



Universidade Estadual de Feira de Santana
Programa de Pós-Graduação em Computação Aplicada

Uma Proposta para Configuração Dinâmica de Sensores em Cidades Inteligentes

Felipe Pinheiro de Oliveira

Feira de Santana

2019



Universidade Estadual de Feira de Santana
Programa de Pós-Graduação em Computação Aplicada

Felipe Pinheiro de Oliveira

Uma Proposta para Configuração Dinâmica de Sensores em Cidades Inteligentes

Dissertação apresentada à Universidade Estadual de Feira de Santana como parte dos requisitos para a obtenção do título de Mestre em Computação Aplicada.

Orientador: Daniel G. Costa

Feira de Santana

2019

Ficha Catalográfica - Biblioteca Central Julieta Carteado - UEFS

O47 Oliveira, Felipe Pinheiro de
Uma proposta para configuração dinâmica de sensores em cidades inteligentes / Felipe Pinheiro de Oliveira. – 2018.
86 f.: il.

Orientador: Daniel Gouveia Costa.
Dissertação (mestrado) – Universidade Estadual de Feira de Santana, Programa de Pós-graduação em Computação Aplicada, 2018.

1. Redes de sensores sem fio. 2. Redes – Configuração dinâmica.
3. Cidades inteligentes. I. Costa, Daniel Gouveia, orient. II. Universidade Estadual de Feira de Santana. III. Título.

CDU: 004.73

Felipe Pinheiro de Oliveira

Uma Proposta para Configuração Dinâmica de Sensores em Cidades Inteligentes

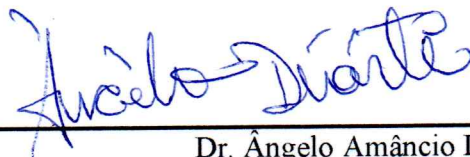
Dissertação apresentada à Universidade Estadual de Feira de Santana como parte dos requisitos para a obtenção do título de Mestre em Computação Aplicada.

Feira de Santana, 21 de dezembro de 2018

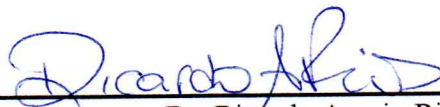
BANCA EXAMINADORA



Dr. Daniel Gouveia Costa (Orientador)
Universidade Estadual de Feira de Santana



Dr. Ângelo Amâncio Duarte
Universidade Estadual de Feira de Santana



Dr. Ricardo Araujo Rios
Universidade Federal da Bahia

Abstract

The development of more efficient communication technologies and the affordability of powerful embedded hardware platforms have brought the smart cities dreams to a more realist setting. Actually, this fertile scenario has fostered the development of different monitoring and control applications aimed at quality life improvement in modern urban areas, often implemented as sensors and actuators networks. Nevertheless, multiple concurrent applications may have different levels of criticality concerning the impact of the performed tasks for the protection of people integrity and thus they may be treated in different ways. This work then proposes an architecture to guide the development of monitoring and control sensor-based applications in smart city scenarios, which will be differentiated based on their monitoring nature when applying prioritization policies. This architecture will also define security mechanisms to ensure that all sensor nodes will respect the defined prioritization for the considered smart cities.

Palavras-chave: Wireless Sensor Network, Prioritization, Smart City.

Resumo

Esta dissertação de mestrado aborda o tema de Configuração Dinâmica em Redes de Sensores Sem Fio propondo uma nova metodologia que visa aprimorar a forma com que as redes que compõem uma Cidades Inteligentes reagem a partir de eventos capturados levando em consideração fatores presentes em uma cidade que influenciam diretamente no resultado final, como clima, tráfego de veículos, temperatura, dia da semana e etc. Cidades são sistemas dinâmicos, ou seja, seu espaço está em constante mudança, seja pela ação do homem ou da natureza. Um evento é um acontecimento com determinada relevância para um sistema de Cidade Inteligente, como uma batida de trânsito, um incêndio, uma ocorrência de assalto, etc. Somente o tipo de evento não é suficiente para mensurar o nível de relevância que uma informação tem perante uma cidade. Por essa razão, esses fatores presentes em uma cidade devem ser levados em consideração em projetos de Cidades Inteligentes, podendo ser determinantes na ação que o nó deve tomar perante ao evento percebido.

O grande desafio que essa dissertação visa solucionar é o de desenvolver um modelo que possa ser utilizado em qualquer projeto de Cidade Inteligente onde seja possível configurar previamente os nós para que esses possam se adaptar ao dinamismo dos centros urbanos. As configurações realizadas nos nós sensores podem ser alterados a depender da necessidade de cada perfil de cidade, pois determinados eventos podem possuir níveis altos de relevância em algumas cidades, porém, para outras esses eventos podem não ter tanta relevância. Por exemplo, um evento identificado como um acidente de trânsito pode ter seu nível de relevância divergente a depender da região da cidade onde ele esteja ocorrendo ou do horário em que ele foi detectado. Para cada cidade, a ação a ser tomada diante desse evento pode variar a depender das situações apresentadas.

Palavras-chave: Redes de Sensores Sem Fio, Priorização, Cidades Inteligentes.

Sumário

Abstract	i
Resumo	ii
Sumário	iv
Lista de Tabelas	v
Lista de Figuras	vi
Lista de Abreviações	vii
1 Introdução	1
1.1 Motivação	5
1.2 Objetivos	6
1.3 Organização do Documento	6
2 Fundamentação Teórica	7
2.1 Rede de Sensores Sem Fio	7
2.1.1 Características de RSSF	8
2.1.2 Aplicações	10
2.1.3 Desafios na Implantação de RSSF	11
2.2 Cidades Inteligentes	12
2.3 Priorização em RSSF	13
2.3.1 Qualidade de Serviço	14
2.3.2 Qualidade de Experiência	16
2.3.3 Escopo em RSSF	16
2.4 Configuração Dinâmica	17
2.5 Mecanismos de autenticação em RSSF	18
3 Metodologia	21
3.1 Modelos de priorização e estado da arte	22
3.2 Formalização do Problema	24
3.3 Síntese da Modelagem Proposta	27
3.4 Metodologia de Desenvolvimento do Trabalho	27

4	Modelo Proposto de Configuração Dinâmica	29
4.1	Cenário considerado	29
4.2	Conceitos fundamentais	30
4.2.1	Estrutura do modelo	30
4.2.2	O conceito de Aplicação	31
4.2.3	Eventos de interesse	33
4.2.4	Parâmetros de priorização	35
4.2.5	Tabelas de priorização	37
4.2.6	Cálculo da prioridade final	40
4.2.7	Transmissões de tabelas e interações dos nós	41
5	Resultados	44
5.1	Especificação formal do modelo	44
5.1.1	Funcionamento Geral do Modelo	44
5.1.2	Análise por Rede de Petri	47
5.2	Simulação utilizando <i>CupCarbon</i>	50
5.2.1	Resultados de simulação (casos isolados)	59
5.3	Experimentação em sensores reais com a plataforma Raspberry	63
5.4	Conclusões da validação	72
6	Conclusão	73
	Referências Bibliográficas	74

Lista de Tabelas

2.1	Parâmetros de Priorização.	17
3.1	Alguns trabalhos sobre priorização em RSSF.	23
4.1	Exemplo de eventos de interesse detectados e prioridades relacionadas, considerando $A = 3$	34
4.2	Exemplos de cálculo de $Pi_{(c)}$	36
4.3	Tabelas de configuração para gerenciamento da priorização.	38
4.4	Exemplo de Tabelas de Eventos para $A = 4$ e $E = 4$	39
4.5	Exemplo de Tabela de Parâmetros para $C = 5$	39
4.6	Exemplos de otimizações baseadas em prioridade.	41
5.1	Tabela de Parâmetros Nó 1 (Período do dia) - Simulação.	52
5.2	Tabela de Parâmetros Nó 1 (Dia da semana) - Simulação.	52
5.3	Tabela de Parâmetros Nó 2 (Período do dia) - Simulação	52
5.4	Tabela de Parâmetros Nó 2 (Dia da semana) - Simulação	52
5.5	Tabela de Parâmetros Nó 3 (Período do dia) - Simulação.	52
5.6	Tabela de Parâmetros Nó 3 (Dia da semana) - Simulação.	53
5.7	Tabela de Ação - Simulação.	53
5.8	Cenário 1: Período do Dia	60
5.9	Cenário 1: Estação do Ano	60
5.10	Cenário 1: Eventos	60
5.11	Cenário 1: Ação	61
5.12	Cenário 2: Clima	62
5.13	Cenário 2: Zona	62
5.14	Cenário 2: Feriado	62
5.15	Cenário 2: Eventos	62
5.16	Cenário 2: Ação	63

Lista de Figuras

3.1	Exemplo de um modelo simplificado para priorização de informação em Cidades Inteligentes.	26
4.1	Topologia Básica de uma RSSF	31
4.2	Exemplo de uma Cidade Inteligente sendo monitorada por três <i>Aplicações</i> diferentes.	33
5.1	Início da Rede	45
5.2	Registro de Novo Nó.	46
5.3	Detecção de Evento.	47
5.4	Rede de Petri - Cadastro.	48
5.5	Rede de Petri - Ação.	49
5.6	Simulação - Primeira Parte.	51
5.7	Simulação - Resultado	54
5.8	Simulação - Cadastro.	57
5.9	Simulação - Nó Cadastrado	59
5.10	Esquema <i>RaspBerry</i>	65
5.11	Funcionamento <i>RaspBerry</i>	66

Lista de Abreviações

Abreviação	Descrição
RSSF	Rede de Sensores Sem Fio
IoT	Internet das Coisas
QoS	Quality of Service
QoE	Quality of Experience
RSSF _M	Rede de Sensores sem Fio Multimídia

Capítulo 1

Introdução

Com os avanços na área de microcontroladores, dispositivos eletrônicos sensoriais e comunicação de redes sem fio, o número de aplicações onde sensores são utilizados para a aquisição de dados do ambiente tem crescido bastante. No geral, torna-se cada vez mais possível o desenvolvimento de sistemas complexos e robustos capazes de realizar diversas formas de sensoriamento, alavancando os estudos na área de Redes de Sensores sem Fio (RSSF).

De maneira geral, uma rede de sensores sem fio consiste em diversos dispositivos (chamados de nós) capazes de realizar algum tipo de sensoriamento, processamento e comunicação, tornando possível a aquisição de dados do mundo real [Sen 2010]. Os conceitos de RSSF vêm sendo aplicados em diversos setores, como na área militar [Hussain et al. 2009], na saúde [Zhang et al. 2014], para aplicações de monitoramento [Papán et al. 2012], entre outras. Porém, diversos desafios relacionados ao desempenho dessas aplicações ainda persistem. Dessa forma, torna-se necessário realizar pesquisas com o objetivo de tratar problemáticas trazidas pela implementação dessa tecnologia, seja com sua estrutura, com questões ligadas à gerência de recursos ou mesmo na segurança da rede e de seus dados.

As RSSF são a base para a implementação de tecnologias em um cenário que está em bastante evidência nos dias de hoje, a Internet das Coisas (*Internet of Things* - IoT). Em [Bin et al. 2010], IoT representa a próxima geração da Internet, onde uma determinada quantidade de nós sensores irá se comunicar, sendo considerada como o núcleo da teoria de *Smart Planet*, proposta pela IBM [Lampkin et al. 2012]. Nessas tecnologias objetos inteligentes são aptos a se comunicar pela Internet, baseado nas novas tecnologias de informação e comunicação.

Com a IoT, o campo das RSSF ganhou ainda mais representatividade, devido ao fato dessas redes de sensores serem responsáveis por gerenciar diversos setores que envolvem o dia a dia da sociedade, como compras, tráfego de automóveis, monitoramento inteligente, suporte a serviços de saúde, etc. De fato, o cenário IoT define a interconexão dos objetos do cotidiano de maneira inteligente, aumentando a ubiquidade da Internet e integrando diversos sistemas embarcados de forma a facilitar

muitas atividades do cotidiano [Xia et al. 2012]. A IoT está abrindo oportunidades para o desenvolvimento de novas tecnologias com o objetivo de melhorar a qualidade de vida da população, porém muitos desafios ainda persistem, sobretudo no que tange a operacionalidade das redes de sensores sem fio em grande escala, como em Cidades Inteligentes.

Seguindo a evolução tecnológica dos últimos anos, as Cidades Inteligentes surgem como espaços urbanos onde o uso de tecnologias de comunicação e informação acontece de forma intensa. Esse conceito envolve três grandes áreas: IoT, *Big Data* (processamento de grandes quantidades de informação) e Governança Algorítmica (gestão e planejamento levando em consideração as ações construídas por algoritmos aplicados à vida urbana). A junção desses três conceitos tem o objetivo de melhorar a qualidade de vida das pessoas que integram os centros urbanos. Com o desenvolvimento das RSSF, grandes cidades já estão utilizando esta tecnologia de forma a melhorar a qualidade de vida da sociedade. Sensores são espalhados pela cidade com o objetivo de funcionarem como captadores ou atuadores na região em que foram instalados.

Uma Cidade Inteligente é composta por diversas situações que acontecem o tempo todo e ao mesmo tempo, dentre essas situações podemos citar: acidentes de carro, incêndio, assalto, engarrafamento ou qualquer outro acontecimento que impacte na dinâmica de uma cidade, a essas situações damos o nome de evento. Para identificar e classificar um evento é necessário distinguir o conceito de dado e evento. Um evento é classificado a partir de um dado que foi obtido, por exemplo, para classificar um evento de incêndio é possível capturar a temperatura do ambiente por meio de algum sensor e considerar que uma temperatura acima de determinado valor é considerado como incêndio. Com esses conceitos, é possível entender a dinâmica da classificação de um evento: o nó captura um determinado dado, esse dado é analisado e, quando cabe classificação, esse dado permite que seja classificado um ou mais eventos.

Um evento pode ser considerado mais importante que outro a depender do nível de criticidade que aquela situação pode impactar em uma cidade. Por essa razão, além de classificar um evento, é necessário também definir a prioridade que esse evento representa para todo o sistema de uma Cidade Inteligente. Quanto maior o nível de prioridade de um evento, maior importância deve ser dada a esse evento de forma que ele possa ser detectado e que seja tomada uma atitude o mais rápido possível para impedir que ele cause consequências catastróficas.

Ao se falar em cidades Inteligentes, alguns cuidados devem ser levados em consideração e assim surgem os questionamentos:

1. Como garantir a segurança dessas interconexões de redes de sensores?
2. Como garantir o funcionamento pleno deste sistema levando em consideração os recursos disponíveis?
3. Como determinar a prioridade de eventos que ocorram nessa cidade de forma a alocar recursos levando em consideração a criticidade da situação?

4. Como definir qual o comportamento de um nó ao detectar um evento?

Este trabalho visa propor uma solução para o questionamento do item 4. A solução proposta se baseia no conceito de configuração dinâmica de um nó sensor para que seja definido de forma dinâmica qual o comportamento do nó baseado na prioridade do evento capturado, no tipo de evento capturado e na aplicação a qual o nó faz parte. Considerando o nó como um sistema isolado, ele é composto por um conjunto de entradas e uma saída. O objetivo da configuração é definir qual a saída desse sistema. A saída deste sistema é baseada em uma prioridade final interna calculada a partir da prioridade do evento capturado e nos parâmetros dinâmicos relevantes. Esta priorização interna pode alterar o comportamento do nó mediante a um evento.

Por ser um sistema considerado bastante dinâmico, outras situações além do evento podem ser consideradas importantes no momento de definir qual a ação que o nó deve desempenhar. A saída do nó pode variar a depender de diversos parâmetros externos, por exemplo: a falta de luz em um bairro pode gerar saídas diferentes para o nó a depender do período do dia. Digamos que essa situação possui maior relevância a noite, nesse tipo de situação a ação tomada pelo nó pode ser se comunicar com a companhia de energia da cidade e a corporação da polícia, devido à periculosidade do local durante a noite. Enquanto a falta de luz durante o dia não necessitasse comunicar a corporação da polícia. Para que esse tipo de configuração funcione de forma apropriada, os nós já devem ser previamente configurados dessa forma, não cabe que esses nós sejam atualizados a todo o momento. Dessa forma, o nó deve perceber todos os parâmetros externos configurados para ele antes de gerar uma saída em seu sistema.

Em [Costa et al. 2017a], é definido o conceito de configuração como centralizado em três diferentes elementos: sensoreamento, programação e transmissão:

1. Sensoreamento: consiste na forma em que o dado é obtido;
2. Programação: algoritmo implantado no nó de forma a definir qual o seu comportamento após obter o dado;
3. Transmissão: como deve ser transmitida a informação levando em consideração fatores como latência e oscilações da rede.

Um fator importante a ser discutido no contexto de *Smart City* é o conceito de “aplicação”. Os nós que compõem o conjunto de redes de uma Cidade Inteligente capturam os dados e classificam eventos a fim de atender alguma aplicação específica. Os mesmos tipos de dados podem atender a finalidades diferentes, ou seja, esses dados podem atender a aplicações diferentes. Uma Cidade é composta por diversos órgãos, entidades, instituições públicas e privadas, etc. E cada entidade pode possuir um número de sensores espalhados pela cidade. Dessa forma a aplicação das informações obtidas pode divergir a depender do interesse e finalidade da entidade. Por essa razão, o tipo de aplicação também interfere no comportamento dos nós em uma *Smart City*. Como exemplos de aplicação podemos citar: monitoramento de

um estacionamento e monitoramento de uma rodovia. Nesses dois exemplos de aplicação são apresentados sensores que utilizam dados visuais capturados por câmeras em dois ambientes diferentes, porém o tipo de dado é o mesmo, mas esse dado é utilizado para aplicações diferentes. Uma aplicação é monitorar um estacionamento e a outra é monitorar uma rodovia.

O sistema proposto utiliza a configuração dinâmica de um nó sensor como forma de recalculá-lo a partir da detecção de um evento. Esse evento passa a não ser o único parâmetro do sistema para definir o comportamento do nó, parâmetros externos podem ser utilizados como entrada que podem ou não alterar esse comportamento. Como esses parâmetros são dinâmicos, o nó deve ser capaz de se autoconfigurar se adaptando a essas mudanças de parâmetros pré-configurados. O sistema proposto apresenta uma modelagem matemática onde é capaz de aplicar a configuração dinâmica em qualquer Rede de Sensores Sem Fio.

Para garantir a funcionalidade do sistema é necessário garantir a autenticidade dos nós. Dessa forma, deve ser levada em consideração a possibilidade de nós falsos tentarem sequestrar informações da rede, o que compromete a autenticação e confiabilidade, partindo da perspectiva onde deve ser garantida o ininterrupto funcionamento da rede, impedindo que nós falsos possam se apoderar de informações que trafegam na rede ou enviar dados maliciosos que não sejam condizentes com o objetivo da rede.

Em geral, os elementos que compõem uma RSSF possuem limitações de recursos. Por serem de tamanho reduzido e por questões de custo, tais elementos não possuem grande quantidade de memória, processamento e energia, porém, a depender do objetivo da rede, a utilização desses recursos pode alcançar níveis elevados e tornar o funcionamento dessa rede impossível [Sen 2010]. Sabendo das limitações de recursos em uma rede de sensores sem fio, os algoritmos pensados como soluções devem ser eficientes. As soluções utilizadas em uma rede de computadores “tradicional” podem não se adequar às demandas e exigências das RSSF.

Um elemento importante para a busca de eficiência em redes de sensores sem fio é a priorização. De modo geral, a priorização consiste na diferenciação de sensores de acordo com alguma característica, e essa diferenciação pode ser explorada para diferentes otimizações da rede. Como a rede possui recursos limitados, tal diferenciação pode ser relevante por garantir mais recursos para os dados mais importantes em determinado momento. Por exemplo, pode acontecer de mais de um nó sensor necessitar utilizar a rede para o envio de alguma informação, porém, se vários sensores resolverem enviar informações ao mesmo tempo, alguns problemas como perda de pacotes podem ocorrer, prejudicando a qualidade de monitoramento da rede [Sharif et al. 2010]. A priorização vem então como uma alternativa para melhoria da eficiência de monitoramento da rede.

Como já investigado em outros trabalhos [Costa et al. 2015] e [Costa et al. 2012], um sensor pode transmitir, em determinado momento, informações mais relevantes para os requisitos de monitoramento da aplicação. Dessa forma, tal sensor deveria

ter uma maior prioridade para enviar os dados coletados do ambiente. E tal prioridade pode ser refletida em diversas situações, como no uso dos recursos de rede (largura de banda, prioridade no encaminhamento de pacotes, gerenciamento de congestionamento, recuperação de erros, etc) e no processamento de dados (algoritmos de codificação e criptografia).

Existem dois escopos bem definidos de priorização: local e global [Costa et al. 2015]. A priorização local (ou de fluxo) tem efeito apenas no mesmo fluxo de informações enviado pelo sensor. Por exemplo, a priorização local ocorre quando pacotes contendo dados visuais (ex: imagens) são mais importantes que pacotes contendo dados escalares (ex: temperatura), considerando a mesma fonte, mas tal esquema depende da aplicação, que pode aplicar a regra de priorização que melhor se adequa às suas especificidades. Por outro lado, a priorização global diz respeito a relevância de um nó como um todo, e essa relevância tem significado para toda a rede (ou parte bem definida dela). Tal escopo de priorização é útil então para definir diferentes relevâncias para os sensores, que são diferenciados pelo potencial impacto que os dados recuperados possuem na qualidade global da aplicação. De maneira geral, ambos os escopos de priorização podem ser combinados em RSSF.

1.1 Motivação

A definição de criticidade é o que pode definir qual o comportamento dos nós que integram uma *Smart City*. E essa decisão pode gerar consequências catastróficas caso seja tomada de forma arbitrária, podendo até mesmo colocar em risco a vida de pessoas que dependam dela. Por exemplo, analisemos a seguinte situação: está acontecendo um incêndio em determinado ponto da cidade, porém a rede definiu que outro evento, não tão grave, tivesse maior relevância. Consequentemente a informação da ocorrência do incêndio seria negligenciada e recebida com atraso ou sem os dados necessários para a tomada de uma decisão imediata, o que geraria consequências negativas. Por essa razão a configuração dinâmica é importante por definir a partir do cálculo da prioridade interna do nó, qual ação deve ser tomada por ele. Essa ação pode ser enviar um determinado tipo de dado na rede, utilizar uma largura de banda maior para que a informação possa trafegar mais rápido, etc.

Devido ao crescimento da utilização das redes de sensores sem fio nas mais diversas aplicações, surgiu a necessidade em se pensar em medidas efetivas que visem solucionar os mais diversos problemas encontrados em RSSF. Sendo o comportamento de um nó diante a um evento um fator de alta relevância em uma rede, percebeu-se a falta de pesquisas efetivas nessa área e estratégias que tivessem como objetivo definir o nível de relevância das informações que circulam em uma RSSF. Então houve a percepção de como esse comportamento pode ser definido a partir de um conjunto de parâmetros.

Por conta do comportamento de um nó diante a um evento em uma *Smart City* ser

uma temática bastante crítica para o funcionamento ideal do sistema, este trabalho apresenta uma modelagem matemática baseada no conceito de configuração dinâmica de um nó sensor que visa solucionar esse problema de forma ampla, para que possa ser adotada por qualquer cidade inteligente, assim, minimizando o risco de que os recursos do sistema sejam alocados para situações não relevantes, o que poderia levar a situações catastróficas por conta do negligenciamento de eventos mais críticos.

1.2 Objetivos

O objetivo geral deste trabalho é propor uma nova abordagem no contexto de configuração dinâmica em Cidades Inteligentes capaz de garantir alocação dinâmica de recursos e priorização de eventos de forma que esse sistema entregue melhores resultados a partir de parâmetros externos inerentes nesses centros urbanos.

