iot wsn-based applications. *Sensors*, v. 16, n. 11, p. 1816, 2016. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5134475/>. Acesso em: 26 de abril de 2018. Citado 2 vezes nas páginas 9 e 11.

CENTENARO, M. et al. Long-Range Communications in Unlicensed Bands: the Rising Stars in the IoT and Smart City Scenarios. *IEEE Wireless Communications*, v. 23, October 2016. Disponível em: <https://arxiv.org/pdf/1510.00620.pdf>. Acesso em: 5 de maio de 2018. Citado na página 14.

DIO, P. D. et al. Exploiting state information to support qos in software-defined wsns. In: IEEE. *Ad Hoc Networking Workshop (Med-Hoc-Net), 2016 Mediterranean*. [S.l.], 2016. p. 1–7. Citado 4 vezes nas páginas 3, 24, 25 e 26.

GALLUCCIO, L. et al. Sdn-wise: Design, prototyping and experimentation of a stateful sdn solution for wireless sensor networks. In: IEEE. *Computer Communications (INFOCOM), 2015 IEEE Conference on*. 2015. p. 513–521. Disponível em: <https://ieeexplore.ieee.org/abstract/document/7218418/>. Acesso em: 24 de maio de 2018. Citado 5 vezes nas páginas 3, 19, 20, 21 e 23.

GUY, C. Wireless sensor networks. *Sixth International Symposium on Instrumentation and Control Technology: Signal Analysis, Measurement Theory, Photo-Electronic Technology, and Artificial Intelligence*, v. 6357, 2006. Disponível em: <https://www.spiedigitallibrary.org/conference-proceedings-of-spie/6357/63571I/Wireless-sensor-networks/10.1117/12.716964.short>. Acesso em: 2 de maio de 2018. Citado na página 11.

HAMID, Z.; HUSSAIN, F. B. Qos in wireless multimedia sensor networks: A layered and cross-layered approach. *Wireless Personal Communications*, v. 75, n. 1, p. 729–757, 2014. Disponível em: <https://link.springer.com/article/10.1007%2Fs11277-013-1389-0>. Acesso em: 30 de abril de 2018. Citado na página 12.

KARAKUS, M.; DURRESI, A. Quality of service (qos) in software defined networking (sdn): A survey. *Journal of Network and Computer Applications*, Elsevier, v. 80, p. 200–218, 2017. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1084804516303186>. Acesso em: 18 de junho de 2018. Citado na página 18.

KLAUBERG, A. F. Isolamento das redes virtuais (vlan) em uma infraestrutura em nuvem usando uma abordagem sdn/openflow. Monografia (Tecnólogo em Sistemas de Telecomunicações) - Instituto Federal de Santa Catarina, R. José Lino Kretzer, 608 - Praia Comprida, São José - SC, 88103-310, 2016.

Disponível em: <https://wiki.sj.ifsc.edu.br/wiki/index.php/Isolamento_das_Redes_Virtuais_(VLAN)_em_uma_Infraestrutura_em_Nuvem_Usando_uma_Abordagem_SDN/OpenFlow>. Acesso em: 18 de junho de 2018. Citado na página 18.

MBARUSHIMANA, C.; SHAHRABI, A. Comparative study of reactive and proactive routing protocols performance in mobile ad hoc networks. *Advanced Information Networking and Applications Workshops*, 2007. Disponível em: <https://ieeexplore.ieee.org/document/4224182/>. Acesso em: 30 de abril de 2018. Citado na página 17.

MICAZ, M. Wireless sensor networks. *Datasheet*, 2013. Disponível em: <http://www.memsic.com/userfiles/files/Datasheets/WSN/micaz_datasheet-t.pdf>. Acesso em: 18 de junho de 2018. Citado na página 30.

NASIR, H. J. A.; KU-MAHAMUD, K. R. Wireless sensor network: A bibliographical survey. *Indian Journal of Science and Technology*, v. 9, n. 38, 2006. Disponível em: <https://www.spiedigitallibrary.org/conference-proceedings-of-spie>. Acesso em: 5 de maio de 2018. Citado na página 11.

NDIAYE, M.; HANCKE, G. P.; ABU-MAHFOUZ, A. M. Software defined networking for improved wireless sensor network management: A survey. *Sensors*, v. 17, n. 5, 2017. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5469636/>. Acesso em: 20 de maio de 2018. Citado 3 vezes nas páginas 9, 18 e 19.

