Sistema Multi-Agente para o Restabelecimento de Subestações Elétricas

CLAUDIO RIBEIRO LOPES JUNIOR

Itajubá, Fevereiro de 2002

ESCOLA FEDERAL DE ENGENHARIA DE ITAJUBÁ

Programa de Pós-graduação em Engenharia Elétrica

SISTEMA MULTI-AGENTE PARA O RESTABELECIMENTO DE SUBESTAÇÕES ELÉTRICAS

Dissertação apresentada à Escola Federal de Engenharia de Itajubá para a obtenção do grau de Mestre em Ciências em Engenharia Elétrica.

Cláudio Ribeiro Lopes Júnior

Itajubá, Fevereiro de 2002

ESCOLA FEDERAL DE ENGENHARIA DE ITAJUBÁ

Programa de Pós-graduação em Engenharia Elétrica

SISTEMA MULTI-AGENTE PARA O RESTABELECIMENTO DE SUBESTAÇÕES ELÉTRICAS

Dissertação apresentada à Escola Federal de Engenharia de Itajubá para a obtenção do grau de Mestre em Ciências em Engenharia Elétrica.

Cláudio Ribeiro Lopes Júnior

Itajubá, Fevereiro de 2002

Cláudio Ribeiro Lopes Júnior

SISTEMA MULTI-AGENTE PARA O RESTABELECIMENTO DE SUBESTAÇÕES ELÉTRICAS

Dissertação apresentada à Escola Federal de Engenharia de Itajubá para a obtenção do grau de Mestre em Ciências em Engenharia Elétrica.

Área de concentração:
Sistemas Elétricos de Potência

Orientador:
Germano Lambert Torres

Co-orientador:
Luiz Eduardo Borges da Silva

Itajubá, Fevereiro de 2002

AGRADECIMENTOS

Ao Prof. Germano Lambert Torres pela orientação e incentivo, sempre disposto a ajudar e esclarecer minhas dúvidas.

Ao Prof. Luiz Eduardo Borges da Silva, pela disponibilidade e pelas sugestões que ajudaram a encontrar os caminhos corretos.

Ao amigo Alexandre Rasi Aoki pelo incentivo e apoio, e pela companhia ao longo de toda a caminhada.

Ao amigo Jonas Guedes Borges da Silva, pela ajuda na programação.

À minha mãe e família, que sempre torceram pelo meu sucesso e me incentivaram nos momentos mais difíceis.

À Íris, companheira de todos os momentos, que suportou as dificuldades enfrentadas durante o tempo de trabalho.

Aos amigos Ahmed Amin Abdala Esmin, Carlos Henrique Valério de Moraes e Rodrigo de Oliveira Gonçalves, pela atenção e amizade, e em especial pela companhia durante a jornada.

À todos que, direta ou indiretamente, contribuíram para a realização desse trabalho.

À Deus, que ilumina nossos caminhos e nossas mentes, dando-nos meios para que possamos participar dos processos de melhoria do mundo e da vida de nossos semelhantes.

