UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
DEPARTAMENTO ACADÊMICO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

FÁBIO CÉSAR SCHUARTZ

SIMULADOR PARA CÁLCULO DE PROBABILIDADE DE FALHA PARA REDES COOPERATIVAS SEM FIO COM RESTRIÇÕES DE SIGILO

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2013

FÁBIO CÉSAR SCHUARTZ

SIMULADOR PARA CÁLCULO DE PROBABILIDADE DE FALHA PARA REDES COOPERATIVAS SEM FIO COM RESTRIÇÕES DE SIGILO

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso II, do Curso Superior em Engenharia de Computação dos Departamentos Acadêmicos de Eletrônica e Informática da Universidade Tecnológica Federal do Paraná, apresentado como requisito parcial para obtenção do título de Engenheiro de Computação.

Orientador: Prof. Dr. João Luiz Rebelatto

CURITIBA

2013

FÁBIO CÉSAR SCHUARTZ

SIMULADOR PARA CÁLCULO DE PROBABILIDADE DE FALHA PARA REDES COOPERATIVAS SEM FIO COM RESTRIÇÕES DE SIGILO

Este trabalho de conclusão de curso foi julgado e aprovado como requisito parcial para obtenção do título de Engenheiro de Computação pela Universidade Tecnológica Federal do Paraná.

Curitiba, 11 de outubro de 2013.

_____

Prof. Dr. Hugo Vieira Neto
Coordenador de Curso
Departamento Acadêmico de Eletrônica

_____

Prof. Dr. Dario Eduardo Amaral Dergint
Responsável pela Disciplina de Trabalho de Conclusão de Curso II
Departamento Acadêmico de Eletrônica

BANCA EXAMINADORA

_____ Prof. Dr. João Luiz Rebelatto Orientador	_____ Prof. Dr. Richard Demo Souza
_____ Prof. Dr. Dario Eduardo Amaral Dergint	

RESUMO

SCHUARTZ, Fábio César. **Simulador para Cálculo de Probabilidade de Falha para Redes Cooperativas Sem Fio com Restrições de Sigilo.** 2013. Monografia (Graduação) – Curso de Engenharia de Computação, UTFPR, Curitiba, Brasil.

Segundo estudos publicados na Conferência Internacional IEEE sobre Comunicação em *Smart Grids*, em 2011, é necessário modernizar as linhas de transmissão de energia, transformando-as em um meio de controle da transmissão e distribuição de energia. Neste ambiente - um exemplo entre diversas aplicações que requerem sigilo no envio de informações - informações privadas circulam pela rede elétrica, sujeitas a escuta não autorizada e interferências, existindo a necessidade de proteção da informação. O objetivo geral deste trabalho é construir um simulador para calcular a probabilidade de falha de sigilo em uma rede cooperativa sem fio com restrições de sigilo. Este projeto é composto de três fases. A primeira fase é a modelagem da aplicação e a definição da interface gráfica. A segunda fase é a construção do código executável do *software* e a terceira fase são os testes e a coleta de dados provenientes da execução da aplicação. A aplicação será modelada utilizando UML, desenvolvida em linguagem de programação Java e apresentará ao usuário uma topologia de rede sem fio, composta por dois emissores, um receptor e um nodo de escuta passiva. Podendo variar a distância entre os nodos, para cada emissor é possível observar em um gráfico a probabilidade de perda de segredo (*Secrecy Outage Probability*) utilizando a transmissão direta de dados (DT), *Decode-and-Forward* (DF) e *Generalized Dynamic Network Coding* (GDNC), com e sem o nodo espião conhecendo a topologia da rede. Como resultado, obteve-se uma aplicação capaz de gerar as curvas de probabilidade de falha de sigilo para cada estratégia que condizem com a teoria, ficando para trabalhos futuros incorporar novas estratégias de comunicação e formas de medição de desempenho.

Palavras-chave: Simulador. Codificação de rede. Redes sem fio. Probabilidade de falha de sigilo. Orientado a objetos.

ABSTRACT

SCHUARTZ, Fábio César. **Simulator for Calculation of Outage Probability in Cooperative Wireless Networks with Secrecy Restrictions.** 2013. Monograph (Undergraduate) – Computer Engineering Course, UTFPR, Curitiba, Brazil.

According with studies published at IEEE International Conference about Smart Grids, in 2011, it's necessary update the power lines, making them a way to control and distribute energy. In this scenario – an example like many others applications that require secrecy during information communication – private informations goes through the energy line, susceptible to unauthorized eavesdropping and interferences, resulting in a necessity to protect the information. The general goal of this paper is build a simulator able to calculate the secrecy outage probability ina a cooperative wireless network with secrecy restrictions. This project has three phases. First phase is the application modeling and the graphic interface definition. The second phase is building the runnable software code and the third phase are the tests and collecting data from the execution of the application. The application will be modeled using UML, developed in Java programing language and will show to the user a wireless network topology, with two sources, one destination and one passive eavesdropper node. The distances can change between nodes and for each source is possible to watch a secrecy outage probability graph using direct transmission, decode-and-forward and generalized dynamics network coding, with and without a dumb eavesdropper knowing the network topology. As the result, it is built an application capable of generating secrecy outage probability graphs for each strategy that are expected by the theory, resting for future works implement new communication strategies and ways to measure performance.

Keywords: Simulator. Network Coding. Wireless. Secrecy Outage Probability. Object-Oriented.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

BER	Binaty Error Rate (Taxa de Erro Binário)
BPL	Broadband over Power Lines (Banda Larga sobre Linhas de Energia)
CA	Autoridade Certificadora
COM	Critical Path Method (Método do Caminho Crítico)
DNC	Dynamic-Network Coding (Codificação de Rede Dinâmica)
DT	Direct Transmission (Transmissão direta)
EF	Environmental Factor (Fator de Complexidade de Ambiente)
FD	Decode-and-Forward (Decodificar e Repassar)
GDNC	Generalized Dynamic Network Coding (Codificação de Rede Dinâmica Generalizada)
IF	Information Frames (Quadros de Informações)
JVM	Java Virtual Machine (Máquina Virtual Java)
MAC	Message Authentication Code (Código de Autenticação de Mensagens)
PERT	Program Evaluation and Review Technique (Método de Avaliação e Revisão de Programa)
PF	Parity Frames (Quadros de Paridade)
PKI	Public Key Infrastructure (Infraestrutura de Chave Pública)
PLC	Power Line Communication (Comunicação por Linha de Energia)
PLNC	Physical Layer Network Coding (Codificação de Rede de Camada Física)
RA	Autoridade de Registro
SCADA	Supervisory Control and Data Acquisition (Controle Supervisório e Aquisição de Dados)
SDG	Smart Distribution Grid (Rede de Distribuição Inteligente)
SNR	Relação Sinal-Ruído
SOP	Secrecy Outage Probability (Probabilidade de Falha de Sigilo)
TCC	Trabalho de Conclusão de Curso
TCF	Technical Complexity Factor (Fator de Complexidade Técnica)
UAW	Unadjusted Actor Weight (Peso Não Ajustado dos Atores)
UCP	Use Case Points (Pontos de Casos de Uso)

UML	Unified Modeling Language (Linguagem de Modelagem Unificada)
UUCP	Unadjusted Use Case Point (Pontos de Caso de Uso Não Ajustados
UUCW	Unadjusted Use Case Weight (Peso Não Ajustado dos Casos de Uso
WMN	Wireless Mesh Network (Rede Mesh sem Fio)
WSN	Wireless Sensor Networks (Redes de Sensores sem Fio)
WTC	Wiretap Channel (Canal de Escuta)

SUMÁRIO

1 INTRODUÇÃO

Em nossa sociedade tecnológica atual, a comunicação entre dispositivos eletrônicos é essencial para manter o padrão de vida esperado nas grandes comunidades.

Com o passar dos anos, diversos meios de comunicação foram criados e utilizados para este fim. Inicialmente foi criado o telégrafo, capaz de transmitir dois tipos de sinais, formando um código que podia ser interpretado como palavras [HUURDEMAN, 2003]. A seguir veio o telefone, chamado originalmente por seu criador, *Innocenzo Manzetti*, de “telégrafo falante” [QUÉTÀND, 1865]. Pouco mudou em seu conceito até os dias de hoje. A evolução seguinte foi a transmissão por ondas de rádio, uma tecnologia utilizada em grande escala atualmente, seja em telefones celulares ou *laptops*.

Atualmente diversos meios de comunicação entre dois dispositivos eletrônicos existem. Conexão por cabos coaxiais, fibra óptica, ondas de rádio (por exemplo, *WI-FI* e *Bluetooth*), microondas, infravermelho, ultrassom e linhas de energia são alguns exemplos.

A comunicação através das linhas de energia, também chamada *PLC* (*Power Line Communication*), começou a ser estudada no início do século 20, onde as primeiras patentes foram registradas. Desde então, surgiram dispositivos eletrônicos capazes de transmitir dados em baixa velocidade (*narrowband*), na faixa de poucos *kbps*. À medida que esta tecnologia foi amadurecendo e suas aplicações foram difundidas, começaram a aparecer sistemas de *PLC* capazes de operar em banda larga (*broadband*), alcançando velocidades de até 200 Mbps. Com o incremento da velocidade de transmissão, o *PLC* proporciona hoje em dia o serviço de internet por banda larga para clientes residenciais, conectividade entre escritórios e comando e controle remoto para automação e medidas remotas. Este crescimento na utilização do *PLC* se deve em grande parte pelos benefícios que a infraestrutura das linhas de energia proporciona, sendo mais extensiva que outros meios de comunicação [GALLI; SCAGLIONE; WANG, 2011].

Embora a comunicação por *PLC* não tenha espaço amplo no mercado, preocupações com o meio-ambiente levam a necessidade de modernizar as

linhas de energia, transformando-as em um meio de controle da transmissão e distribuição de energia, denominado *Smart Grid*. Uma das metas é permitir uma métrica avançada para adquirir leituras de consumo de eletrodomésticos residenciais em uma central e enviar informações sobre preços e tarifas diretamente ao consumidor através desses mesmos dispositivos. Outra meta é permitir o controle de sistemas de distribuição avançadas, os quais coletam medidas de sensores distribuídos, permitindo estimar o comportamento de fornecedores e consumidores de uma maneira autônoma, aumentando assim a eficiência, economia, confiabilidade e sustentabilidade da produção e distribuição de energia elétrica [PHULPIN; BARROS; LUCANI, 2011].

A infraestrutura de comunicação de um *Smart Grid* deve permitir uma comunicação confiável e eficiente de uma grande quantidade de dados em um tempo reduzido. Uma solução para atingir estas metas é através da codificação de redes (*Network Coding*), uma técnica objetivando obter máximo fluxo de informações em redes. Esta técnica permite um aumento do tráfego na rede e reduz as operações de controle, aumentando assim sua eficiência na comunicação. A ideia por trás da codificação de redes é permitir os nodos intermediários da comunicação a executar transformações algébricas nos pacotes de dados, codificando-os, ao invés apenas de repassá-los adiante. Assim, o receptor, tendo informações lineares suficientes, é capaz de decodificar o pacote e extrair as informações mandadas originalmente para ele.

A codificação de rede também engloba o aspecto de segurança. A informação transmitida deve manter a sua integridade contra interferências durante a transmissão, pois um simples pacote que seja corrompido irá afetar os cálculos algébricos e contaminar outros pacotes, tornando-os irrecuperáveis. Outro aspecto é manter a informação sigilosa para pessoas não autorizadas a recebê-las, sendo assim resistente a ataques maliciosos.

A codificação de redes em sua forma básica é vulnerável a vários tipos de ataques, sendo assim necessário considerar diversos aspectos de segurança. Diversos estudos sobre segurança foram feitos, considerando vários tipos de ataques que a rede pode sofrer. Por exemplo, a confidencialidade de mensagens pode ser obtida através da criptografia dos coeficientes selecionados pelo remetente [VILELA; LIMA; BARROS, 2008], por meio de criptografia homomórfica [FAN; JIANG; SHEN, 2009] ou permutando

os símbolos dos dados [ZHANG et al., 2010]. Já ataques de poluição podem ser prevenidos através da teoria da informação [HO et al., 2004], criptografia incluindo *MAC* (*Message Authentication Code* – Código de Autenticação de Mensagens) homomórficos [AGRAWAL; BONEH, 2009], *hashes* homomórficos [GKANTSIDIS; RODRIGUEZ, 2006] ou assinaturas homomórficas [BONEH; FREEMAN, 2009].

Neste trabalho será construído um simulador para demonstrar as diferentes estratégias de transmissão de informação em uma rede cooperativa sem fio com restrição de sigilo, calculando a probabilidade de falha de sigilo para cada um dos nodos emissores. Para este estudo serão consideradas as variáveis de coeficiente de perda do meio, relação sinal-ruído (SNR) entre os nodos e as distâncias entre os nodos emissores, destino e espião. Serão analisadas as estratégias de transmissão direta (DT), *decode-and-forward* (DF), *Generalized Dynamic Network Coding* (GDNC – Codificação de Rede Dinâmica Generalizada) e *Generalized Dynamic Network Coding with Dumb Eavesdropper* (GDNC *with Dumb Eavesdropper* – Codificação de Rede Dinâmica Generalizada com Espião Desinformado), utilizando uma topologia de rede com dois nodos emissores, um nodo destino e um nodo espião passivo.

1.1 Motivação e justificativa

A motivação para este projeto é fornecer uma ferramenta de simulação para visualização da probabilidade de falha de sigilo em redes cooperativas, de forma que possa ser utilizado futuramente como plataforma de estudo para comparação de diversas estratégias de comunicação entre nodos em uma rede.

Este trabalho oferece uma ferramenta para analisar uma rede cooperativa sem fio, porém não é limitada a apenas esta topologia. Redes com fio e redes PLC também podem aproveitar das simulações abordadas neste trabalho, sendo assim, um recurso para trabalhos envolvendo estratégias de transmissão de informação, com e sem codificação de rede.

1.2 Objetivo geral

O objetivo geral deste trabalho é construir um simulador para calcular a probabilidade de falha de sigilo em uma rede cooperativa sem fio com restrições de sigilo.

1.3 Objetivos específicos

Como objetivos específicos, têm-se:

- ✓ Desenvolver um simulador utilizando a linguagem de programação Java;
- ✓ Realizar simulações utilizando dados entrados pelo usuário e considerando a estratégia de transmissão direta;
- ✓ Realizar simulações utilizando dados entrados pelo usuário e considerando a estratégia DF;
- ✓ Realizar simulações utilizando dados entrados pelo usuário e considerando a estratégia GDNC;
- ✓ Realizar simulações utilizando dados entrados pelo usuário e considerando a estratégia GDNC com *Dumb Eavesdropper*;
- ✓ Exibir os resultados obtidos para cada nodo emissor em gráficos separados, comparando as curvas obtidas em cada estratégia.

1.4 Requisitos do sistema

O projeto para a simulação da probabilidade de falha de sigilo utilizará como recurso de *hardware* um computador (CPU, monitor, teclado, *mouse*, etc.) e como recursos de *software* programas específicos para a programação em linguagem *Java*.

O *software* apresentado será capaz de demonstrar as probabilidades de falha de sigilo para quatro estratégias diferentes de transmissão de informação em uma rede cooperativa. Na interface será mostrada uma janela com a topologia da rede, duas janelas de resultados com os gráficos referentes a cada nodo emissor, os botões de comando para ativação e desativação do

GDNC, GDNC com *Dumb Eavesdropper* e DF; e caixas de entradas para os sete parâmetros controlados pelo usuário.

A Figura 1 mostra o protótipo da interface homem-máquina da aplicação.

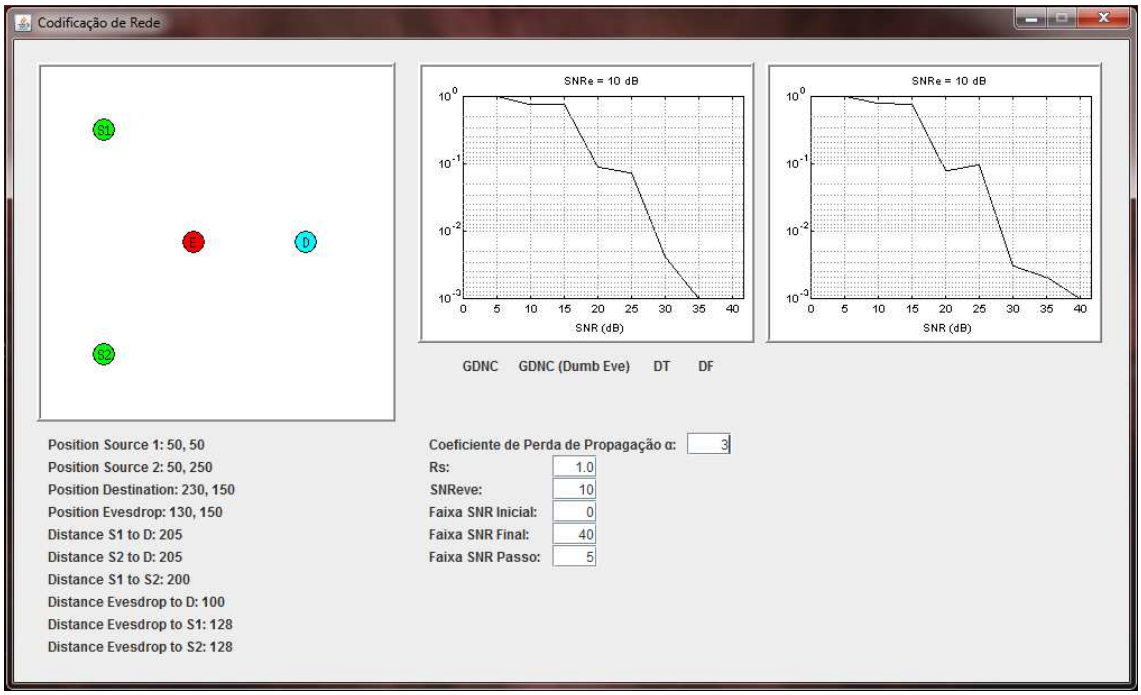

Figura 1: Protótipo da interface homem-máquina. Autoria própria.