NUNES, B. A. A. et al. A survey of software-defined networking: Past, present, and future of programmable networks. *IEEE Communications Surveys & Tutorials*, IEEE, v. 16, n. 3, p. 1617–1634, 2014. Citado 2 vezes nas páginas 18 e 19.

OLIVEIRA, G. C. de. Localização indoor utilizando a tecnologia lorawan e aprendizado de máquina. Monografia (Bacharel em Engenharia de Telecomunicações) - Instituto Federal de Santa Catarina, R. José Lino Kretzer, 608 - Praia Comprida, São José - SC, 88103-310, 2017. Disponível em: <https://wiki.sj.ifsc.edu.br/wiki/images/a/ad/TCC290_Giulio_Cruz_de_Oliveira.pdf>. Acesso em: 12 de maio de 2018. Citado na página 15.

RADI, M. et al. Multipath routing in wireless sensor networks: survey and research challenges. *Sensors*, Molecular Diversity Preservation International, v. 12, n. 1, p. 650–685, 2012. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3279234/>. Acesso em: 18 de junho de 2018. Citado 2 vezes nas páginas 3 e 17.

SARAIVA, L. da S. Plataforma de desenvolvimento de aplicações iot utilizando redes lorawan. Monografia (Bacharel em Engenharia de Telecomunicações) - Instituto Federal de Santa Catarina, R. José Lino Kretzer, 608 - Praia Comprida, São José - SC, 88103-310, 2017. Disponível em: <https://wiki.sj.ifsc.edu.br/wiki/index.php/Plataforma_de_desenvolvimento_de_aplica%C3%A7%C3%B5es_IoT_utilizando_redes_LoRaWAN>. Acesso em: 12 de maio de 2018. Citado 2 vezes nas páginas 14 e 15.

SDN-WISE. Sdn-wise. 2017. Disponível em: <http://sdn-wise.dieei.unict.it/>. Acesso em: 24 de maio de 2018. Citado 2 vezes nas páginas 3 e 22.

SDN-WISE. Sdn-wise core. 2017. Disponível em: <http://sdn-wise.dieei.unict.it/docs/guides/Core.html>. Acesso em: 24 de maio de 2018. Citado na página 21.

SEMPREBOM, T. *Explorando descartes de ativações periódicas para provimento de qualidade de serviço em redes IEEE 802.15.4*. Tese (Doutorado) — Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Engenharia de Automação e Sistemas, Campus Reitor João David Ferreira Lima, s/n - Trindade, Florianópolis - SC, 88040-900, 2012. Disponível em: <http://repositorio.ufsc.br/xmlui/handle/123456789/100591>. Acesso em: 5 de maio de 2018. Citado 4 vezes nas páginas 3, 12, 13 e 14.

SHAKSHUKI, E.; XING, X.; MALIK, H. An introduction to wireless multimedia sensor networks. *Handbook of Research on Mobile Multimedia*, v. 2, p. 16, 2009. Disponível em: <https://pdfs.semanticscholar.org/4437/0558cadd9c05916947bd65c2d952f0b217a9.pdf>. Acesso em: 26 de abril de 2018. Citado na página 12.

SHEN, H.; BAI, G. Routing in wireless multimedia sensor networks: A survey and challenges ahead. *Journal of Network and Computer Applications*, v. 71, p. 30–49, 2016. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1084804516301102>. Acesso em: 26 de abril de 2018. Citado na página 12.

SILVA, T. H. B. da. Mapeamento de qualidade de serviço em redes ieee 802.15.4. Monografia (Bacharel em Engenharia de Telecomunicações) - Instituto Federal de Santa Catarina, R. José Lino Kretzer, 608 - Praia Comprida, São José - SC, 88103-310, 2017. Disponível em: <https://wiki.sj.ifsc.edu.br/wiki/index.php/Mapeamento_de_Qualidade_de_Servi%C3%A7o_em_Redes_IEEE_802.15.4>. Acesso em: 2 de maio de 2018. Citado na página 13.

SINGH, S. K.; SINGH, M. P.; SINGH, D. K. Routing protocols in wireless sensor networks – a survey. *International Journal of Computer Science and Engineering Survey (IJCSES)*, v. 1, n. 2, p. 63–83, 2010. Disponível em: <https://pdfs.semanticscholar.org/f376/57052b3c8dc64052a1df8ae2f1959f438c19.pdf>. Acesso em: 16 de maio de 2018. Citado na página 16.