SUMÁRIO

Agradecimentos ii

Resumo vii

Abstract viii

Lista de Figuras ix

Lista de Tabelas xi

Capítulo 1 - Introdução 1

1.1 – Organização da Dissertação 3

Capítulo 2 – Sistemas Multiagentes 5

2.1 – Introdução 5

2.2 – Inteligência Artificial 5

2.2.1 – Teste de Turing 8

2.2.2 – Campos da IA 9

2.2.2.1 – Redes Neurais Artificiais 9

2.2.2.2 – Lógica Fuzzy 10

2.2.2.3 – Algoritmos Genéticos 10

2.2.2.4 – Sistemas Especialistas 11

2.3 – Agentes Inteligentes 12

2.3.1 – Propriedades do Agente 13

2.3.1.1 – Autonomia 13

2.3.1.2 – Racionalidade 14

2.3.1.3 – Continuidade Temporal 15

2.3.1.4 – Reatividade 16

2.3.1.5 – Habilidade Social 16

2.3.1.6 – Iniciativa Própria 16

2.3.2 – Estrutura dos Agentes Inteligentes 16

2.3.2.1 – Tipos de Agente 18

2.3.2.2 – Agentes Cognitivos e Reativos 21

2.3.3 – Conhecimento e Raciocínio 21

2.4 – Sistemas Multiagentes 23

2.4.1 – Comunicação entre Agentes 24

2.4.2 – Ambiente 25
2.4.3 – Aplicações de Sistemas Multi-Agentes 26
2.5 – Conclusões 27

Capítulo 3 – Subestações Elétricas 29

3.1 - Introdução 29

3.2 – Equipamentos de Manobra 30
3.2.1 - Chaves 30
3.2.1.1 - Seccionadoras 30
3.2.1.2 – Chaves de Operação em Carga 30
3.2.1.3 – Chaves de Aterramento Rápido 31
3.2.1.4 – Chaves de Terra 31
3.2.2 - Disjuntores 31
3.3 – Sistema de Medição 31
3.4 – Sistema de Proteção 33
3.4.1 – Proteção das Linhas 35
3.4.2 – Proteção de Transformadores 35
3.4.3 – Proteção de Barras 36
3.5 – Restabelecimento 37
3.5.1 – Linhas de Transmissão e Alimentadores de Distribuição 38
3.5.1.1 – Critérios de Restabelecimento 39
3.5.2 – Transformadores e Autotransformadores 40
3.5.3 – Equipamentos de Controle de Tensão 40
3.5.4 – Cargas 41

3.6 – Filosofia de Restabelecimento de Subestações 41
3.6.1 – Bases Filosóficas 41
3.6.2 – Hierarquia e Prioridades de Restabelecimento 42
3.6.2.1 – Falta de Tensão 42
3.6.2.2 – Atuação de Proteção 42
3.6.2.3 – Falha de Disjuntor 45
3.6.3 – Restabelecimento da SE 45
3.6.3.1 – Após um Colapso Total 46
3.6.3.2 – Após uma Perturbação no Sistema que Envolva a Subestação .47
3.6.4 – Influência das Condições Operativas do Sistema 48
3.7 – Subestação Modelo 49