1.4.1 Recursos de *Hardware*

O único recurso de *hardware* utilizado no projeto é um computador de mesa completo, incluindo os periféricos necessários para o funcionamento do mesmo: teclado, mouse e monitor.

O equipamento é de propriedade pessoal, não gerando custos para o projeto.

1.4.2 Recursos de *Software*

Os recursos de *software* incluem o sistema operacional Windows 7, o *kit* de desenvolvimento Java SDK e o ambiente de desenvolvimento IDE Eclipse. A linguagem de programação utilizada é o Java.

O Java SDK e o Eclipse são *softwares* de uso gratuito e, portanto, não geram gastos financeiros. A linguagem Java não é proprietária e também não existe custo para sua utilização. Somente o sistema operacional utilizado – Windows 7 – requer a aquisição de licença de uso, porém será utilizado o computador pessoal do aluno, já contendo o sistema operacional instalado, não gerando, assim, custos para o projeto.

1.4.3 Requisitos funcionais

- O usuário pode iniciar ou parar a estratégia GDNC;
- O usuário pode iniciar ou parar a estratégia GDNC Dumb;
- O usuário pode iniciar ou parar a estratégia DF;
- O usuário pode inserir um valor para o coeficiente de perdas α;
- O usuário pode inserir um valor para a taxa de sigilo desejado Rs;
- O usuário pode inserir um valor para o nível de sinal-ruído do nodo espião SNReve;
- O usuário pode inserir um valor inicial para a faixa de sinal-ruído SNR;
- O usuário pode inserir um valor final para a faixa de sinal-ruído SNR;
- O usuário pode inserir um valor de passo para a faixa de sinal-ruído SNR;
- O usuário pode inserir um valor para o número de iterações;
- O usuário pode controlar a posição dos nodos na janela da topologia de rede;

1.4.4 Requisitos Não-Funcionais

- Não é possível controlar a posição de mais de um nodo por vez;
- O sistema de processar as alterações feitas nos dados de entrada pelo usuário e apresentar os gráficos resultantes em menos de um segundo, para um número de iterações igual ou menor que 10.000;

1.5 Metodologia

Este trabalho apresentará um simulador capaz de calcular a probabilidade de falha de sigilo para diversas estratégias de transmissão, dentro de uma rede cooperativa sem fio, com restrições de sigilo. Utilizando a linguagem de programação Java, sobre uma plataforma Windows, será desenvolvido um *software*, composto por uma interface gráfica, elementos de entrada para diversas variáveis do sistema e botões que controlam a visualização dos resultados, simulando uma rede com quatro nodos: dois nodos emissores, um nodo receptor e um nodo espião passivo.

Inicialmente será calculada a probabilidades de falha de sigilo das informações transmitidas entre dois nodos emissores, um nodo destino e um nodo espião, utilizando apenas a transmissão direta de informações. A partir deste cenário, serão observados dois gráficos resultantes, em janelas distintas, referentes à probabilidade de cada nodo emissor possuir falha de sigilo, considerando o coeficiente de perdas de propagação, a taxa de sigilo desejada, o nível de sinal-ruído do nodo espião e as distâncias dos nodos na topologia de rede. Nestes gráficos, será variado o nível de sinal-ruído dos canais emissores e observado a probabilidade de, em um determinado nível de sinal-ruído, acontecer falha de sigilo das informações transmitidas.

Na segunda etapa do trabalho, serão adicionadas as estratégias de codificação de rede GDNC e GDNC com *dumb eavesdrop* e a estratégia DF no cenário descrito anteriormente. Agora com as informações sendo codificadas dentro da rede, serão observados novamente os gráficos resultantes, segundo as mesmas métricas adotadas na etapa anterior.

A partir dos dados coletados em ambas as etapas, será analisado qualitativamente o aspecto de segurança fornecido pela codificação de rede GDNC sobre uma rede cooperativa, demonstrando assim, a validade da aplicação quando comparado com os resultados teóricos.

O ponto examinado para determinar a segurança da rede será o nível de sinal-ruído necessário para determinada estratégia de transmissão atingir um determinado valor de falha da capacidade de sigilo. Por exemplo, se desejarmos obter um valor de probabilidade de falha de sigilo de 1%, cada

forma de transmissão possuirá um valor de nível de sinal-ruído necessário para tal ser atingido. A estratégia de transmissão com o menor valor SNR será, assim, mais vantajosa.

O nível de segurança de cada estratégia dependerá da razão sinal-ruído entre os nodos; a distância entre eles; o poder de transmissão dos nodos; o ruído termal do meio; o coeficiente de falhas de transmissão; e da taxa de sigilo desejada.

O aplicativo será construído utilizando a linguagem de programação Java, sendo assim executável em diferentes plataformas (Windows, Linux, Mac, por exemplo) através da máquina virtual JVM. Será utilizado o *kit* de desenvolvimento da *Oracle* [ORACLE, 2013], o *Java SE JDK versão 7, update* 17 [ORACLE, 2013]. O sistema operacional escolhido é o Windows, executando em uma plataforma compatível com o padrão IBM-PC.

A aplicação será responsável por toda a simulação envolvida no projeto, sem a necessidade de ferramentas extras para este fim. Em trabalhos futuros, poderão ser incluídas novas estratégias para comparar com a segurança da codificação de rede, conforme propostos em trabalhos relacionados.

1.6 Organização do documento

Este documento propõe um trabalho de conclusão de curso na área de Engenharia de Computação. Ele está dividido em sete capítulos, sendo o primeiro capítulo a introdução. O segundo capítulo refere-se ao levantamento bibliográfico e ao estado da arte. O terceiro capítulo traz o desenvolvimento do trabalho. A seguir, no capítulo quatro, os procedimentos de teste e validação. O capítulo cinco trata da gestão do projeto. O capítulo seis refere-se a trabalhos futuros e finalmente o capítulo sete contém as considerações finais.

2 ESTUDOS

Nas últimas décadas, a demanda por energia elétrica cresceu acima da taxa em que podemos gerar esta energia por meios tradicionais. Resultado de investimentos limitados feitos no passado e pelo surgimento de novas necessidades, a rede de energia elétrica necessita ser modernizada.

Balancear a geração e a demanda de energia elétrica requer a integração adicional de tecnologias de proteção e controle que garantem a estabilidade do sistema, e esta tarefa não é trivial no atual modelo de distribuição de energia elétrica. Conforme [GALLI; SCAGLIONE; WANG, 2011], as redes de energia elétrica são concebidas para serem gerenciadas através de um arcaico modelo de infraestrutura centralizada, conhecido como *SCADA* (*Supervisory Control and Data Acquisition* – Controle Supervisório e Aquisição de Dados). Assim, surge o conceito de redes inteligentes (*Smart Grids*). As redes inteligentes visam suprir as deficiências da rede atual de distribuição de energia, através do monitoramento de amplas áreas, comunicação bidirecional e funcionalidades de controle avançadas.

O sistema de distribuição elétrica evolui para um novo paradigma graças às técnicas avançadas de comunicação e controle, que aumentam a eficiência operacional e econômica. Esta evolução engloba dois pontos principais:

a) Um sistema avançado de medição de energia capaz de coletar informações sobre o consumo individual de cada cliente, na estação-base, e retornar informações gerais sobre preços e tarifas diretamente aos dispositivos dos consumidores;
b) O gerenciamento de um sistema de distribuição avançado com funcionalidades que incluem sensores de medição capazes de estimarem o estado de distribuição do sistema e controle da troca de mensagens entre diversos centros de controle, tais como subestações automatizadas, microredes e depósitos de energia distribuída.

Para dar suporte a este sistema de troca de informações dos sensores e controle de mensagens, diversos meios de comunicação foram propostos: comunicação por cabos, comunicação sem fio (*wireless*) e comunicação por

linhas de energia (*PLC – Power Line Communication*). A construção de uma infraestrutura auxiliar para a comunicação por cabos requer custos altos adicionais; a comunicação sem fio enfrenta perda de sinal, ruído e obstáculos no caminho; e o *PLC* possui forte atenuação do sinal de comunicação e interferência no espectro [PHULPIN; BARROS; LUCANI, 2011].

O *PLC* utiliza a infraestrutura da rede de distribuição de energia para a comunicação, fornecendo um canal que, por sua própria natureza, liga todo consumidor à central de distribuição de energia. Entretanto, este meio de comunicação aberto está sujeito a potenciais ataques maliciosos e não garante a privacidade da informação transmitida.

Na década de 90, desenvolveu-se o uso da banda larga em redes elétricas (*PLC*). A ideia básica da tecnologia *BPL* (*Broadband over Power Lines* – Banda Larga sobre Linhas de Energia) é modular um sinal de rádio com dados e enviar através da rede elétrica em frequências que não são utilizadas para distribuição de energia, permitindo a coexistência dos sinais no mesmo meio de transmissão.

Entretanto, estas novas capacidades apresentadas pelas redes e sistemas do *Smart Grid* – inteligência distribuída e capacidade de banda larga, também criam novas vulnerabilidades se utilizadas sem um controle de segurança apropriado. Estas vulnerabilidades podem comprometer a confidencialidade, integridade ou disponibilidade da informação, seja sua causa ocorrida por agentes passivos (interferências) ou ativos (ataques maliciosos) [METKE; EKL, 2010].

Normalmente, aumentar a segurança implica em custos adicionais, pois mecanismos de segurança requerem computações adicionais, introduzem atrasos e aumentam os requisitos de armazenamento.

O conceito de codificação de rede foi proposto em 2000, por Ahlswede [AHLSWEDE et al., 2000]. O conceito principal por trás da codificação de rede é que, ao se permitir que os nodos intermediários em uma rede possam misturar diferentes pacotes de dados através de combinações algébricas de vários datagramas, o fluxo de dados e a robustez da rede podem ser melhorados [PHULPIN; BARROS; LUCANI, 2011].

Embora a codificação de rede seja uma solução para o aumento da eficiência em transmissão de dados, ele também é vulnerável a vários ataques.

Introduzem-se então mecanismos de segurança que fornecem proteção contra ataques de poluição, ao preço de custos maiores.

Entretanto, Franz [FRANZ; PFENNING; FISCHER, 2012] mostra em seu trabalho que métodos seguros de codificação de rede podem ainda fornecer benefícios sobre o fluxo e largura de banda como intencionado pela codificação de rede simples.

2.1 Estado da arte

No trabalho de Wang, Lin e Zhang (WANG; LIN; ZHANG, 2010), são apresentadas as aplicações de redes de sensores em sistemas de energia elétrica para *Smart Grids*. Estas redes WSN (*Wireless Sensor Networks*) não podem utilizar as soluções de segurança convencionais para redes de sensores sem fio genéricas, e uma nova arquitetura de segurança é proposta, considerando os requisitos de segurança da informação nos sistemas de energia elétrica.

Já Metke e Ekl (METKE; EKL, 2010) discutem as tecnologias de segurança chaves para um sistema de *Smart Grid*, incluindo infraestruturas de chave pública e computação confiável.

Wang e Yi (WANG; YI, 2011) analisam a questão de segurança da SDG (*Smart Distribution Grid*), inicialmente propondo uma arquitetura de comunicação sem fio baseado em WMN (*Wireless Mesh Network*) e depois analisando potenciais ataques à segurança e possíveis medidas para contra-atacar, incluindo a construção de um novo método de detecção e resposta a invasões.

Outro estudo, de Li, Lai e Zhang (LI; LAI; ZHANG, 2011), analisa o problema de como estimar seguramente o estado do sistema e o controle do *smart grid*. No cenário estudado, os sensores e o controlador comunicam-se através de um canal sem fio sujeito à escuta de um intruso.

Franz, Pfenning e Fischer (FRANZ; PFENNING; FISCHER, 2012) discutem parâmetros para medir a eficiência de codificação de rede seguras, considerando que para garantir a transmissão confiável de dados é necessário

introduzir aspectos de segurança contra ataques de poluição, aumentando assim o custo computacional do sistema.

Li *et al* (LI; GONG; LAI; HAN; QIU; YANG, 2012) propõem um método de comunicação sem fio para a infraestrutura de medição avançada em *Smart Grids* que pode significativamente aumentar o espectro de eficiência, transmitindo apenas quando ocorre uma troca significativa de consumo de energia.

A equipe de Yamaguchi (YAMAGUCHI; TAKYU; FUJII; OHTSUKI; SASAMORI; HANDA, 2013) analisam a capacidade de segurança de PLNCs (*Physical Layer Network Coding*) com estações de nodo não confiáveis. No PLNC, dois terminais independentes acessam o nodo da estação, simultaneamente, para então o nodo da estação transmitir o sinal combinado dos dois terminais. Cada terminal detecta a informação do outro terminal devido o cancelamento do seu próprio sinal no sinal transmitido.

Em seu trabalho, Gabry (GABRY, F., 2012) propõe o cálculo da probabilidade de falha de sigilo utilizando as estratégias de transmissão direta e *decode-and-forward* (DF) para uma rede cooperativa sem fio.

Já o estudo do grupo de Rebelatto (REBELATTO, et al., 2012) propõe o cálculo das probabilidades de falha de sigilo para as estratégias GDNC e GDNC com *Dumb Eavesdropper*.

Este trabalho propõe analisar a probabilidade de falha de sigilo utilizando as estratégias de transmissão direta, DF, GDNC e GDNC com *Dumb Eavesdropper* em uma rede cooperativa sem fio. Serão utilizados parâmetros para configuração do sistema, tais como o coeficiente de perdas do meio e a taxa de sigilo desejada, entre outros, o as curvas obtidas serão visualizadas em dois gráficos, um para cada nodo emissor. Será utilizada uma topologia de rede composta por dois nodos emissores, um nodo espião e um nodo destino, sendo esta uma rede sem fio. Entretanto, os conceitos podem ser aplicados em redes ligadas por cabos, PLC ou outros meios de comunicação.

2.2 Viabilidade

O projeto proposto é composto por duas partes principais: estudo das tecnologias de codificação de rede e segurança na transmissão de informações; e implementação, em linguagem Java, de um simulador de cálculo da probabilidade de falha em redes cooperativas sem fio com restrições de sigilo para análise da teoria apresentada.

O único elemento que poderia inviabilizar o projeto seria o tempo disponível para a conclusão do mesmo, existindo o prazo aproximado de dois semestres para a realização do mesmo. Entretanto, conforme demonstrado no tópico destinado ao cronograma, o tempo disponível não será um fator de risco que possa prejudicar o projeto.

2.2.1 Cronograma Preliminar

O projeto deve ser concluído durante o período de dois semestres, referentes às disciplinas TCC 1 e TCC 2. Estas duas disciplinas fornecem o quadro de tempo entre 3 de dezembro de 2012 e 14 de outubro de 2013.

Para a execução do projeto, este foi dividido em quatro etapas básicas, sendo elas: estudo das tecnologias a serem utilizadas; modelagem do *software*; implementação do *software*, incluindo testes, correções e modificações necessárias; e produção do documento final a ser entregue ao final do segundo semestre. Estas etapas ocorrem em paralelo e os prazos para cada etapa podem ser visualizados na Tabela 1:

Tabela 1: Cronograma Preliminar.

	Fev.		**Mar.**		**Abr.**		**Mai.**		**Jun.**		**Jul.**		**Ago.**		**Set.**		**Out.**	
Estudo das tecnologias																		
Modelagem do *software*																		
Implementação do *software*																		
Redação do TCC																		

2.2.2 Viabilidade financeira

Com relação aos custos financeiros, os recursos de *hardware* e *software* são de propriedade do aluno, e não será necessário o gasto com outros equipamentos ou ferramentas.

2.2.3 Viabilidade Técnica

As tecnologias estudadas possuem fontes e referências científicas suficientes para sua compreensão e análise, não sendo um fator inviabilizante do projeto. A linguagem Java é de profundo conhecimento do aluno e não será necessário treinamento ou tempo extra para aprendizagem do mesmo.

2.3 Tecnologias

O projeto é composto por três partes distintas: um computador, um sistema operacional e o aplicativo de simulação da codificação de rede.

O computador utilizado é compatível com a arquitetura IBM-PC e estará executando um sistema operacional que dá suporte as funções principais exigidas pela aplicação desenvolvida neste projeto.

O *software* desenvolvido fornece a interface de comunicação com o usuário e consiste em quatro partes principais, sendo estas:

a) Janela de visualização da topologia de rede do ambiente simulado, composto por quatro nodos dentro de um *grid*;

b) Lista de posições dos nodos e distância entre cada nodo e os demais;

c) Botão de ativação e desativação da visualização das estratégias de transmissão - codificação de rede GDNC, GDNC com *Dumb Eavesdropper* e DF;

d) Caixas de entrada para as variáveis de controle da simulação.