A partir do objetivo geral, os objetivos específicos que permeiam este estudo, são:

1. Propor uma solução para Cidades Inteligentes de configuração dinâmica de um nó de forma a definir o comportamento desse nó levando em consideração parâmetros externos.
2. Garantir a eficácia do método desenvolvido de forma que ele possa definir o comportamento de um nó a depender da criticidade definida pelo sistema.

1.3 Organização do Documento

Este documento discute a proposição de uma solução baseada no conceito de configuração dinâmica e autenticação de nós sensores em uma RSSF, com o objetivo de garantir a QoS e QoE de uma RSSF, permitindo um melhor funcionamento e entrega de resultados por essa rede. O restante deste documento encontra-se dividido do seguinte modo: o Capítulo 2 descreve a fundamentação teórica pertinente ao tema; o Capítulo 3 descreve a metodologia aplicada no desenvolvimento do projeto; o Capítulo 4 discute o modelo de priorização proposto para solucionar o problema de priorização em cidades inteligentes; o Capítulo 5 discute os resultados obtidos a partir de uma especificação formal por Rede de Petri, simulação utilizando o *software CupCarbon* e experimentação com *Raspberry*; e o Capítulo 6 discute as conclusões diante do modelo desenvolvido e resultados obtidos.

Capítulo 2

Fundamentação Teórica

Neste capítulo, serão apresentados os conceitos básicos relacionados ao desenvolvimento desse trabalho. Inicialmente, na seção 2.1, serão formalizados os conceitos a respeito de Redes de Sensores Sem Fio, com o objetivo de conceituar melhor esse tema que é a base deste projeto. Em 2.1.1, são discutidas as características de uma RSSF. Em 2.1.2, são demonstradas aplicações onde as RSSF são utilizadas. Em 2.1.3, são discutidos os desafios na implantação de uma RSSF. Na seção 2.2 é definido o conceito de Cidades Inteligentes. Em seguida, na seção 2.3, são discutidos os conceitos de priorização de nós sensores. Em 2.3.1 e 2.3.2, são apresentados os conceitos de QoS e QoE respectivamente, relacionando com a ideia de priorização. Em 2.3.3 são discutidos os tipos de escopo de priorização em RSSF. Em 2.4 são apresentados os principais conceitos de configuração dinâmica. Em 2.5 são apresentados mecanismos de autenticação em RSSF.

2.1 Rede de Sensores Sem Fio

Segundo [Hadim e Mohamed 2006], com a evolução da computação e dos dispositivos eletrônicos, o campo de redes compostas por pequenos sensores vem emergindo. Estes sensores são considerados de baixo custo, baixo poder energético e de fácil implementação. Porém, quando esses sensores são combinados, eles podem fornecer diversas vantagens em relação a redes tradicionais, como:

- Arquitetura flexível
- Alta resolução na aquisição dos dados
- Adaptação ao tipo de aplicação

Com a combinação desses sensores cria-se o conceito de Rede de Sensores Sem Fio. Esses sensores possuem limitações de capacidade de processamento, baixa capacidade de armazenamento e limitação de comunicação devido a baixa largura de

banda. A limitação energética é um fator crítico em uma RSSF, pois ela interfere em todas as outras, por essa razão, existem diversos estudos onde o consumo de energia é considerado o fator principal [Sen 2010].

A ideia de uma RSSF é que sensores sejam distribuídos com a capacidade de realizar o sensoriamento, detectar ocorrências de mudanças e se comunicar a outros dispositivos em uma específica área geográfica para realizar um propósito específico, como segurança, monitoramento do meio ambiente, rastreamento e etc [Pathan et al. 2006].

Cada dispositivo que pertence a uma RSSF é chamada de nó, esse nó tem a capacidade de realizar algum tipo de sensoriamento, coletar e enviar informações. Esses nós podem se comportar de duas formas, a depender de como eles coletam a informação, eles podem ser nós sensores ou nós atuadores [Loureiro et al. 2003].

- Nós Sensores: são dispositivos autônomos com capacidade de sensoriamento, processamento e comunicação. Os dados são coletados, processados e enviados para um *data sink*
- Nós Atuadores: são nós capazes de alterar valores do ambiente que está sendo monitorado. Um exemplo onde esse tipo de sensor pode ser utilizado é em sistemas de segurança contra incêndio, caso o foco do incêndio seja detectado, o próprio atuador se encarrega por liberar água para tentar acabar com o fogo.

Para realizar a comunicação entre esses sensores, alguma tecnologia de comunicação sem fio deve ser utilizada. Para a escolha dessa tecnologia deve-se levar em consideração o quanto de recurso ela pode consumir, sendo que grande parte do consumo de energia se dá no processo de comunicação. O Bluetooth é um candidato natural para interconexões sem fio entre os nós. Uma alternativa ao Bluetooth bastante utilizada tem sido o ZigBee, onde o objetivo é reduzir o custo energético [Cantoni et al. 2006].

Os nós de uma RSSF são sistemas embarcados com recursos como memória, processador, energia e sensores. A depender dessas variáveis, até mesmo um SO (Sistema Operacional) pode ser embarcado nesses sensores. Um dos SO mais utilizados é o TinyOS. Esses sistemas devem ser bastante compactos para se adequarem às limitações dos sensores.

A depender da finalidade da RSSF a configuração da rede pode mudar, seja quanto a topologia ou quanto aos tipos de sensores implantados. Isso significa que uma rede dificilmente será igual a outra. A vasta quantidade de possibilidades de utilização desse tipo de rede é o que traz a motivação para avançar nos estudos em RSSF, para que cada vez mais essa tecnologia possa fazer parte da vida da sociedade.

2.1.1 Características de RSSF

Algumas características provenientes das RSSF são utilizadas em sua classificação. Elas podem ser classificadas quanto ao tipo de dado que os sensores são capazes de

capturar e processar (escalar ou multimídia) ou quanto à infraestrutura (estruturado e não-estruturado).

Os sensores ditos escalares, são aqueles que provêem informações escalares como temperatura, pressão, luminosidade e etc. Já os sensores multimídia, provêem dados em formato multimídia como fotos, vídeos, áudio e etc. Em [Guerrero-Zapata et al. 2010], são discutidas as características de redes multimídia. Os recursos necessários para operar uma RSSFM (Rede de Sensores sem Fio Multimídia) devem ser maiores se comparados aos escalares, isso se dá pela necessidade maior de processamento, memória, largura de banda e bateria. Dados multimídia são mais complexos de serem obtidos, como por exemplo um *streaming* de vídeo. Em redes escalares, as tarefas de processamento extração dos dados são considerados como baixos consumidores de energia, sendo a tarefa de transporte da informação considerada como a mais custosa. Testes mostram que a comunicação representa 91% e 62% do total da energia consumida nos sensores escalares *Telos* e *MicaZ*, respectivamente. Em um sensor multimídia essa tarefa é responsável por apenas 22%, como acontece no *MicroEye*. Esses dados podem variar a depender da arquitetura que essa RSSFM possui, que segundo [Guerrero-Zapata et al. 2010] pode ser:

- *Single-tier flat homogeneous*: nessa arquitetura o *sink* é um *gateway* sem fio conectado ao dispositivo de armazenamento. O processamento ocorre nos próprios sensores de câmera. Realizam funções simples, como detecção de movimento
- *Single-tiered clustered network with heterogenous sensors*: neste tipo de arquitetura, todos os sensores, multimídia ou escalares, retransmitem os dados para um *cluster-head* para realizar o processamento. Esse *cluster-head* está conectado a um *gateway*. Com essa arquitetura é possível os sensores realizarem tarefas mais complexas como reconhecimento de objetos, já que o processamento é passado para um *cluster-head* que possui mais recursos para processar a informação.
- *Multi-tier with heterogenous sensors*: essa é uma arquitetura mais robusta, capaz de processar dados mais pesados como vídeos.

Em [Yick et al. 2008] são discutidas as diferenças entre redes não-estruturadas e estruturadas. As redes que são classificadas como não-estruturadas possuem uma grande quantidade de nós sensores lançadas em um campo sem se preocupar com uma disposição dos sensores, com esse tipo de configuração é possível que existam áreas que não sejam cobertas por esses sensores, além disso, manter esse tipo de rede é bastante complicado, pois não há o gerenciamento efetivo da conectividade e a detecção de falhas é bastante difícil, principalmente quando há muitos nós. Em redes estruturadas, todos os sensores são lançados de forma pré-planejada. A vantagem dessa rede é que uma quantidade menor de nós pode ser lançada com baixa necessidade de manutenção e custo reduzido se comparado às não-estruturadas. Isto acontece pelo fato dessa estrutura de rede utilizar nós dispostos de forma estratégica, otimizando o sensoreamento da região.

2.1.2 Aplicações

As RSSF são bastante utilizadas em diversas áreas, essas aplicações podem ser divididas em duas categorias: monitoramento e rastreamento, essa classificação considera a finalidade da aplicação. Aplicações de monitoramento incluem monitoramento *indoor/outdoor*, nas áreas de saúde e bem estar, energia, fábrica, processos de automação e etc [Yick et al. 2008]. Aplicações de rastreamento incluem rastreamento de objetos, animais, humanos e veículos. Em [Yick et al. 2008] é possível verificar algumas aplicações reais que já estão em funcionamento, como:

- PinPtr: é um sistema experimental para detecção de *snipers*, capaz de detectar a localização desses atiradores. Ele se baseia pelo tempo de chegada do barulho e das ondas de choque do tiro para determinar a localização do atirador.
- *Macroscope of redwood*: é um caso de estudo em RSSF que monitora e grava as árvores do tipo sequoia em Sonoma, Califórnia. Cada sensor mede a temperatura do ar, umidade relativa e radiação solar. Os sensores são posicionados em diferentes alturas da árvore. Esses dados são utilizados por biólogos para estudos.
- Monitoramento subaquático: essa aplicação funciona com o objetivo de monitorar os efeitos da pesca nos recifes de corais a longo prazo. Os nós da rede se comunicam através de *links* ponto a ponto utilizando comunicação óptica de alta velocidade, além de utilizar o TinyOS como sistema operacional.
- CenWits: é uma aplicação que visa auxiliar o processo de busca e resgate utilizando pequenas frequências de rádio baseada em sensores RF um pequeno número de dispositivos de armazenamento e processamento.
- Monitoramento de vulcões: aproveita-se que o custo para se montar uma RSSF é relativamente baixo, dessa forma eles podem ser instalados em vulcões sem a preocupação de perda de algum desses dispositivos. A aplicação em questão conta com 16 sensores que monitoram as atividades do vulcão Reventador no norte do Equador. Em 19 dias, observou-se 230 erupções e outras atividades vulcânicas. 61% dos dados foram retirados dessa RSSF.
- Aplicações de monitoramento de saúde: cinco protótipos foram lançados com o objetivo de monitorar a saúde de crianças, alertando sintomas de surdez e pressão sanguínea.

Diversas são as áreas onde as aplicações de RSSF tem sido bastante efetivas, provendo dados que seriam muito difíceis de serem obtidos sem o uso de uma rede de sensores, como aplicações militares, aplicações para o meio ambiente, aplicações que auxiliam em atividades domiciliares e comerciais [Akyildiz et al. 2002]. É difícil imaginar o mundo sem as RSSF, suas aplicações tornam os processos de monitoramento e rastreamento mais efetivos, quanto mais a tecnologia evolui, mais aplicações surgem.

2.1.3 Desafios na Implantação de RSSF

Com o avanço da tecnologia *wireless*, tornou-se possível estabelecer conexões sem fio cada vez mais leves e baratas, com isso, foi possível tornar o sensoreamento por RSSF algo viável. A grande vantagem do uso de RSSF é o fácil lançamento dos sensores em ambientes externos, como florestas, desertos e selva, além disso, devido ao tamanho reduzido, os sensores realizam um monitoramento de forma não evasiva ao ambiente que está inserido [Cantoni et al. 2006]. Porém, ao mesmo tempo que essas características tornam o uso de RSSF algo positivo, existem alguns desafios que precisam ser enfrentados para que a utilização dessa tecnologia possa ser feita de forma plena, para que seja mantida a continuidade do processo de sensoreamento.

Existem algumas limitações para a implantação de RSSF [Akyildiz et al. 2002]. Elas são:

- Tolerância a falhas: em algumas redes, o funcionamento pode ser interrompido por conta de falhas de seus sensores, seja por falta de energia, danos físicos ou interferências do meio ambiente. Essas falhas não deveriam causar a suspensão do funcionamento de toda a rede, ela deveria estar preparada para tratar essas falhas e manter seu funcionamento. Tolerância a falhas é a habilidade de manter o funcionamento independente de falhas que possam ocorrer.
- Escalabilidade: o número de sensores que são utilizados, a depender da aplicação, pode ser muito grande, centenas, milhares ou até milhões. O sistema deve ser capaz de suportar essa densidade de conexões.
- Custos de produção: o custo de um simples sensor é importante para justificar a utilização de RSSF. Se o custo da RSSF se torna caro, ela perde seu propósito. Dessa forma, o custo de um sensor deve ser baixo e ele deve realizar um número considerável de funcionalidades.
- Limitações de *Hardware*: um nó sensor é construído contendo quatro componentes básicos: unidade de sensoreamento, unidade de processamento, um *transceiver* e um fornecedor de energia. Para algumas aplicações, são adicionados outros componentes a esses nós sensores. Esses componentes ainda possuem outros subcomponentes internamente, como um dos princípios de um nó sensor é que seu tamanho deve ser consideravelmente pequeno, manter o tamanho desses dispositivos no padrão determinado é um desafio. Outros desafios ligados ao *hardware* e que causam limitações a uma RSSF são: a rede deve consumir baixa energia, suportar um grande número de sensores, deve possuir um baixo custo de produção, ser autônomo, se adaptar ao ambiente.
- Topologia: devido a flexibilidade desse tipo de rede, muitos nós podem se tornar inacessíveis, pois aquele nó pode deixar de funcionar e a rede deve continuar sua tarefa, por essa razão a topologia dessas redes devem se adaptar à mudanças frequentes. [Akyildiz et al. 2002] divide a manutenção da topologia em três fases, pré-lançamento, pós-lançamento e relançamento. O pré-lançamento

é a fase em que os sensores são lançados ao campo de sensoriamento, seja jogado de um avião, lançados por algum equipamento, colocado manualmente em locais específicos e etc. O pós-lançamento é fase onde a topologia da RSSF muda, seja por conta da posição dos sensores, seja por falta de acessibilidade, disponibilidade energética, mal funcionamento e etc. O relançamento é a fase onde sensores adicionais são lançados ao campo para substituir os sensores com mal funcionamento, dessa forma a RSSF deve ser capaz de adicionar esses nós à rede.

- Ambiente: as aplicações de nós sensores podem ser utilizadas em diversos terrenos diferentes, seja em ambientes com alta temperatura, úmidos, subaquáticos e etc. Os sensores devem suportar essas situações adversas caso sejam lançados nessas áreas, o que cria um desafio aos fabricantes desses *hardwares*.

2.2 Cidades Inteligentes

Segundo [Atzori et al. 2010], a Internet das Coisas é um novo conceito que rapidamente ganhou o cenário das telecomunicações sem fio. A ideia de IoT é estar presente em diversos objetos que fazem parte da rotina da sociedade, onde sua principal força se encontra na ideia de que sua presença causa um grande impacto na vida das pessoas, seja no campo doméstico, do trabalho, da saúde, da indústria e etc. [Bin et al. 2010] diz que IoT é a próxima geração da Internet, onde trilhões de nós sensores serão conectados para prover algum serviço que tenha impacto na vida da sociedade. O conceito de IoT pode se confundir com o de RSSF, lembrando que o IoT é um paradigma que pode utilizar como base as RSSF para um objetivo específico.

IoT está na lista da *US National Intelligence Council* como uma das seis tecnologias disruptivas (que tem capacidade de derrubar uma tecnologia já estabelecida) com maior impacto no poder nacional dos Estados Unidos. A previsão é que em 2025 a Internet das Coisas esteja em elementos básicos do cotidiano, como nos pacotes de comida, móveis domésticos, documentos e etc [National Intelligence Council NIC 2008]. Porém, para que isso ocorra, muitas questões desafiadoras devem ser tratadas. Em [Atzori et al. 2010] é possível entender quais as medidas que estão sendo tomadas em relação a IoT, de forma a tornar essa tecnologia cada vez mais possível propondo soluções em relação a protocolos, algoritmos, estrutura e etc.

Com o grande impacto causado pela ideia revolucionária trazida pela Internet das Coisas, criou-se um conceito que eleva a IoT a um patamar ainda maior. Essa nova abordagem visa ampliar as fronteiras da IoT de forma que ela consiga abranger um grande centro urbano inteiro, impactando diretamente na vida das pessoas. Esse novo conceito é chamado de *Smart City* ou Cidade Inteligente.

O conceito de Cidades Inteligentes na literatura possui diversas definições, isso ocorre pela generalidade de significados que este termo pode representar. Muitos autores costumam chamar de *Digital City*, *Intelligent City*, *Ubiquitous City* ou *Sustainable City*.

Com o avanço da urbanização, da informação e comunicação e do crescimento econômico nos anos de 1980 e 1990, as pessoas começaram a abandonar as áreas rurais e migrarem para os grandes centros urbanos por conta das oportunidades de emprego e condições econômicas. Para prover uma boa qualidade de vida para essa grande sociedade que se forma, as cidades precisam ser "inteligentes", de forma a fornecer os recursos necessários para tornar a vida da população cada vez mais descomplicada.

O objetivo das *Smart Cities* é utilizar as tecnologias de informação e comunicação disponíveis para tornar os serviços e monitoramento nas cidades mais consciente, interativo e eficiente [Jin et al. 2014]. Dessa forma, as cidades poderão prover uma melhor qualidade de vida a seus cidadãos, melhorando problemas como espaço, mobilidade, energia e etc.

A implementação de Cidades Inteligentes vem fazendo progresso em anos recentes. Uma Cidade Inteligente pode ser considerada como um sistema de sistemas, ou seja, ela é um grande sistema que é composto de sistemas menores com diferentes aplicações. Um sistema de sistemas totalmente integrado deve ter sensoreamento, armazenamento, análise e interpretação. O sistema integrado deve possuir a capacidade de ser reconfigurável, possuir uma política de segurança da informação, possuir Qualidade de Serviço e fácil de ser modificada sem afetar sua operação [Jin et al. 2014].

Uma Cidade Inteligente deve ser projetada para ser capaz de tratar diversos tipos de dados circulem pelos sistemas que a compõe. Esses dados devem ser sincronizados, interpretados, adaptados para propósitos específicos, interconectados com outros dados e distinguido para ser utilizado por uma aplicação específica.

2.3 Priorização em RSSF

A natureza dos dados transmitidos em uma RSSF tem despertado o desenvolvimento de aplicações baseadas em uma abordagem de priorização, onde dados relevantes devem possuir prioridades em sua entrega. Por exemplo, a ideia de priorização em uma rede visual de sensores sem fio, onde o objetivo é priorizar as informações relevantes de dados multimídias, como vídeos e imagens, descartando aquelas informações que não são necessárias no processamento da informação [Costa et al. 2015].

Com o crescimento de aplicações de RSSF, a demanda por métodos eficazes de priorização de dados tem se tornado crescente, esses métodos buscam prover um alto desempenho para a rede. Segundo [Costa et al. 2015], para certas aplicações,

a priorização é um método efetivo quando uma rede possui elementos com significâncias diferentes. Dessa forma, é possível exercer prioridades para cada nó da rede de forma a controlar a utilização dos recursos da rede por aquele nó. Em uma situação em que mais de um nó necessite utilizar a rede, aquele definido como de maior prioridade terá mais recursos disponíveis de forma a realizar sua tarefa com maior desempenho. Alguns outros trabalhos como [Yaghmaee et al. 2013], utiliza a priorização baseada no tipo de dado, diversos sensores são aplicados a pacientes clínicos, eles realizam a tarefa de monitorar alguns sinais vitais desse paciente. A depender da relevância do sinal capturado por esses sensores, essa informação poderá ter prioridade para ser entregue até um computador central que processa essas informações. Em [Lecuire et al. 2007] foi desenvolvido um método de transmissão de imagens onde o consumo de energia e qualidade de imagem foram escolhidos como parâmetro de priorização, a imagem é analisada em forma de um espectro discreto de onda, sendo possível analisar quais as informações relevantes a serem transmitidas e quais poderiam ser descartadas sem causar prejuízos na reconstrução da imagem.

Um fator crítico quando se fala em rede de sensores sem fio é o consumo de energia. Soluções relacionadas a priorização que tenham como foco reduzir o custo de energia são bastante procuradas. Porém, outros fatores que agregam valor a uma rede como disponibilidade, autenticidade, confiabilidade e segurança da informação também devem ser levados em consideração.

Aplicações que utilizam métodos de priorização, devem levar em consideração a utilização dos recursos para atingir o QoS. A camada de transporte é de fundamental importância para garantir a confiabilidade da rede, onde a perda de pacotes pode ser crucial para o bom andamento das tarefas. Os principais fatores que causam a perda de pacotes são congestionamento, colisões quebra de link por meio de falha do nó, e má qualidade do canal de transmissão. A retransmissão de pacote exige um consumo de energia muito alto, o que se torna inviável para aplicações que levam esse fator como prioridade [Sharif et al. 2010].

Priorização é definida como a relação do fluxo de dados com o evento capturado [Costa et al. 2015]. A depender da aplicação, o peso desses valores pode variar, essas duas variáveis atendem os requisitos de definição de prioridade para os escopos local e global.

2.3.1 Qualidade de Serviço

Qualidade de Serviço ou QoS é um termo bastante utilizado com diversos significados e perspectivas [Chen e Varshney 2004]. A depender da aplicação, esse termo pode possuir diversas interpretações, geralmente QoS se refere a qualidade percebida pelo usuário enquanto está utilizando a rede. QoS é aceita como medida da qualidade de serviço que a rede oferece a aplicações ou usuários. Em [Crawley et al. 1998] QoS é citado como um conjunto de requisitos de serviço a serem atendidos ao transportar um fluxo de pacotes da fonte para o seu destino. Essa medida foi criada como

um modo de mensurar medidas de qualidade, como largura de banda, aplicações fim-a-fim, atrasos de pacotes, perda de pacotes e etc.

Em redes cabeadas o QoS geralmente leva em consideração menos parâmetros que em redes sem fio, isso ocorre pela complexidade e diversas variáveis que podem influenciar na qualidade de uma rede quando ela não é cabeada. RSSF faz parte da família das rede sem fio, possuindo características e requisitos próprios.

Como já foi discutido nesse documento, uma RSSF pode atuar coletando diversos tipos de informação, ou seja, existem inimagináveis formas de aplicações de RSSF, cada tipo de aplicação pode ter um parâmetro de QoS próprio, o que torna impossível uma análise de todas individualmente, além de ser improvável que exista uma solução geral que se adeque a tudo.

Em [Chen e Varshney 2004] são apresentadas algumas análises a respeito do tipo de RSSF, por exemplo, em aplicações de rastreamento de objetos, a falha em detectar o objeto ou rastrear um objeto errado pode ser causado por um problema físico, pode acontecer da cobertura da área não está completa e existem momentos em que o objeto é perdido, nesse caso podemos dizer que o número de sensores cobrindo a área de sensoreamento é um parâmetro mensurável de QoS da rede. Em [Chen e Varshney 2004] são definidas duas perspectivas para se classificar o QoS de uma RSSF:

- QoS para aplicações específicas: esta perspectiva considera os parâmetros de QoS como cobertura, exposição, quantidade de erros e número otimizado de sensores.
- Rede QoS: nesta perspectiva a rede pode fornecer os dados do sensor com restrição de QoS, ao mesmo tempo em que utiliza eficientemente recursos de rede.