3.8 - Conclusões 50

Capítulo 4 – Modelagem Orientada a Objeto 51

4.1 - Introdução 51

4.2 – Classes e Objetos 53

4.3 – Diagrama de Classe 55

4.4 – Relacionamento entre Classes 56

4.4.1 – Associação 56

4.4.2 – Agregação 58

4.4.3 – Generalização e Herança 59

4.4.4 – Outras Relações 60

4.5 – Encapsulamento 61

4.6 – Java 62

4.7 – Modelagem de Subestações via Orientação a Objeto 64

4.4.1 – O Modelo de Subestação Elétrica 64

4.8 – Conclusões 70

Capítulo 5 – Sistemas Multiagentes 71

5.1 – Introdução 71

5.2 – Modelo Multi-Agente para Subestação Elétrica 72

5.2.1 – Pacote Agentes de Decisão 74

5.2.1.1 – Agente Identificador de Eventos 75

5.2.1.2 – Agente Planejador 75

5.2.1.3 – Agente Modelo 75

5.2.2 – Pacote Agentes Ferramentas 77

5.2.2.1 – Agente Proteção 77

5.2.2.2 – Agente Equipamentos 78

5.2.2.3 – Agente Medidas 78

5.2.2.4 – Agente Chaves 78

5.3 – Sistema Multi-Agente para Restabelecimento de Subestações Elétricas 79

5.3.1 – Java 79

5.3.2 – O MARSE 79

5.3.2.1 – Abrindo um Caso Existente 80

5.3.2.2 – Abrindo um Novo Caso 81
5.3.2.3 – Menus 83
5.3.2.4 – Edição e Visualização 85
5.3.2.5 – Salvando Casos 88
5.3.2.6 – Funcionamento Passo a Passo 89
5.3.2.7 – Fluxograma de Informações 92
5.4 – Conclusões 93
Capítulo 6 – Testes e Resultados 95
6.1 – Introdução 95
6.2 – Falha na Barra BA-1 95
6.2.1 – Simulação 95
6.2.2 – Discussão dos Resultados 101
6.3 – Falha no Transformador TR-1 101
6.3.1 – Simulação 101
6.3.2 – Discussão dos Resultados 108
6.4 – Colapso Total na Subestação 109
6.4.1 – Simulação 109
6.4.2 – Discussão dos Resultados 120
6.5 – Conclusões 121
Capítulo 7 – Conclusões 122
Referências Bibliográficas 126
Anexo 1 – Banco de Dados do Modelo Completo da Subestação 132
Anexo 2 – Código do Modelo da Subestação 137
Anexo 3 – Código dos Agentes 144

RESUMO

O setor elétrico brasileiro vem sofrendo diversas ações de reestruturação, com a realização de estudos e a utilização de ferramentas modernas, na tentativa de melhorar o desempenho e minimizar as perdas do sistema elétrico de potência.

A tarefa de restabelecimento de uma subestação é executada através da intervenção de um operador humano, muitas vezes com o auxílio de um sistema automatizado de controle e supervisão.

Esse trabalho consiste na construção de um sistema multi-agente usando modelagem orientada a objeto, para gerar um sistema de apoio à decisão capaz de auxiliar o operador de uma subestação na tarefa de restabelecimento.

Um sistema de apoio à decisão, para atuar no restabelecimento de uma subestação elétrica, deve ser capaz de: avaliar o agente da ocorrência; definir a área defeituosa que foi isolada pela atuação da proteção; caracterizar se o defeito é permanente ou transitório; identificar os componentes envolvidos e os afetados; e gerar um plano de restabelecimento.

O modelo orientado a objeto garante uma poderosa abstração da subestação elétrica, permitindo a reprodução virtual das características distribuídas e topologia descentralizada, requisitos indispensáveis para a aplicação da tecnologia de agentes.

Já o sistema multi-agente representa a ferramenta computacional inteligente, que irá usar o conhecimento dos agentes de medição, proteção, equipamentos, identificador de eventos e chaves para elaborar a estratégia de restabelecimento da subestação elétrica.

A implementação desenvolvida consiste de um programa off-line de supervisão e restabelecimento em Java. Foram realizados estudos de caso visando confirmar a viabilidade da aplicação de um sistema multi-agente com modelagem orientada a objeto em sistemas de apoio à decisão para restabelecimento de subestações elétricas.

ABSTRACT

The Brazilian electric sector is under several reorganization actions, with the accomplishment of studies and the use of modern tools, in the attempt to improve the performance and to minimize the losses of the power system.

The substation restoration' task is executed through the intervention of a human operator, many times with the aid of a control and supervision' automated system.

This work consists on the construction of a multi-agent system using object-oriented modeling, to generate a decision support system able to assist the substation operator in the restoration task.

A decision support system that aids the power substation restoration should have means to: evaluate the agent of the occurrence; define the defective area which was isolated by the protection action; characterize if the defect is permanent or transitory; identify the involved components and the affected ones; and to generate a restoration plan.

The object-oriented model provides a powerful abstraction for the power substation, allowing the virtual reproduction of the distributed features and decentralized topology, indispensable requirements for the application of the agents' technology.

The multi-agent system performs the intelligent computational tool, which will use the knowledge of the measurement agent, protection agent, equipment agent, event identification agent and switch agent to elaborate the power substation restoration strategy.

The developed system consists of a supervision and restoration' off-line program in Java. Case studies were achieved aiming at confirm the viability of a multi-agent system application with object-oriented modeling in decision support system for power substation restoration.