2.3.1 Linguagem de Programação

A linguagem de programação que será utilizada no projeto deve conter os requisitos básicos necessários para a construção e execução da aplicação. Os requisitos são listados a seguir:

a) Deve possuir uma ferramenta de desenvolvimento com uma interface gráfica de fácil manuseio;

b) Deve ser capaz de gerar código executável para multiplataforma;

c) Deve ser orientada a objeto;

A Tabela 2 resume as principais necessidades que a linguagem de programação deve atender.

Tabela 2: Tabela comparativa entre as linguagens de programação pesquisadas.

Características	Java	C++
Ferramenta de desenvolvimento com interface gráfica de fácil manuseio	Sim	Sim
Multiplataforma	Sim	Não
Orientada a objeto	Sim	Sim

Entre as principais linguagens de programação existentes hoje no mercado, optou-se por pesquisar as duas mais comumente empregadas em projetos que utilizam requisitos similares ao proposto neste trabalho.

2.3.1.1 Java

O Java é uma linguagem de programação desenvolvida para a programação orientada a objeto, que facilita a modelagem, a organização e a documentação do projeto. Seu código executável é executado em cima de uma máquina virtual, tornando-a independente do sistema operacional e sendo, assim, multiplataforma.

Esta linguagem conta com diversas ferramentas de desenvolvimento, sendo o aplicativo Eclipse [ECLIPSE, 2013] uma excelente escolha. Além de ser um programa gratuito, ele possui uma interface gráfica amigável e que facilita a construção do projeto e da interface homem-máquina. Incorpora *plug-ins* que permitem criar elementos gráficos com facilidade, entre outras funcionalidades.

Por ser uma linguagem executada sobre uma máquina virtual, a velocidade de execução do código não é tão rápida quanto uma linguagem nativa. Entretanto a linguagem Java executa em velocidade mais do que suficiente para os requisitos do projeto.

Assim, conforme pesquisado, a linguagem Java atende a todos os requisitos básicos exigidos pelo projeto.

2.3.1.2 C++

O C++ é uma extensão da linguagem de programação C. A este foram incorporadas diversas funcionalidades, tais como orientação a objeto. Entretanto, o código executável gerado pela compilação do C++ é maior e a sua execução é mais lenta do que um código escrito em C – embora ainda execute mais rapidamente comparado ao Java.

Entretanto o C++ não é uma linguagem multiplataforma, ou seja, ele necessita ser compilado em cada sistema operacional em que ele for executado, utilizando compiladores específicos para essas plataformas. Escrever um código para *cross-plataform* limita a utilização de funções e bibliotecas e aumenta consideravelmente a complexidade da aplicação.

Esta linguagem possui muitos compiladores disponíveis, alguns de baixo custo, porém com sérias limitações (por exemplo, o Dev-C++ [BLOODSHED SOFTWARE, 2013]) e outras ferramentas de desenvolvimento realmente poderosas, mas de custo muito elevado (Visual Studio [MICROSOFT VISUAL STUDIO, 2013], por exemplo). Outras ferramentas disponíveis são o Eclipse [ECLIPSE, 2013] com um *plug-in* para compilação de código C++ e o NetBeans [ORACLE, 2013], ambos com interface gráfica amigável.

Outro ponto do C++ é a difícil programabilidade dos elementos gráficos, tais como janelas, botões e painéis, dificultando a construção da aplicação e podendo impactar no tempo necessário dedicado ao desenvolvimento do *software* de simulação.

A linguagem Java é a opção tecnológica escolhida como linguagem de programação, pois atende o maior número de requisitos básicos, permitindo assim desenvolver o *software* de simulação de maneira mais eficiente.

2.3.2 Sistema Operacional

O sistema operacional a ser escolhido deve apresentar algumas características importantes para o desenvolvimento do projeto. Estes requisitos são:

a) Capacidade de executar a linguagem de programação escolhida;
b) Possuir alguma ferramenta de desenvolvimento de boa qualidade e custo baixo para a linguagem de programação escolhida;
c) Possibilidade de ser instalado em um computador com arquitetura PC.

A Tabela 3 abaixo ilustra os principais requisitos que o sistema operacional deve possuir para ser apropriado ao projeto desenvolvido.

Tabela 3: Tabela comparativa entre os sistemas operacionais pesquisados.

Características	Windows 7	Linux	Mac OS
Possui Java VM	Sim	Sim	Sim
Executa o ambiente de programação Eclipse	Sim	Sim	Sim
Compatível com a arquitetura PC	Sim	Sim	Sim
Facilidade de instalação, configuração e manuseio	Fácil	Complexo	Moderado
Baixo custo de aquisição	Não	Sim	Não

Os três sistemas operacionais estudados como opções viáveis são: Windows 7, Linux e Mac OS. Inicialmente todos os três sistemas operacionais possuem a máquina virtual Java, permitindo assim executar o código executável da linguagem escolhida.

Ainda, todos eles possuem uma versão da ferramenta que será utilizada para a programação da aplicação e todos podem ser instalados em uma arquitetura PC.

O sistema Linux possui baixo custo de aquisição, porém apresenta elevada complexidade de instalação, configuração e manuseio.

Optou-se pelo Windows 7 pela facilidade de uso que o sistema oferece, comparado as demais opções.

2.4 Parte Teórica

Para a construção da aplicação utilizada neste trabalho, será necessário utilizar técnicas de segurança para proteger as informações transmitidas entre diversos pontos da rede.

Algumas técnicas promissoras para este fim são o uso de uma infraestrutura de chaves públicas e codificação de rede. Entretanto, neste trabalho não foi utilizado a infraestrutura de chaves públicas.

2.4.1 Infraestrutura de Chaves Públicas

A infraestrutura de chaves públicas – também chamadas de *PKI* (*Public Key Infrastructure*) – não é composta apenas por equipamentos e aplicações de um sistema, mas inclui também as políticas e procedimentos que descrevem a configuração, gerenciamento, atualização e revogação dos certificados.

Uma *PKI* une chaves públicas com a identidade de usuários através do uso de certificados digitais. A união é estabelecida através de um processo de registro, onde após uma autoridade de registro (*RA*) garantir a veracidade da união, a autoridade certificadora (*CA*) emite o certificado ao usuário. Os usuários ou dispositivos podem autenticar uns aos outros através dos

certificados digitais, estabelecer chaves simétricas de sessão e criptografar e descriptografar mensagens entre ambos.

A *PKI* permite uma cadeia de confiança, onde a primeira *CA* estende a confiança para uma segunda *CA* simplesmente emitindo um certificado de confiança para ela. Isto permite que recursos seguros que confiam na primeira *CA* também confiarem na segunda *CA*.

Embora *PKI* sejam conhecidas por serem complexas, muitos dos itens responsáveis por esta complexidade podem ser significativamente reduzidos através da inclusão de elementos técnicos, tais como padronização *PKI*; ferramentas *PKI* para redes inteligentes; atributos certificadores e mecanismos de confiança automáticos [METKE; EKL, 2010].

2.4.2 Codificação de Rede Linear

O modelo convencional para transmissão de informação por uma rede de comunicação é através da técnica de roteamento, onde os nodos intermediários usam o processo de armazenar-e-enviar (*store-and-*foward) dados. Assim, cada pacote transmitido é uma cópia do pacote recebido.

O conceito principal por trás da codificação de rede é que o fluxo de dados e a robustez do sistema podem ser aprimorados ao se permitir nodos intermediários codificar diferentes fluxos de dados através de combinações algébricas de múltiplos diagramas. A Figura 2 abaixo ilustra um exemplo típico de codificação de rede.

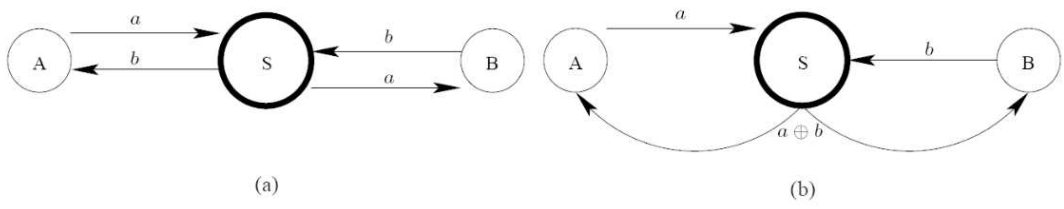

Figura 2: Exemplo de codificação de rede [PHULPIN; BARROS; LUCANI, 2011].

Para trocar as mensagens *a* e *b*, os nodos A e B utilizam o nodo S para rotear seus pacotes. No modelo tradicional, sem codificação, pode-se observar

em (a) que são necessários quatro transmissões. Com codificação, em (b) observa-se apenas três transmissões. O nodo S utiliza um método simples de *XOR* para combinar *a* e *b* e transmitir por *broadcast* o pacote, economizando assim uma transmissão.

Os nodos A e B são capazes de extrair a informação do pacote recebido, pois cada um deles possui o conhecimento do dado transmitido por ele (A conhece *a* e B conhece *b*) [PHULPIN; BARROS; LUCANI, 2011].

No método linear, um bloco de dados é visto como um vetor sobre certo campo base e permite um nodo aplicar uma transformação linear sobre um vetor antes de passá-lo adiante [LI; YEUNG; CAI, 2003].

Protocolos de codificação mais sofisticados tratam pacotes como uma coleção de símbolos de um campo finito particular e transmitem combinações lineares destes símbolos através da rede, fornecendo características básicas de códigos lineares tais como capacidade de correção de perdas e conhecimento de algoritmos de codificação e decodificação [PHULPIN; BARROS; LUCANI, 2011].

Para a codificação de rede ser usável na prática, é necessário considerar aspectos de sua segurança. A confidencialidade, integridade e disponibilidade das mensagens (os dados originais) têm que ser garantidos mesmo em caso de ataques intencionais. Para garantir a confidencialidade, deve-se prevenir o atacante de conhecer pacotes de dados independentes lineares suficientes para sua decodificação. Para garantir a integridade e disponibilidade, uma quantia suficiente de pacotes de dados deve estar disponível ao destinatário para que este possa decodificar a mensagem com sucesso.

Devem-se considerar ataques passivos e ativos na rede. Atacantes passivos apenas observam o sistema (ouvinte) enquanto atacantes ativos realizam ações específicas (modificação, remoção ou poluição de pacotes).

Se apenas pacotes codificados são enviados através da rede, um ouvinte com acesso limitado aos *links* não oferece risco a confidencialidade. Entretanto, não é possível excluir ataques mais fortes com certeza. Em particular, se um atacante é capaz de controlar um nodo na rede, ele pode observar e modificar todos os pacotes que passam por este nodo. Ataques de poluição afetam a integridade dos pacotes de dados, pois influenciam os

resultados de todas as combinações subsequentes computadas nos nodos adiantes [FRANZ; PFENNING; FISCHER, 2012].

Modelos de codificação de rede que permitem nodos intermediários detectar pacotes poluídos são principalmente baseados em criptografia. Para isso, são necessárias algumas informações secretas que podem ser usadas para verificar a validade dos pacotes recebidos. Entretanto, soluções criptográficas conhecidas não podem ser aplicadas diretamente à codificação de rede, pois os pacotes de dados são modificados por nodos na rede, invalidando assinaturas usadas para verificar a integridade e origem da mensagem. Além disso, autenticações simétricas exigiriam a troca de chaves entre o emissor e todos os nodos intermediários e destinos.

Para resolver estes problemas, *hashes* e assinaturas homomórficos são indicadas para segurança da codificação de rede contra ataques de poluição. A propriedade homomórficas destes métodos permitem nodos intermediários computarem *hashes* ou assinaturas válidas para os pacotes de dados combinados.

O modelo utilizado para a segurança da rede será o de codificação de rede devido a sua menor complexidade de implementação e gerenciamento em comparação ao *PKI*. A codificação de rede utilizará o método dinâmico generalizado, visto que a topologia da rede será dinâmica, com e sem o nodo espião conhecendo a topologia da rede (*Dumb Eavesdropper*).

2.4.3 Sigilo Teórico de Informações

Conforme Gabry [GABRY, 2012], diferente das técnicas convencionais de criptografia, o sigilo teórico de informações procura proteger redes de comunicação, podendo utilizar o auxílio de chaves de criptografia.

O modelo mais simples para estudar o sigilo em comunicações foi proposto por Wyner [WYNER, 1975], denominado canal de escuta (*wiretap channel*, ou WTC). A Figura 3 representa este canal.

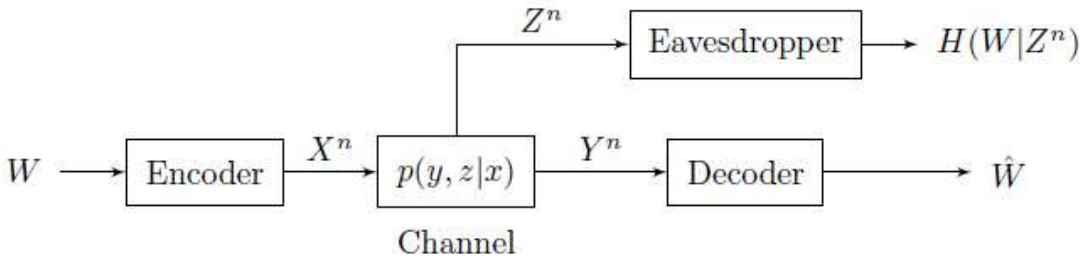

Figura 3: O canal de escuta [WYNER, 1975].

O WTC é uma rede formada por três nodos: emissor, receptor legítimo e espião. Para o canal ponto-a-ponto, a mensagem original W é escolhida uniformemente de um conjunto de mensagens $\mathcal{W}$. O codificador então escolhe (estocasticamente) uma palavra codificada $x^n(w) \in X^n$ para cada mensagem $w \in \mathcal{W}$. x^n é transmitido por um canal discreto sem memória (*discrete memoryless channel*, ou DMC) com probabilidade de transição $p_{Y\,Z|X}(\cdot\,|\,\cdot)$, gerando duas sequências de saída y^n e z^n recebidas pelo receptor legítimo e pelo espião, respectivamente. O decodificador escolhe um valor estimado $\hat{w} \in \mathcal{W}$ da mensagem para cada sequência $y^n \in Y^n$ recebida.

O modelo do canal de escuta generaliza o modelo introduzido por Wyner, onde é assumido que o canal de comunicação do emissor ao receptor e ao espião foi fisicamente degradado, isto é, o espião recebeu uma versão com ruído do canal de saída para o receptor legítimo.

Segundo Gabry [GABRY, 2012], para o canal de escuta, são de interesse duas medições de desempenho: confiança e sigilo.

A confiança é medida pela probabilidade média de erro, definida por

$$P_e^{(n)} = P\{\hat{W} \neq W\} = \frac{1}{|W|}\sum_{w \in W} P\{\hat{w} \neq w\}$$

O sigilo é medido pela taxa de equívoco $R_e^{(n)}$, definida como

$$R_e^{(n)} = \frac{1}{n}H(W|Z^n)$$

A taxa de equívoco descreve o nível de confusão do espião sobre a mensagem *W*, dado suas observações Z^n. Assim, o nível de sigilo aumenta quando a taxa de equívoco aumenta.

2.4.4 Cooperação para Sigilo em Rede sem Fio

Quando considerado um ambiente com perdas nos canais de transmissão por um método particular de distribuição estatística Rayleigh, não é possível assumir um conhecimento perfeito do canal espião. Assim, os desempenhos das estratégias de transmissão devem ser analisados de uma perspectiva de perdas [GABRY, 2012].

Um desses métodos é a cooperação. A cooperação pode aumentar o sigilo de duas maneiras: aumentando a qualidade da transmissão legítima ou diminuindo a quantidade de informações obtidas pelo espião.

2.4.5 O Canal *Relay*

O canal *relay* (ou canal de retransmissão), o modelo mais simples de uma rede cooperativa, foi proposta há mais de quatro décadas atrás por van der Meulen [VAN DER MEULEN, 1971]. Esta rede, mostrada na Figura 4, consiste de três nodos: um transmissor, um retransmissor e um receptor. O principal objetivo do *relay* é auxiliar o aumento da taxa de comunicação entre o transmissor e o receptor.

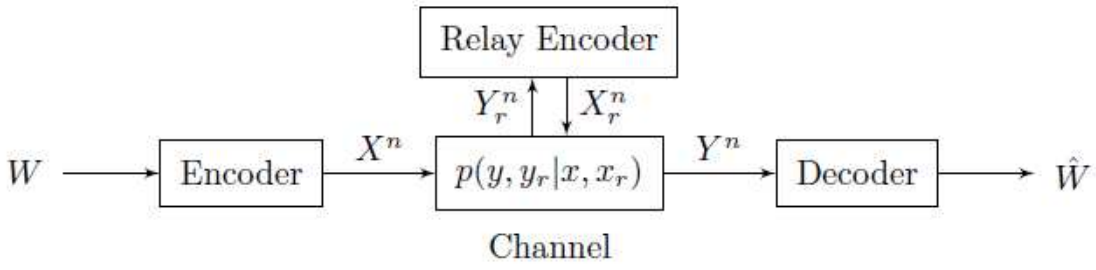

Figura 4: O canal de retransmissão [VAN DER MEULEN, 1971].

2.4.6 Retransmissão *Decode-and-Forward*

Na estratégia *Decode-and-Forward* (DF), o retransmissor decodifica a transmissão da origem e recodifica a mensagem antes de retransmitir para seu destino.

A taxa alcançada pela estratégia DF é limitada pela qualidade da ligação origem-*relay*.