Pelo fato das RSSF terem que se adequar ao ambiente em que irão atuar, a entrega do QoS pode ficar comprometida, assim, alguns desafios devem ser levados em consideração, [Chen e Varshney 2004] cita alguns:

- Limitação de recursos: as limitações de recursos envolvem energia, largura de banda, memória, tamanho de *buffer*, capacidade de processamento e limitada transmissão de energia. Esses fatores podem ser parâmetros de QoS para a rede, que caso não forem entregues, a qualidade da rede fica comprometida.
- Tráfego desbalanceado: em muitas aplicações de RSSF o tráfego dos dados pode ficar comprometido por conta da grande quantidade de nós e um pequeno conjunto de nós *sink*.
- Redundância de dados: RSSF são caracterizados pela alta redundância dos dados no sensor. Mesmo que a redundância ajude na confiabilidade dos dados, ela também causa um gasto de energia desnecessário. Uma solução seria a fusão ou agregação dos dados, porém, essa opção também resulta em latência e complica o projeto de QoS da rede.

- Balanceamento de energia: a energia é necessária para manter a rede funcionando, distribuir o consumo de energia entre os nós pode manter a rede funcionando por mais tempo do que concentrar todo o consumo em um nó ou um conjunto pequeno de nós.
- Escalabilidade: o QoS da rede não pode se degradar com o aumento de nós da rede, pois esse tipo de implementação pode contar com centenas ou milhares de nós.
- Múltiplos *sinks*: a existência de múltiplos *sinks* pode requerer que diversos tipos de dados estejam circulando entre os *sinks*, o QoS da rede deve ser capaz de associar a relação desses diferentes *sinks*.
- Múltiplos tipos de tráfego: a rede pode conter diversos conjuntos de sensores heterogêneos, onde cada conjunto é responsável por prover um tipo de informação. O QoS da rede deve suportar esses conjuntos de dados heterogêneos.

2.3.2 Qualidade de Experiência

Assim como o conceito de QoS, QoE também possui diversas definições na literatura. QoE é uma medida subjetiva para mensurar a qualidade de experiência do usuário de forma que o produto seja aceito pelo consumidor final. Em [Muhammad et al. 2006] tenta-se diferenciar QoS e QoE. Para ele, QoE visa alcançar o maior número de usuários por meio da experiência ao utilizar os serviços, enquanto QoS visa melhorar a qualidade do serviço de forma a alcançar boas taxas de QoE.

QoE também depende especificamente da aplicação, não existe um método de QoE ótimo para todos os tipos de aplicações, para cada uma em particular, certas características devem ser mensuradas e analisadas quais delas se encaixam melhor para definir sua relevância.

2.3.3 Escopo em RSSF

Nesta subseção são discutidos os conceitos de escopo e parâmetro em redes de sensores sem fio.

Em [Costa et al. 2015] é discutida uma abordagem de priorização que leva em consideração duas variáveis, o escopo e o parâmetro, para definir qual o nível de relevância de um sensor em relação ao escopo abordado. Quanto ao escopo, ele pode ser classificado como:

- Local: nesse tipo de escopo o que é analisada é a relevância de determinado parâmetro para o nó, por exemplo, um nó pode obter dois tipos diferentes de informação, o que deve ser verificado nesse caso é qual delas possui uma maior relevância para o nó.

- Global: nesse tipo de escopo é analisada a relevância do parâmetro de determinado nó para a rede, a depender desse fator, um nível de prioridade é definido para aquele nó em relação aos demais elementos da rede.

Quanto aos parâmetros, [Costa et al. 2015] definiu alguns deles na tabela 2.1.

Tabela 2.1: Parâmetros de Priorização.

Parâmetros	Escopo	Descrição
Energia	Global	Nível de energia do nó sensor
Capacidade de Hardware	Global	Recursos de hardware provenientes para a realização da tarefa, como resolução de câmera, capacidade de memória, etc
Tipo de Dado	Local/Global	Tipo de dado a ser transmitido, como temperatura, pressão, vídeo, imagem, áudio e etc
Codificação da mídia	Local	Codificação e configuração
Relevância da Informação	Global	A relevância da informação obtida pelo sensor
Confidencialidade	Local/Global	Confidencialidade do dado a ser transmitido
Criticidade	Local/Global	O nível de criticidade associado ao dado transmitido

Fonte: [Costa et al. 2015].

Algo que até então não havia sido levado em consideração é como o fator prioridade atua na interação entre redes, ou seja, qual o nível de significância de determinado nó para as demais redes conectadas com a sua e quais parâmetros poderiam definir esse nível de prioridade.

2.4 Configuração Dinâmica

O conceito de configuração dinâmica de nós em Cidades Inteligentes é um tema recente, o que possibilita o desenvolvimento de diversas soluções que envolvam esse tema. Um trabalho que discute bem esse conceito é [Costa et al. 2017a]. Segundo esse trabalho devido as Cidades Inteligentes serem sistemas dinâmicos, a configuração dos elementos da rede devem se adaptar a essas mudanças constantes. Em

[Costa et al. 2017a] o trabalho apresentado utiliza a lógica fuzzy de forma a tornar o sistema adaptativo às mudanças nas cidades Inteligentes.

Os elementos que compõem o conceito de configuração dinâmica são: sensoriamento, codificação e transmissão. Para que a configuração dinâmica ocorra, esses três elementos devem estar contidos no processo. Primeiro o dado é capturado durante o sensoriamento dos elementos da rede, logo após algum tipo de programação deve ocorrer para definir o tipo de configuração que o elemento da rede deverá ter naquele momento e depois a informação deve ser transmitida para a rede.

A configuração dinâmica permite que os nós sensores possam se adaptar às mudanças que ocorrem dinamicamente em uma cidade. A partir desse conceito é possível realizar uma pré-configuração do nó para que ele possa perceber e se adaptar a essas mudanças, de forma a gerarem resultados diferentes a partir dessas mudanças sem a necessidade de interferência humana toda vez que algum parâmetro sofra mudança.

2.5 Mecanismos de autenticação em RSSF

Segurança é algo que deve ser priorizado em qualquer tipo de rede. Os serviços de segurança em RSSF devem proteger as informações que trafegam na rede e os recursos de possíveis ataques que se aproveitam do mal comportamento de um nó. Os principais requisitos para uma RSSF são integridade dos dados, disponibilidade, atualização dos dados, auto-organização, localização segura, tempo de sincronização e autenticação [Sen 2010]. Em uma RSSF, maiores cuidados devem ser tomados devido a exposição que essas redes são submetidas, se tornando alvos fáceis de tentativas de ataque.

Segundo [Pathan et al. 2006], diversos são os ataques a uma RSSF, muitos são comuns a redes de computadores, como DoS, *Blockhole*, *Wormhole*, *Spoofing*, *Sybil Attack* e etc. Cada tipo de ataque visa explorar alguma deficiência da rede para obter dados ou prejudicar seu funcionamento. Ao propor um modelo baseado em priorização para Cidades Inteligentes, a escolha dos métodos de autenticação se torna um fator crítico para o funcionamento da rede. Uma decisão errada, pode acarretar em sérios danos à integridade das informações que estão circulando pelo sistema. Dessa forma, foi realizada uma revisão bibliográfica a fim de analisar alguns métodos de autenticação em RSSF, que sejam possíveis de serem aplicados em uma Cidade Inteligente. Como a estrutura do sistema nas Cidades Inteligentes podem variar de cidade para cidade, não foi definido qual o melhor método a ser aplicado, pois um método pode ser bastante efetivo em uma cidade, porém, ineficiente em outra, essa decisão deve ser tomada no momento da implantação.

Dentro da área de priorização em RSSF, um tipo de ataque que deve ser evitado são os que buscam atingir a autenticidade dos elementos da rede, é de extrema importância que o nó que está atuando na rede seja realmente quem ele se apresenta,

pois a existência de um nó intruso pode acarretar em roubo de informação. Existem diversas propostas que visam se proteger de ataques que exploram a falta de autenticação de uma RSSF.

Em [Melo Jr et al.], é proposto um modelo onde o nó receptor verifica a autenticidade do nó que está enviando a informação por meio de eventos físicos, o identificador do evento deve conhecer, de alguma forma, que tal evento ocorreu. Como um evento é imprevisível, o atacante não sabe de tal situação e quando ele tentar obter algum dado poderá ser identificado como um nó falso na rede.

Em [Patil et al. 2012] são citadas algumas técnicas de autenticação que podem ser utilizadas em RSSF, para ele, autenticação é garantir que os nós sensores, os *cluster head* e as *base station* sejam autenticados antes de conceder recursos limitados ou informações reveladoras. Os processos de autenticação discutidos são:

- *One-way authentication*: uma mensagem é enviada do nó remetente para o nó receptor, essa mensagem é responsável por estabelecer a comunicação entre os dois, ela carrega a identidade do nó garantindo que ela foi enviada por quem diz ser.
- *Two-way / mutual authentication*: é um processo em que ambos os nós se autenticam um com o outro.
- *Implicit authentication*: ele utiliza uma chave para garantir a autenticação. Esse tipo de autenticação pode reduzir a complexidade operacional e minimizar o consumo de energia.

Em [Rajeswari e Seenivasagam 2016] o conceito de autenticação é apresentado de forma ampla, além de trazer alguns métodos de autenticação que podem ser aplicados em RSSF. Autenticação é um processo de identificação de um nó em uma rede, além de garantir que os dados ou as mensagens de controle são originárias de uma fonte autenticada. Alguns populares métodos de autenticação são apresentados a seguir:

- *Lightweight Dynamic User Authentication Scheme*: a RSSF é disposta em uma área confinada e separada em zonas variadas. Esse tipo de método é pode ser utilizado quando um usuário deseja ter acesso à rede. Ela consiste em três fases:
 1. fase de registro,
 2. fase de login,
 3. fase de autenticação.
- *Lightweight Key Management Scheme*: esquema baseado em gerenciamento de chaves. Aplicado para reduzir o consumo de recursos e agir como uma barreira para manter a segurança da rede. Ele se baseia em sequências numéricas para permitir que cada sensor possa estimar um par de chaves distintas com os seus vizinhos. Este método traz como benefício o baixo consumo de memória, baixo consumo de energia e baixo cálculo para geração de chaves.

- EIBAS: *An Efficient Identity-Based Broadcast Authentication*: a estrutura da rede deste modelo possui uma *sink* fixo e diversos sensores móveis. O *sink* é responsável por gerar chaves privadas para os usuários, limitando-se pela capacidade de armazenamento. Este esquema é considerado seguro e leve.

Capítulo 3

Metodologia

O desenvolvimento deste trabalho foi fundamentado nos conceitos de RSSF com o objetivo de propor uma solução baseada em configuração dinâmica e relevância da informação para ser aplicada em Cidades Inteligentes, levando em consideração o tipo da aplicação e de parâmetros dinâmicos comuns a qualquer cidade. Partindo deste ponto, este trabalho propõe uma nova abordagem no contexto de configuração dinâmica para Cidades Inteligentes baseada em parâmetros externos inerentes a esses ambientes, sendo essa abordagem inédita nessa área de pesquisa.

As Cidades Inteligentes definem um “ambiente” composto por múltiplos sistemas concorrentes, que operam em conjunto ou de forma independente para a realização de tarefas que objetivam melhorar a vida das pessoas. Esses sistemas podem ser influenciados por diversos parâmetros que variam a depender da situação. A partir dessa ideia, sistemas implantados nesses ambientes devem ter a habilidade de se auto-configurar dinamicamente, se adequando às mais diversas situações que acontecem nas cidades para que possam entregar resultados satisfatórios que impactem positivamente na vida dos seus habitantes.

Quando consideramos que uma cidade é “inteligente”, assumimos que um conjunto de sensores eventualmente implantados devem ser capazes de capturar eventos e realizar uma ou mais ações em prol da aplicação a qual ele foi designado. Assim, a partir desta ideia de Cidade Inteligente, surge neste trabalho o conceito de *Aplicação*, que é o propósito específico que um conjunto de nós de uma Cidade Inteligente desempenha visando solucionar ou prevenir algum tipo de problema.

Como exemplos de *Aplicação* no contexto deste trabalho, podemos citar:

- Monitorar vagas livres de estacionamento;
- Identificar número e locais de assaltos na cidade;
- Monitorar a temperatura e a umidade relativa do ar;
- Controlar os semáforos de uma cidade, de acordo com o nível de tráfego;

Somente com o conceito de *Aplicação* teríamos um modelo engessado, pois a relevância desse monitoramento seria considerada de forma fixa e padronizada, independente da cidade considerada. A relevância das informações capturadas em uma Cidade Inteligente deve poder variar de acordo com diversos fatores, como por exemplo, a entidade que possui o sensor, o dia da semana, o período do dia, o clima, entre outros. Por essa razão, um modelo de priorização deve estabelecer parâmetros de *Aplicação* diferentes para cada entidade detentora do nó. Partindo deste princípio, o desenvolvimento do modelo proposto visou estabelecer estratégias para atender essa demanda levando em consideração as necessidades de cada cidade.

A abordagem de configuração dinâmica proposta neste trabalho teve como motivação a ideia de desenvolver um modelo capaz de funcionar em qualquer Cidade Inteligente, independente do nível da configuração ou peculiaridades das redes que compõem cada cidade. A partir do modelo desenvolvido, seria possível então definir quais as situações de maior relevância para cada Cidade Inteligente, de forma a priorizar determinadas aplicações onde a informação capturada é considerada mais crítica.

O desenvolvimento do modelo seguiu, então, as etapas definidas a seguir:

- Definição do problema de priorização;
- Definição do escopo do modelo a ser desenvolvido;
- Definição dos parâmetros do sistema a serem considerados;
- Modelagem estrutural;
- Modelagem matemática;
- Validação do modelo por meio de simulação e Rede de Petri;

3.1 Modelos de priorização e estado da arte

Na última década, as redes de sensores sem fio constituíram-se em um dos principais tópicos de pesquisa na área de redes de computadores e comunicação digital. Com essas redes, uma grande gama de aplicações pôde ser desenvolvida, impulsionando o amadurecimento da Internet das Coisas. Contudo, a complexidade relacionada às comunicações nessas redes vem fomentando o desenvolvimento de otimizações na transmissão de dados entre sensores, com atuação direta no roteamento, controle de fluxo, recuperação de erros, acesso ao meio e codificação de dados. Encontrar a melhor forma de realizar tais otimizações é de grande importância para o desenvolvimento ainda maior das redes de sensores.

A otimização de sensores pode ser fortemente baseada na diferenciação das suas relevâncias temporais, como discutido neste trabalho. E, de fato, alguns artigos foram publicados nos últimos anos abordando esse tema em diversos aspectos diferentes [Costa et al. 2015]. A Tabela 3.1 apresenta alguns desses trabalhos.

Tabela 3.1: Alguns trabalhos sobre priorização em RSSF.

Trabalho	Ano	Descrição
[Costa e Guedes 2011]	2011	Sensores mais relevantes, que também participam como nós em rotas de transmissão, recebem menos tráfego de outros sensores, em comparação com sensores menos relevantes. Rotas de transmissão disjuntas são, portanto, definidas de acordo com a relevância dos sensores.
[Costa et al. 2012]	2012	O roteamento/encaminhamento de pacotes em nós sensores é realizado de acordo com a prioridade dos pacotes transmitidos e o nível de energia residual nos nós. A relevância dos pacotes é definida de acordo com o tipo do dado transmitido.
[Costa et al. 2013]	2013	A forma como imagens são codificadas pode produzir pacotes de dados com diferentes níveis de prioridades para as aplicações. Essa prioridade é então explorada na forma como pacotes de dados são encaminhados em redes de sensores sem fio.
[Costa et al. 2014]	2014	Confiabilidade em redes de sensores sem fio é definida de acordo com a prioridade dos pacotes, que é uma função do nível de relevância dos dados transmitidos para a aplicação. Esquemas diferentes de recuperação são então definidos para cada tipo de pacote.
[Peixoto e Costa 2015]	2015	<i>Sinks</i> móveis são posicionados de acordo com áreas que concentram sensores mais relevantes, sendo que a relevância é definida pelo potencial impacto que os dados capturados terão para a qualidade da aplicação.

[Duran-Faundez et al. 2016]	2016	A priorização dos sensores é definida de acordo com a parte dos alvos que está sendo vista pelos sensores considerados. Assim, como alvos podem ter partes mais importantes (como, por exemplo, zona com informações e códigos em um pacote), a priorização de sensores pode ser calculada considerando essa característica.
[Costa et al. 2017a]	2017	A priorização de sensores nesse trabalho considera não apenas a importância dos dados transmitidos, mas também informações externas à rede, como dia da semana, dia do mês, hora do dia, previsão climática, entre outros fatores. Lógica <i>fuzzy</i> é utilizada para calcular a relevância final.
[Costa et al. 2018a]	2018	A partir da mineração de dados em redes sociais, notadamente do Twitter, eventos de interesse podem ser identificados e sensores na área do evento podem ser associados ao nível de priorização calculado.

Os trabalhos apresentados na Tabela 3.1 são apenas alguns exemplos para destacar a relevância dessa área de pesquisa, evidenciando que ainda existem desafios que precisam ser tratados.

3.2 Formalização do Problema

As *Smart Cities*, como foi explicado no Capítulo 2, podem ser definidas como sendo “sistemas de sistemas”, ou seja, elas são compostas por diversos sistemas menores onde há uma comunicação entre eles. Nesse cenário potencialmente complexo, um dos grandes desafios quando se fala em Cidades Inteligentes é quanto a questão da priorização das informações. Levando-se em consideração um sistema adaptativo de acordo com a ocorrência de eventos de interesse, este sistema deve ser capaz de identificar cada evento e transmitir a informação coletada da forma mais eficiente e eficaz possível. De fato, em uma Cidade Inteligente, diversos eventos podem estar ocorrendo ao mesmo tempo, e, portanto, uma política de priorização se faz necessária

para definir o grau de relevância de cada um desses eventos, alocando mais recursos àqueles considerados mais críticos.

Como premissa básica adotada neste trabalho, uma Cidade Inteligente deve ser capaz de se autoconfigurar diante dos eventos percebidos por ela. Um sistema estático que não está apto a se adaptar em face de mudanças é ineficaz nesse contexto, pois os eventos que acontecem em uma cidade nem sempre podem ser previstos, ou melhor, dificilmente podem ser previstos. Uma Cidade Inteligente que não possui uma metodologia dinâmica de priorização pode alocar recursos de forma menos eficiente, podendo ocasionar em consequências graves que irão impactar na qualidade de vida de seus cidadãos.

Uma preocupação em relação às Cidades Inteligentes é quanto a questão da autenticação, sobretudo quando priorizações de nós estão sendo implementadas. O objetivo da autenticação é garantir que uma entidade que esteja solicitando recursos do sistema realmente possa ter acesso a essas informações. Essa verificação é importante por impossibilitar que nós maliciosos tomem o controle do sistema, roubando informações ou enviando dados incoerentes. No contexto de uma Cidade Inteligente, esses nós maliciosos podem impedir que informações relevantes sejam entregues, o que pode comprometer toda a estrutura da rede. Dessa forma, fica claro como a questão da autenticação é primordial para um sistema como uma Cidade Inteligente. Por ser algo bastante variável, a qualidade do método de autenticação pode variar de um sistema para outro, pois o responsável pelo planejamento de um sistema desse nível pode considerar alguns parâmetros mais relevantes que outros.

A proposta apresentada visa complementar sistemas de priorização já utilizados em Cidades Inteligentes, oferecendo um novo nível de definição de relevâncias que atenda especificamente às necessidades do monitoramento nas cidades. Em geral, a forma mais comum de se avaliar a prioridade de um evento é definir um nível de prioridade para cada tipo de evento. Assim, os nós capturam o evento e a partir da prioridade associada, os nós executam uma ação. Para tanto, um modelo simplificado de priorização interno ao nó poderia ser adotado, como apresentado na Figura 3.1.

Essa solução é bastante prática e objetiva, mas uma cidade não funciona, necessariamente, de forma tão objetiva. De fato, diversos são os fatores que podem alterar a relevância de uma informação, como dia, hora, clima, etc. Então ao se pensar em uma Cidade Inteligente é necessário modelar também essas situações subjetivas, tornando-as variáveis da equação. As abordagens propostas na maioria dos trabalhos como em [Yaghmaee et al. 2013], [Avelar et al. 2015], [Djahel et al. 2013] não levam em consideração essa gama de parâmetros externos que podem influenciar no processo de priorização em Cidades Inteligentes. A esses modelos mais simples podemos modelá-los partindo da visão de que eles são compostos por duas estruturas de dados em forma de tabelas, elas são:

- A Tabela de Eventos é composta pelos eventos que podem ser capturados pelos sensores que compõem o nó, juntamente com a prioridade associada.

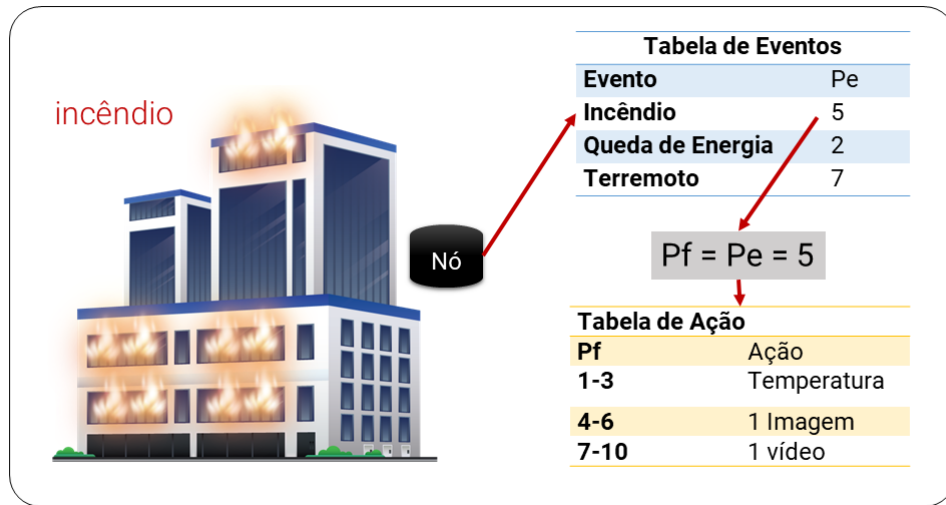


Figura 3.1: Exemplo de um modelo simplificado para priorização de informação em Cidades Inteligentes.

- A Tabela de Ação é utilizada para definir qual a ação a ser tomada pelo nó a partir da prioridade calculada.

Tomando como base o exemplo apresentado na Figura 3.1, é possível notar um nó onde os seus sensores embarcados (unidades de monitoramento) capturaram um evento de incêndio. Levando em conta uma abordagem tradicional de definição de nível de priorização, o evento é classificado por conta da Tabela de Eventos; neste caso específico um incêndio possui um nível de prioridade igual a 5. Como nesse modelo a prioridade final (Pf) se baseia somente na prioridade do evento (Pe), então a prioridade final definida pelo nó também é 5. Esse valor é utilizado para definir a ação que será tomada pelo nó mediante à Tabela de Ação. A partir dos tipos de informações contidas nesta tabela (temperatura, imagem e vídeo), o nó possui pelos menos um sensor de temperatura e um sensor visual. Como é possível notar, a ação tomada pelo nó é a de enviar uma imagem que foi capturada pelo sensor, isso porque a Pf foi definida com o valor 5, e na Tabela de Ação esse valor corresponde ao envio de 1 imagem. O modelo proposto apresentado (que será detalhado no Capítulo 4) visa então aperfeiçoar o critério de análise da Pf a partir do conceito de Aplicação, proposto neste trabalho. A Aplicação leva em consideração o propósito de um conjunto de nós em uma rede, sendo que parâmetros pré-definidos pelo projetista podem interferir na priorização final obtida pelo nó.