LISTA DE FIGURAS

2.1 – Teste de Turing 8
2.2 – Estrutura e Funcionamento do SE 11
2.3 – Agentes Reflexivos Simples 18
2.4 – Agentes que Acompanham o Mundo 19
2.5 – Agentes Baseados em Metas 20
2.6 – Agentes Baseados em Satisfação 20
3.1 – Proteção de Linha da Subestação Modelo 35
3.2 – Proteção de Transfomadores da Subestação Modelo 36
3.3 – Proteção de Barras da Subestação Modelo 36
3.4 – Subestação Modelo 49
4.1 – Diagrama de Classe 55
4.2 – Multiplicidade Um para Um 57
4.3 – Multiplicidade Muitos para Muitos 58
4.4 – Agregação 59
4.5 – Herança 60
4.6 – Modelo de Subestação via Orientação a Objeto 66
4.7 – Classe Dispositivos 66
4.8 – Classe Manobras 68
4.9 – Classe Disjuntores 68
4.10 – Classe Proteções 69
4.11 – Classe Medidas 69
5.1 – Modelo Multi-Agente para Subestação Elétrica 73
5.2 – Comunicação entre Agentes 74
5.3 – Estrutura de Dados no Formato Árvore 76
5.4 – Função Load-Devices 77
5.5 – Janela Inicial do MARSE 80
5.6 – Abrindo um Caso Existente 81
5.7 – Abrindo um Novo Caso 81
5.8 – Tela Principal do MARSE 82
5.9 – Janela Representativa de Proteções do Nó TR-1 83
5.10 – Janela de Comunicação dos Agentes 84
5.11 – Modelo Orientado a Objeto da Subestação 84

5.12 – Janela de Saída dos Agentes 85
5.13 – Escolhendo o Equipamento para Visualizar as Medições 85
5.14 – Visualizando Dados de Medições 86
5.15 – Escolhendo o Equipamento para Visualizar as Proteções 86
5.16 – Janela Representativa de Proteções do Barramento BA-3 87
5.17 – Janela de Diálogo para Alterar o Estado das Proteções 87
5.18 – Janela de Diálogo para Alterar o Estado dos Disjuntores 88
5.19 – Salvando um Caso Existente 88
5.20 – Sugestões do Planejador 90
5.21 – Processamento Passo a Passo do MARSE 91
5.22 – Fluxograma de Informações do MARSE 92
6.1 – Falha em BA-1 – Passo 1 96
6.2 – Falha em BA-1 – Passo 2 96
6.3 – Falha em BA-1 – Passo 3 97
6.4 – Falha em BA-1 – Passo 4 99
6.5 – Falha em BA-1 – Passo 5 100
6.6 – Falha em TR-1 – Passo 1 102
6.7 – Falha em TR-1 – Passo 2 102
6.8 – Falha em TR-1 – Passo 3 104
6.9 – Falha em TR-1 – Passo 4 106
6.10 – Falha em TR-1 – Passo 5 107
6.11 – Colapso Total – Passo 1 109
6.12 – Colapso Total – Passo 2 110
6.13 – Colapso Total – Passo 3 113
6.14 – Colapso Total – Passo 4 117
6.15 – Colapso Total – Passo 5 118

LISTA DE TABELAS

2.1 – Exemplos de Agentes 17
3.1 – Principais Relés de Proteção Encontrados em Subestações 34
3.2 – Hierarquia de Restabelecimento 38
6.1 – Ajuste de Medidas Falha na Barra BA-1 Passo 4 99
6.2 – Ajuste de Medidas Falha na Barra BA-1 Passo 5 100
6.3 – Ajuste de Medidas Falha no Transformador TR-1 Passo 3 105
6.4 – Ajuste de Medidas Falha no Transformador TR-1 Passo 4 106
6.5 – Ajuste de Medidas Falha no Transformador TR-1 Passo 5 108
6.6 – Ajuste de Medidas Colapso Total Passo 3 115
6.7 – Ajuste de Medidas Colapso Total Passo 4 117
6.7 – Ajuste de Medidas Colapso Total Passo 5 119

1 – INTRODUÇÃO

A atual crise energética brasileira apresenta um cenário sem precedentes na história do país, que reflete a falta de investimento no setor elétrico. A situação foi agravada em 2001 pela ocorrência esparsa de chuvas e a população, o comércio e as indústrias têm sofrido com uma estratégia de racionamento que visa remediar temporariamente o problema.