2.4.7 Canal Retransmissor-Espião

O canal retransmissor-espião é representado na Figura 5. Ele foi introduzido e investigado por Lai e El Gamal [LAI; EL GAMAL, 2008]. Esta rede de quatro nodos é composta por uma origem, um retransmissor, um destino e um espião. Este modelo combina a rede cooperativa mais simples – o canal de retransmissão – e a rede mais simples com restrições de sigilo – o canal de escuta.

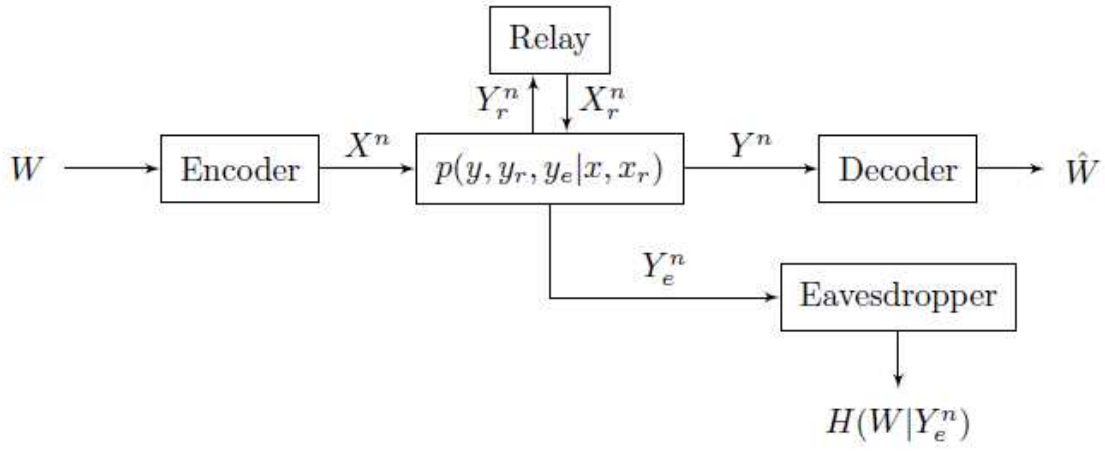

Figura 5: O canal de retransmissão-espião [LAI; EL GAMAL, 2008].

As taxas de sigilo alcançadas são derivadas de diferentes estratégias para o retransmissor. Estas estratégias caem em duas categorias. Na primeira categoria, o retransmissor tenta confundir o espião através do envio de palavras codificadas independentemente da mensagem da origem. Na outra

categoria, o retransmissor implementa uma estratégia de retransmissão, como, por exemplo, DF.

2.4.8 Codificação de Rede Dinâmica

Segundo Rebelatto et al. [REBELATTO et al., 2012], a codificação de rede, proposto inicialmente por Ahlswede [AHLSWEDE et al., 2000] para obter o máximo fluxo de informações em uma rede, tem sido aplicado recentemente em sistemas cooperativos de comunicação sem fio para melhorar o desempenho da taxa de erro binário (BER). Em um sistema de rede codificado, os retransmissores processam a informação de diferentes usuários e realizam combinações lineares dos sinais recebidos, utilizando coeficientes escolhidos dentre um campo finito GF(q).

Em um sistema cooperativo de dois usuários, cada usuário transmite uma soma binária (XOR) de sua própria mensagem e da mensagem recebida do seu parceiro – caso tenha decodificado a mensagem corretamente.

A codificação de rede dinâmica (*Dynamic-Network Coding*, ou DNC), em vez de considerar retransmissores dedicados, utiliza os usuários da rede para atuarem como retransmissores entre si. A estratégia considera um código de rede não binário fixo. Na primeira fatia de tempo, cada usuário transmite um único pacote próprio para o destino e para todos os outros usuários, os quais tentam decodificá-lo. Da segunda fatia de tempo, até a M-ésima fatia, cada usuário transmite para o destino M-1 combinações lineares não binárias de pacotes que ele pode decodificar com sucesso [REBELATTO et al., 2010].

O sistema é chamado de “dinâmico” no sentido que a codificação de rede é projetada para um bom desempenho dentre a possível ocorrência de erros nos canais entre usuários.

2.4.9 Codificação de Rede Dinâmica Generalizada

A codificação de rede dinâmica generalizada (*Generalized Dynamic-Network Coding*, ou GDNC) foi proposta inicialmente por Rebelatto et al. [REBELATTO et al., 2012]. Ele considera uma rede de acesso múltipla, onde

múltiplos nodos possuem informações independentes para transmitir a um destino comum.

Na estratégia GDNC, a transmissão é dividida em duas fases. Inicialmente ocorre a fase de transmissão, onde cada usuário transmite um número arbitrário de k_1 quadros de informações (*information frames*, ou IF) independentes próprios. Estes IFs, devido à natureza de transmissão do meio sem fio, também podem ser recebidos por outros nodos na rede. A segunda fase é denominada fase de cooperação, onde cada usuário transmite um número de k_2 quadros de paridade (*parity frames*, ou PF) para o destino.

As Figuras 6 e 7 ilustram as fases de transmissão e cooperação, respectivamente, para uma rede com M = 2 usuários e sujeitos a uma estratégia GDNC com k_1 = k_2 = 2. I_i^t representa o quadro de informação transmitido pelo nodo *i* na fração de tempo *t*. $P_i^{t\prime}$ representa o quadro de paridade transmitido pelo nodo *i* na fração de tempo *t'*.

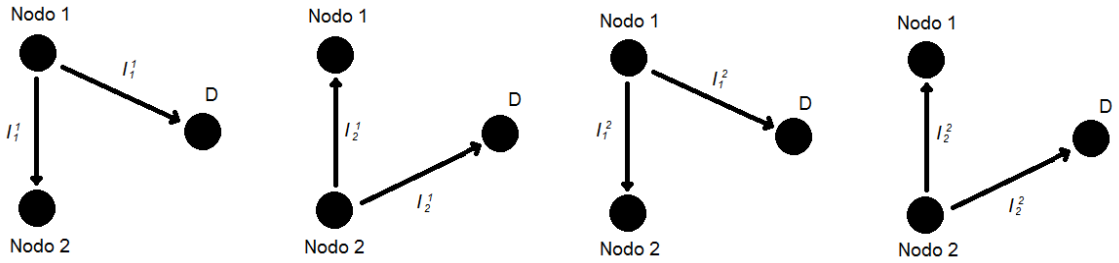

Figura 6: Fase de transmissão da estratégia GDNC, com M = 2 usuários e k_1 = 2. Autoria própria.

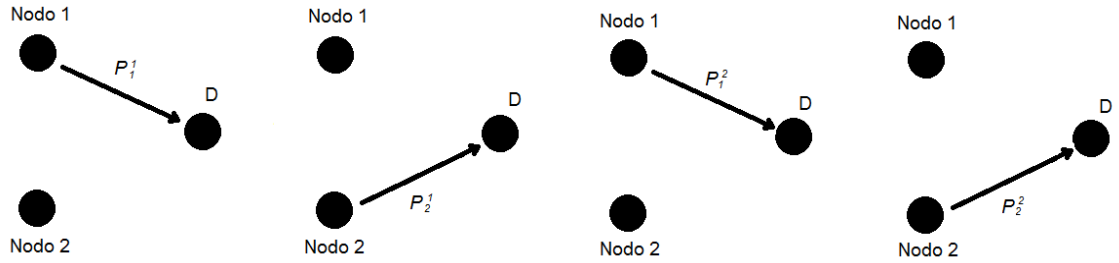

Figura 7: Fase de cooperação da estratégia GDNC, com M = 2 usuários e k_2 = 2. Autoria própria.

2.4.10 Falha de Sigilo

Para canais sem fio onde não existem restrições de sigilo, um evento de falha ocorre quando a taxa de comunicação *R* escolhida excede a capacidade do canal. Se este evento ocorrer, a comunicação confiável não é mais possível, de acordo com a definição da capacidade do canal [GABRY, 2012]. A probabilidade de falha é então definida como a probabilidade de tal evento ocorrer.

Tse e Viswanath [TSE, D.; VISWANATH, P., 2010] mostram que para um canal com perdas e relação sinal-ruído SNR γ_{sd} instantâneo entre a origem e o destino, a probabilidade de falha é definida como:

$$P_{out}(R) = Pr\{\log(1 + \gamma_{sd}) < R\},$$

sendo:

$$\gamma_{sd} = \frac{P_i |h_{sd}|^2}{d_{sd}^{\alpha} \sigma^2}$$

Se o transmissor conhecer perfeitamente γ_{sd}, ou seja, o coeficiente do canal, ele pode determinar *R* de tal maneira que nunca ocorra uma falha. Entretanto, se a propriedade do canal é desconhecida, uma falha ocorre com uma chance que depende da distribuição de probabilidade do coeficiente do canal.

2.4.11 Probabilidade de Falha de Sigilo

Em uma maneira mais realística, assume-se que os nodos legítimos possuem conhecimento completo sobre os canais legítimos, isto é, os canais entre origem, destino e retransmissores, e o transmissor possui apenas informações limitadas do estado do canal espião, ou seja, o seu SNR médio.

Isto corresponde ao modelo de perda de caminho para cenários de canais de perda Rayleigh, onde apenas a localização do espião é conhecida.

Para este modelo, usa-se a probabilidade de falha de sigilo (*secrecy outage probability*, ou SOP) como uma medida de desempenho do sistema. Similar à probabilidade de falha sem restrição de sigilo, é definido como a probabilidade da capacidade de sigilo instantâneo ser menor que uma taxa de sigilo desejada.

Para Gabry [GABRY, 2012], a probabilidade de falha de sigilo é uma medida de segurança particularmente adotada em situações onde os nodos legítimos tem conhecimento limitado sobre o canal do espião e o transmissor deve escolher uma taxa de sigilo *R* sem a informação exata do canal espião.

2.5 Considerações

Existem diversos trabalhos científicos e artigos publicados sobre a transmissão de informações em redes com fio, *wireless* e PLC, utilizando diversas estratégias de codificação. Entretanto, os resultados obtidos são apresentados em gráficos com elementos pré-determinados.

Este trabalho oferece uma maneira fácil e sem custos financeiros de analisar os resultados de diversas propostas de comunicação, proporcionando ao usuário o controle de variáveis importantes relativas à comunicação em uma rede.

3 DESENVOLVIMENTO

O *software* apresentado será capaz de demonstrar o cálculo das probabilidades de falha de sigilo em uma rede cooperativa sem fio com restrições de sigilo para diferentes estratégias de transmissão. Na interface será mostrada uma janela com a topologia da rede, duas janelas com os resultados, os botões de comando e diversos campos para entrada de valores.

3.1 Modelagem UML

Esta seção apresenta a modelagem UML do *software* da simulação. São apresentados apenas os componentes da modelagem relevantes para o planejamento e desenvolvimento da aplicação.

3.1.1 Ator Identificado

O ator identificado durante a análise do *software* é listado a seguir e ilustrado pela Figura 8.

- Usuário

Figura 8: Ator identificado. Autoria própria.

Requisitos funcionais identificados no *software* simulador

- Inicio da simulação GDNC pelo usuário. Representado pelo requisito funcional **“Usuário pode iniciar a simulação da estratégia GDNC – RF01”**;

- Término da simulação GDNC pelo usuário. Representado pelo requisito funcional **“Usuário pode parar a simulação da estratégia GDNC – RF02”**;
- Inicio da simulação GDNC Dumb pelo usuário. Representado pelo requisito funcional **“Usuário pode iniciar a simulação da estratégia GDNC Dumb – RF03”**;
- Término da simulação GDNC Dumb pelo usuário. Representado pelo requisito funcional **“Usuário pode parar a simulação da estratégia GDNC Dumb – RF04”**;
- Inicio da simulação DF pelo usuário. Representado pelo requisito funcional **“Usuário pode iniciar a simulação da estratégia DF – RF05”**;
- Término da simulação DF pelo usuário. Representado pelo requisito funcional **“Usuário pode parar a simulação da estratégia DF – RF06”**;
- Inserção do valor para o coeficiente de perdas pelo usuário. Representado pelo requisito funcional **“Usuário pode inserir o valor do coeficiente de perdas – RF07”**;
- Inserção do valor para a taxa de sigilo desejado pelo usuário. Representado pelo requisito funcional **“Usuário pode inserir o valor da taxa de sigilo desejado – RF08”**;
- Inserção do valor para o nível de sinal-ruído do nodo espião pelo usuário. Representado pelo requisito funcional **“Usuário pode inserir o valor do nível de sinal-ruído do nodo espião – RF09”**;
- Inserção do valor inicial para a faixa de sinal-ruído pelo usuário. Representado pelo requisito funcional **“Usuário pode inserir o valor inicial para a faixa de sinal-ruído – RF10”**;
- Inserção do valor final para a faixa de sinal-ruído pelo usuário. Representado pelo requisito funcional **“Usuário pode inserir o valor final para a faixa de sinal-ruído – RF11”**;
- Inserção do valor de passo para a faixa de sinal-ruído pelo usuário. Representado pelo requisito funcional **“Usuário pode inserir o valor de passo para a faixa de sinal-ruído – RF12”**;

- Inserção do valor do número de iterações pelo usuário. Representado pelo requisito funcional **“Usuário pode inserir o valor do número de iterações – RF13”**;
- Mudança na posição de um nodo na topologia de rede pelo usuário. Representado pelo requisito funcional **“Usuário pode modificar a posição de um nodo na topologia de rede – RF14”**;

3.1.2 Casos de uso identificados

- Inicio da simulação GDNC pelo usuário. Representado pelo caso de uso **“Usuário pode iniciar a simulação da estratégia GDNC – CU01”**;
- Término da simulação GDNC pelo usuário. Representado pelo caso de uso **“Usuário pode parar a simulação da estratégia GDNC – CU02”**;
- Inicio da simulação GDNC Dumb pelo usuário. Representado pelo caso de uso **“Usuário pode iniciar a simulação da estratégia GDNC Dumb – CU03”**;
- Término da simulação GDNC Dumb pelo usuário. Representado pelo caso de uso **“Usuário pode parar a simulação da estratégia GDNC Dumb – CU04”**;
- Inicio da simulação DF pelo usuário. Representado pelo caso de uso **“Usuário pode iniciar a simulação da estratégia DF – CU05”**;
- Término da simulação DF pelo usuário. Representado pelo caso de uso **“Usuário pode parar a simulação da estratégia DF – CU06”**;
- Inserção do valor para o coeficiente de perdas pelo usuário. Representado pelo caso de uso **“Usuário pode inserir o valor do coeficiente de perdas – CU07”**;
- Inserção do valor para a taxa de sigilo desejado pelo usuário. Representado pelo caso de uso **“Usuário pode inserir o valor da taxa de sigilo desejado – CU08”**;
- Inserção do valor para o nível de sinal-ruído do nodo espião pelo usuário. Representado pelo caso de uso **“Usuário pode inserir o valor do nível de sinal-ruído do nodo espião – CU09”**;

- Inserção do valor inicial para a faixa de sinal-ruído pelo usuário. Representado pelo caso de uso **“Usuário pode inserir o valor inicial para a faixa de sinal-ruído – CU10”**;
- Inserção do valor final para a faixa de sinal-ruído pelo usuário. Representado pelo caso de uso **“Usuário pode inserir o valor final para a faixa de sinal-ruído – CU11”**;
- Inserção do valor de passo para a faixa de sinal-ruído pelo usuário. Representado pelo caso de uso **“Usuário pode inserir o valor de passo para a faixa de sinal-ruído – CU12”**;
- Inserção do valor do número de iterações pelo usuário. Representado pelo caso de uso **“Usuário pode inserir o valor do número de iterações – CU13”**;
- Mudança na posição de um nodo na topologia de rede pelo usuário. Representado pelo caso de uso **“Usuário pode modificar a posição de um nodo na topologia de rede – CU14”**;

3.1.3 Diagramas de casos de uso

A Figura 9 mostra a visão geral do sistema, incluindo o ator e os casos de usos identificados no projeto. Em seguida, nas Tabelas 4 a 17 são detalhadas as especificações de cada caso de uso da aplicação.