3.3 Síntese da Modelagem Proposta

A relevância dos eventos pode variar de cidade para cidade dependendo do que é considerado como sendo mais importante. Por essa razão, utilizar somente a relevância do evento como única variável do sistema de priorização pode gerar um resultado insatisfatório.

Visando aumentar o nível de especificidade desse sistema, foi desenvolvido um novo modelo onde o evento passa a ser apenas uma variável do sistema que será utilizada em conjunto com diversos outros fatores que poderão alterar o resultado final do sistema a depender de situações dinâmicas. Assim que a rede é iniciada, o módulo central de processamento, que pode ser qualquer estrutura computacional capaz de processar as informações recebidas da rede, envia as tabelas para os respectivos nós autenticados na rede para que eles possam realizar o processamento e definir o valor de Pf quando necessário.

De forma a tornar a modelagem do problema mais simples, foram realizadas algumas simplificações matemáticas. É definido que para qualquer parâmetro de priorização, t define um nível de prioridade $P_{(t)}$, como sendo um número positivo considerando o intervalo de $Pmin_{(t)}$ e $Pmax_{(t)}$. Então, $Pmin_{(t)} \leq P_{(t)} \leq Pmax_{(t)}$. Esse conceito é a base para o entendimento a respeito de parâmetro de priorização neste trabalho. Para simplificar, é definido $Pmin_{(t)} = 1$ e $Pmax_{(t)} = 10$ para todos os parâmetros de priorização definidos.

Os parâmetros estabelecidos pelo projetista podem variar o valor da priorização interna final em relação a redes que utilizam apenas a relevância do evento como determinante para definir o valor de Pf . Com a priorização interna final obtida, é possível definir, por meio da Tabela de Ação, qual a ação a ser tomada pelo nó.

A priorização interna final passa a ser obtida pela relação do tipo de evento com os parâmetros considerados como sendo relevantes pelo projetista em determinado momento ou situação.

3.4 Metodologia de Desenvolvimento do Trabalho

Para o desenvolvimento deste trabalho foram definidas etapas que seguiram uma ordem cronológica. Os passos que foram seguidos até a fase atual do trabalho foram:

- Levantamento bibliográfico

Nesta etapa foram levantados os conceitos de Redes de Sensores Sem Fio e Cidades Inteligentes. Foram analisados trabalhos que discutiam o conceito de priorização em RSSF e sua aplicação em Cidades Inteligentes. A partir dos trabalhos analisados, o modelo proposto neste projeto começou a ser estruturado visando atacar o problema tomando como base o aperfeiçoamento de

modelos que utilizam os eventos como parâmetro para a realização de ações pelos nós da rede.

- Definição da solução baseada em configuração dinâmica

Nesta fase foram tomadas as principais decisões de projeto como forma de melhor resolver o problema. Foi desenvolvido um modelo conceitual a partir dos parâmetros definidos. Este modelo foi evoluindo durante o desenvolvimento do projeto a medida que novas necessidades eram percebidas. A base desse modelo é constituída por um conjunto de nós e um módulo de processamento central. Foram definidas as responsabilidades de cada um desses elementos, onde cada nó é responsável por definir a ação a ser tomada a partir dos eventos capturados. O módulo de processamento central pode ser qualquer estrutura capaz de realizar o processamento das informações recebidas e controlar a segurança da rede, incluindo a autenticação dos nós.

Foi desenvolvida uma modelagem matemática com o objetivo de tornar o modelo flexível e robusto para ser adotado por qualquer projeto de Cidades Inteligentes. Para que isso seja possível, foi pensado em um modelo abrangente que não exigisse da rede requisitos além dos necessários para qualquer RSSF operar. A modelagem matemática leva em consideração o nível de prioridade do evento percebido pelo nó e parâmetros voláteis que influenciam na priorização interna final baseado no contexto em que o nó está inserido naquele momento, como, por exemplo, o período do dia, estação do ano, etc.

- Validação baseada em cenários reais

Nesta fase foram projetados cenários baseados em cidades reais onde foram simulados eventos capturados por nós sensores. A partir daí foi possível analisar o comportamento dos mesmos ao processar os eventos. Como os eventos possuem relevâncias já definidas, o papel do modelo proposto é processá-lo e definir uma priorização interna final baseado no conceito de aplicação.

Para atestar a implementação do modelo foram realizados três análises que consistem em: analisar o fluxo de informação e possíveis *deadlocks* por meio de Rede de Petri. Simulação por meio do *software CupCarbon*, onde foi possível simular eventos e sensores reais. Experimentação utilizando *Raspberry*, onde ficou comprovado o funcionamento do modelo utilizando dispositivos que podem ser utilizados como nós em uma Cidade Inteligente.

Capítulo 4

Modelo Proposto de Configuração Dinâmica

Neste capítulo será descrito detalhadamente todo o funcionamento do modelo proposto, desde os conceitos básicos até como deve ser estruturado uma RSSF em Cidades Inteligentes para que o modelo possa responder da forma esperada.

Na seção 4.1 é descrito o cenário onde o modelo de priorização pode ser aplicado e qual a estrutura necessária. Na seção 4.2 são apresentados os conceitos fundamentais para a implantação do modelo proposto, os módulos que fazem parte da estrutura e as decisões de projeto que foram tomadas para apresentar um modelo que possa atender os requisitos de uma Cidade Inteligente.

4.1 Cenário considerado

Nesta seção será descrito o cenário geral onde o modelo de priorização em Cidades Inteligentes proposto pode ser aplicado. O modelo proposto foi desenvolvido com a ideia de ser versátil, o objetivo é que ele possa ser aplicado em qualquer arquitetura de RSSF independente da estrutura da cidade onde está sendo implantada.

Uma Cidade Inteligente é composta por um conjunto de subredes interconectadas formando uma única e grande rede. O modelo proposto não se limita ao número de redes: seja uma cidade com uma rede ou outra com centenas de redes, a arquitetura do modelo de configuração dinâmica atende as necessidades da mesma forma.

Para cada tipo de ocorrência de um evento, diferentes tipos de dados podem ser capturados. Por essa razão, uma cidade pode conter diversos tipos de sensores monitorando um mesmo evento. O modelo de configuração dinâmica não é alterado por conta da heterogeneidade dos sensores em uma RSSF. Isso acontece por conta do modelo levar em consideração o evento já capturado e definido e não o tipo de dado. Por exemplo, a entrada para o modelo deve ser a ocorrência de um incêndio

com prioridade 5 e não que a temperatura está a 100°C. Essa solução para detecção e classificação do evento deve ser implantada previamente na rede, pois não é objetivo do modelo proposto fornecê-la.

O modelo proposto funciona independente da tecnologia utilizada para a comunicação ou topologia da RSSF. O modelo apenas exige elementos que são comuns a qualquer tipo de RSSF, que são os nós e um elemento central que recebe os dados enviados pelos nós e os processa.

4.2 Conceitos fundamentais

Além dos nós que compõem uma RSSF, existe um outro elemento que é responsável por gerenciar todas as redes que compõem uma Cidade Inteligente. Esse elemento é definido neste trabalho como o Módulo de Processamento Central (MPC). Esse módulo gerencia todos os elementos das redes que ele faz parte, podendo ser um *sink*, conjunto de computadores ou qualquer outra unidade computacional capaz de realizar o processamento das informações que circulam pelas redes. O MPC também é responsável por criar políticas de segurança para os nós que compõem a Cidade Inteligente considerada.

4.2.1 Estrutura do modelo

No modelo proposto, o processamento do sistema se divide em dois locais: no conjunto de nós e no MPC. Cada nó é responsável por calcular a prioridade interna a partir do evento capturado e dos parâmetros externos pré-configurados. Como já foi dito anteriormente, o modelo de configuração dinâmica proposto pode ser implantado independente da topologia da rede. Em uma rede de topologia estrela o nó possui comunicação direta com o MPC, com isso o envio das informações não necessita passar por outros nós. Na topologia Mesh é necessário que os nós repassem os dados recebidos do MPC para os nós adjacentes. A Figura 4.1 apresenta exemplos das topologias esperadas.

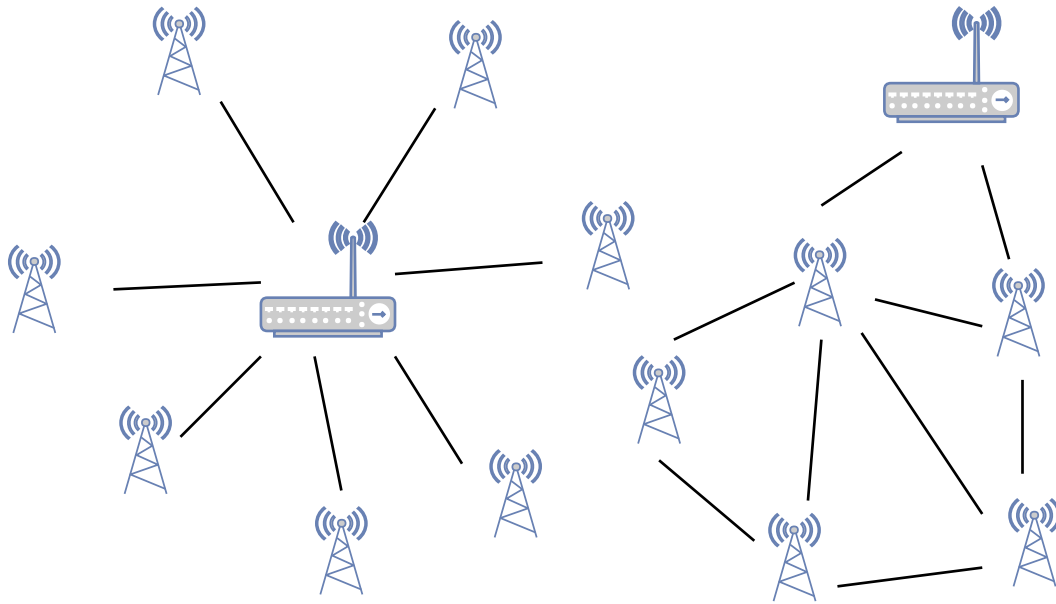


Figura 4.1: Topologia Básica de uma RSSF

A decisão de utilizar o próprio nó que capturou o evento para realizar o processamento e definir o nível de priorização interna final foi tomada com base em uma análise onde foi considerado mais relevante manter o sistema em funcionamento em detrimento de centralizar toda a informação em um único nó mais robusto. A definição da priorização final depende apenas dos dados inerentes ao próprio nó, sendo ele capaz de realizar o processamento, não sendo necessário retransmitir esses dados para que um outro nó realize o cômputo. A partir dessa decisão de projeto é possível garantir o funcionamento do sistema mesmo que uma parte dele seja prejudicada. Caso algum nó seja danificado, o sistema como um todo não será prejudicado, sendo que apenas um nó da RSSF estará inoperante.

4.2.2 O conceito de Aplicação

Rede de sensores podem operar individualmente e ter uma percepção particular de algum evento em uma cidade. Esta percepção individual da rede é bem comum de ser encontrada em trabalhos anteriores na literatura de forma a tentar otimizar redes individuais baseando-se em parâmetros de priorização, como eventos de interesse ([Costa e Guedes 2013], [Goel et al. 2012] e [Ai e Abouzeid 2006]) e parâmetros ambientais/contextuais ([Costa et al. 2017b] e [Zanella et al. 2014]). Em uma perspectiva diferente, este trabalho considera que múltiplas redes de sensores concorrentes podem operar no mesmo cenário e cada uma dessas redes de sensores pode ter uma percepção particular dos eventos. Este novo paradigma é uma das contribuições deste trabalho.

Neste artigo, uma cidade inteligente é definida como um sistema genérico composto de um ou mais sensores, como sensores de temperatura, poluição, radiação UV, som, imagem, etc. Cada sensor também possui informação sobre sua localização [Sahoo e Hwang 2011][Pescaru e Curiac 2014] e endereço de rede, assim essas informações podem ser requisitadas em serviços de priorização, garantindo que dois sensores nunca sejam posicionados no mesmo local. Além disso, na abordagem 2D adotada neste trabalho, em uma cidade inteligente (CI), podemos representá-la como um plano cartesiano com dimensões $CI_m \times CI_n$, assumindo $(0, 0)$ como a origem no vértice superior esquerdo do retângulo criado por CI , é definido o cenário para ser otimizado. Neste cenário, um número Z de redes de sensores serão operadores, onde $z = 0, \dots, Z$, cada rede de sensor z será composto por $S(z)$ sensores. É definido também que todos os sensores possuam energia para executar os protocolos e processos definidos neste trabalho e que cada sensor $s \in S(z)$ esteja localizado na posição $(x_{(s)}, y_{(s)})$, onde $0 \leq x_{(s)} \leq CI_m$ e $0 \leq y_{(s)} \leq CI_n$.

Baseado nesta ideia, foi definido o conceito de “Aplicação”. Uma Aplicação a , onde $0 \leq a \leq A$ sendo A as aplicações definidas. Com isso é definido um subconjunto de sensores que operam sob o mesmo escopo de priorização para sensores pertencentes a qualquer rede de sensores. Portanto, o impacto de qualquer parâmetro de priorização é uma função da definição de Aplicação. Isto é, de fato, uma inovação conceitual de alto nível na percepção de nós sensores.

Na Figura 4.2 é possível ter uma visão macro de como o conceito de Aplicação atua em uma *Smart City*. Na imagem é possível notar alguns prédios com nós conectados às redes dos bombeiros (vermelho), polícia (azul) e prefeitura (amarelo). Alguns eventos estão ocorrendo nesta cidade, como o incêndio no posto de gasolina, a chuva forte numa parte da cidade e um redemoinho.



Figura 4.2: Exemplo de uma Cidade Inteligente sendo monitorada por três *Aplicações* diferentes.

4.2.3 Eventos de interesse

Nos grande centros urbanos costumam acontecer diversos eventos ao longo do dia, continuamente e de forma inesperada. Eventos podem ser qualquer acontecimento que possa afetar o comportamento natural e esperado dos vários elementos que compõem uma cidade, como seus habitantes, veículos, animais e construções. Tradicionalmente, a importância de cada evento dependerá das características de cada sensor, mas isto pode ser considerado de formas diferentes. Na verdade, este trabalho propõem o uso do conceito de Aplicação para guiar as formas em que os eventos de interesse serão processados.

Para a definição de um cenário de uma cidade inteligente, um evento e é qualquer evento de interesse que possa ser percebido em uma cidade inteligente, para $e = 0, \dots, E$, assumi-se E como o último evento detectado. Entretanto, empregando o conceito de “Aplicação”, é definido que um evento genérico será associado a uma Aplicação particular a e que qualquer evento pode se encontrar em dois possíveis estados: detectado ou não-detectado. Logicamente um evento só deve ser analisado em termos de priorização quando ele é detectado. Portanto, para todo evento detectado, haverá uma prioridade associada que é definida como $Pe_{(e,a)}$. Este índice é um valor numérico positivo definido por cada Aplicação, e $Pmin_{(e)} \leq Pe_{(e,a)} \leq Pmax_{(e)}$. A fim de manter a normalização da equação, é aconselhável manter o intervalo de Pe entre os valores de 0 a 10.

A Tabela 4.1 apresenta alguns exemplos de eventos comuns que podem ocorrer em cidades inteligentes. Nesta tabela são mostrados três diferentes aplicações: Departamento de Bombeiros ($a = 1$), Estação Policial ($a = 2$) e Sistema de Mobilidade da Cidade ($a = 3$). Deve-se notar que cada uma dessas aplicações pode incluir sensores de diversas redes de sensores. Os valores utilizados na tabelas são meramente a título de exemplificação.

Tabela 4.1: Exemplo de eventos de interesse detectados e prioridades relacionadas, considerando $A = 3$.

e	Evento	$Pe_{(e,1)}$	$Pe_{(e,2)}$	$Pe_{(e,3)}$
1	Incêndio - Temperatura (Celsius)	10	8	5
2	Engarrafamento - Velocidade média (km/h)	4	6	10
3	Chuva forte - Precipitação (mm/h)	9	2	8
4	Poluição sonora - Som (Decibel)	2	9	2

Os valores de $Pe_{(e,a)}$ dependem das características de cada aplicação que operam em uma cidade inteligente e portanto caso seja encontrado um valor como “10”, deve-se saber que ele possui significado apenas para um escopo particular. Contudo, para a cidade inteligente considerada, qualquer valor de $Pe_{(e,a)}$ terá significado para todo o escopo da cidade inteligente.

Embora eventos possam ser detectados em centro urbanos, diferentes técnicas podem ser empregadas para isto. Na verdade, essas técnicas tem explorado diferentes tipos de dados (*thresholds*, processamento de imagens, mineração de dados, etc.) [Costa et al. 2015, Costa et al. 2018b], cada uma com seus desafios particulares. Neste trabalho, a forma com que os eventos são detectados não faz parte do escopo, a abordagem deste trabalho é com o uso das prioridades de eventos já detectados.

Após a detecção e classificação (definido pelo conceito de Aplicação) de um evento de interesse, o evento pode ser utilizado para definir a prioridade dos sensores. Neste caso, é esperado que a) sensores que detectem um evento sejam definidos com a mesma prioridade do evento, ou b) sensores na mesma área de influência do evento (mesmo não monitorando-o diretamente) sejam definidos com a mesma prioridade. Qualquer que seja a abordagem escolhida, um procedimento que deverá ser feito será associar sensores a eventos e este trabalho atribui a mesma prioridade baseada no evento (Pe) para todos os nós da mesma Aplicação.

Considerando a associação de sensores à prioridades de eventos no contexto de priorização global, já que as prioridades atribuídas são significativas para todo o cenário de cidade inteligente, uma rede de sensores pode ser naturalmente guiada por esse paradigma, potencializando seu desempenho. Contudo, para cidades inteligentes com múltiplas redes de sensores concorrentes, esta abordagem de priorização sensor-evento pode ser ineficiente. A abordagem proposta vem como uma alternativa para enriquecer a percepção de eventos em cenários urbanos.

4.2.4 Parâmetros de priorização

Além da exploração do conceito de eventos de interesse na definição de prioridades, outros parâmetros relevantes também podem ser definidos. Geralmente esses parâmetros são auxiliares e podem contribuir para o refinamento na interpretação de eventos [Costa et al. 2017b]. Alguns parâmetros de priorização comuns em cidades inteligentes são definidos a seguir:

- Condições ambientais: parâmetros como o dia da semana e horário do dia, por exemplo, podem impactar diferentemente no comportamento de uma cidade inteligente e conseqüentemente na forma que as prioridades de sensoreamento são computados;
- Capacidade de sensoreamento: os nós podem possuir diferentes capacidades de sensoreamento, por exemplo: câmeras, microfones e diferentes unidades de sensoreamento. A depender das demandas da aplicação, tais capacidades podem afetar o nível da prioridade final;
- Recursos energéticos: como nós sensores podem operar usando baterias que fornecem uma fonte de energia finita, a energia residual pode ser usada como parâmetro para refinar a prioridade final do nó.

Para uma cidade inteligente genérica CI , espera-se que hajam C diferentes parâmetros de priorização para serem considerados em diferentes eventos. Estes parâmetros (C) são definidos como um conjunto de pares ordenados, associando um intervalo de valores a um índice de prioridade (P_i). Os parâmetros de priorização esperados podem ser definidos considerando diferentes intervalos, possuindo como valor de referência d . Por exemplo, para dias da semana com intervalo de 1 (Segunda) a 7 (Domingo), podemos definir que se $1 \leq d \leq 5$, $P_i = 1$, se $6 \leq d \leq 7$, $P_i = 2$, assegurando neste exemplo que haverá uma prioridade diferente para os finais de semana.

Assumindo que P_i é sempre um número positivo e maior que 0, e que um parâmetro de contexto é definido como c , $c = 0, \dots, C$, para C diferentes parâmetros, é possível definir que $R_{(c)} = \{(V_1 \leq d \leq V_2, P_i = \frac{P_{max}}{W}), (V_2 \leq d \leq V_3, P_i = \frac{2*P_{max}}{W}), \dots, (V_{(W_{(c)})} \leq d \leq V_{(W_{(c)}+1)}, P_i = P_{max})\}$, considerando que $R_{(c)}$ define $W_{(c)}$ diferentes intervalos. Por razões de normalização, o valor de P_i pode variar entre 0 e 10.

Os valores de V_1 a $V_{W_{(c)}+1}$ são intervalos definidos para intervalos específicos do parâmetro c , e assim eles têm que ser configurados adequadamente para cada caso. Portanto, as definições de parâmetro c são feitas em torno da associação de V_n e $V_{W_{(c)}+1}$ para um valor de $Pi_{(c)}$.

Algo a ser considerado nos Parâmetros de Priorização é que eles podem ter uma significância de forma local ou global, como definido em [Costa et al. 2015], mas eles não são associados a uma Aplicação neste artigo. Em outras palavras, “parâmetros” são processados igualmente por todos os nós, uma vez definidos para todo o CI . Essa decisão foi tomada com base no princípio de que os Eventos deveriam orientar a operação das redes e que muitos nós podem até decidir ignorar todos os PC . Contudo, trabalhos futuros podem estender os significados dos parâmetros de priorização adicionais.

A Tabela 4.2 apresenta três exemplos meramente ilustrativos de diferentes parâmetros de priorização.

Tabela 4.2: Exemplos de cálculo de $Pi_{(c)}$.

c	$W_{(c)}$	Descrição	Intervalos ($R_{(c)}$)	$Pi_{(c)}$
1	3	Dia da Semana	$V_1 = 1; V_2 = 4$	$Pi_{(1)} = \frac{Pmax}{W_{(c)}}$
			$V_2 = 4; V_3 = 5$	$Pi_{(1)} = \frac{2 * Pmax}{W_{(c)}}$
			$V_3 = 6; V_4 = 7$	$Pi_{(1)} = Pmax$
2	4	Hora do Dia	$V_1 = 0; V_2 = 6$	$Pi_{(2)} = \frac{Pmax}{W_{(c)}}$
			$V_2 = 6; V_3 = 12$	$Pi_{(2)} = \frac{2 * Pmax}{W_{(c)}}$
			$V_3 = 12; V_4 = 18$	$Pi_{(2)} = \frac{3 * Pmax}{W_{(c)}}$
			$V_4 = 18; V_5 = 24$	$Pi_{(2)} = Pmax$
3	5	Energia Residual	$V_1 = 0J;$ $V_2 = 400J$	$Pi_{(3)} = \frac{Pmax}{W_{(c)}}$
			$V_2 = 400J;$ $V_3 = 800J$	$Pi_{(3)} = \frac{2 * Pmax}{W_{(c)}}$
			$V_3 = 800J;$ $V_4 = 1200J$	$Pi_{(3)} = \frac{3 * Pmax}{W_{(c)}}$

			$V_4 = 1200J;$ $V_5 = 1600J$	$Pi_{(3)} = \frac{4 * Pmax}{W_{(c)}}$
			$V_5 = 1200J;$ $V_6 = 1600J$	$Pi_{(3)} = Pmax$

4.2.5 Tabelas de priorização

Existem algumas informações importantes no processo de priorização que requer o uso do elemento MPC. Primeiro, a configuração dos parâmetros de priorização deve ser realizada de forma central, uma vez que desejamos torná-la dinâmica e válida para todo o escopo da cidade inteligente. Em segundo lugar, como unidade central, o MPC realiza a autenticação dos nós, para que esses possam ser inseridos na rede. Embora um processamento central seja razoável, pode de alguma forma comprometer a escalabilidade e, portanto, algumas cidades inteligentes podem empregar mais de um MPC com conteúdo replicado.