O setor elétrico busca modos alternativos de produção de energia e a melhoria do sistema operante. Estudos são realizados e novas ferramentas são utilizadas na tentativa de melhorar o desempenho e minimizar as perdas do Sistema Elétrico de Potência (SEP).

A energia elétrica necessita ser consumida no mesmo instante em que é gerada, não podendo ser estocada. Isso exige que o SEP tenha, a todo o momento, condições de gerar, transportar e distribuir a energia que lhe é solicitada pelos consumidores. Quaisquer alterações de topologia ou interrupção de funcionamento de seus componentes podem afetar o SEP como um todo e, conseqüentemente, comprometer o fornecimento de energia elétrica aos consumidores.

Uma parte importante do SEP é a subestação elétrica (SE), que atende às necessidades de interconexão, transformação e controle do sistema. O restabelecimento de uma subestação, depois da ocorrência de uma falta, é executado através da intervenção de um operador humano, muitas vezes com o auxílio de um sistema automatizado de controle e supervisão.

Considerando a complexidade das topologias das subestações, e ainda que o processo de restauração do sistema não é uma tarefa totalmente

automática, a probabilidade de falha humana e o tempo gasto na execução das ações de restabelecimento são grandes e devem ser evitados.

A viabilidade técnica da utilização de sistemas automatizados em subestações é hoje comprovada, e a aplicação desses sistemas é fato, havendo vasta oferta de equipamentos e produtos no mercado. Esse tipo de procedimento se baseia na automação das ações de supervisão (sinalização e medição), controle (comando e seleção) e proteção.

Os sistemas automatizados de controle e supervisão são utilizados com sucesso, atuando em um grande número de usinas e subestações. Com o funcionamento desses sistemas, são obtidas grandes quantidades de dados e informações, sendo que para a aquisição de dados e para a supervisão da rede elétrica utilizam-se computadores em rede. Estes dados e informações podem ser utilizados não somente pelos sistemas automatizados, como também por outros sistemas computacionais que possam acessá-los da rede.

O desenvolvimento de técnicas aplicadas aos procedimentos operativos que proporcionem maior rapidez e eficiência na recomposição do sistema constituem o caminho para se obter uma minimização dos impactos impostos pelas perturbações ao SEP. Elas levam a soluções que aumentam o desempenho e o grau de confiabilidade na execução do restabelecimento de uma SE.

A integração de sistemas automatizados com sistemas inteligentes tem se mostrado viável tecnologicamente. Mais que isso, essa integração se faz necessária, uma vez que em um sistema de operação desassistida de subestações, onde o operador controla várias unidades ao mesmo tempo, os sistemas inteligentes de apoio à decisão são ferramentas importantes para uma boa operação do sistema elétrico.

Esse trabalho busca estruturar uma ferramenta inteligente, baseada em Sistemas Multi-Agentes (SMA) e utilizando modelagem Orientada a Objeto (OO), para gerar um sistema de apoio a decisão capaz de contribuir com o operador de uma subestação no processo de restabelecimento. Esta aplicação deverá apresentar flexibilidade na manutenção e permitir a adição

de novos conhecimentos, assim como fornecer subsídios para aplicação em sistemas automatizados de supervisão e controle de subestações elétricas.

Em primeira instância, o projeto consta de um módulo "offline" de supervisão contendo estratégia de restabelecimento desenvolvido em Java. Através da análise dos resultados encontrados na aplicação desse módulo, busca-se confirmar a viabilidade da aplicação de SMA usando modelagem OO em sistemas de apoio à decisão em restabelecimento de subestações elétricas.