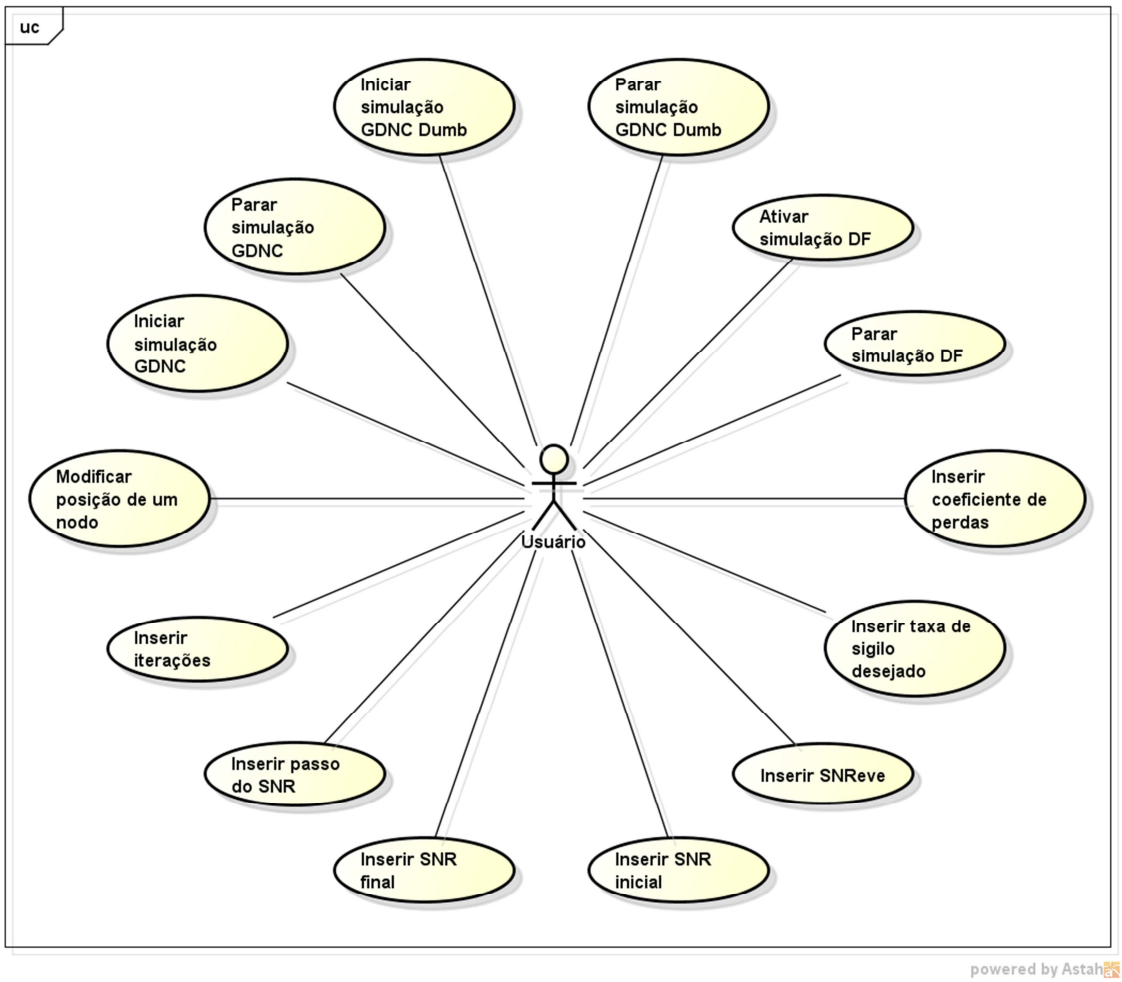

Figura 9: Diagrama dos casos de uso do sistema. Autoria própria.

3.1.4 Especificação dos casos de uso

Tabela 4: Caso de uso: iniciar simulação GDNC.

Nome do caso de uso	Iniciar simulação GDNC.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para iniciar a simulação da estratégia GDNC.
Pré-condições	A simulação da estratégia GDNC deve estar parada.
Pós-condições	A simulação da estratégia GDNC deve estar sendo executada.
Ações do ator	Ações do sistema
1. Usuário seleciona a opção de iniciar a simulação da estratégia GDNC.	
Restrições/Validações	

Tabela 5: Caso de uso: parar simulação GDNC.

Nome do caso de uso	Parar simulação GDNC.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para parar a simulação da estratégia GDNC.
Pré-condições	A simulação da estratégia GDNC deve estar sendo executada.
Pós-condições	A simulação da estratégia GDNC deve estar parada.
Ações do ator	Ações do sistema
1. Usuário seleciona a opção de parar a simulação da estratégia GDNC.	
Restrições/Validações	

Tabela 6: Caso de uso: iniciar simulação GDNC Dumb.

Nome do caso de uso	Iniciar simulação GDNC Dumb.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para iniciar a simulação da estratégia GDNC Dumb.
Pré-condições	A simulação da estratégia GDNC Dumb deve estar parada.
Pós-condições	A simulação da estratégia GDNC Dumb deve estar sendo executada.
Ações do ator	Ações do sistema
1. Usuário seleciona a opção de iniciar a simulação da estratégia GDNC Dumb.	
Restrições/Validações	

Tabela 7: Caso de uso: parar simulação GDNC Dumb.

Nome do caso de uso	Parar simulação GDNC Dumb.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para parar a simulação da estratégia GDNC Dumb.
Pré-condições	A simulação da estratégia GDNC Dumb deve estar sendo executada.
Pós-condições	A simulação da estratégia GDNC Dumb deve estar parada.
Ações do ator	Ações do sistema
1. Usuário seleciona a opção de parar a simulação da estratégia GDNC Dumb.	
Restrições/Validações	

Tabela 8: Caso de uso: iniciar simulação DF.

Nome do caso de uso	Iniciar simulação DF.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para iniciar a simulação da estratégia DF.
Pré-condições	A simulação da estratégia DF deve estar parada.
Pós-condições	A simulação da estratégia DF deve estar sendo executada.
Ações do ator	Ações do sistema
1. Usuário seleciona a opção de iniciar a simulação da estratégia DF.	
Restrições/Validações	

Tabela 9: Caso de uso: parar simulação DF.

Nome do caso de uso	Parar simulação DF.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para parar a simulação da estratégia DF.
Pré-condições	A simulação da estratégia DF deve estar sendo executada.
Pós-condições	A simulação da estratégia DF deve estar parada.
Ações do ator	Ações do sistema
1. Usuário seleciona a opção de parar a simulação da estratégia DF.	
Restrições/Validações	

Tabela 10: Caso de uso: inserir coeficiente de perdas.

Nome do caso de uso	Inserir coeficiente de perdas.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para inserir o valor do coeficiente de perdas α.
Pré-condições	
Pós-condições	O sistema deve atualizar o valor do coeficiente de perdas α.
Ações do ator	Ações do sistema
1. Usuário digita no campo "Coeficiente de Perda de Propagação α:" o novo valor desejado.	
2. Usuário pressiona a tecla ENTER para confirmar o valor.	
	3. O sistema verifica a informação conforme Regras de Negócio 1 e 2.
	4. O sistema atualiza os gráficos resultantes nas duas janelas de saída do simulador.
Restrições/Validações	**Regras de Negócio**:
	1. O campo de entrada é obrigatório.
	2. O valor deve ser positivo e maior que zero.

Tabela 11: Caso de uso: inserir taxa de sigilo desejado.

Nome do caso de uso	Inserir taxa de sigilo desejado.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para inserir o valor da taxa de sigilo desejada Rs.
Pré-condições	
Pós-condições	O sistema deve atualizar o valor da taxa de sigilo desejada Rs.
Ações do ator	Ações do sistema
1. Usuário digita no campo "Rs:" o novo valor desejado.	
2. Usuário pressiona a tecla ENTER para confirmar o valor.	
	3. O sistema verifica a informação conforme Regras de Negócio 1 e 2.
	4. O sistema atualiza os gráficos resultantes nas duas janelas de saída do simulador.
Restrições/Validações	**Regras de Negócio**:
	1. O campo de entrada é obrigatório.
	2. O valor deve ser positivo e maior que zero.

Tabela 12: Caso de uso: inserir nível de sinal-ruído do nodo espião.

Nome do caso de uso	Inserir nível de sinal-ruído do nodo espião.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para inserir o valor do nível de sinal-ruído do nodo espião SNReve.
Pré-condições	
Pós-condições	O sistema deve atualizar o valor do nível de sinal-ruído do nodo espião SNReve.
Ações do ator	Ações do sistema
1. Usuário digita no campo "SNReve:" o novo valor desejado.	
2. Usuário pressiona a tecla ENTER para confirmar o valor.	
	3. O sistema verifica a informação conforme Regras de Negócio 1 e 2.
	4. O sistema atualiza os gráficos resultantes nas duas janelas de saída do simulador.
Restrições/Validações	**Regras de Negócio**:
	1. O campo de entrada é obrigatório.
	2. O valor deve ser positivo e maior que zero.

Tabela 13: Caso de uso: inserir valor inicial da faixa de sinal-ruído.

Nome do caso de uso	Inserir valor inicial da faixa de sinal-ruído.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para inserir o valor inicial da faixa de sinal-ruído SNR.
Pré-condições	
Pós-condições	O sistema deve atualizar o valor inicial da faixa de sinal-ruído SNR.
Ações do ator	Ações do sistema
1. Usuário digita no campo "Faixa SNR Inicial:" o novo valor desejado.	
2. Usuário pressiona a tecla ENTER para confirmar o valor.	
	3. O sistema verifica a informação conforme Regras de Negócio 1 e 2.
	4. O sistema atualiza os gráficos resultantes nas duas janelas de saída do simulador.
Restrições/Validações	**Regras de Negócio**:
	1. O campo de entrada é obrigatório.
	2. O valor deve ser positivo e maior que zero.

Tabela 14: Caso de uso: inserir valor final da faixa de sinal-ruído.

Nome do caso de uso	Inserir valor final da faixa de sinal-ruído.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para inserir o valor final da faixa de sinal-ruído SNR.
Pré-condições	
Pós-condições	O sistema deve atualizar o valor final da faixa de sinal-ruído SNR.
Ações do ator	Ações do sistema
1. Usuário digita no campo "Faixa SNR Final:" o novo valor desejado.	
2. Usuário pressiona a tecla ENTER para confirmar o valor.	
	3. O sistema verifica a informação conforme Regras de Negócio 1 e 2.
	4. O sistema atualiza os gráficos resultantes nas duas janelas de saída do simulador.
Restrições/Validações	**Regras de Negócio**:
	1. O campo de entrada é obrigatório.
	2. O valor deve ser positivo e maior que zero.

Tabela 15: Caso de uso: inserir valor do passo da faixa de sinal-ruído.

Nome do caso de uso	Inserir valor do passo da faixa de sinal-ruído.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para inserir o valor do passo da faixa de sinal-ruído SNR.
Pré-condições	
Pós-condições	O sistema deve atualizar o valor inicial da faixa de sinal-ruído SNR.
Ações do ator	Ações do sistema
1. Usuário digita no campo "Faixa SNR Passo:" o novo valor desejado.	
2. Usuário pressiona a tecla ENTER para confirmar o valor.	
	3. O sistema verifica a informação conforme Regras de Negócio 1 e 2.
	4. O sistema atualiza os gráficos resultantes nas duas janelas de saída do simulador.
Restrições/Validações	**Regras de Negócio**:
	1. O campo de entrada é obrigatório.
	2. O valor deve ser positivo e maior que zero.

Tabela 16: Caso de uso: inserir valor do número de iterações.

Nome do caso de uso	Inserir valor do número de iterações.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para inserir o valor do número de iterações.
Pré-condições	
Pós-condições	O sistema deve atualizar o valor do número de iterações.
Ações do ator	Ações do sistema
1. Usuário digita no campo "Iterações (10^n):" o novo valor desejado.	
2. Usuário pressiona a tecla ENTER para confirmar o valor.	
	3. O sistema verifica a informação conforme Regras de Negócio 1 e 2.
	4. O sistema atualiza os gráficos resultantes nas duas janelas de saída do simulador.
Restrições/Validações	**Regras de Negócio**:
	1. O campo de entrada é obrigatório.
	2. O valor deve ser positivo e maior que zero.

Tabela 17: Caso de uso: modificar posição de um nodo.

Nome do caso de uso	Modificar posição de um nodo.
Caso de uso geral	
Ator principal	Usuário
Atores secundários	
Resumo	Este caso de uso descreve as etapas percorridas pelo usuário para modificar a posição de um nodo da topologia de rede.
Pré-condições	
Pós-condições	O sistema deve atualizar a posição do nodo na janela de topologia de rede.
Ações do ator	Ações do sistema
1. Usuário pressiona o botão esquerdo do *mouse* sobre um dos nodos na janela de topologia de rede.	
2. Usuário mantém o botão esquerdo do *mouse* pressionado enquanto arrasta o nodo para a nova posição desejada.	
3. Usuário solta o botão esquerdo do *mouse*.	
	4. O sistema atualiza os gráficos resultantes nas duas janelas de saída do simulador.
Restrições/Validações	

3.2 Descrição do *Software*

Ao ser inicializado, a aplicação mostrará no canto superior esquerdo a janela de topologia contendo os dois nodos emissores (S1 e S2), o nodo receptor (D) e o nodo espião (E). A Figura 10 mostra a janela de topologia.

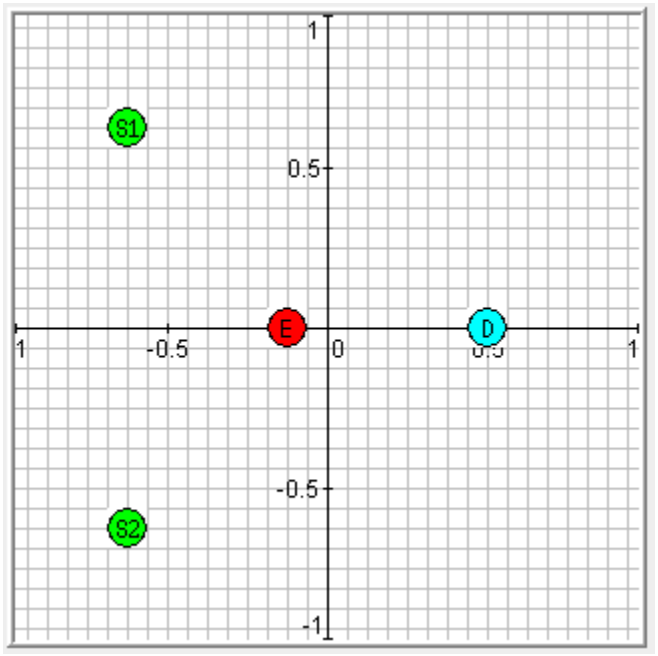

Figura 10: Janela de topologia de rede. Autoria própria.

Abaixo da janela de topologia, no canto inferior esquerdo, encontram-se informações sobre a posição de cada nodo, com as coordenadas x e y sendo números reais variando entre -1 e 1. Também são mostradas as distâncias entre cada nodo em relação aos demais. Um exemplo das coordenadas e distâncias mostradas pela aplicação pode ser observado na Figura 11.

Posição Source 1 (S1): -0,67; 0,67
Posição Source 2 (S2): -0,67; -0,67
Posição Destination (D): 0,53; 0
Posição Eavesdropper (E): -0,13; 0
Distância S1 ao D: 1,37
Distância S2 ao D: 1,37
Distância S1 ao S2: 1,33
Distância Eavesdropper ao D: 0,67
Distância Eavesdropper ao S1: 0,85
Distância Eavesdropper ao S2: 0,85

Figura 11: Exemplo de coordenadas e distâncias entre nodos. Autoria própria.

No centro superior e no canto direito superior encontram-se as janelas de resultados dos nodos emissores um (S1) e dois (S2), respectivamente. Nesta janela inicialmente é mostrado apenas a curva referente a transmissão direta (DT), sendo as demais curvas desativadas. A Figura 12 mostra as duas

janelas de saída, sendo a janela da esquerda referente ao nodo um e a janela da direita referente ao nodo dois.

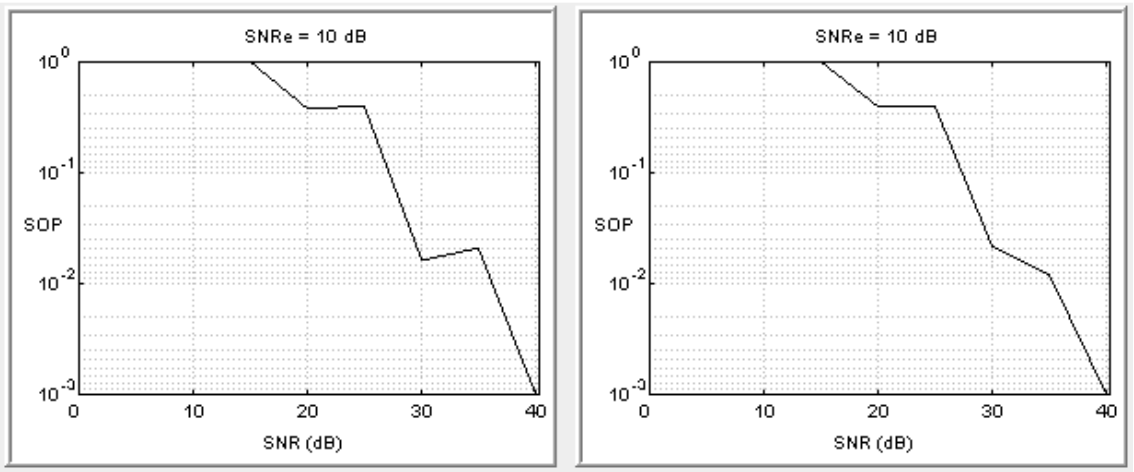

Figura 12: Janelas de saída. A janela da esquerda representa o nodo S1 e a janela de direita representa o nodo S2. Autoria própria.

Na parte central inferior encontram-se sete campos controlados pelo usuário para modificar o valor dos parâmetros utilizados nos cálculos de probabilidade. São estes parâmetros:

1) Coeficiente de perda de propagação α;
2) Taxa de sigilo desejada Rs;
3) Nível de sinal-ruído de nodo espião SNReve;
4) Faixa inicial do sinal-ruído SNR;
5) Faixa final do sinal-ruído SNR;
6) Passo da faixa de sinal-ruído SNR;
7) Número de iterações.

A Figura 13 ilustra os campos controlados pelo usuário.