A principal função do MPC é armazenar e distribuir as Tabelas de Prioridade entre os Nós, elas são de dois tipos: Tabela de Eventos (TE) e Tabela de Parâmetros (TP). Além disso, o MPC possui uma tabela de nós (TN), já que deve haver uma maneira de associar os nós a um aplicativo específico. Juntas, essas três tabelas de configuração são o núcleo do processo de priorização proposto.

Assim, ao todo, são definidas quatro tabelas no modelo proposto, complementando soluções “tradicionais”. Essas quatro tabelas são definidas a seguir:

- A Tabela de Nós se encontra no MPC, sendo responsável por armazenar o endereço de rede de todos os nós cadastrados na rede e a qual aplicação ele faz parte. Assim que a RSSF é iniciada, o módulo de processamento central acessa a tabela do roteador e preenche sua Tabela de Nós com os IPs presentes nele. Caso um nó novo deseje se conectar à rede, ele deverá passar por um processo de autenticação para que seu endereço de rede seja cadastrado na tabela. As tabelas de Ação e Eventos ainda permanecem no modelo proposto. Contudo, a Pf da coluna passa a não ser definida apenas pelo Pe da tabela de Eventos. Para encontrar esse valor foi definida uma equação matemática, que é apresentada no Capítulo 5.
- A Tabela de Parâmetros é enviada pelo MPC para o nó, já preenchida. Ela é responsável por conter os parâmetros de aplicação. Em um nó pode haver mais de uma Tabela de Parâmetros; quanto mais dessas tabelas, mais específicos se tornam os resultados.
- A Tabela de Eventos é enviada pelo MPC para o nó. Ela é composta pelos eventos que podem ser capturados pelos sensores que compõem o nó. Tanto dados escalares quanto dados multimídia podem gerar informação interpretável

pelo nó. Um incêndio, por exemplo, pode ser percebido tanto por um sensor visual quanto por um sensor escalar de temperatura. Essa tabela deve ser preenchida com o nível de prioridade de cada evento, uma vez que essa tabela possui duas colunas: uma referente ao nível de prioridade do evento e a outra contendo o identificador do evento (nome, id, etc). Cada informação capturada pelo nó deve ser processada e identificada. Após essa identificação, o nível de prioridade do evento pode ser definido por meio desta tabela.

- A Tabela de Ação é utilizada para definir qual a ação a ser tomada pelo nó a partir do cálculo da Priorização Interna Final (Pf). Essa tabela possui duas colunas, uma referente às ações que o nó é capaz de tomar e a outra contendo um intervalo dos níveis de Pf . A partir desse intervalo, esta tabela é capaz de informar qual a ação que o nó deve realizar. Após a definição da Pf , basta uma consulta do nó a essa tabela para que a ação correta seja tomada. Essa tabela é preenchida a depender dos sensores que compõem o nó, estando as ações tomadas ligadas ao evento capturado.

A tabela 4.3 resume as três tabelas de configuração definidas neste artigo. Essas tabelas precisam ser definidas previamente no MPC, mas podem ser alteradas constantemente, permitindo dinâmicas para sistemas de cidades inteligentes.

Tabela 4.3: Tabelas de configuração para gerenciamento da priorização.

Tabela	Data	Descrição
Tabela de Nós (TN)	Endereço de rede, a	Associa o endereço de rede dos nós à aplicação relacionada (a).
Tabela de Eventos (TE)	$a, e, Pe_{(e,a)}$	Associa uma aplicação (a) e um tipo de evento (e) a um nível de prioridade ($Pe_{(e,a)}$).
Tabela de Parâmetros (TP)	$c, W_{(c)},$ "Intervalos"	Define os intervalos $\{(V_1 \leq d \leq V_2), (V_2 \leq d \leq V_3), \dots, (V_{W_{(c)}} \leq d \leq V_{W_{(c)}+1})\}$ para priorização.

A tabela 4.4 apresenta alguns exemplos da Tabela de Eventos como eles devem ser armazenados no MPC e transmitidos para os nós. O formato apresentado é apenas uma simples sugestão, mas estas tabelas podem ser implementadas de formas diferentes.

Para as Tabelas de Parâmetros, a abordagem adotada é um pouco diferente. Embora não seja influenciado pelo conceito de Aplicação, os intervalos de valores são definidos

Tabela 4.4: Exemplo de Tabelas de Eventos para $A = 4$ e $E = 4$.

a	e	$Pe_{(e,a)}$
1	1	7
1	2	10
1	3	2
1	4	5
1	5	5

a	e	$Pe_{(e,a)}$
2	1	8
2	2	10
2	3	5
2	4	6
2	5	1

a	e	$Pe_{(e,a)}$
3	1	8
3	2	8
3	3	2
3	4	3
3	5	10

a	e	$Pe_{(e,a)}$
4	1	3
4	2	3
4	3	9
4	4	9
4	5	7

para cada tipo de parâmetro. Portanto, uma única tabela deveria ser armazenada, mas com uma variação no número de “colunas”. Para uma melhor representação, tais tabelas podem ser implementadas como mais de uma tabela (quando há uma relação entre elas) ou usando estruturas dinâmicas como “*arrays*”. O exemplo apresentado na Tabela 4.5 apresenta uma forma de construir este tipo de tabela de configuração. Os valores encontrados nessa tabela são ilustrativos, foram utilizados apenas para demonstrar a estrutura desse tipo de tabela.

Tabela 4.5: Exemplo de Tabela de Parâmetros para $C = 5$.

c	$W_{(c)}$	$R_{(c)}$
1	2	$\{(1,2),(2,5)\}$
2	4	$\{(0,10),(10,20),(20,30),(30,40)\}$
3	6	$\{(1,2),(2,3),(3,4),(4,5),(5,6)\}$
4	4	$\{(10,50),(50,60),(60,200),(200,1000)\}$
5	2	$\{(1,5),(6,7)\}$

As tabelas são encontradas tanto nos nós sensores, quanto no MPC. Um cuidado maior deve ser tomado ao decidir pela estrutura de dados utilizada para representar a tabela em cada nó, uma vez que estruturas de dados muito pesadas podem comprometer o desempenho do nó. Por essa razão é recomendável um conhecimento da linguagem de programação embarcada no nó, buscando as soluções mais eficientes possíveis.

4.2.6 Cálculo da prioridade final

As tabelas de configuração fornecem as informações necessárias sobre as regras de priorização adotadas em uma cidade inteligente específica, válidas para todas as redes de sensores que estão operando nesse escopo definido. Depois de definir os valores apropriados nas tabelas, os nós devem receber as tabelas, conforme apresentado em 4.2.7. Com as tabelas, os nós podem finalmente calcular sua prioridade.

Toda a priorização é computada local e individualmente por cada nó, depois de receber as tabelas de configuração. Cada nó irá computar um único índice de priorização que é válido e significativo para toda a cidade inteligente, referido como $Pf_{(s)}$. Como os valores de Pi e Pe serão algo entre $Pmin$ e $Pmax$, podemos esperar que $(2 * Pmin) \leq Pf \leq (2 * Pmax)$. A formulação em (4.1) apresenta o cálculo de $Pf_{(s)}$. Como o valor de $Pf_{(s)}$ pode ser um número de ponto flutuante, uma abordagem típica será arredondá-lo, de acordo com as preferências de configuração nó. Foram definidas as variáveis fc e fe para serem utilizadas em situações onde um tipo de prioridade tenha maior relevância que outra. Por padrão o valor delas é 1.

$$Pf_{(s)} = \left(\frac{\sum_{(c=0)}^C (Pi_{(c)})}{C} \right) . fc + \left(\frac{\sum_{(e=0)}^E (Pe_{(e)})}{E} \right) . fe \quad (4.1)$$

Há algumas observações interessantes sobre a Equação (4.1). Primeiro, os valores de $Pi_{(c)}$ não vêm diretamente da Tabela de Parâmetros, portanto, eles precisam ser computados como apresentados na Tabela 4.2. O nó os calcula localmente com base em sua configuração. Por exemplo, se $c = 1$ estiver configurado como “dias da semana”, o nó usará sua referência de horário local, os intervalos definidos para $c = 1$ e $W_{(1)}$ (informações apresentadas na Tabela de Parâmetros) para calcular o valor de $Pc_{(1)}$. E o mesmo é verdade para todas as linhas no TP. Por razões de normalização, o valor de Pf pode variar entre 0 e 20.

A segunda observação é sobre os valores de $Pe_{(e)}$. Como os eventos podem ser detectados (“ON”) ou não detectados (“OFF”), os valores apresentados na Tabela de Eventos são válidos somente se o nó detectar o evento correspondente. Por exemplo, para $e = 1$ como o evento “incêndio”, um nó deve saber previamente que um incêndio é detectado se e somente se a temperatura detectada for maior que 50°C (ou qualquer outro valor configurado). Se esse limiar for alcançado, o valor de $Pe_{(e)}$ muda de 0 (não detectado) para o valor em TE para $Pe_{(1)}$, por exemplo, 8. Neste caso, como a informação sobre o aplicativo já foi considerado no MPC, apenas uma tabela de eventos é recebida e processada pelo nó. Todos os detalhes sobre como detectar um evento estão fora do escopo deste trabalho, uma vez que existem muitos trabalhos que já propuseram abordagens para este tipo de situação [Costa e Guedes 2013, Costa et al. 2015].

A terceira e a última observação sobre a importância de $Pi_{(c)}$ e $Pe_{(e)}$ para o cálculo geral de $Pf_{(s)}$. Em termos normais, podemos dizer que ambos os valores contribuem

igualmente. No entanto, algumas cidades podem definir diferentes percepções para $Pi_{(c)}$ e $Pe_{(c)}$. Para isso, definimos dois fatores de multiplicação diferentes na Equação (4.1), fc e fe , e normalmente teremos $fc = 1.0$ e $fe = 1.0$. Portanto, se quisermos atribuir o dobro de relevância para eventos de interesse, uma maneira possível para isso é fazer $fc = 0.5$ e $fe = 1.0$

Tabela 4.6: Exemplos de otimizações baseadas em prioridade.

Pf	Ação
Padrão de captura de imagem	
$0 \leq Pf \leq 5$	Capture 1 foto a cada 10 segundos
$5 < Pf \leq 10$	Capture 1 foto a cada 5 segundos
$10 < Pf \leq 15$	Capture 1 foto a cada 2 segundos
$15 < Pf \leq 20$	Capture 1 foto a cada 1 segundo
Configuração de vídeo	
$0 \leq Pf \leq 10$	H.264 Resolução QCIF
$10 < Pf \leq 15$	H.264 Resolução CIF
$15 < Pf \leq 18$	H.264 Resolução VGA
$19 < Pf \leq 20$	H.264 Resolução HDMI
Garantias de confiabilidade	
$0 \leq Pf \leq 10$	Transmissão baseada em UDP não confiável
$10 < Pf \leq 20$	Transmissão baseada em TCP confiável

4.2.7 Transmissões de tabelas e interações dos nós

Quando a abordagem proposta é implementada, existem duas fases de operação principais: a) autenticação de novos nós na rede e b) transmissão de tabelas. A primeira fase é quando um novo nó solicita sua adição no sistema, que pode ser realizada por meio de um processo de autenticação ou inserido manualmente dentro da Tabela de Nós. O MPC é responsável por controlar a autenticação desses novos

nós e enviar as tabelas de parâmetros e de eventos para os nós. Para garantir a flexibilidade do sistema, diferentes tipos de autenticação podem ser empregados, mas foi definido um mecanismo genérico capaz de realizar a inscrição de novos nós em sistemas de cidades inteligentes.

Como o MPC é responsável por gerenciar as diferentes Aplicações configuradas, e cada nó é associado a uma única Aplicação na Tabela de Nós, deve haver uma forma de indicar que um sensor está vinculado a uma Aplicação. Uma forma simples e factível é definir estaticamente a aplicação na Tabela de Nós, manualmente associado ao seu endereço de rede a uma Aplicação a . Entretanto é possível também gerar essa associação de forma dinâmica, com os novos nós automaticamente envie essa informação em sua requisição. O processo padrão para a solicitação de cadastro dinâmico de um novo nó é apresentado a seguir.

1. Configuração inicial dos nós: Cada nó que é capaz de se comunicar usando os serviços da abordagem proposta deve “pertencer” a uma Aplicação específica a e deve ter a chave de criptografia simétrica dessa aplicação, definida como $Cr_{(a)}$. Qualquer algoritmo de criptografia simétrica pode ser usado e existem algumas soluções que se encaixam melhor em redes de sensores [Goncalves e Costa 2015, Costa et al. 2017c, Zhang et al. 2010].
2. Um novo nó solicita associação: quando o nó é ligado ou entra na área do sistema, ele pode solicitar permissão para ser admitido como um nó válido. Isso pode ser feito usando qualquer protocolo de camada de aplicação de solicitação-resposta e, portanto, não é formalmente definido neste trabalho, especialmente devido ao fato de que a abordagem proposta pretende ser independente de qualquer detalhe da pilha de protocolos. No entanto, nesta solicitação de associação, o nó deve informar sua Aplicação a pré-configurada.
3. O MPC responde com um código criptografado: depois de receber a solicitação do nó, o MPC realiza um teste de desafio-resposta com esse nó. Como a aplicação do nó solicitante foi informada, e o MPC já tem todos os valores de $Cr_{(a)}$, para $a = 1, \dots, A$, tudo o que tem a fazer é para criptografar um “desafio” usando a chave $Cr_{(a)}$, enviando de volta para o nó solicitante.
4. O nó processa a mensagem do MPC: Se o nó for válido, ele terá a chave $Cr_{(a)}$ (pré-configurada) e em seguida poderá responder ao desafio transmitido pelo MPC.
5. O MPC recebe a resposta do nó: Se a mensagem recebida estiver correta, o MPC associa o endereço de rede do nó à Aplicação a , inserindo essas informações na Tabela de Nós ou atualizando-as, nesse caso, o nó é “alterado” para uma Aplicação diferente.
6. Confirmação: Uma quarta mensagem pode ser usada neste processo de autenticação para fins de confirmação final, transmitida do MPC para o nó.

Esse processo descrito pode não ser a abordagem mais eficiente, mas garante um certo nível de autenticidade ao mesmo tempo em que protege contra alguns ataques

de negação de serviço. Como os nós precisam solicitar admissão e esperar que a mensagem do MPC (desafio) seja processada, os ataques “spoofing” são evitados. Na verdade, se não for um problema, as funções hash poderiam ser usadas e os nós poderiam enviar códigos criptografados na mensagem de solicitação inicial, tornando este processo mais rápido (mas talvez um pouco menos seguro).

Após a associação de um nó à Tabela de Nós, ele poderá receber as tabelas de configuração, que serão a Tabela de Parâmetros e a Tabela de Eventos correspondente à Aplicação a . E a transmissão de tais tabelas será realizada da seguinte forma:

- As Tabelas de Eventos e Parâmetros correspondente à Aplicação a é totalmente criptografada usando a chave $Cr_{(a)}$.
- Após a admissão de um novo nó, ele recebe imediatamente o TP e o TE criptografado correspondente.
- Todos os nós devem receber uma nova tabela de configuração (TP ou TE) se qualquer valor em qualquer linha da tabela for alterada, fornecendo um comportamento adaptativo para todo o sistema. Uma Tabela de Eventos é transmitida apenas para nós pertencentes à mesma Aplicação, conforme definido na Tabela de Nós.

As definições apresentadas aqui podem ser implementadas de maneiras diferentes, desde que as regras declaradas sejam preservadas. Isso garante algum nível de flexibilidade à abordagem proposta, mantendo-a eficaz para sistemas de cidades inteligentes.

Capítulo 5

Resultados

A especificação do modelo de priorização proposto pode trazer inúmeros benefícios para aplicações em cidades reais, revolucionando a forma como Cidades Inteligentes serão construídas. Contudo, a verificação do desempenho prático desta solução é bastante complexa e envolve custos que inviabilizaram sua execução. Dessa forma, a validação da solução foi centrada em verificações formais, simulações e experimentações, o que trouxe inúmeros indícios que nos leva a esperar resultados satisfatórios na implementação do modelo em Cidades Inteligentes.

A partir dos requisitos levantados a fim de aprimorar o método de priorização em RSSF que atuam em Cidade Inteligentes, foi possível projetar um modelo funcional para ser implementado em qualquer *Smart City* por meio de seus parâmetros variáveis. Nesta seção é apresentado o modelo proposto completo e as simulações realizadas a fim de validá-lo, levando em consideração sua eficiência, QoS, QoE e segurança. Por fim, experimentos reais com a plataforma Raspberry Pi serão apresentados, demonstrando a viabilidade de implementação do modelo.

5.1 Especificação formal do modelo

A especificação do modelo proposto, como definido anteriormente, teve uma etapa de especificação formal. Essa etapa visa não apenas verificar se existem inconsistências no modelo proposto, mas também criar uma documentação sólida para guiar eventuais implementações. Para tanto, foram considerados dois formalismos diferentes, que são os Fluxogramas e as Redes de Petri, como apresentados nas subseções a seguir.

5.1.1 Funcionamento Geral do Modelo

Para a especificação formal do modelo, foi pensada qual estratégia seria mais benéfica para validar os detalhes operacionais do modelo proposto. Nessa análise, decidiu-

se usar Fluxogramas e Redes de Petri como formalismos para validar o fluxo de execução dos nós sensores e da MPC. De fato, essa é uma parte crítica do modelo que precisa ser apropriadamente validada, evitando inconsistências e dando suporte à eventuais implementações.

Alguns diagramas foram montados para cada situação que a RSSF pode enfrentar, desde o início do seu funcionamento até a captura de um evento. A figura 5.1 explica o procedimento de configuração da rede assim que ela é iniciada.

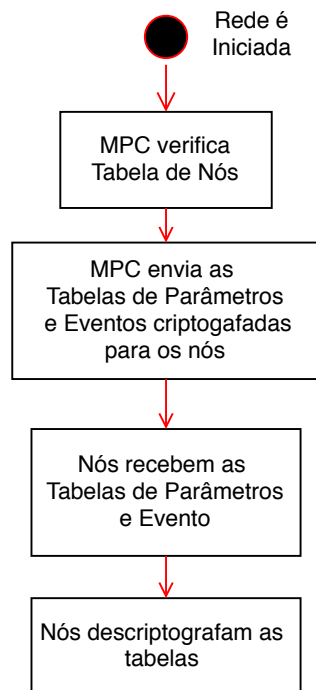


Figura 5.1: Início da Rede

Na Figura 5.2 é apresentado qual o comportamento da RSSF ao receber um novo nó. Medidas de segurança são aplicadas para impedir a entrada de nós maliciosos. A criptografia funciona como uma camada extra onde são acrescentadas políticas de segurança que dificultam a inserção de nós maliciosos na RSSF. Porém, a RSSF não está livre de sofrer ataques de negação de serviço (DDos). Como o módulo de processamento central permite que um nó externo se comunique com ele, mesmo antes de ter passado pelo procedimento de segurança, isso abre margem para que diversas requisições sejam feitas por esse nó e acabe afetando o desempenho da rede. Mesmo com essa deficiência, por decisão de projeto, os riscos de um nó malicioso entrar na rede foram considerados maiores, pois, com isso, um nó malicioso pode obter total controle da rede, basta que ele seja configurado para que suas requisições tenham um nível de prioridade máxima.

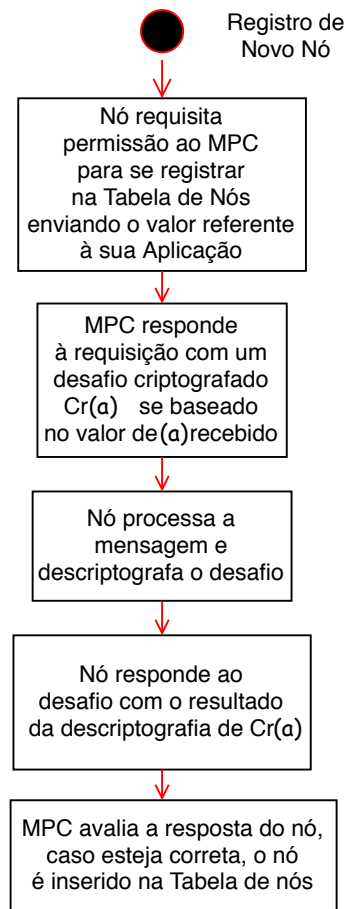


Figura 5.2: Registro de Novo Nó.

O diagrama a Figura 5.3 explica como um nó se comporta ao detectar um evento. O modelo proposto é responsável por realizar o processo após a aquisição do evento. A responsabilidade de como a informação é obtida e classificada como evento não faz parte do escopo do projeto.

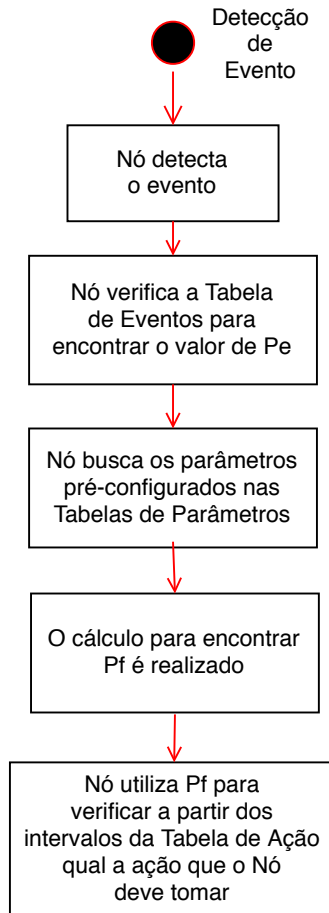


Figura 5.3: Detecção de Evento.

5.1.2 Análise por Rede de Petri

Rede de Petri é uma representação matemática para sistemas distribuídos discretos. Ela permite definir graficamente um sistema distribuído apresentando um formato de grafo. A Rede de Petri constitui de um grafo orientado onde existem dois tipos de nós: transições (retângulo) e posições (círculo). Os arcos do grafo partem de transições para posições ou vice-versa. Elas são representadas por uma quintupla (P, T, A, W, m_0) .

- P: conjunto de posições ou lugares;
- T: conjunto finito de transições;
- A: conjunto finito de arcos pertencentes ao conjunto $(P \times T) \cup (T \times P)$, sendo $(P \times T)$ os arcos de P para T e $(T \times P)$ os arcos de T para P;
- W: é função que atribui um peso w a cada arco;

- m_0 : vetor cuja i -ésima coordenada define um número de marcas na posição p , no início da evolução da rede;

Com a modelagem do sistema via Rede de Petri foi possível realizar algumas análises a respeito do modelo de configuração dinâmica apresentado. Com ele foi possível realizar análises de completude e *deadlock*. A partir dessas análises foi possível verificar se realmente o modelo estaria apto a ser implementado a uma cidade inteligente.

A ferramenta utilizada para o desenvolvimento desse modelo foi a PIPEv4.3.0. Nela é possível montar e simular a rede de forma a realizar algumas verificações.

Na Figura 5.4 é possível visualizar o módulo onde ocorre a inserção de um novo nó na rede. Na Figura 5.5 é apresentado o módulo onde ocorre a distribuição das tabelas de parâmetro e a configuração dos nós da rede.