O modelo orientado a objetos permite uma poderosa abstração da subestação elétrica. De acordo com essa modelagem, a subestação terá características distribuídas e topologia descentralizada, que são requisitos indispensáveis para a aplicação da tecnologia de agentes.

O SMA representa a ferramenta computacional inteligente, que irá usar o conhecimento dos agentes de medição, proteção, equipamentos e alarmes para elaborar a estratégia de restabelecimento da subestação elétrica.

Um sistema de apoio à decisão, para atuar no restabelecimento de uma subestação elétrica, deve ser capaz de:

- Avaliar o agente da ocorrência;
- Definir a área defeituosa que foi isolada pela atuação da proteção;
- Caracterizar se o defeito é permanente ou transitório;
- Identificar os componentes envolvidos e os afetados;
- Gerar um plano de restabelecimento [1].

1.1 - ORGANIZAÇÃO DA DISSERTAÇÃO

Esse documento se divide da seguinte forma: no Capítulo 2, após breve apresentação de referências sobre Inteligência Artificial, são apresentados os conceitos básicos de Sistemas Multi-Agentes, incluindo considerações sobre estrutura, raciocínio, ambiente e comunicação.

O Capítulo 3 descreve as Subestações Elétricas, mostrando os equipamentos de manobra, proteção e medição. São também descritos os

critérios e filosofias de restabelecimento comumente adotados em subestações de 88-138kV, e ainda é apresentada a hierarquia e prioridades de restabelecimento após um colapso total e após uma perturbação no sistema.

No Capítulo 4, são apresentados os conceitos básicos de Programação Orientada a Objeto, sendo dedicada especial atenção à abstração e modelagem. Faz parte desse capítulo a construção de um modelo orientado a objeto de uma subestação elétrica.

No Capítulo 5, são apresentados o modelo multi-agente para restabelecimento de subestações elétricas e o software de apoio à decisão.

O Capítulo 6 é dedicado à análise de resultados de simulações, no estudo de casos escolhidos para a aplicação do software desenvolvido.

Por fim, no Capítulo 7, apresentam-se as principais conclusões e perspectivas futuras desse trabalho.

2 - SISTEMAS MULTI-AGENTES

2.1 – INTRODUÇÃO

Nesse capítulo serão descritas as características dos Agentes Inteligentes, assim como o processo de integração entre as unidades e seu comportamento social. A exposição é iniciada com uma pequena revisão sobre a evolução da Inteligência Artificial. A seguir, são expostos os parâmetros que definem um agente como entidade inteligente independente. Finalmente, são levados em consideração os aspectos de comunicabilidade e comportamento social, definindo os limites de interação entre agentes e entre os agentes e o ambiente no qual estarão atuando.

2.2 - INTELIGÊNCIA ARTIFICIAL

Definir Inteligência Artificial exige antes de tudo uma definição de inteligência. Etimologicamente, a palavra inteligência origina-se da junção das palavras *inter* e *legere*, em latim, que significam entre e escolher, respectivamente. A inteligência consiste no poder de escolha racional do homem. A razão é uma característica peculiar do ser humano, baseada na estruturação lógica de propriedades cognitivas. Registros de estudos relacionados à definição de inteligência e comportamento humano são conhecidos desde a época de Platão, por volta de 450 AC., quando este relatou a preocupação de Sócrates com o estabelecimento de um padrão de piedade, através do qual fosse possível julgar o comportamento dos homens [2]. Desde então, várias correntes filosóficas têm tentado definir padrões que permitam compreender os processos de heurística e inferência da inteligência humana.

A palavra artificial vem do latim artificiale, e significa algo não natural, produzido pelo homem. No contexto de Inteligência Artificial, o significado toma o sentido de similaridade, indicando a tentativa de reprodução de características de inteligência humana, como raciocínio e poder de decisão, em computadores.