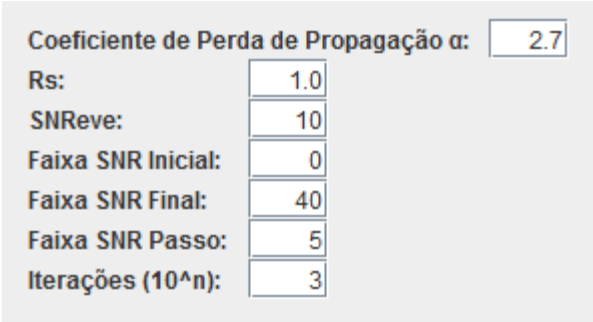

Figura 13: Campos com as variáveis controladas pelo usuário. Autoria própria.

Abaixo das janelas de resultados encontram-se três botões utilizados para ativar ou desativar as codificações de rede GDNC e GDNC com *Dumb Eavesdropper* e DF. As cores destes botões e a legenda para indicar a transmissão direta, ao lado destes botões, possuem cores distintas, utilizadas para diferenciar as curvas nos gráficos das duas janelas de resultados. A Figura 14 mostra os três botões e a legenda.

Figura 14: Exemplo dos botões e legenda. Autoria própria.

A Figura 15 mostra a janela completa da aplicação.

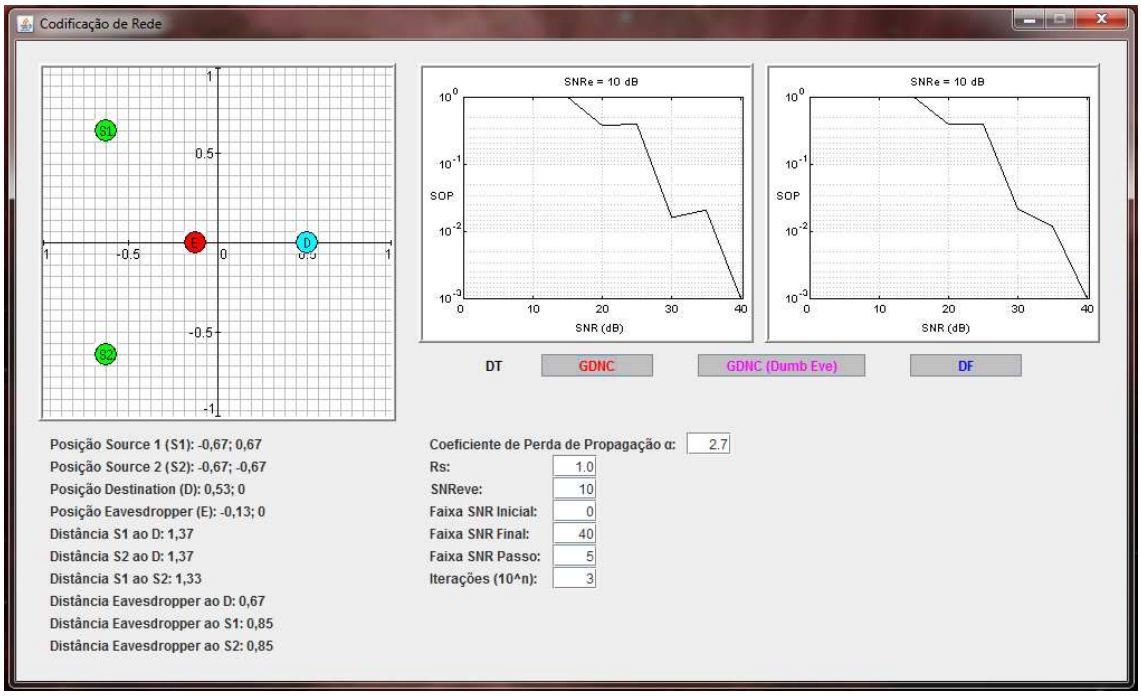

Figura 15: Exemplo completo da janela da aplicação. Autoria própria.

A janela da topologia de rede possui quatro nodos que podem ser movimentados livremente. O usuário clica e segura o botão esquerdo do *mouse* e arrasta o nodo. Cada nodo possui uma coordenada (x, y) que representa sua posição dentro do *grid*. Os valores de x e y podem variar entre -1 e 1.

Cada nodo ao ser movimentado atualiza sua coordenada e a distância entre os demais nodos. Ao término do movimento, o usuário solta o botão esquerdo do *mouse* e a aplicação recalcula as estratégias de transmissão (DT, GDNC, GDNC Dumb e DF) para os dois nodos, atualizando as duas janelas de saída de acordo com qual estratégia esteja ativada para mostrar como resultado. A estratégia de transmissão direta está sempre ativada, sendo assim, sempre visualizada nas janelas de saída.

Para os cálculos da probabilidade de falha de sigilo realizados na simulação foram utilizadas fórmulas apresentadas por Gabry [GABRY, 2012] para as estratégias de transmissão direta e *decode-and-forward*. Para as fórmulas de GDNC e GDNC com *dumb eavesdropper*, utilizaram-se as fórmulas apresentadas por Rebelatto et al. [REBELATTO et al., 2010].

3.2.1 Transmissão Direta

As fórmulas para os cálculos da probabilidade de falha de sigilo da transmissão direta são apresentadas a seguir.

$$I_{DT} = log_2(1 + \gamma_{id})$$

$$I_{DT-Eve} = log_2(1 + \gamma_{ie})$$

$$P_{oDT}(R_s) = \Pr(|I_{DT} - I_{DT-Eve}|^+ < R_s)$$

3.2.2 Transmissão GDNC

As fórmulas para os cálculos da probabilidade de falha de sigilo da transmissão GDNC são apresentadas a seguir.

$$I_a = log_2(1 + \gamma_{id}) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{id}^t)$$

$$I_b = log_2(1 + \gamma_{id}) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{id}^t) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{jd}^t)$$

$$I_{GDNC} = \begin{cases} \left(\frac{k_1}{k_1 + k_2}\right) I_a, se\ I_{ij} < R_s \\ \left(\frac{k_1}{k_1 + k_2}\right) I_b, se\ I_{ij} \geq R_s \end{cases}$$

$$I_{Eve_a} = log_2(1 + \gamma_{ie}) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{ie}^t)$$

$$I_{Eve_b} = log_2(1 + \gamma_{ie}) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{ie}^t) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{je}^t)$$

$$I_{Eve} = \begin{cases} \left(\frac{k_1}{k_1 + k_2}\right) I_{Eve_a}, se\ I_{ij} < R_s \\ \left(\frac{k_1}{k_1 + k_2}\right) I_{Eve_b}, se\ I_{ij} \geq R_s \end{cases}$$

$$P_{o\ GDNC}(R_s) = \Pr(|I_{GDNC} - I_{Eve}|^+ < R_s)$$

3.2.3 Transmissão GDNC com *Dumb Eavesdropper*

As fórmulas para os cálculos da probabilidade de falha de sigilo da transmissão GDNC com *Dumb Eavesdropper* são apresentadas a seguir.

$$I_a = log_2(1 + \gamma_{id}) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{id}^t)$$

$$I_b = log_2(1 + \gamma_{id}) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{id}^t) + \sum_{t=k_1+1}^{k_1+k_2} log_2(1 + \gamma_{jd}^t)$$

$$I_{GDNC} = \begin{cases} \left(\frac{k_1}{k_1 + k_2}\right) I_a, se\ I_{ij} < R_s \\ \left(\frac{k_1}{k_1 + k_2}\right) I_b, se\ I_{ij} \geq R_s \end{cases}$$

$$I_{Dumb-Eve} = \left(\frac{k_1}{k_1 + k_2}\right) log_2(1 + \gamma_{ie})$$

$$P_{o\ GDNC\ Dumb-Eve}(R_s) = \Pr(|I_{GDNC} - I_{Dumb-Eve}|^+ < R_s)$$

3.2.4 Transmissão DF

As fórmulas para os cálculos da probabilidade de falha de sigilo da transmissão DF são apresentadas a seguir.

$$I_{ij} = \frac{1}{2} log_2(1 + \gamma_{ij})$$

$$P_{o-ij} = \Pr(I_{ij} - R_s)$$

$$I_{DFa} = \frac{1}{2} log_2(1 + \gamma_{id})$$

$$I_{DFb} = \frac{1}{2} log_2(1 + \gamma_{id} + \gamma_{jd})$$

$$I_{DF} = \begin{cases} I_{DFa}, se\ I_{ij} < R_s \\ I_{DFb}, se\ I_{ij} \geq R_s \end{cases}$$

$$I_{DF-Evea} = \frac{1}{2} log_2(1 + \gamma_{ie})$$

$$I_{DF-Eveb} = \frac{1}{2} log_2(1 + \gamma_{ie} + \gamma_{je})$$

$$I_{DF-Eve} = \begin{cases} I_{DF-Evea}, se\ I_{ij} < R_s \\ I_{DF-Eveb}, se\ I_{ij} \geq R_s \end{cases}$$

$$P_{o\ DF}(R_s) = \Pr(|I_{DF} - I_{DF-Eve}|^+ < R_s)$$

3.3 Pontos de Casos de Uso

Para determinar os pontos de casos de uso da aplicação e obter, assim, o tamanho funcional do sistema, inicialmente são classificado os atores envolvidos em cada caso de uso, obtendo um somatório de pesos não ajustado.

O peso total dos atores do sistema (*Unadjusted Actor Weight* – UAW) é calculado pela soma dos produtos de atores de cada tipo pelo respectivo peso, onde:

- Ator simples é definido por outro sistema acessado através de uma API de programação;
- Ator médio é outro sistema interagindo através de um protocolo de comunicação, tal como TCP/IP ou FTP;
- Ator complexo é um usuário interagindo através de uma interface gráfica, seja local ou via *web*.

A Tabela 18 mostra o cálculo do somatório de peso para os atores.

Tabela 18: Somatório de pesos para os atores.

Tipo de Ator	Peso	N. de Atores	Resultado
Ator Simples	1	0	0
Ator Médio	2	0	0
Ator Complexo	3	1	3
Total de UAW			**3**

Depois de calculado o peso dos atores, o passo seguinte é calcular o peso não ajustado dos casos de uso (*Unadjusted Use Case Weight* – UUCW), onde:

- Caso de uso é simples se existe até três transações incluindo também o fluxo alternativo. Deve ser possível efetuar o caso de uso acessando menos de cinco objetos ou entidades;
- Caso de uso é médio se existe entre quatro a sete transações incluindo também o fluxo alternativo. Deve ser possível efetuar o caso de uso acessando entre cinco e dez objetos ou entidades.

- Caso de uso é complexo se existe mais de sete transações incluindo também o fluxo alternativo. Deve ser possível efetuar o caso de uso acessando mais de dez objetos ou entidades.

A Tabela 19 mostra o somatório de pesos não ajustados para os casos de uso.

Tabela 19: Somatório de pesos para os casos de uso.

Complexidade	Peso	N. de Casos de Uso	Resultado
Caso de Uso Simples	5	6	30
Caso de Uso Médio	10	8	80
Caso de Uso Complexo	15	0	0
Total de UUCW			**110**

De posse dos pesos para os atores e para os casos de uso, é possível calcular os pontos de caso de uso não ajustados (*Unadjusted Use Case Point* – UUCP), através da fórmula:

UUCP = UAW + UUCW

Obtêm-se assim o valor de UUCP igual a 113. A seguir são calculados os fatores de ajuste, procedimento similar ao adotado pela Análise de Pontos de Função, constituído por duas partes:

- Cálculo de fatores técnicos, cobrindo requisitos funcionais do sistema;
- Cálculo de fatores de ambiente, composto por requisitos não funcionais associados ao processo de desenvolvimento.

O Fator de Complexidade Técnica (*Technical Complexity Factor* – TCF) é calculado pela fórmula:

TCF = 0,6 + (0,01 x TFactor)

O TFactor é obtido pelo somatório dos níveis de influência atribuídos a cada fator (F1 a F13), multiplicados pelos pesos correspondentes. Na Tabela 20, a seguir, F1 a F13 são fatores relacionados a uma escala com valores possíveis de 0, 1, 2, 3, 4 e 5, sendo que o valor 0 significa que o fator é

irrelevante e 5 significa que o fator é essencial dentro do contexto do sistema a ser desenvolvido.

Tabela 20: Cálculo de fatores técnicos.

Fator	Fatores que contribuem para a complexidade	Peso	Valor	Total
F1	Sistemas distribuídos	2	0	0
F2	Tempo de resposta	1	3	3
F3	Eficiência para o usuário final (*online*)	1	2	2
F4	Processamento interno complexo	1	5	5
F5	Código reusável	1	2	2
F6	Facilidade de instalação	0,5	2	1
F7	Facilidade de uso (facilidade operacional)	0,5	2	1
F8	Portabilidade	2	3	6
F9	Facilidade de mudança	1	2	2
F10	Concorrência (acesso simultâneo à aplicação)	1	0	0
F11	Recursos de segurança	1	0	0
F12	Fornece acesso direto para terceiros	1	0	0
F13	Requer treinamento especial para o usuário	1	0	0
TFactor				**22**

Para este projeto, temos o valor de TCF igual a 0,82.

O Fator de Complexidade de Ambiente (*Environmental Factor* – EF) é calculado através da seguinte fórmula:

ECF = 1,4 + (-0,03 x EFactor)

O EFactor é obtido pelo somatório dos níveis de influência atribuídos a cada fator (F1 a F8), multiplicados pelo seu peso correspondente. Na Tabela 21, F1 a F8 são fatores relacionados a uma escala com valores possíveis de 0, 1, 2, 3, 4 e 5, sendo que o valor 0 significa que o fator é irrelevante e 5 significa que o fator é essencial dentro do contexto do sistema a ser desenvolvido.

Tabela 21: Cálculo de fatores ambientais.

Fator	Fatores que contribuem para a complexidade	Peso	Valor	Total
F1	Familiaridade da equipe com o processo formal de desenvolvimento adotado	1,5	4	6
F2	Colaboradores de meio período	-1	0	0
F3	Capacidade do líder do projeto em análise de requisitos e modelagem	0,5	4	2
F4	Experiência da equipe em desenvolvimento de aplicações do gênero em questão	0,5	4	2
F5	Experiência em Orientação a Objetos	1	5	5
F6	Motivação da equipe	1	4	4
F7	Dificuldades com a linguagem de programação	-1	1	-1
F8	Requisitos estáveis	2	4	8
EFactor				**26**

Para este projeto, temos o valor de ECF igual a 0,62. Com estes valores obtidos, é possível calcular o valor total do sistema em *Use Case Points* (UCP), ajustados através da seguinte fórmula:

UCP = UUCP x TCF x ECF

Aplicando a fórmula, obtemos UCP igual a 57,45. Estimando o tempo necessário do projeto com uma média de 20 horas de trabalho por UCP, obtêm-se:

Tempo estimado = UCP x 20 = 1149 horas de trabalho

Conclui-se então que, para esta aplicação ser desenvolvida, serão necessárias 1149 horas de trabalho.

3.4 Considerações

A aplicação foi desenvolvida inicialmente através da linguagem de modelagem UML. Foram criados os diagramas necessários, descrito as sequências para cada situação e projetado a interface ser-humano-máquina.

Foram então calculadas as horas necessárias para atender o projeto e verificou-se que uma quantidade de horas a serem trabalhadas de aproximadamente 600 horas, por apenas um integrante do projeto, era condizente com o projeto proposto e viável de ser executado. Entretanto, no decorrer do trabalho, o objetivo geral do mesmo foi alterado, resultando em nova modelagem e cálculo de horas a serem efetuados, resultando em aproximadamente 1150 horas previstas a serem trabalhadas. No final do projeto, esta estimativa foi superada por aproximadamente 10%, resultante de modificações sugeridas no *software* pelo professor orientador do projeto.

A programação em Java não apresentou obstáculos, sendo executada conforme planejado. As fórmulas utilizadas para o cálculo das probabilidades de falha de sigilo foram executadas através do método de Monte Carlo, com um número de iterações significativas para fornecer um cálculo de probabilidade mais preciso, impactando gravemente no tempo de resposta do sistema. Para um número superior a 10^4 iterações, o sistema falha em cumprir o requisito não funcional de tempo máximo de resposta igual a um segundo.

A janela de topologia da rede fornece uma maneira fácil e precisa e posicionar os nodos, auxiliada pela visualização das coordenadas de cada nodo, permitindo posicioná-los com distâncias entre nodos com grande precisão.

A interface que leitura dos dados inseridos pelo usuário não apresentou problemas, porém foi considerado que o usuário não tentará propositalmente inserir valores não condizentes com o tipo de valores solicitados em cada campo.

Para os botões de mostrar e ocultar as curvas, o cálculo é sempre executado para as quatro estratégias, porém somente são visualizadas as curvas solicitadas pelo usuário, para melhor comparação entre estratégias diferentes.

O tamanho da janela da aplicação é fixo, porém em trabalhos futuros pode-se permitir ao usuário aumentar ou diminuir o tamanho dos elementos, de acordo com sua preferência. Neste trabalho optou-se pelo modelo fixo de tamanho para simplificar o desenvolvimento da aplicação, considerando o número de horas previstas para o trabalho ser superior ao recomendado.

4 PROCEDIMENTOS DE TESTE E VALIDAÇÃO

A atividade de teste é o processo de executar um programa com a intenção de descobrir um erro. O objetivo é projetar testes que descubram sistematicamente diferentes classes de erros e façam-no com uma quantidade de tempo e esforço mínimos.

Os testes funcionais, também denominados de testes de caixa preta, referem-se aos testes que são realizados a partir das especificações das interfaces (entradas e saídas) do programa. Eles são usados para demonstrar que as funções do *software* são operacionais; que a entrada é adequadamente aceita e a saída é corretamente produzida e que a integridade das informações externas é mantida.