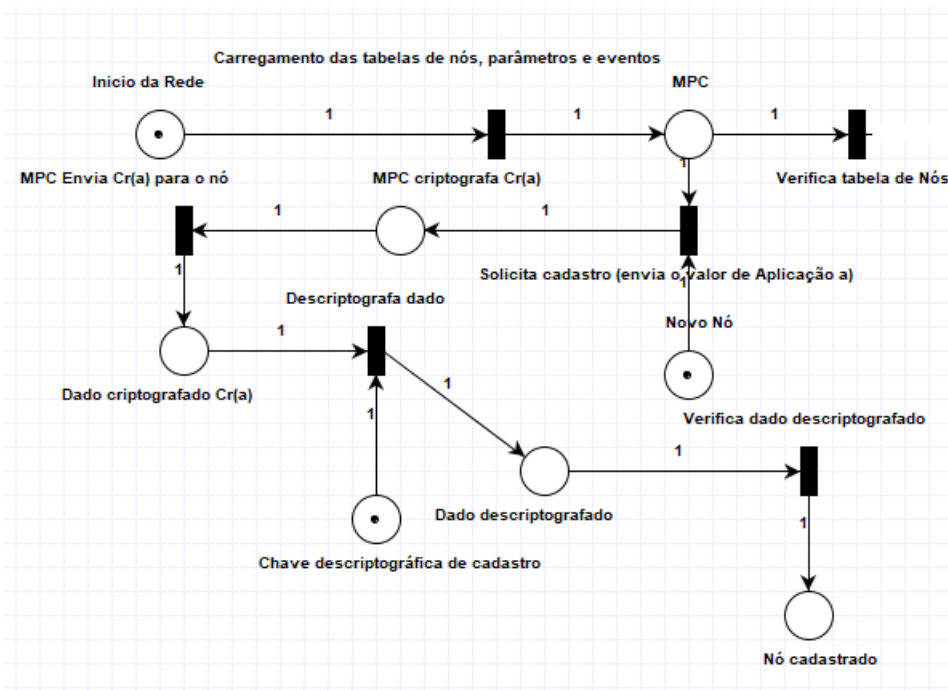


Figura 5.4: Rede de Petri - Cadastro.

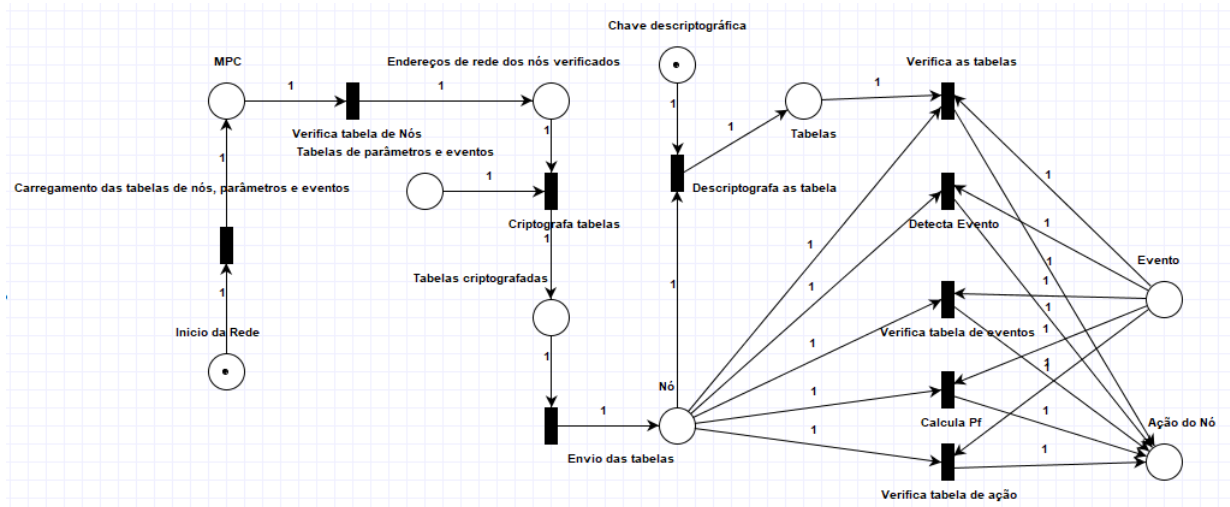


Figura 5.5: Rede de Petri - Ação.

Um *token* foi colocado em alguns pontos estratégicos da rede para que fosse possível verificar algumas funcionalidades do modelo de configuração dinâmica. Quando uma transição é satisfeita a cor dos retângulos muda, isso mostra que todas as posições que apontam para ela estão com o *token*. Por exemplo, para que a transição “Solicita cadastro” seja satisfeita, deve haver um *token* na posição “Novo Nó” e outro na posição “Chave criptográfica”. Isso quer dizer que para que haja uma solicitação de cadastro por um novo nó, o novo nó deve estar requisitando o cadastro e ele deve existir uma chave criptográfica, chave essa que é utilizada para decodificar a mensagem recebida pelo MPC.

Para a montagem desta Rede de Petri foi utilizada a estratégia *top-down*, ou seja, primeiro foram definidos os objetivos da rede e a partir deles foram elaborados os passos para alcançá-los. Os objetivos que o modelo se propõe a realizar é configurar a ação de um nó por meio de alguns parâmetro e cadastrar os nós de forma que eles sejam autenticados. Dessa forma, na Rede de Petri para alcançar esses objetivos as posições “Ação do Nó” e “Nó cadastrado” devem ser alcançáveis.

A rede é iniciada carregando as tabelas de nós, parâmetros e eventos dentro do MPC. Com isso o MPC está apto a realizar suas operações, ele pode verificar sua tabela de nós e obter os endereços de rede dos nós que devem receber as tabelas de parâmetros e eventos. As tabelas devem ser criptografadas para que possam ser enviadas para os nós cadastrados na Tabela de Nós. O nó, ao receber as tabelas de parâmetros e eventos, deve descriptografar essas tabelas (lembrando que as tabelas de parâmetros recebidas pelo nó são as referentes à aplicação daquele nó). Quando um evento é detectado, é necessário que o nó realize algumas operações para que ele possa realizar uma ação. As operações são: identificar Pa nas tabelas de parâmetros, detectar qual o evento, identificar Pe na na tabela de evento, calcular o valor de Pf , identificar a ação a ser tomada na tabela de ação.

Um outro caminho que pode ser tomado por MPC é cadastrar um novo nó na rede desde que ele seja identificado como um nó não malicioso. O novo nó deve solicitar o cadastro da rede ao MPC, ele envia o valor a referente à sua aplicação para o MPC. O MPC criptografa esse valor $Cr(a)$ e o envia de volta para o nó. Esse valor deve ser descriptografado e enviado de volta para o MPC para que assim seu endereço de rede e aplicação sejam cadastrados na tabela de nós.

Após a construção da Rede de Petri e a realização das verificações desejadas pôde-se perceber que a solução proposta está funcionalmente correta, podendo então ser considerada para novas etapas de validação.

5.2 Simulação utilizando *CupCarbon*

O CupCarbon [Mehdi et al. 2014] é um simulador utilizado em projetos de Cidades Inteligentes. Nele é possível inserir elementos como nós, eventos, *sinks* e diversos outros elementos utilizados em *SmartCities*. Ele foi desenvolvido utilizando a linguagem de programação Java, dessa forma, é necessário possuir a máquina virtual Java (JVM) para utilizar o simulador.

O CupCarbon permite que cada elemento seja configurado de forma que o usuário possui total liberdade para programar a sua rede da forma que achar necessário, desde cada elemento até a topologia. Pelo CupCarbon foi possível programar os nós e o MPC utilizando uma linguagem de programação própria da plataforma.

A simulação foi dividida em duas partes, na primeira o simulador deve ser capaz de realizar o processo de configuração do nó para realizar uma ação a depender do evento e de fatores externos capturados. Na segunda parte foi construído um modelo que deve permitir que nós sejam autenticados e possam solicitar participação na rede.

O primeiro passo para a construção do modelo no CupCarbon foi adicionar um mapa e adicionar elementos a esse mapa. A API escolhida para a representação do mapa foi a do *Open Street Maps*, por essa ser uma ferramenta *open source*. A localização do mapa escolhida para realizar a implantação do sistema foi avenida Getúlio Vargas na cidade de Feira de Santana. Foram acrescentados quatro nós, onde o ip de três deles estavam cadastrados na tabela de nós do MPC. Dessa forma a comunicação do MPC deve acontecer apenas com esses três nós previamente cadastrados, ignorando o quarto nó.

No desenvolvimento da primeira parte do sistema foram incluídos eventos próximos desses nós cadastrados. O evento escolhido foi o de um incêndio. Para os nós, foi configurado que um evento é classificado como incêndio quando há o registro de uma temperatura que ultrapasse os 100°C. Foi programado para que esse valor de temperatura varie com o tempo. Esse valor varia entre 60°C e 140°C. O elementos dispostos no simulador podem ser visualizados na Figura 5.6

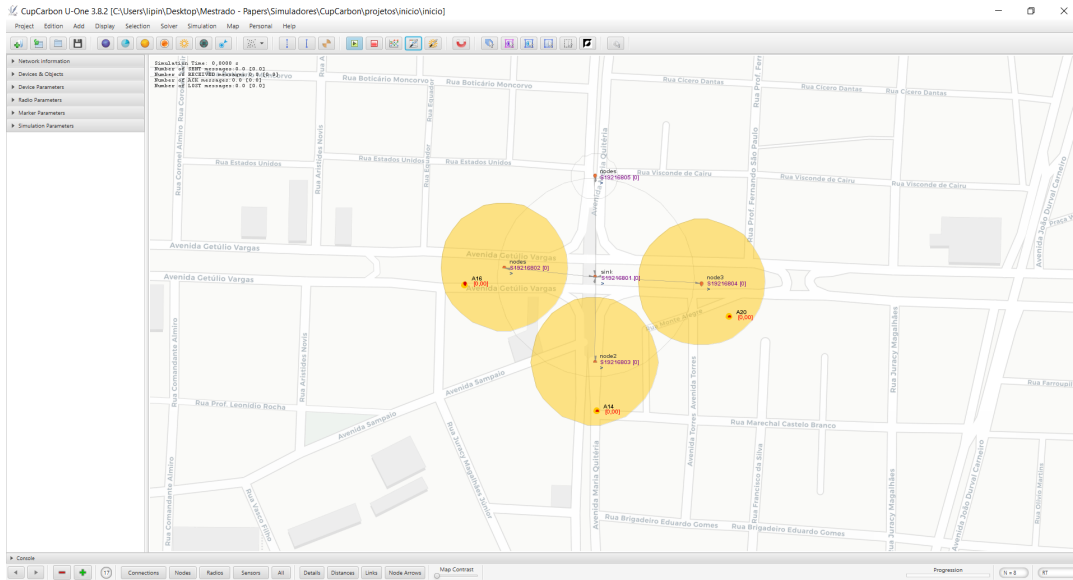


Figura 5.6: Simulação - Primeira Parte.

Os três nós possuem o mesmo comportamento, mas diferem nos valores em suas tabelas de parâmetros. Essa diferença foi criada propositalmente para mostrar como os fatores externos podem influenciar ação que o nó pode realizar.

Para a geração dos valores de temperatura, a própria ferramenta auxilia na geração de valores aleatórios que se alternam com o passar do tempo. É possível definir o valor médio e o quanto os valores podem se distanciar desse valor médio. No caso utilizado para a simulação, o valor da média escolhido foi 100 e o valor máximo de distanciamento da média foi 40.

O sistema inicia com o MPC enviando as tabelas de evento e parâmetros para os nós cadastrados na Tabela de Nós. Esses nós capturam as tabelas referentes à sua aplicação e realiza o processo de descritografia dessas tabelas. Os nós começam a monitorar a temperatura, quando essa ultrapassa os 100°C o nó realiza alguma ação. As ações definidas podem ser três: soar alarme, disparar os sistema anti-incêndio e acionar o corpo de bombeiros. A tabela de parâmetros leva em consideração dois parâmetros: período do dia e se é final de semana ou não. Os três nós foram definidos como de aplicações diferentes, ou seja, os valores de P_i para cada um dos parâmetros é diferente de nó para nó. A simulação foi construída dessa forma para provar que com o mesmo valor de prioridade de evento P_e para os três nós, os valores dos parâmetros podem influenciar no resultado final. Os valores das tabelas de parâmetros dos três nós podem ser visualizados nas tabelas abaixo. O monitoramento foi realizado em um final de semana no período noturno.

Tabela 5.1: Tabela de Parâmetros Nó 1 (Período do dia) - Simulação.

<u>Período do Dia</u>	<u>Valor</u>
Dia	0.5
Noite	0.9

Tabela 5.2: Tabela de Parâmetros Nó 1 (Dia da semana) - Simulação.

<u>Semana</u>	<u>Valor</u>
Final de Semana	0.8
Dia de Semana	0.6

Tabela 5.3: Tabela de Parâmetros Nó 2 (Período do dia) - Simulação

<u>Período do Dia</u>	<u>Valor</u>
Dia	0.3
Noite	0.4

Tabela 5.4: Tabela de Parâmetros Nó 2 (Dia da semana) - Simulação

<u>Semana</u>	<u>Valor</u>
Final de Semana	0.1
Dia de Semana	0.6

Tabela 5.5: Tabela de Parâmetros Nó 3 (Período do dia) - Simulação.

<u>Período do Dia</u>	<u>Valor</u>
Dia	0.5
Noite	0.5

Tabela 5.6: Tabela de Parâmetros Nó 3 (Dia da semana) - Simulação.

Semana	Valor
Final de Semana	0.6
Dia de Semana	0.6

O Pf é calculado em cima desses valores. O valor Pe do evento incêndio é 8. Ao se calcular Pf , o valor obtido pode se encontrar em um dos intervalos da tabela de ação apresentada em 5.7.

Tabela 5.7: Tabela de Ação - Simulação.

Ação	Intervalo
Soar Alarme	0 - 0.3
Disparar o Sistema Anti-Incêndio	0.4 - 0.7
Acionar o Corpo de Bombeiros	0.8 - 1

Ao executar a simulação e os nós identificarem a ocorrência de um incêndio, os resultados obtidos foram: o nó 1 acionou o corpo de bombeiros, o nó 2 soou o alarme e o nó 3 disparou o sistema anti-incêndio. Esses resultados são apresentado na Figura 5.7.

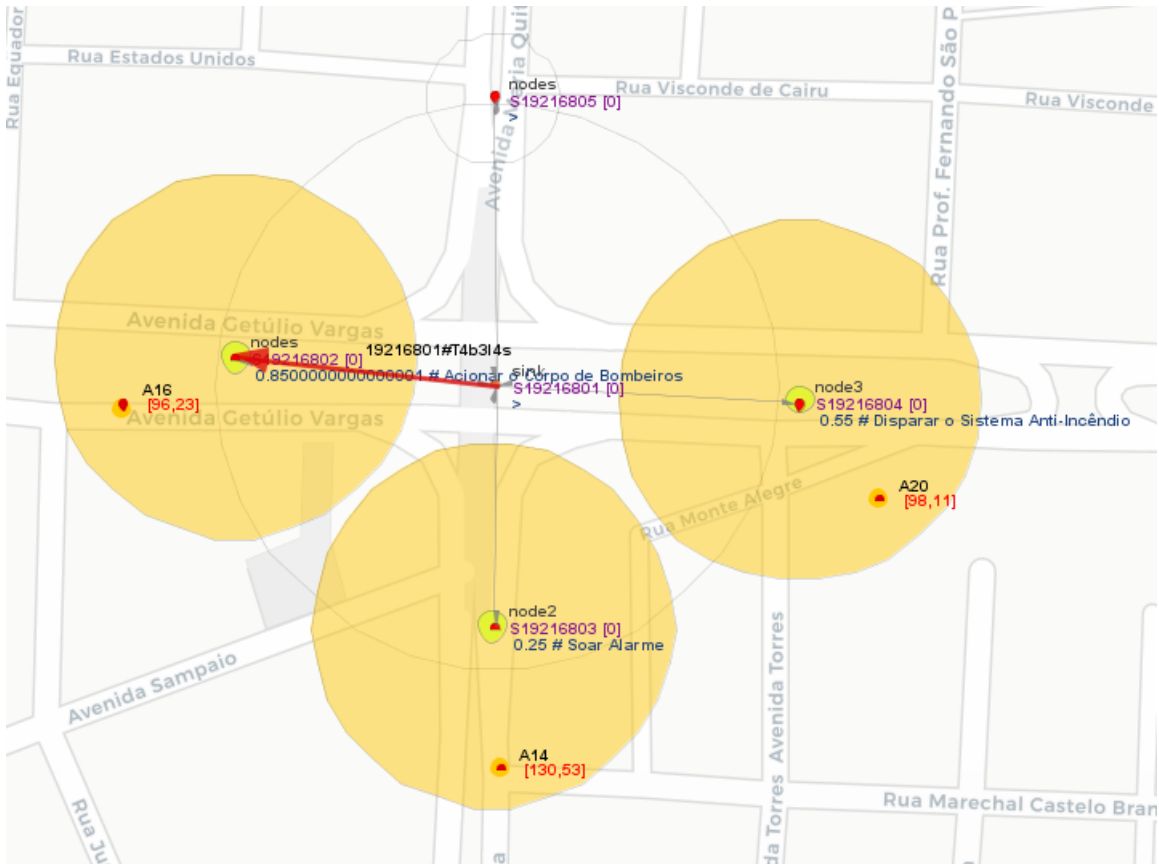


Figura 5.7: Simulação - Resultado

Como o algoritmo implementado nos nós é o mesmo mudando apenas a tabela de parâmetros, então o algoritmo dos nós é basicamente o mesmo. Abaixo é possível visualizar o código fonte implementado em um dos nós.

Código 5.1: Nó 1

```

1 //nodel
2 loop
3 wait
4 read rp
5 rdata $rp rid v
6
7 //verifica e descriptografa as tabelas e seleciona a que eh de seu
  interesse
8 if ($v==T4b314s)
9     //reconhecimento das tabelas
10    set Pa tabelaPa1
11    //print $Pa
12
13 end
14
15 if ($Pa==tabelaPa1)
16    mark = 1

```

```

17         //periodo do dia
18         set Padia 0.5
19         set Panoite 0.9
20         //semana (final de semana ou nao)
21         set Padiasemana 0.6
22         set Pafds 0.8
23     end
24
25     if ($Pa==tabelaPa2)
26         //periodo do dia
27         set Padia 0.6
28         set Panoite 0.7
29         //semana (final de semana ou nao)
30         set Padiasemana 0.5
31         set Pafds 0.6
32     end
33
34     //Sensor le a temperatura
35     areadsensor x
36     if ($x!=X)
37         //print $x
38         rdata $x a b c
39     end
40     delay 1000
41
42
43     //acima de 100 graus eh considerado um evento de incendio
44     if ($c >= 100)
45         set Pe incendio
46         //print $Pe
47
48         if ($Pe == incendio)
49             set Peincendio 8
50             //caso seja em um final de semana de noite
51             set Pf (($Panoite+$Pafds)/2) * $Peincendio
52
53             if ($Pf > 0)
54                 if ($Pf <= 0.3)
55                     print $Pf # Soar Alarme
56                 end
57             end
58
59             if ($Pf > 0.4)
60                 if ($Pf <= 0.7)
61                     print $Pf # Disparar o Sistema Anti-
62                         Incendio
63                 end
64             end
65
66             if ($Pf > 0.8)
67                 if ($Pf <= 1)
68                     print $Pf # Acionar o Corpo de
69                         Bombeiros

```

```

68             end
69         end
70     end
71     end
72 end
73 mark = 0

```

Um fator que deve ser explicado nos códigos apresentados é quanto à descryptografia das tabelas. Pelo fato da linguagem de script fornecida pela ferramenta ser limitada, não foi possível a utilização de estruturas de dados mais complexas como arrays, por exemplo. Então definiu-se que a *string* “T4b3l4s” seriam as tabelas encriptadas. Elas são enviadas pelo MPC para os nós. A condição apresentada linha 8 do código 5.1 faz alusão à descryptografia, mesmo que de forma rudimentar, é possível representar essa operação para que possa dar prosseguimento à simulação.

No código 5.1, após a descryptografia o nó reconhece qual tabela foi enviada para ele, a essa tabela foi dado o nome de “tabelaPa1”, a partir disso o nó consegue identificar o valor Pa dos seus parâmetros externos. Na linha 38 o nó lê os valores de temperatura a longo do tempo. Quando esse valor é igual ou superior a 100, ele começa a realizar o cálculo de Pf baseado nos valores de Pa da “tabelaPa1”. Como foi dito anteriormente, o monitoramento foi realizado em um final de semana no período noturno. Na linha 51 o cálculo de Pf é realizado de fato. As condições subsequentes analisam em qual intervalo está o valor de Pf encontrado.

É importante analisar também no que ocorre no código 5.1 na parte de realizar o cálculo para definir o valor de Pf . Como existe apenas um evento acontecendo, não foi necessário inserir o somatório de eventos no cálculo. Outra observação é que ambos os fatores de configuração (parâmetros e eventos) possuem relevâncias semelhantes, com isso fc e fe possuem valores iguais a 1, com isso eles foram suprimidos da fórmula no *script*.

O código 5.2 é referente ao algoritmo implementado no MPC. Esse algoritmo é bastante simples, o MPC basicamente envia as tabelas criptografadas para os nós que possuem os ips cadastrados na tabela de nós.

Código 5.2: MPC

```

1 //MPC
2 atget id id
3
4 loop
5 //Envio de conjunto de tabelas para os nos
6 data p $id T4b3l4s
7 send $p 19216802
8 delay 1000
9 send $p 19216803
10 delay 1000
11 send $p 19216804
12 delay 1000

```

Na parte de autenticação do nó, a simulação foi desenvolvida utilizando um nó e um MPC. Essa estrutura pode ser visualizada na Figura 5.8.

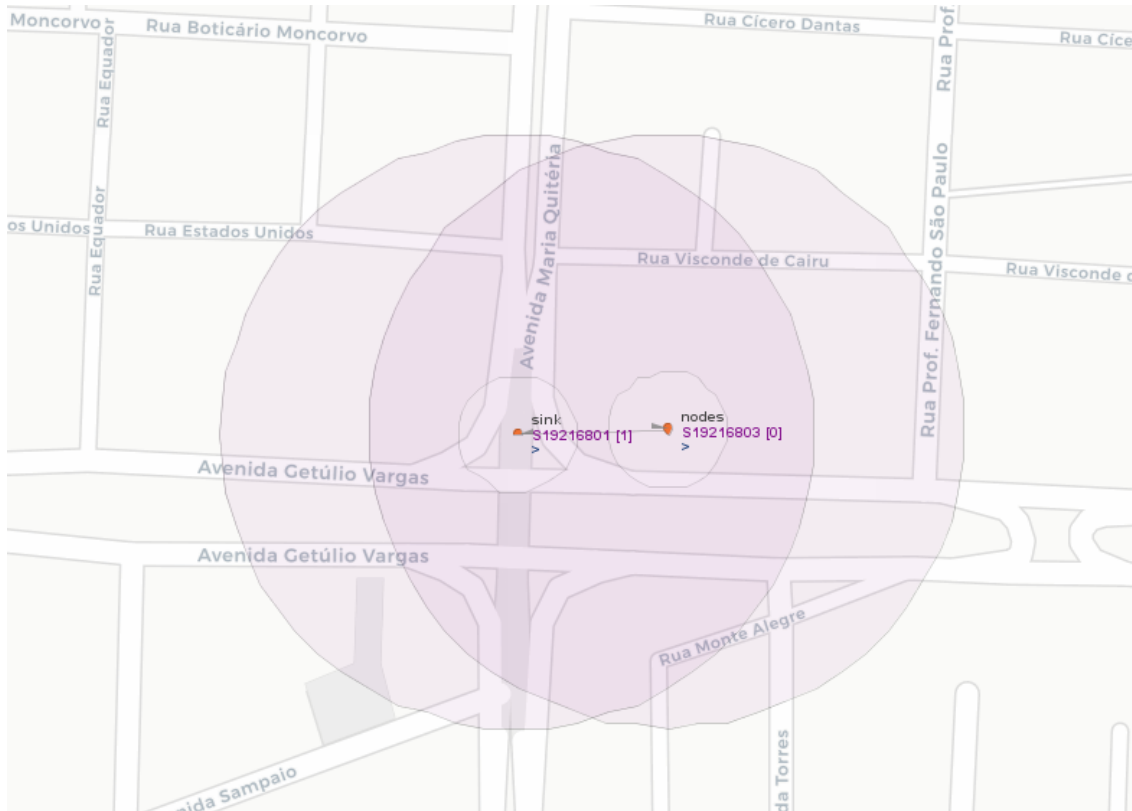


Figura 5.8: Simulação - Cadastro.