A especificação de requisitos de software contém os critérios que formam a base para uma abordagem ao teste de validação. A validação demonstra a conformidade com os requisitos e é bem sucedida quando o software funciona de maneira esperada pelo cliente.

Dentre os tipos de testes, o *particionamento de equivalência* é um método de teste que divide o domínio de entrada de um programa em classe de dados (equivalência), a partir dos quais os casos de teste podem ser derivados.

- Se o domínio de entrada exigir um intervalo, são definidas uma classe de equivalência válida e duas classes de equivalência inválidas;
- Se o domínio de entrada exigir um valor específico, são definidas uma classe de equivalência válida e duas classes de equivalência inválidas;
- Se o domínio de entrada especificar um membro de um conjunto, são definidas uma classe de equivalência válida e uma classe de equivalência inválida;
- Se o domínio de entrada for *booleano*, são definidas uma classe de equivalência válida e uma classe de equivalência inválida.

No teste de validação, é necessário garantir que todos os elementos da configuração de *software* tenham sido adequadamente desenvolvidos, estejam catalogados e possuem os detalhes necessários para apoiar a fase de manutenção do ciclo de vida do *software*.

Neste projeto, a interface com o usuário será gráfica e numérica, existindo entradas que serão feitas com valores reais, positivos e maiores que zero. As entradas referentes ao uso ou não uso das estratégias GDNC e DF serão feitas através de botões que executarão uma ação específica e apresentarão uma saída determinada.

Cada módulo do projeto será testado individualmente através da execução do mesmo e observando-se a saída resultante. O módulo será validado se a saída apresentada estiver coerente com o resultado esperado. Em conjunto, os módulos deverão apresentar os resultados esperados sem interferência, ou seja, a execução de um módulo não deve interferir com o resultado de outro módulo.

Para a aceitação do projeto, cada módulo deve apresentar um tempo de resposta não superior a um segundo e deve cumprir sua função esperada. O projeto também deve obedecer aos requisitos não funcionais especificados.

A Tabela 22 contém a descrição dos testes de caixa preta para cada Caso de Uso do sistema.

Tabela 22: Descrição dos testes de caixa preta para cada Caso de Uso.

Caso de Uso	Descrição do teste de caixa preta
CU01	A simulação da estratégia GDNC deve ser iniciada, atualizando os resultados nas janelas de resultados, mostrando a curva resultante da estratégia GDNC. O botão “GDNC” deve ser modificado para parar a simulação GDNC se pressionado novamente.
CU02	A simulação da estratégia GDNC deve ser finalizada, atualizando os resultados nas janelas de resultados, removendo a curva resultante da estratégia GDNC. O botão “GDNC” deve ser modificado para iniciar a simulação GDNC se pressionado novamente.
CU03	A simulação da estratégia GDNC Dumb deve ser iniciada, atualizando os resultados nas janelas de resultados, mostrando a curva resultante da estratégia GDNC Dumb. O botão “GDNC Dumb” deve ser modificado para parar a simulação GDNC Dumb se pressionado novamente.
CU04	A simulação da estratégia GDNC Dumb deve ser finalizada, atualizando os resultados nas janelas de resultados, removendo a curva resultante da estratégia GDNC Dumb. O botão “GDNC Dumb” deve ser modificado para iniciar a simulação GDNC Dumb se pressionado novamente.
CU05	A simulação da estratégia DF deve ser iniciada, atualizando os resultados nas janelas de resultados, mostrando a curva resultante da estratégia DF. O botão “DF” deve ser modificado para parar a simulação DF se pressionado novamente.
CU06	A simulação da estratégia DF deve ser finalizada, atualizando os resultados nas janelas de resultados, removendo a curva resultante da estratégia DF. O botão “DF” deve ser modificado para iniciar a simulação DF se pressionado novamente.
CU07	O valor da perda de propagação e os gráficos resultantes nas janelas de resultados devem ser atualizados.
CU08	O valor da taxa de sigilo desejada e os gráficos resultantes nas janelas de resultados devem ser atualizados.
CU09	O valor do nível de sinal-ruído do nodo espião e os gráficos resultantes nas janelas de resultados devem ser atualizados.
CU10	O valor inicial da faixa de sinal-ruído e os gráficos resultantes nas janelas de resultados devem ser atualizados.
CU11	O valor final da faixa de sinal-ruído e os gráficos resultantes nas janelas de resultados devem ser atualizados.
CU12	O valor de passo da faixa de sinal-ruído e os gráficos resultantes nas janelas de resultados devem ser atualizados.

CU13	O valor do número de iterações e os gráficos resultantes nas janelas de resultados devem ser atualizados.
CU14	A posição do nodo deve ser atualizada, mostrando as novas distâncias entre cada nodo, e os gráficos resultantes nas janelas de resultados devem ser atualizados.

4.1 Considerações

Os Casos de Uso foram testados individualmente e analisados os resultados. Cada teste de caixa preto verificou, de maneira satisfatória, o plano de teste proposto para cada módulo. Não existiu interferência e todos os requisitos funcionais e não funcionais foram avaliados.

Entretanto, para um número de iterações superiores a 10^4, o requisito não funcional que especifica um tempo de resposta do sistema inferior a um segundo, não é satisfeito.

5 GESTÃO

Este capítulo é destinado ao gerenciamento do projeto, incluindo o cronograma e a análise de risco.

5.1 Cronograma

O cronograma do projeto visa organizar as atividades e o tempo gasto para sua realização. O desenvolvimento do projeto foi dividido em seis tarefas, sendo elas: levantamento bibliográfico, levantamento dos requisitos, diagramas de modelagem, desenvolvimento da aplicação, testes e validação e plano de projeto.

O levantamento bibliográfico contém a fundamentação teórica do trabalho, com referências às pesquisas e aos conhecimentos já construídos e publicados, situando a evolução do assunto e dando sustentação ao tema trabalhado. Também fazem parte a análise do estado da arte do problema abordado, com uma sistematização de ideias, fundamentos, conceitos e proposições de diversos autores, apresentados de forma lógica, encadeada e descritiva, embasando a formulação do problema e sua justificativa.

O levantamento dos requisitos engloba os requisitos funcionais e não funcionais. Os requisitos funcionais descrevem o comportamento do sistema, suas ações para cada entrada, mostrando o que deve ser feito pelo sistema. Os requisitos não funcionais expressam como o sistema será feito, relacionando-se com padrões de qualidade, confiabilidade, desempenho, robustez, entre outros, apresentando restrições e especificações de uso para os requisitos funcionais.

Os diagramas de modelagem contêm os diagramas de casos de uso, diagramas de classe, diagramas de objeto e dicionário de informações do *software* a ser desenvolvido.

O desenvolvimento da aplicação é a construção da aplicação. Utilizando a modelagem construída anteriormente, engloba a programação da interface gráfica e a implementação dos algoritmos de codificação de rede.

Os testes e validações indicam como os diversos módulos do projeto serão testados e validados individualmente e em conjunto. Serão descritos os

critérios de aceitação do projeto em termos de desempenho e do cumprimento de seus respectivos aspectos funcionais, assim como os testes de caixa preta para cada caso de uso do sistema.

O plano de projeto constitui o documento do trabalho e reúne a introdução, objetivos gerais e específicos, metodologia, recursos de *hardware* e *software*, viabilidade, pontos de caso de uso, cronograma, análise de riscos, conclusões e referências bibliográficas.

Para a determinação do cronograma aplicado no desenvolvimento do *software* serão utilizados os métodos PERT (*Program Evaluation and Review Technique*, ou método de avaliação e revisão de programa) e CPM (*Critical Path Method*, ou método do caminho crítico). A rede PERT-CPM é a representação gráfica do cronograma, no qual se apresenta a sequência lógica do planejamento com as interdependências das tarefas, tendo por finalidade a construção do projeto.

No diagrama estão representados:

- Todas as tarefas do início ao fim do projeto;
- A sincronização das atividades (tarefas);
- As dependências entre as atividades;
- O caminho crítico (sequência de atividades que determinam a duração do projeto);
- Uma estimativa de duração das atividades e os limites de tempo para elas.

A notação utilizada na rede PERT-COM é vista na Figura 16. O tempo necessário para que o evento seja atingido sem atrasos é denominado *Cedo* do evento. O tempo limite para realização do evento sem que atrase o projeto é denominado *Tarde* do evento. Com estes dois valores é possível calcular a *Folga* de cada evento, o qual representa a diferença entre o *Tarde* e o *Cedo* de cada evento. O caminho crítico será representado por setas na cor vermelha.

Figura 16: Notação da Rede PERT-CPM.

A Tabela 23 mostra as identificações das atividades do projeto, as durações das tarefas, em semanas, e as dependências entre elas. Uma atividade é dependente de outra se para ela ser iniciada é necessário que a outra tenha sido concluída. Duas atividades são paralelas se elas podem ser executadas simultaneamente. As atividades Levantamento Bibliográfico e Plano de Projeto estarão ocorrendo durante todo o desenvolvimento do projeto, não sendo, assim, representadas no diagrama PERT-CPM.

Tabela 23: Identificação e dependências das atividades.

Código	Descrição	Duração	Dependência
A	Levantamento dos requisitos	2	
B	Diagramas de modelagem	4	A
C	Desenvolvimento da aplicação	20	B
D	Testes e validação	20	B

A Figura 17 mostra o diagrama PERT-CPM para o trabalho proposto, contendo o Cedo, o Tarde, a Folga e o Caminho Crítico (em vermelho).

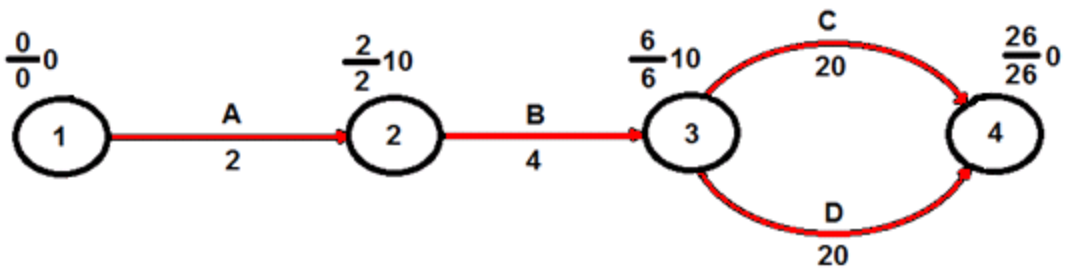

Figura 17: Diagrama PERT-CPM do projeto.

5.2 Análise de Risco

Um risco é qualquer variável do projeto, que dentro de sua distribuição normal de valores possíveis, pode assumir um valor que afete o projeto.

A probabilidade de o risco ocorrer é dividida em cinco partes: a) alta; b) médio-alta; c) média; d) médio-baixa; e) baixa. Uma probabilidade alta ocorre em quase todas as circunstâncias. A médio-alta vai ocorrer na maioria das circunstâncias. A média deve ocorrer em algum momento. A médio-baixa pode ocorrer em algum momento. E a baixa ocorre somente em circunstâncias excepcionais.

O fator de impacto é dividido em cinco partes, igualmente à probabilidade de risco, variando de alta até baixa. Um impacto alto significa um efeito sobre o projeto que o torna inviável. Um impacto baixo possui pouco ou nenhum efeito relevante ao projeto.

A probabilidade e o impacto definem o índice de risco de uma ameaça. Uma probabilidade e um impacto alto indicam um alto risco ao projeto, enquanto probabilidades e impactos baixos mostram baixo risco. O grau de risco pode ser observado na Tabela 24. O símbolo **X** indica alto risco; o símbolo ***** indica médio risco; e o símbolo **O** indica baixo risco.

Tabela 24: Classificação do grau de risco baseado na probabilidade e impacto. O símbolo **O** significa baixo risco; o símbolo ***** significa médio risco; e o símbolo **X** significa alto risco.

Probabilidade					
Alta	**O**	*****	*****	**X**	**X**
Médio-alta	**O**	**O**	*****	**X**	**X**
Média	**O**	**O**	*****	**X**	**X**
Médio-baixa	**O**	**O**	**O**	*****	**X**
Baixa	**O**	**O**	**O**	**O**	*****
	Baixo	Médio-baixo	Médio	Média-alto	Alto
	Impacto				

1ª. Etapa: Identificação do Risco

Denominação do risco: Não cumprimento dos prazos estabelecidos pelo cronograma.	Nº. Identificação **01**
Descrição do risco: O aluno pode não atender aos prazos estipulados e assim comprometer o andamento do projeto.	

2ª. Etapa: Avaliação do Risco

Impacto: ▪5(alto) □4(média/alto) □3(médio) □2(médio/baixo) □1(baixo) **Explique**: O não cumprimento dos prazos das etapas intermediárias irá interferir no prazo final do projeto podendo ocasionar a inviabilização do mesmo.
Probabilidade: □5(alta) □4(média/alta) □3(média) ▪2(média/baixa) □1(baixa) **Explique**: O professor orientador irá acompanhar o andamento das tarefas de forma a auxiliar o aluno para uma melhor condução das mesmas.

3ª. Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): **Prevenir**: Deve haver comprometimento do aluno com o projeto. O orientador irá auxiliar o aluno com dúvidas e realizar cobranças parciais sobre o andamento do projeto, reduzindo assim a chance de atraso das etapas intermediárias. **Transferir**: - **Mitigar**: Conversar com o orientador e professores envolvidos com o projeto para encontrar uma solução prática e satisfatória evitando assim maiores danos ao projeto. **Aceitar**: -

Impacto reavaliado: 1(baixo)	**Probabilidade reavaliada**: 1(baixa)
Elaborado por: **Fábio César Schuartz**	Data: **30/04/2013**

1ª. Etapa: Identificação do Risco

Denominação do risco: Subestimação do prazo para execução de tarefas.	Nº. Identificação **02**
Descrição do risco: Determinação de prazo insuficiente para a execução de tarefas.	

2ª. Etapa: Avaliação do Risco

Impacto: □5(alto) ▪4(média/alto) □3(médio) □2(médio/baixo) □1(baixo) **Explique**: Uma alteração muito grande no prazo de execução de uma determinada tarefa pode comprometer o prazo final.
Probabilidade: □5(alta) □4(média/alta) ▪3(média) □2(média/baixa) □1 (baixa) **Explique**: A inexperiência do aluno com determinadas tarefas poderá comprometer o prazo determinado.

3ª. Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): **Prevenir**: Superestimar o tempo programado para a realização de cada tarefa. **Transferir**: - **Mitigar**: Disponibilizar um tempo maior para a execução da tarefa com o objetivo de minimizar o atraso. **Aceitar**: -

Impacto reavaliado: 2(médio/baixo)	**Probabilidade reavaliada**: 2(média/baixa)
Elaborado por: **Fábio César Schuartz**	Data: **30/04/2013**

1ª. Etapa: Identificação do Risco

Denominação do risco: Subestimação da complexidade ou dificuldade na implementação do software – conhecimento insuficiente em determinado assunto pelo aluno.	Nº. Identificação **03**
Descrição do risco: Conhecimento insuficiente do aluno inviabilizará a conclusão do projeto.	

2ª. Etapa: Avaliação do Risco

Impacto: ▪5(alto) □4(média/alto) □3(médio) □2(médio/baixo) □1(baixo) **Explique**: Sem os conhecimentos necessários, o projeto poderá não ser concluído e o artefato não poderá ser construído.
Probabilidade: □5(alta) □4(média/alta) ▪3(média) □2(média/baixa) □1 (baixa) **Explique**: O aluno possui uma base da tecnologia empregada e dos conceitos envolvidos no projeto.

3ª. Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): **Prevenir**: Procurar uma tecnologia que seja compatível com o nível de conhecimento para a construção do artefato, satisfazendo os requisitos determinados pelo projeto. **Transferir**: - **Mitigar**: Estender a carga de estudo em cima do projeto, procurar tecnologias mais simples, caso o aluno esteja no estágio inicial do projeto. Procurar suporte emergencial com algum professor. **Aceitar**: -

Impacto reavaliado: 2(médio/baixo)	**Probabilidade reavaliada**: 2(média/baixa)
Elaborado por: **Fábio César Schuartz**	Data: **30/04/2013**

1ª. Etapa: Identificação do Risco

Denominação do risco: Falta de tempo para a conclusão do projeto – término do prazo.	Nº. Identificação **04**
Descrição do risco: Tempo não pode ser adquirido, portanto o prazo final não pode ser estendido.	

2ª. Etapa: Avaliação do Risco

Impacto: ▪5(alto) □4(média/alto) □3(médio) □2(médio/baixo) □1(baixo) **Explique**: Caso o tempo disponível não seja suficiente, o artefato não poderá ser completado, invalidando o projeto.
Probabilidade: □5(alta) □4(média/alta) ▪3(média) □2(média/baixa) □1 (baixa) **Explique**: É feita uma avaliação e planejamento prévios, mas imprevistos podem ocorrer durante o tempo estimado para cada tarefa.

3ª. Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade): **Prevenir**: Durante o planejamento, estimar os prazos para cada tarefa e considerar folgas entre as mesmas, para que se algum imprevisto ocorra, o aluno tenha tempo hábil para corrigir os problemas, sem atrasar o cronograma. **Transferir**: - **Mitigar**: Diminuir o escopo do trabalho e reduzir os requisitos finais do artefato. **Aceitar**: -

Impacto reavaliado: 3(média)	**Probabilidade reavaliada**: 1(baixa)
Elaborado por: **Fábio César Schuartz**	Data: **30/04/2013**

5.3 Considerações

Embora o objetivo geral do projeto tenha sido alterado durante a execução do trabalho, foi possível completá-lo sem grandes atrasos ou problemas, não impactando no cronograma planejado. Não existiram gastos monetários envolvidos na construção do projeto, considerando que o artefato

final é uma aplicação de simulação, envolvendo apenas programação em linguagem Java.