Na simulação o nó realiza a solicitação para ser autenticado e adicionado à tabela de nós. O MPC envia uma mensagem criptografada ao nó, esse último deve descriptografar a mensagem e enviar de volta ao MPC. Assim o nó é cadastrado na tabela de nós. Abaixo segue os algoritmos implementados no nó e no MPC.

No código 5.3 o nó envia uma solicitação ao MPC nas linhas 5 e 6. Na linha 9, ele recebe uma resposta do MPC criptografada. A mensagem criptografada foi representada por "c4d4str0", assim essa mensagem é descriptografada e enviada novamente ao MPC, que por sua vez envia as tabelas para o novo nó.

Código 5.3: Nó solicita cadastro

```

1 //No
2 atget id id
3 print Nao Cadastrado
4 loop
5 data p $id Solicitacao
6 send $p
7 delay 100000
8

```

```

9  read rs
10 //mensagem criptografada
11 if ($rs==c4d4str0)
12     print Descriptografando
13     data desc $id cadastro
14     send $desc
15     delay 100000
16 //envio de tabelas para o novo no
17 else if ($rs==tabelas)
18     print Cadastrado
19     stop
20 end

```

No código 5.4 é possível ter uma visão da autenticação de um novo pelo lado do MPC. O MPC recebe uma mensagem de solicitação de um novo nó que está requisitando acesso a rede. O MPC recebe a mensagem de solicitação e o ip do nó que enviou a solicitação. Então a mensagem criptografada "c4d4str0" é enviada para o nó que deve descriptografar essa mensagem e enviar a resposta ao MPC. Por fim, se a mensagem enviada for a correta, que nesse caso foi definida como "cadastro", as tabelas são enviadas ao nó e esse o id desse nó é adicionado à tabela de nós.

Código 5.4: MPC cadastra nó

```

1 //MPC
2 loop
3 wait
4 read rp
5 rdata $rp rid v
6 if ($v==Solicitacao)
7     mark 1
8     //chave para ser descriptografada
9     send c4d4str0 $rid
10    delay 100000
11 else if ($descr==cadastro)
12     print $rid Cadastrado
13     send tabelas $rid
14     delay 100000
15 end

```

Quando um nó for cadastrado, o MPC exibe o ip desse nó e a mensagem de cadastrado. Essa representação pode ser visualizada na Figura 5.9

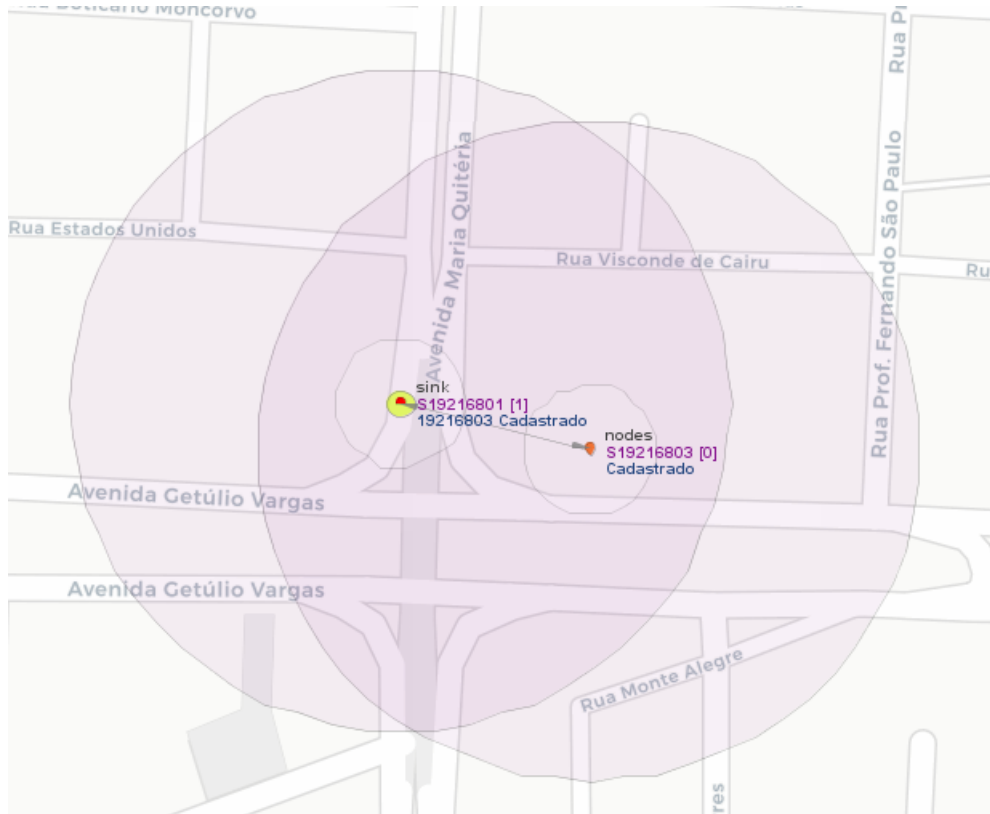


Figura 5.9: Simulação - Nó Cadastrado

5.2.1 Resultados de simulação (casos isolados)

A partir da formalização do modelo, foi possível simular o comportamento da RSSF em face alguns eventos que podem ocorrer em uma *Smart City*. Foram criados cenários fictícios onde foram analisadas as atitudes dos nós diante os eventos capturados. Por ser um modelo focado na priorização de forma local, ele não leva em consideração a ação dos outros nós na rede, por esse motivo as simulações limitam-se ao processamento que está ocorrendo internamente em um nó. Vale ressaltar que nos experimentos apresentados a seguir considera-se que o evento e os parâmetros possuem a mesma influência sobre o modelo, com isso temos $fe = 1$ e $fc = 1$.

Cenário 1

- Parâmetros: período do dia e estação do ano;
- Eventos Capturados: Queda de Energia e Incêndio;
- Sensores que capturaram o evento: sensor de luminosidade, sensor de temperatura e câmera;

- Período do Dia: Noite;
- Estação do Ano: Outono;

As tabelas 5.8, 5.9 e 5.10 representam as tabelas preenchidas para a situação descrita no Cenário 1.

Tabela 5.8: Cenário 1: Período do Dia

Situação	Pi
Dia	0.6
Tarde	0.8
Noite	1.0

Tabela 5.9: Cenário 1: Estação do Ano

Situação	Pi
Primavera	0.6
Verão	1.0
Outono	0.8
Inverno	0.4

Tabela 5.10: Cenário 1: Eventos

Pe	Evento
8	Acidente de Carro
10	Incêndio
3	Queda de Energia

Analisando as informações obtidas das tabelas, a priorização final foi obtida, com ela foi possível definir qual ação a ser tomada pelo nó utilizando a Tabela de Ação 5.11. Neste primeiro caso, como é apresentado na equação 5.1, houve diferença na ação tomada ao utilizar o modelo e sem utilizá-lo, os dois ficaram em intervalos diferentes

na Tabela de Ação, pois a Pf encontrada utilizando o modelo foi de 3.9, e a Pf sem utilizar o modelo é 3, nesse caso, a ação tomada pelo nó seria enviar os níveis de luminosidade e temperatura e uma imagem para o MPC.

$$Pf_{(s)} = \left(\frac{\sum_{(c=0)}^C (Pi_{(c)})}{C} \right) . fc + \left(\frac{\sum_{(e=0)}^E (Pe_{(e)})}{E} \right) . fe \quad (5.1)$$

$$Pf_{(s)} = \frac{1 + 0.8}{2} + \frac{3 + 10}{2} \quad (5.2)$$

$$Pf_{(s)} = 7.4 \quad (5.3)$$

Tabela 5.11: Cenário 1: Ação

Pf	Ação
0 ~ 3	Nível de Luminosidade/Temperatura
3.1 ~ 6	Nível de Luminosidade/Temperatura + 1 imagem
6.1 ~ 10	Nível de Luminosidade/Temperatura + 2 imagens

Cenário 2

- Parâmetros: clima, zona da cidade, feriado;
- Evento Capturado: Chuva;
- Sensores que capturaram o evento: sensor pluviométrico;
- Clima: Chuvoso;
- Zona da cidade: litoral;
- Feriado: sim;

O cenário 2 foi pensado em um contexto onde o nó analisado pode definir se a população poderá se deslocar de suas casas para aproveitar a praia no fim de semana. Podemos pensar que esse sensor faz parte de uma estação meteorológica. A partir do modelo apresentado, podemos ver qual a informação a ser enviada para o módulo de processamento central. As tabelas de Parâmetros e Evento são representadas nas tabelas 5.12, 5.13, 5.14 e 5.15 contendo os dados que compõem esse cenário.

Tabela 5.12: Cenário 2: Clima

Situação	Pi
Chuvoso	0.6
Ensolarado	0.8

Tabela 5.13: Cenário 2: Zona

Situação	Pi
Interior	0.2
Litoral	0.7

Tabela 5.14: Cenário 2: Feriado

Situação	Pi
Sim	0.7
Não	0.1

Tabela 5.15: Cenário 2: Eventos

Pe	Evento
9	Chuva
7	Sol

Mesmo antes de realizar qualquer tipo de cálculo é possível inferir algumas características do sistema:

1. A informação de chuva é mais crítica, por essa razão seu nível de prioridade é maior que a do sol;
2. A informação litoral é mais crítica para o contexto desse nó por conta dele possuir um sensor pluviométrico, o que pode influenciar, por exemplo, no fluxo do trânsito da cidade, pois uma informação de dia ensolarado no litoral pode significar que a população irá aproveitar o sol na praia.

3. No feriado significa que a população está com tempo livre para ir à praia, por essa razão sua prioridade é maior do que quando não é feriado. Porém, para um outro tipo de nó, essa informação poderia vir inversa. Pensemos em um nó que monitora o tráfego no centro de uma grande cidade, a informação de um acidente neste local pode influenciar na rotina de todos os trabalhadores, por essa razão, nesse possível contexto a prioridade da informação em dias não feriado deveria ser maior.

A partir do cálculo da equação 5.4, é possível encontrar o valor da prioridade final do sistema.

$$Pf_{(s)} = \left(\frac{\sum_{(c=0)}^C (Pi_{(c)})}{C} \right) .fc + \left(\frac{\sum_{(e=0)}^E (Pe_{(e)})}{E} \right) .fe \quad (5.4)$$

$$Pf_{(s)} = \frac{0.7 + 0.7 + 0.6}{3} + \frac{9}{1} \quad (5.5)$$

$$Pf_{(s)} = 9.667 \quad (5.6)$$

Neste caso, baseando-se na tabela 5.16, a partir do Pf calculado, é possível notar que a ação tomada pelo nó ao utilizar o modelo proposto é diferente da que se ele não estivesse usando o modelo. Neste caso, o nó possui um sensor pluviométrico, a ação tomada independente do Pf encontrado, é a de enviar o índice pluviométrico, mas o Pf entra neste caso para indicar a largura de banda que o nó utilizará para enviar a informação. Quanto maior o Pf , maior a velocidade com que a informação será transmitida.

Tabela 5.16: Cenário 2: Ação

Pf	Ação
0 ~ 3	1x velocidade de transmissão
3.1 ~ 6	2x velocidade de transmissão
6.1 ~ 10	3x velocidade de transmissão

5.3 Experimentação em sensores reais com a plataforma Raspberry

Para comprovar a viabilidade do modelo apresentado, através de um nível adicional de validação, foi realizada uma experimentação com componentes reais que podem

compor uma RSSF e fazer parte de um sistema mais amplo, como uma Cidade Inteligente.

Para a experimentação foram utilizados dois dispositivos *Raspberry*, realizando o papel de nós sensores, e um computador portátil, desempenhando o papel de processamento de um MPC.

Raspberry são mini computadores com capacidade computacional reduzida em relação a um PC (*Personal Computer*). Porém, mesmo com essa limitação computacional, eles são capazes de suportar diversas aplicações, inclusive embarcar sistemas operacionais desenvolvidos especialmente para esse tipo de arquitetura. Além de funcionar como um computador, esses dispositivos possuem interfaces periféricas onde os tornam capazes de realizar a comunicação com diversos componentes eletrônicos, incluindo componentes que podem realizar o papel de sensores.

Foram desenvolvidas duas aplicações utilizando a linguagem de programação Python. Uma das aplicações foi desenvolvida para funcionar no notebook realizando o papel de um MPC e a outra aplicação foi desenvolvida para ser embarcada nos *Raspberrys* realizando o papel de sensoreamento e comunicação com o MPC.

A aplicação que executa no notebook possui as funções de cadastrar e enviar as tabelas de parâmetros e eventos para os *Raspberrys*, enquanto os *Raspberrys* detectam um evento e realizam determinada ação a depender de algumas variáveis externas pré-configuradas. Para tornar a experimentação possível, por não possuir um ambiente controlado onde poderiam ser trabalhadas diversas variáveis como temperatura, umidade, pressão e etc, optou-se por utilizar um *pushbutton* como acionador de um evento, então, ao perceber o pressionar desse botão, o *Raspberry* deve:

- Analisar o evento;
- Analisar as variáveis externas;
- Analisar as tabelas de parâmetros enviadas pelo notebook;
- Realizar o cálculo de Pf ;
- Executar uma ação;

O modelo de *Raspberry* utilizado na experimentação foi o *Raspberry PI 2*. Essa versão possui algumas melhorias em relação à sua versão anterior, porém não possui um módulo de conexão de rede já implementado. Em contrapartida, ele possui interfaces USB onde é possível conectar um módulo WIFI e assim tornar possível a comunicação com uma rede sem fio.

A comunicação entre o notebook e os *Raspberrys* aconteceu por meio de uma rede doméstica via transferências de pacotes. O protocolo utilizado nessa comunicação foi o UDP (*User Datagram Protocol*). O UDP é considerado um protocolo simples em que comunicação ocorre pela troca de datagramas, o cabeçalho desses datagramas UDPs se resume a:

- Porta origem (opcional);
- Porta destino;
- Comprimento da mensagem;
- Checksum (opcional);

Por ser um protocolo simples ele acaba se tornando um protocolo rápido, ideal para troca de mensagens instantâneas. Por a aplicação desenvolvida ser simples, optou-se por esse protocolo em detrimento de outros orientados a conexão, onde determinadas verificações, como a verificação de entrega da mensagem poderia tornar o processo lento, mesmo que em determinadas situações reais seja imprescindível essa verificação devido à relevância da informação.

O circuito montado para a geração do evento é bem simples. Ele se resume em conectar um *pushbutton* ao *Raspberry*. Para realizar essa conexão foram utilizados 2 *jumpers* macho-fêmea conectado aos pinos 6 e 12 do *Raspberry* e também aos dois pinos do *pushbutton* que se encontram do mesmo lado do botão. Esse esquema pode ser visualizado na na Figura 5.10. As outras conexões presentes aqui são referentes à conexão de um *display* ao *Raspberry* para que seja possível visualizar os resultados. Na Figura 5.11 é possível visualizar uma captura do *Raspberry* em funcionamento.

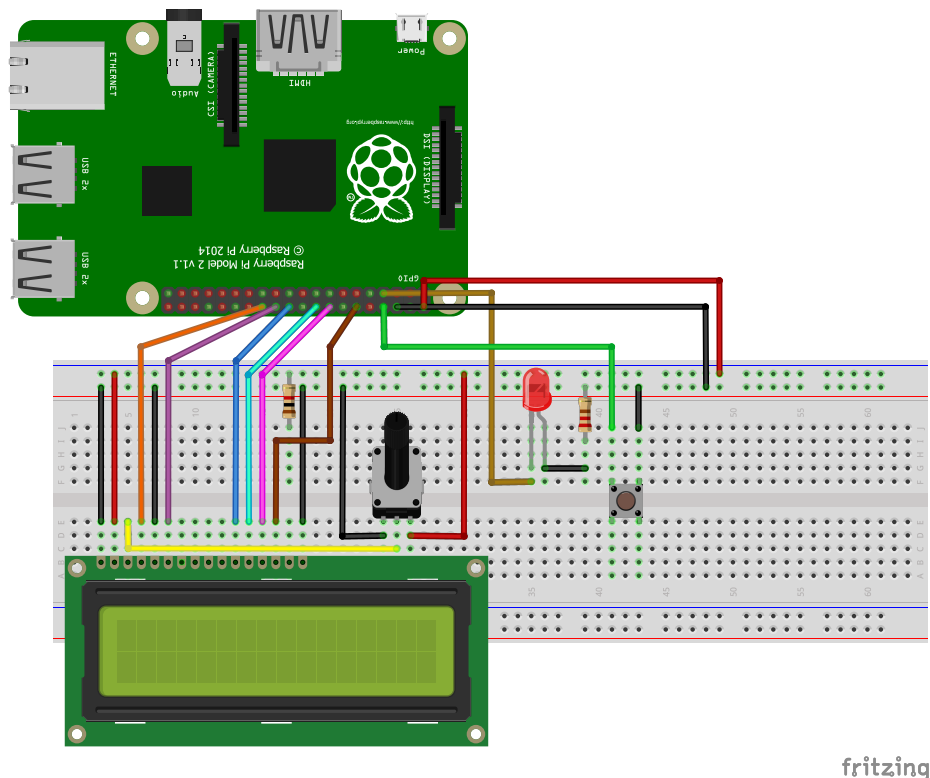
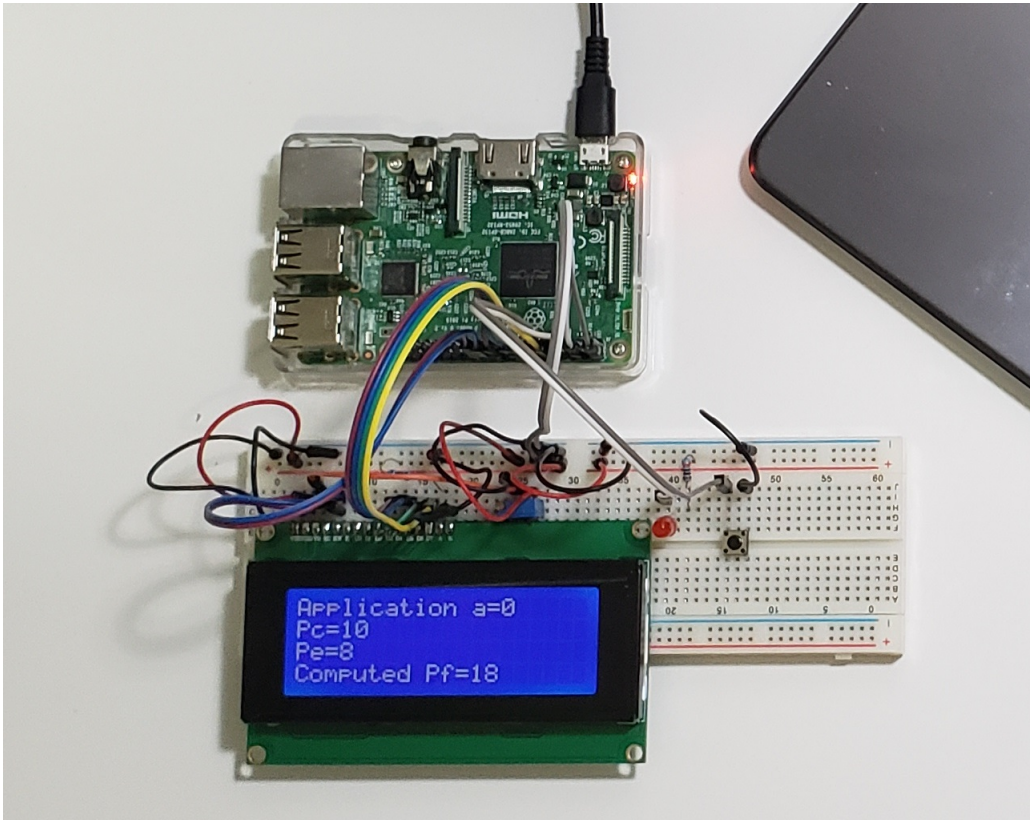


Figura 5.10: Esquema *RaspBerry*

Figura 5.11: Funcionamento *RaspBerry*

Para explicar o funcionamento da experimentação passo-a-passo é necessário analisar os códigos embarcados tanto no MPC quanto nos nós. O Código 5.5 contém o *script* de funcionamento do MPC e o Código 5.6 o *script* de funcionamento dos nós. Os códigos apresentados a seguir foram desenvolvidos utilizando a linguagem de programação Python.

No Código 5.5 é possível notar a definição de algumas variáveis importantes no desenvolvimento da aplicação:

- A variável UDPIP é o IP do próprio MPC (ou seja do notebook);
- O vetor TABELASNO é o vetor onde estão cadastrados os IPs dos nós (nesse caso um dos nós já está cadastrado previamente);
- CHAVECADASTRO é a chave descritografada que os nós precisam ser capazes de gerar para se cadastrarem na tabela de nós;

São basicamente duas as funções do MPC nessa aplicação:

- Cadastrar o novo nó;
- Enviar as tabelas de parâmetros e eventos para os nós;

Por funcionar como um servidor onde pode desempenhar mais de uma função ao mesmo tempo a depender das requisições dos clientes, cada função do MPC é configurada para rodar como uma *thread*, dessa forma ele consegue realizar suas operações em "paralelo"sem necessitar travar toda a aplicação.

Assim que o MPC é executado ele verifica se há algum IP cadastrado em sua tabela de nós e envia as tabelas de parâmetros para esses nós como uma mensagem via protocolo UDP.

Na função de cadastro de novo nó é possível notar qual o procedimento que o MPC toma ao receber uma requisição do tipo cadastro. Primeiro ele verifica se o solicitante já não está cadastrado no vetor referente à tabela de nós, caso esteja, ele apenas envia as tabelas de parâmetros e eventos para esse nó. Como ele já existe na tabela de nós, significa que ele já passou pelo processo de verificação e autenticação. Caso o nó ainda não exista nessa tabela, ele deve enviar junto com a requisição uma chave de cadastro, essa chave enviada pelo nó é comparada com a chave CHAVECADASTRO definida previamente, caso elas sejam iguais, o nó é cadastrado na tabela de nós e as tabelas de parâmetros e eventos são enviadas para ele.

Código 5.5: Raspberry MPC

```

1 import socket
2 import time
3 import threading
4 import _thread as thread
5
6 UDP_IP = "192.168.0.107"
7 TABELAS_NO = ["192.168.0.108"]
8 TABELAS_PARAMETROS = "PeriodoDia_1=[0.5,0.6]%DiaSemana_1=[0.3,0.5];
   PeriodoDia_2=[0.9,0.8]%DiaSemana_2=[0.1,0.7];PeriodoDia_3
   =[0.4,0.6]%DiaSemana_3=[0.5,0.8];"
9 CHAVECADASTRO = ""
10 TABELAEVENTOS = [5 , 7 , 8] # invasao , incendio , terremoto
11
12 for C in "Cadastro": # Monto a chave
13     CHAVECADASTRO = CHAVECADASTRO + str(ord(C))
14
15 UDP.PORT = 5005
16 sock = socket.socket(socket.AF_INET, # Internet
17                     socket.SOCK_DGRAM) # UDP
18 sock.bind((UDP_IP, UDP.PORT))
19
20 #----- FUNCAO DE CADASTRO DE NOVO NO // THREAD -----
21 def newNode():
22     print ('SERVER IS RUNNING: ', UDP_IP)
23     while True:
24         msg, addr = sock.recvfrom(1024) # Buffer size is 1024
25         bytes
26         print(addr)

```

```

27         if addr[0] in TABELAS_NO:
28             sendTabelaParametroNewNode(addr[0])
29
30         elif int(msg.decode('utf-8')) == int(CHAVE_CADASTRO) :
31             # Mensagem criptografada de cadastro
32             print('cadastrando: '+ addr[0]+'...')
33             TABELAS_NO.append(addr[0])
34             print(TABELAS_NO)
35             sendTabelaParametroNewNode(addr[0]) # Envio de
36             # tabela de parametros para esse no
37         else:
38             print('Credenciais de cadastros sao invalidas.')
39             )
40 #----- FUNCAO PARA ENVIO DE TABELA DE PARAMETROS E EVENTOS
41
42 def sendTabelas():
43     time.sleep(3)
44     for NO_IP in TABELAS_NO : # Envio as tabelas para os nos da
45         # tabela de nos
46         sock.sendto(TABELAS_PARAMETROS.encode('utf-8')+
47                     TABELAS_EVENTOS.encode('utf-8'), (NO_IP, UDP_PORT))
48         print("Envio de Tabelas de Parametros para ", NO_IP)
49 #-----
50 #----- ENVIA AS TABELAS DE PARAMETROS E EVENTOS PARA O NOVO NO
51
52 def sendTabelaParametroNewNode(newNode_IP):
53     sock.sendto(TABELAS_PARAMETROS.encode('utf-8')+TABELAS_EVENTOS.
54                 encode('utf-8'), (newNode_IP, UDP_PORT))
55
56 thread.start_new_thread(sendTabelasParametros, ())
57 thread.start_new_thread(newNode, ())

```

O Código 5.6 referente aos nós, permite aos nós realizarem as funções de:

- Detecção de eventos;
- Receber as tabelas de parâmetros e eventos enviadas pelo MPC;
- Solicitar registro na tabela de nós do MPC

A tabela de eventos desses nós já devem vir preenchidas desde o MPC. Nesta aplicação ela é representada por um vetor onde cada elemento representa o valor de prioridade de cada evento Pe . A tabela de ação vem preenchida com as ações que o nó pode tomar a depender do cálculo de priorização final Pf . As variáveis SOLICITATABELAS E MESSAGE são as bases para a montagem da chave para a comunicação com o MPC.

A função responsável por solicitar o cadastro na tabela de nós é bastante simples, o nó envia uma mensagem solicitando seu cadastro para o MPC, o MPC envia uma mensagem criptografada de desafio para esse nó, assim o nó deve descriptografar essa mensagem desafio e enviar o resultado para o MPC para que esse possa verificar se a

mensagem de resposta está correta. Caso a mensagem de resposta esteja correta, o nó é adicionado à tabela de nós e são enviadas as tabelas de parâmetros e eventos para o nó. A função *ord()* é utilizada como encriptador, ela transforma cada caractere da mensagem para o seu respectivo código ASCII, o MPC por sua vez faz o processo inverso, transforma de código ASCII para caractere normal. Ao receber as tabelas de parâmetros, o nó as armazena na variável TABELASPARAMETROS.

Como foi dito previamente, a detecção de um evento é captada com o pressionar de um *pushbutton*. Quando esse botão é pressionado, ele chama a função *executaAcao()*, nesse caso ele envia como parâmetro o número do evento que foi detectado. Esse número é referente ao *index* do vetor TABELAEVENTOS, que no caso apresentado no código, é referente a um evento de invasão.

Dentro da função *executaAcao()* é onde todo o cálculo para a definição da ação do nó acontece. O valor de *Pe* é obtido pela tabela de eventos, os valores dos parâmetros são obtidos pela tabela de parâmetros correspondente ao nó. As tabelas de parâmetros enviadas nessa situação pelo MPC contém valores para os seguintes parâmetros:

- Período do dia (dia, noite);
- Dia da semana (semana, final de semana);

Com os parâmetros identificados, o nó verifica por meio das funções do seu sistema operacional onde ele consegue obter o horário e o dia da semana atual. Caso seja identificado um horário superior às 18h ele considera como noite e se for sábado ou domingo ele considera como final de semana. Com os valores de *Pe* e *Pa* obtidos, o nó realiza o cálculo para encontrar *Pf*. Como são três as ações possíveis a serem tomadas pelo nó, é definido um intervalo de valores para cada ação. A depender do seu valor, o nó toma uma ação diferente, imprimindo na tela qual a decisão tomada.

Código 5.6: Raspberry Nó

```

1 import socket
2 import time
3 import _thread as thread
4 import datetime
5 import RPi.GPIO as GPIO
6
7 UDP_IP = "192.168.0.107"
8 MY_IP = "192.168.0.108"
9 UDP_PORT = 5005
10 MESSAGE = "Cadastro"
11 MESSAGE_Encrypted = ""
12 TABELAS_PARAMETROS = ""
13 TABELAS_EVENTOS = ""
14 TABELA_ACAO = ["Disparar Alarme", "Enviar SMS", "Enviar Foto"]
15 SOLICITA_TABELAS = "Tabelas"
16 PARAMETROS_DEFINIDOS = []
17
18 sock = socket.socket(socket.AF_INET,
```



```

19             socket.SOCK_DGRAM)
20 sock.bind((MY_IP, UDP_PORT))
21
22 #----- BUTTON CLICK -----
23
24 GPIO.setmode(GPIO.BCM)
25
26 GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)
27
28 def buttonClick():
29     while True:
30         input_state = GPIO.input(18)
31         if input_state == False:
32             print('Invasao Detectada')
33             executaAcao(0)
34             time.sleep(0.2)
35
36 #-----
37
38 #----- FUNCAO PARA RECEBER TABELA DE PARAMETROS E EVENTOS -----
39
40 def receiveTabelas(): # Recebe tabelas de parametros e eventos
41     while True:
42         msg, addr = sock.recvfrom(1024)
43         if (msg.decode('utf-8') != ""):
44             global TABELAS_PARAMETROS
45             global TABELAS_EVENTOS
46             TABELAS_PARAMETROS = msg.decode('utf-8').split(' ')[0]
47             TABELAS_EVENTOS = msg.decode('utf-8').split(' ')[1]
48             print(TABELAS_PARAMETROS)
49
50
51 #-----
52
53 #----- FUNCAO DE SOLICITACAO DE CADASTRO -----
54
55 def solicitaCadastro():
56     for M in MESSAGE: # Criptografar a Mensagem
57         global MESSAGE_Encrypted
58         MESSAGE_Encrypted = MESSAGE_Encrypted + str(ord(M))
59
60     sock.sendto(MESSAGE_Encrypted.encode('utf-8'), (UDP_IP, UDP_PORT))
61     print("Cadastro")
62
63 #-----
64
65
66 def executaAcao(evento):
67     # O evento capturado foi detectado como sendo o alarme
68     Pe = TABELA_EVENTOS[evento]
69     defineParametros()
70     global PARAMETROS_DEFINIDOS

```

```

71     print(PARAMETROS_DEFINIDOS)
72
73     sum_parametros = 0
74     for pd in PARAMETROS_DEFINIDOS:
75         sum_parametros += float(pd)
76
77     print("Soma dos Parametros: ", sum_parametros)
78
79     Pf = ((sum_parametros)/2) * Pe
80
81     if Pf < 3:
82         print(TABELA_ACAO[0], " PF:", Pf)
83     if Pf >= 3 and Pf < 6:
84         print(TABELA_ACAO[1], " PF:", Pf)
85     if Pf >= 6:
86         print(TABELA_ACAO[2], " PF:", Pf)
87
88 def defineParametros():
89     aplicacao = "1" # Define o contexto de aplicacao (define qual tipo
90                     de tabela pertence a esse no)
91
92     now = datetime.datetime.now() # Verifica o horario e dia da semana
93     para definir os parametros
94     if(now.hour < 18): # Se for menor que 18h, entao eh dia
95         periodoDia = 0
96     else:
97         periodoDia = 1
98
99     if(now.today().weekday() >= 5): # 0: segunda, 5: sabado, 6: domingo
100         diaSemana = 0
101     else:
102         diaSemana = 1
103
104     global PARAMETROS_DEFINIDOS
105     PARAMETROS_DEFINIDOS = []
106     perDia = TABELAS_PARAMETROS.split("PeriodoDia_"+aplicacao+"=[", 1)
107     perDia = perDia[1].split("]", 1)
108     perDia = perDia[0].split(",", 1)
109     PARAMETROS_DEFINIDOS.append(perDia[periodoDia])
110
111     diaSem = TABELAS_PARAMETROS.split("PeriodoDia_"+aplicacao+"=[", 1)
112     diaSem = perDia[1].split("]", 1)
113     diaSem = perDia[0].split(",", 1)
114     PARAMETROS_DEFINIDOS.append(diaSem[diaSemana])
115
116 if TABELAS_PARAMETROS == "": # Solicita cadastro se nao estiver
117     cadastrado
118     solicitaCadastro()
119
120 thread.start_new_thread(receiveTabelasParametros,())
121 thread.start_new_thread(buttonClick,())

```

5.4 Conclusões da validação

A validação se mostrou viável ao abordar três formas de apresentar o funcionamento do modelo proposto. Inicialmente foi apresentado o fluxo de como as informações circulam dentro do modelo por meio da Rede de Petri, além de demonstrar que não existem *deadlocks* durante a execução do modelo. Depois foi apresentado uma simulação onde foi possível a representação de eventos e dispositivos reais que foram programados para atuar como o modelo determina, foi possível visualizar, mesmo que de forma simulada, como o modelo pode atuar em uma Cidade Inteligente. Por último foram configurados elementos que podem compor uma Cidade Inteligente para comprovar a real possibilidade de implementação desse modelo.

A partir destas três análises, é possível afirmar diante da completude dos resultados obtidos um ganho no refinamento no tratamento de eventos capturados por um nó.

Capítulo 6

Conclusão

Esta dissertação de mestrado apresentou um novo modelo de configuração dinâmica para cidades inteligentes com foco em redes de sensores de diferentes tipos. Para tanto, foi abordada desde a concepção das aplicações até os conceitos estruturantes da solução proposta. Por fim, foram realizadas simulações e experimentações comprovando a real funcionalidade e os ganhos que este modelo pode prover quando se fala em priorização de informação em cidades inteligentes. A solução proposta foi pensada para ser implementada em RSSF quaisquer, independente de suas especificidades. Por meio dos cenários apresentados nas simulações, foi possível comprovar que o modelo não é dependente características particulares da rede, como topologia, número de nós e latência.

Em termos científicos, este trabalho apresentou diversas contribuições na área de RSSF e Cidades Inteligentes, sobretudo por está alinhado com artigos bastante recentes encontrados na literatura. E embora esse fato tenha aumentando sobremaneira a complexidade para a realização das pesquisas, os resultados obtidos podem contribuir para os avanços das pesquisas nessa área a nível mundial; ao final deste trabalho, artigos científicos foram escritos e submetidos para periódicos científicos internacionais.

Após a conclusão deste trabalho de mestrado, pode-se dizer que os resultados obtidos foram satisfatórios, tanto matematicamente quanto na questão prática. O modelo se mostrou viável para aquilo que foi proposto, embora experimentos e validações adicionais, especialmente em ambientes maiores e com a presença de muitos nós, ainda possam ser realizados em trabalhos futuros.

No geral, trabalhos futuros estarão relacionados a adaptações do modelo proposto para novos cenários de monitoramento. Adicionalmente, questões de tempo de resposta e adaptação serão particularmente centrais em trabalhos futuros. Por fim, experimentos adicionais, sobretudo em cidades reais, são esperados como evolução natural deste trabalho.

Referências Bibliográficas

- [Ai e Abouzeid 2006] Ai, J. e Abouzeid, A. A. (2006). Coverage by directional sensors in randomly deployed wireless sensor networks. *Journal of Combinatorial Optimization*, 11(1):21–41.
- [Akyildiz et al. 2002] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., e Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer networks*, 38(4):393–422.
- [Atzori et al. 2010] Atzori, L., Iera, A., e Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15):2787–2805.
- [Avelar et al. 2015] Avelar, E., Marques, L., dos Passos, D., Macedo, R., Dias, K., e Nogueira, M. (2015). Interoperability issues on heterogeneous wireless communication for smart cities. *Computer Communications*, 58:4–15.
- [Bin et al. 2010] Bin, S., Yuan, L., e Xiaoyi, W. (2010). Research on data mining models for the internet of things. In *Image Analysis and Signal Processing (IASP), 2010 International Conference on*, pp. 127–132. IEEE.
- [Cantoni et al. 2006] Cantoni, V., Lombardi, L., e Lombardi, P. (2006). Challenges for data mining in distributed sensor networks. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 1, pp. 1000–1007. IEEE.
- [Chen e Varshney 2004] Chen, D. e Varshney, P. K. (2004). Qos support in wireless sensor networks: A survey. In *International conference on wireless networks*, volume 233, pp. 1–7.
- [Costa et al. 2017a] Costa, D. G., Collotta, M., Pau, G., e Duran-Faundez, C. (2017a). A fuzzy-based approach for sensing, coding and transmission configuration of visual sensors in smart city applications. *Sensors*, 17(1):93.
- [Costa et al. 2017b] Costa, D. G., Collotta, M., Pau, G., e Duran-Faundez, C. (2017b). A fuzzy-based approach for sensing, coding and transmission configuration of visual sensors in smart city applications. *Sensors*, 17:1–17.
- [Costa et al. 2018a] Costa, D. G., Duran-Faundez, C., Andrade, D. C., Rocha-Junior, J. B., e Just Peixoto, J. P. (2018a). Twittersensing: An event-based approach for wireless sensor networks optimization exploiting social media in smart city applications. *Sensors*, 18(4).

- [Costa et al. 2018b] Costa, D. G., Duran-Faundez, C., Andrade, D. C., Rocha-Junior, J. B., e Just Peixoto, J. P. (2018b). Twittersensing: An event-based approach for wireless sensor networks optimization exploiting social media in smart city applications. *Sensors*, 18(4).
- [Costa et al. 2017c] Costa, D. G., Figueredo, S., e Oliveira, G. (2017c). Cryptography in wireless multimedia sensor networks: A survey and research directions. *Cryptography*, 1(1).
- [Costa e Guedes 2011] Costa, D. G. e Guedes, L. A. (2011). Coverage-aware node-disjoint multipath selection in wireless multimedia sensor networks. In *2011 4th IFIP International Conference on New Technologies, Mobility and Security*, pp. 1–5.
- [Costa e Guedes 2013] Costa, D. G. e Guedes, L. A. (2013). Exploiting the sensing relevancies of source nodes for optimizations in visual sensor networks. *Multimedia tools and applications*, 64:549–579.
- [Costa et al. 2012] Costa, D. G., Guedes, L. A., Vasques, F., e Portugal, P. (2012). A routing mechanism based on the sensing relevancies of source nodes for time-critical applications in visual sensor networks. In *Wireless Days (WD), 2012 IFIP*, pp. 1–6. IEEE.
- [Costa et al. 2013] Costa, D. G., Guedes, L. A., Vasques, F., e Portugal, P. (2013). Energy-efficient packet relaying in wireless image sensor networks exploiting the sensing relevancies of source nodes and dwt coding. *Journal of Sensor and Actuator Networks*, 2(3):424–448.
- [Costa et al. 2014] Costa, D. G., Guedes, L. A., Vasques, F., e Portugal, P. (2014). Relevance-based partial reliability in wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2014(1):142.
- [Costa et al. 2015] Costa, D. G., Guedes, L. A., Vasques, F., e Portugal, P. (2015). Research trends in wireless visual sensor networks when exploiting prioritization. *Sensors*, 15(1):1760–1784.
- [Crawley et al. 1998] Crawley, E., Sandick, H., Nair, R., e Rajagopalan, B. (1998). A framework for qos-based routing in the internet.
- [Djahel et al. 2013] Djahel, S., Salehie, M., Tal, I., e Jamshidi, P. (2013). Adaptive traffic management for secure and efficient emergency services in smart cities.
- [Duran-Faundez et al. 2016] Duran-Faundez, C., Costa, D. G., Lecuire, V., e Vasques, F. (2016). A geometrical approach to compute source prioritization based on target viewing in wireless visual sensor networks. In *2016 IEEE World Conference on Factory Communication Systems (WFCS)*, pp. 1–7.
- [Goel et al. 2012] Goel, A., Ray, S., e Chandra, N. (2012). Intelligent traffic light system to prioritized emergency purpose vehicles based on wireless sensor network. *International Journal of Computer Applications*, 40(12):36–39.

- [Goncalves e Costa 2015] Goncalves, D. d. O. e Costa, D. G. (2015). A survey of image security in wireless sensor networks. *Journal of Imaging*, 1(1):4–30.
- [Guerrero-Zapata et al. 2010] Guerrero-Zapata, M., Zilan, R., Barceló-Ordinas, J. M., Bcakci, K., e Tavli, B. (2010). The future of security in wireless multimedia sensor networks. *Telecommunication Systems*, 45(1):77–91.
- [Hadim e Mohamed 2006] Hadim, S. e Mohamed, N. (2006). Middleware for wireless sensor networks: A survey. In *Communication System Software and Middleware, 2006. Comsware 2006. First International Conference on*, pp. 1–7. IEEE.
- [Hussain et al. 2009] Hussain, M. A., kyung Sup, K., et al. (2009). Wsn research activities for military application. In *Advanced Communication Technology, 2009. ICACT 2009. 11th International Conference on*, volume 1, pp. 271–274. IEEE.
- [Jin et al. 2014] Jin, J., Gubbi, J., Marusic, S., e Palaniswami, M. (2014). An information framework for creating a smart city through internet of things. *IEEE Internet of Things Journal*, 1(2):112–121.
- [Lampkin et al. 2012] Lampkin, V., Leong, W. T., Olivera, L., Rawat, S., Subrahmanyam, N., Xiang, R., Kallas, G., Krishna, N., Fassmann, S., Keen, M., et al. (2012). *Building smarter planet solutions with mqtt and ibm websphere mq telemetry*. IBM Redbooks.
- [Lecuire et al. 2007] Lecuire, V., Duran-Faundez, C., e Krommenacker, N. (2007). Energy-efficient transmission of wavelet-based images in wireless sensor networks. *Journal on Image and Video Processing*, 2007(1):15–15.
- [Loureiro et al. 2003] Loureiro, A. A., Nogueira, J. M. S., Ruiz, L. B., Mini, R. A. d. F., Nakamura, E. F., e Figueiredo, C. M. S. (2003). Redes de sensores sem fio. In *Simpósio Brasileiro de Redes de Computadores*, volume 21, pp. 19–23.
- [Mehdi et al. 2014] Mehdi, K., Lounis, M., Bounceur, A., e Kechadi, T. (2014). Cupcarbon: A multi-agent and discrete event wireless sensor network design and simulation tool. In *7th International ICST Conference on Simulation Tools and Techniques, Lisbon, Portugal, 17-19 March 2014*, pp. 126–131. Institute for Computer Science, Social Informatics and Telecommunications
- [Melo Jr et al.] Melo Jr, W. S., Prado, C. B., e Carmo, L. F. Autenticação de sensores usando eventos físicos.
- [Muhammad et al. 2006] Muhammad, N., Chiavelli, D., Soldani, D., e Li, M. (2006). Introduction. *QoS and QoE Management in UMTS Cellular Systems*, pp. 1–8.
- [National Intelligence Council NIC 2008] National Intelligence Council NIC, D. C. T. (2008). Six technologies with potential impacts on us interests out to 2025. *Conference Report CR 2008-07, April 2008*, <http://www.dni.gov/nic/NIC_home.html> .

- [Papán et al. 2012] Papán, J., Jurečka, M., e Púchyová, J. (2012). Wsn for forest monitoring to prevent illegal logging. In *Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on*, pp. 809–812. IEEE.
- [Pathan et al. 2006] Pathan, A.-S. K., Lee, H.-W., e Hong, C. S. (2006). Security in wireless sensor networks: issues and challenges. In *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, volume 2, pp. 6–pp. IEEE.
- [Patil et al. 2012] Patil, S., Kumar, V., Singha, S., e Jamil, R. (2012). A survey on authentication techniques for wireless sensor networks. *International Journal of Applied Engineering Research*, 7(11):2012.
- [Peixoto e Costa 2015] Peixoto, J. P. J. e Costa, D. G. (2015). Qoe-aware multiple sinks mobility in wireless sensor networks. In *2015 7th International Conference on New Technologies, Mobility and Security (NTMS)*, pp. 1–4.
- [Pescaru e Curiac 2014] Pescaru, D. e Curiac, D.-I. (2014). Anchor node localization for wireless sensor networks using video and compass information fusion. *Sensors*, 14:4211–4224.
- [Rajeswari e Seenivasagam 2016] Rajeswari, S. R. e Seenivasagam, V. (2016). Comparative study on various authentication protocols in wireless sensor networks. *The Scientific World Journal*, 2016.
- [Sahoo e Hwang 2011] Sahoo, P. K. e Hwang, I.-S. (2011). Collaborative localization algorithms for wireless sensor networks with reduced localization error. *Sensors*, 11:9989–10009.
- [Sen 2010] Sen, J. (2010). A survey on wireless sensor network security. *arXiv preprint arXiv:1011.1529*.
- [Sharif et al. 2010] Sharif, A., Potdar, V., e Rathnayaka, A. (2010). Prioritizing information for achieving qos control in wsn. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pp. 835–842. IEEE.
- [Xia et al. 2012] Xia, F., Yang, L. T., Wang, L., e Vinel, A. (2012). Internet of things. *International Journal of Communication Systems*, 25(9):1101.
- [Yaghmaee et al. 2013] Yaghmaee, M. H., Bahalgardi, N. F., e Adjeroh, D. (2013). A prioritization based congestion control protocol for healthcare monitoring application in wireless sensor networks. *Wireless personal communications*, 72(4):2605–2631.
- [Yick et al. 2008] Yick, J., Mukherjee, B., e Ghosal, D. (2008). Wireless sensor network survey. *Computer networks*, 52(12):2292–2330.
- [Zanella et al. 2014] Zanella, A., Bui, N., Castellani, A., Vangelista, L., e Zorzi, M. (2014). Internet of things for smart cities. *IEEE Internet of Things journal*, 1(1):22–32.

-
- [Zhang et al. 2010] Zhang, X., Heys, H. M., e Li, C. (2010). Energy efficiency of symmetric key cryptographic algorithms in wireless sensor networks. In *2010 25th Biennial Symposium on Communications*, pp. 168–172.
- [Zhang et al. 2014] Zhang, Y., Sun, L., Song, H., e Cao, X. (2014). Ubiquitous wsn for healthcare: Recent advances and future prospects. *IEEE Internet of Things Journal*, 1(4):311–318.