As horas trabalhadas estimadas eram de 1149 horas, mas foi necessário estender este total, resultando em aproximadamente 1400 horas trabalhadas, ao decorrer de dois semestres. Isto se deve ao fato do foco da aplicação ter sido modificado durante o decorrer da disciplina de TCC II, resultando em refazer a modelagem UML, diagramas e parte do código já desenvolvido. Esta mudança, porém, não impediu a conclusão do projeto dentro do prazo máximo permitido pela disciplina.

6 TRABALHOS FUTUROS

Este projeto analisa a probabilidade de falha de quatro estratégias de comunicação entre dois nodos emissores e um nodo destino, existindo um nodo espião passivo, dentro de uma rede cooperativa sem fio, com restrições de sigilo.

Embora a rede utilizada seja sem fio, os conceitos são válidos para outras topologias de rede.

Em trabalhos futuros, podem ser acrescentadas novas estratégias para a transmissão de informações, assim como o aumento do número de nodos emissores e retransmissores, tornando a rede mais complexa. Outras variáveis podem ser adicionadas, tais como a potência de transmissão de cada nodo e o ruído térmico do meio, fornecendo assim uma plataforma mais completa.

Outras implementações podem calcular mais opções de saída, além da probabilidade de falha de sigilo. Por exemplo, pode-se calcular o *secure throughput*, que é outra medida de desempenho e permite obter algumas informações sobre a taxa que podemos transmitir acima da taxa de sigilo desejada R_s. Outra medida de desempenho é a *secrecy capacity*, que é a taxa de transmissão máxima no qual o espião não consegue decodificar informação alguma.

7 CONSIDERAÇÕES FINAIS

O objetivo geral do projeto é o desenvolvimento de uma aplicação capaz de mostrar visualmente, em dois gráficos distintos, a probabilidade de falha de sigilo na transmissão entre duas fontes e um destino, considerando um agente de escuta passiva.

Para este fim, foram utilizados quatro estratégias de transmissão. A primeira forma de transmissão é a transmissão direta, onde informações são enviadas diretamente de um emissor ao destino, sem nenhuma alteração no caminho, por parte de retransmissores. A segunda estratégia é a *decode-and-forward*, onde o retransmissor irá decodificar a mensagem recebida; recodificar e enviar ao próximo nodo. A terceira e quarta forma de transmissão utilizam a codificação de rede conhecida como *Generalized Dynamic Netwok Coding*, sendo que uma delas considera que o agente espião não possui informações sobre a topologia da rede, sendo então denominado de *Dumb Eavesdropper*.

A rede utiliza uma topologia cooperativa sem fio, onde a posição dos dois nodos emissores, do nodo destino e do nodo espião pode ser alterada, dentro de uma escala normalizada com coordenadas x e y variando entre -1 e 1. A distância entre cada nodo afeta a probabilidade de falha de sigilo, pois cada nodo possui uma potência de transmissão e o meio possui um coeficiente de perdas. Quanto mais afastado o emissor do receptor, maior será a probabilidade de falha de sigilo.

Para realizar a simulação foram utilizadas fórmulas propostas por Gabry [GABRY, 2012], para as estratégias de transmissão direta e *decode-and-forward*, e por Rebelatto et al. [REBELATTO et al., 2012], para as estratégias de GDNC e GDNC com *dumb eavesdropper*. A aplicação foi desenvolvida em linguagem Java e utilizou-se da biblioteca *Java Statistical Class* (JSC) para gerar o modelo de ruído Gaussiano e o canal de perdas de uma distribuição Rayleigh entre os canais. As curvas resultantes destas fórmulas são visualizadas em duas janelas de saída distintas, um gráfico para cada nodo emissor.

As coordenadas dos nodos e as distâncias entre eles são mostradas na aplicação, fornecendo uma ferramenta de maior precisão para o

posicionamento dos mesmos. O usuário pode entrar com valores para o coeficiente de perdas do meio, a taxa de sigilo desejada e a relação sinal-ruído do nodo espião, que afetam a probabilidade de falha de sigilo durante as transmissões. As variáveis de SNR inicial, final e passo, assim como o número de iterações, controlam a precisão dos gráficos e o tempo de resposta do sistema. Para cada ponto da reta é calculada a probabilidade de falha de sigilo para as quatro estratégias, tomando-se a média resultante do cálculo por um número de iterações solicitadas pelo usuário. Um número maior de iterações fornece resultados mais precisos, porém consome mais tempo para ser executado.

Por último, o usuário pode escolher qual curva resultante é visualizada nos gráficos, com exceção da estratégia de transmissão direta, que sempre será visualizada. Cada curva possui uma cor diferente, facilitando a identificação da mesma.

Para a aplicação ser satisfatória, duas condições principais devem ser satisfeitas: os gráficos resultantes devem estar de acordo com a teoria apresentada; e o tempo de resposta do sistema deve estar dentro de um padrão aceitável, neste caso, menos que um segundo.

A construção do *software* foi dividida em cinco partes, sendo cada parte testada separadamente.

A primeira parte desenvolvida foi a janela de topologia da rede, onde os quatro nodos são visualizados e suas posições podem ser alteradas. Apesar de simples em termos de complexidade, o uso de elementos gráficos e interação com o *mouse* exigiram considerável parte do tempo gasto no desenvolvimento da aplicação. Como resultado obteve-se uma interface intuitiva e de fácil uso, permitindo o usuário posicionar cada nodo dentro da janela de topologia de maneira independente dos demais. Inicialmente, ao arrastar um nodo, as janelas de resultados eram atualizadas em tempo real, mas testes exaustivos mostraram que para um número muito elevado de iterações, não era possível realizar o posicionamento dos nodos de forma satisfatória. Assim, as janelas de resultados são atualizadas somente após o posicionamento final do nodo, ou seja, após o usuário soltar o botão do *mouse*. As coordenadas e distâncias entre nodos, entretanto, são atualizadas em tempo real, permitindo assim o

usuário controlar com maior precisão a posição de um nodo e a distância entre dois ou mais nodos.

A segunda parte desenvolvida foi a visualização das coordenadas dos nodos e a distância entre cada um dos nodos. O *grid* da topologia de rede possui coordenada que variam entre -1 e 1. Esta parte foi rapidamente desenvolvida e testada, não exigindo tempo além do previsto para sua execução.

A terceira parte foi a construção de campos para as variáveis de entrada do sistema. Inicialmente projetada cinco entradas – coeficiente de perdas, taxa de sigilo desejada, SNR do *eavesdropper*, SNR inicial, final e passo – e utilizado um número de iterações fixo, igual a 1000. Entretanto, para uma precisão maior do método de Monte Carlo, foi decidido permitir o usuário escolher o número de iterações utilizadas para os cálculos estatísticos, resultando em seis campos para as variáveis de entrada. A construção e teste destes campos foram simples e rapidamente desenvolvidos, não exigindo tempo além do previsto.

A quarta parte é referente aos cálculos das probabilidades de falha de sigilo para as quatro estratégias escolhidas. Os cálculos são realizados através de fórmulas estabelecidas na literatura e que utilizam o ruído Gaussiano, distribuição de Rayleigh e o método de Monte Carlo para obter o valor de probabilidade para cada ponto da curva, de acordo com o número de passos escolhido. Para validar esta etapa do projeto, os valores obtidos para cada estratégia foram comparados com gráficos existentes na literatura, procurando observar o formato das curvas e comparando quais estratégias resultam melhores resultados, conforme previamente esperado. Esta etapa apresentou dificuldades em termos de desenvolvimento, pois o tempo de resposta do sistema depende grandemente do número de iterações. Para obter um tempo de resposta abaixo de um segundo, o número de iterações não pode ser superior a 10.000, para o sistema computacional utilizado. Este tempo varia de acordo com a velocidade do processador do equipamento utilizado. Para 100.000 iterações o sistema tem um tempo de resposta de aproximadamente dez segundos, o que ainda é aceitável, porém não desejado. Os resultados obtidos, embora dentro do esperado – e, portanto validados – apresentaram uma tendência comum entre as quatro estratégias. Esta tendência é

visualizada nas curvas resultantes, sugerindo que os valores de Rayleigh possam estar utilizando uma mesma semente para a geração dos números aleatórios. Entretanto, nada consta na documentação da biblioteca do JSC sobre a geração da distribuição de Rayleigh.

A quinta e última etapa da construção da aplicação foi as duas janelas de resultados, com os botões de ativar e desativar a visualização de três estratégias. Cada janela de resultado apresenta gráficos referentes a um nodo emissor único (S1 e S2). A modificação no posicionamento de um nodo qualquer produz resultados diferentes para cada nodo emissor, pois suas posições e distancias são diferentes em relação aos nodos destino e espião. Embora simples em complexidade, a utilização de elementos gráficos para visualização das curvas exigiu tempo acima do planejado. Com os gráficos mostrados nas janelas de resultados, podem-se comprovar visualmente os cálculos das fórmulas da etapa anterior. Os botões de controle de visualização ativam ou desativam a visualização das estratégias DF, GDNC d GDNC com *Dumb Eavesdropper* e servem como legenda para as cores de cada curva.

O sistema como um todo cumpre com os dois objetivos principais: realiza com esperada precisão os cálculos das estratégias de transmissão e apresenta os resultados com tempo de resposta satisfatório, para um número de iterações menor ou igual a 10^4. O projeto foi desenvolvido inicialmente utilizando a linguagem de modelagem UML. Depois de criados os diagramas, iniciou-se a construção da aplicação e teste de cada módulo. Finalizado o desenvolvimento, compararam-se os resultados obtidos graficamente com gráficos apresentados na literatura, validando a aplicação proposta neste projeto.

O cronograma cobre as atividades planejadas e precisou ser estendido para acomodar a necessidade no decorrer do projeto. O desenvolvimento do projeto seguiu o modelo em espiral, ocorrendo em forma modular, permitindo a execução de algumas etapas em paralelo. Entretanto, o tempo total de trabalho em horas excedeu o previsto de 1149 horas em aproximadamente 10%, totalizando aproximadamente 1300 horas de trabalho. Terminado o desenvolvimento, o plano de testes foi executado com sucesso e contemplou todos os requisitos funcionais e não funcionais previstos.

Este projeto exigiu o conhecimento de diversas áreas de conhecimento do curso de Engenharia de Computação, mais especificamente a de Programação I e II, Redes I e II, Cálculo e Estatística. Outros conceitos como Redes sem Fio, Codificação de Redes e Redes Cooperativas foram de grande importância para o desenvolvimento do trabalho.

As maiores dificuldades encontradas são relacionadas à complexidade da codificação de rede em si. Esta é uma técnica proposta recentemente (menos de 15 anos desde sua apresentação) e que exige um nível avançado de estudo, considerando como ponto de partida o conhecimento adquirido durante o curso de graduação e diversos trabalhos apresentados na literatura. Os detalhes operacionais do *software* requerem um conhecimento da linguagem de programação escolhida, o que não apresentou um empecilho durante o desenvolvimento do projeto. Não existiu custo financeiro para a execução do projeto e a análise de riscos do projeto garantiu que o mesmo fosse completado sem grandes problemas.

REFERÊNCIAS BIBLIOGRÁFICAS

AGRAWAL, S.; BONEH, D. Homomorphic MACs: MAC-based integrity for network coding. **Applied Cryptography and Network Security**, 2009, pp. 292-305.

AHLSWEDE, R.; CAI, N.; LI, S.-Y. R.,YEUNG, R. W. Network information flow. **IEEE Transactions on Information Theory**, v. 46, n. 4, p. 1204-1216, 2000.

BONEH, D.; FREEMAN, D. M.; Katz, J.; Waters, B. **Signing a linear subspace:** signature schemes for network coding. Public Key Criptography, 2009, pp. 68-87.

Dev-C++. **Bloodshed Software**. Disponível em: <http://www.bloodshed.net/devcpp.html>. Acesso em: 06 mar. 2013.

Eclipse. **The Eclipse Foundation**. Disponível em: <http://www.eclipse.org/>. Acesso em: 06 mar. 2013.

FAN, Y.; JIANG, Y.; SHEN, X. An efficient privacy-preserving scheme against traffic analysis in network coding. **Porc. IEEE INFOCOM'09**, pp. 2213-2221, Abr. 2009.

FRANZ, E.; PFENNING, S.; FISCHER, A. Efficiency of secure network coding schemes. **IFIP International Federation for Information Processing**, p. 145-159, 2012.

GABRY, Frederic. **Cooperation for secrecy in wireless networks**. 2012. 139 f. Tese (Doutorado em Telecomunicações) – Communication Theory Laboratory, School of Electrical Engineering, Stockholm, Sweden, 2012.

GALLI, S.; SCAGLIONE, A.; WANG, Z. For the grid and through the grid: the role of power line communications in the smart grid. **Proceedings of the IEEE**, v. 99, n. 6, p. 998-1027, jun. 2011.

GKANTSIDIS, C.; RODRIGUEZ, P. R. Cooperative security for network coding file distribution. **Proc. IEEE Int. Conf. on Computer Communications**, 2006.

HO, T.; LEONG, B.; KOETTER, R.; MEDARD, M.; EFFROS, M.; KARGER, D. Byzantine modification detection in multicast networks using randomized network coding. **Proc. of ISIT**, 2004,

HUURDEMAN, A. A. **Electrical telegraphy**. The worldwide history of telecommunications. John Wiley & Sons, Inc. Hoboken, New Jersey. 2003.

Java SE JDK. Versão 7, *update* 17. Estados Unidos: Oracle, 2013. Disponível em: <http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html>. Acesso em: 06 mar. 2013.

Java Statistical Class. **JSC**. Disponível em: <http://www.jsc.nildram.co.uk/index.htm>. Acesso em: 06 mai. 2013.

LAI, L.; EL GAMAL, H. The relay-eavesdropper channel: Cooperation for secrecy. **IEEE Transactions on Information Theory**, v. 54, n. 9, p. 4005-4019, 2008.

LI, H.; GONG, S.; LAI, L.; HAN, Z.; QIU, R.; YANG, D. Efficient and secure wireless communications for advanced metering infrastructure in smart grids. **IEEE Transactions on Smart Grid**, v. 3, n. 3, p. 1540-1551, set. 2012.

LI, H.; LAI, L.; ZHANG, W. Communication requirement for reliable and secure state estimation and control in smart grid. **IEEE Transactions on Smart Grid**, v. 2, n. 3, p. 476-486, set. 2011.

LI, S. R.; YEUNG, R. W.; CAI, N. Linear Network Coding. **IEEE Transactions on Information Theory**, v. 49, n. 2, p. 371-381, fev. 2003.

METKE, A. R.; EKL, R. L. Security technology for smart grid networks. **IEEE Transactions on Smart Grid**, v. 1, n. 1, p. 99-107, jun. 2010.

NetBeans. **Oracle**. Disponível em: <http://netbeans.org/features/cpp/>. Acesso em: 06 mar. 2013.

Oracle. **Oracle**. Disponível em: <http://www.oracle.com/index.html>. Acesso em: 06 mar. 2013.

PHULPIN, Y.; BARROS, J. LUCANI, D. Network coding in smart grids. **IEEE International Conference on Smart Grid Communications**, p. 49-54, out. 2011.

QUÉTÀND, È.. Curiosités de la science. **Le Petit Journal**. nov. 22, 1865, n. 1026, p.3. Disponível em: <http://gallica.bnf.fr/ark:/12148/bpt6k589123j.image.r=Manzetti.f3.langEN>. Acessado em: 19 fev. 2013.

REBELATTO, J. L.; UCHÔA-FILHO, B. F.; LI, Y.; VUCETIC, B. Multiuser cooperative diversity through network coding based on classical coding theory. **IEEE Transactions**, v. 60, n. 2, p. 916-926, Fev. 2012.

TSE, D.; VISWANATH, P. **Fundamentals of wireless communications**. Cambridge University Press, 2010.

VAN DER MEULEN, E. C. Three-terminal communication channels. **Advances in Applied Probability**, n. 3, p. 120-154, 1971.

VILELA, J. P.; LIMA, L.; BARROS, J. Lightweight security for network coding. **Proc. IEEE Int. Conf. on Communications**, 2008.

Visual Studio. **Microsoft Visual Studio**. Disponível em: <http://www.microsoft.com/visualstudio/en-us>. Acesso em: 06 mar. 2013.

WANG, X.; YI, P. Security framework for wireless communications in smart distribution grid. **IEEE Transactions on Smart Grid**, v. 2, n. 4, p. 809-818, dez. 2011.

WANG, Y.; LIN, W.; ZHANG, T. Study on security of wireless sensor networks in smart grid. **IEEE International Conference on Power System Technology**, p. 1-7, 2010.

WYNER, A. D. The wire-tap channel. **Bell Systems Technical Journal**, v. 54, n. 8, p. 1355-1387, 1975.

YAMAGUCHI, K.; TAKYU, O.; FUJII, T.; OHTSUKI, T.; SASAMORI, F.; HANDA, S. Physical layer security in physical layer network coding system with untrusted relay. 2013.

ZHANG, P.; JIANG, Y.; LIN, C.; FAN, Y. S., X. P-coding: secure network coding against eavesdropping attacks. **Proc. of IEEE INFOCOM**, Mar. 2010.