O interesse pela Inteligência Artificial vem de longa data, e tem a oficialização de seu nascimento como ciência no verão de 1956, com um workshop em Dartmouth College, New Hampshire, nos Estados Unidos, organizado pelo idealizador John McCarthy e por Marvin Minsky, Claudi Shannon e Nathaniel Rochester. Eles reuniram pesquisadores interessados nos ramos de redes de neurônios, em automação e no estudo de inteligência. Esse evento teve duração de dois meses, e contou com a presença de outros seis cientistas. São eles Trendchard More, Arthur Samuel, Ray Solomonoff, Oliver Selfridge, Allen Newell e Herbert Simon. Desse encontro surgiu organizado como campo da ciência a Inteligência Artificial [2].

As definições de Inteligência Artificial podem ser agrupadas em quatro tipos distintos. São eles os sistemas que pensam como humanos, que agem como humanos, que pensam racionalmente e que agem racionalmente.

1 – Sistemas que pensam como humanos:

> "O excitante esforço novo para fazer computadores pensar... máquinas com mente, no sentido completo e literal" [3].

> "(A automação de) atividades que associamos com o pensamento humano, tais como tomada de decisão, resolução de problemas, aprendizado..." [4].

2 – Sistemas que pensam racionalmente:

> "O estudo de faculdades mentais através do uso de modelos computacionais" [5].

"O estudo da computação que torna possível perceber, raciocinar e agir" [6].

3 – Sistemas que agem como humanos:

"A arte de criar máquinas que executem funções que exijam inteligência quando executadas por pessoas" [7].

"O estudo de como fazer os computadores realizarem coisas que, no momento, as pessoas fazem melhor" [8].

4 – Sistemas que agem racionalmente:

"Um campo de estudo que busca explicar e emular o comportamento inteligente em termos de processos computacionais" [9].

"O ramo de ciência da computação que está interessado com a automação de comportamento inteligente" [10].

O campo da Inteligência Artificial (IA) vem se tornando uma das áreas mais estudadas da ciência moderna. Desde o advento dos computadores o homem se preocupa com a possibilidade de reprodução em software dos processos de raciocínio e comportamento humano e a conseqüente criação de núcleos de processamento que respondam como pessoas às necessidades do meio no qual atuam. Para tal, esses núcleos devem conhecer o comportamento do ambiente no qual estão inseridos, devem saber como esse meio evolui e como, através de seus atos, melhorar essa evolução.

De uma forma geral, a IA pode ser vista como o estudo de computação direcionado ao desenvolvimento de sistemas que exibem características de inteligência, tais como compreensão de linguagem, raciocínio lógico, comportamento inteligente e habilidade em resolução de problemas.

A Inteligência Artificial se apóia em dois paradigmas, que a dividem em duas grandes áreas. A primeira delas é a Inteligência Artificial Simbólica,

que enfoca o estudo e a simulação do comportamento inteligente humano, sem considerar os mecanismos físicos, químicos e biológicos do cérebro. A segunda área é a Inteligência Artificial Conexionista, que trabalha com o desenvolvimento de sistemas que simulam as características estruturais do cérebro humano e os mecanismos físicos e químicos de comunicação entre os neurônios.

2.2.1 - O TESTE DE TURING

Uma das perguntas mais importantes que se faz com relação à Inteligência Artificial é quando o computador na verdade estará pensando e não simplesmente processando uma lista de informações.

Alan Turing, em 1950, propôs um teste que definiria o comportamento inteligente de um computador. Para tal, ele sugeriu um ambiente distribuído da seguinte forma: em uma sala fica uma pessoa A em um terminal, trocando mensagens datilografadas com dois terminais situados em outra sala, Figura 2.1. Em um desses terminais está uma pessoa B, e no outro, um computador. A pessoa A faz perguntas aos dois outros terminais. Se a pessoa A, analisando as mensagens trocadas, não conseguir distinguir quem é a pessoa B e quem é o computador da outra sala, pode-se dizer que o computador age de forma inteligente.

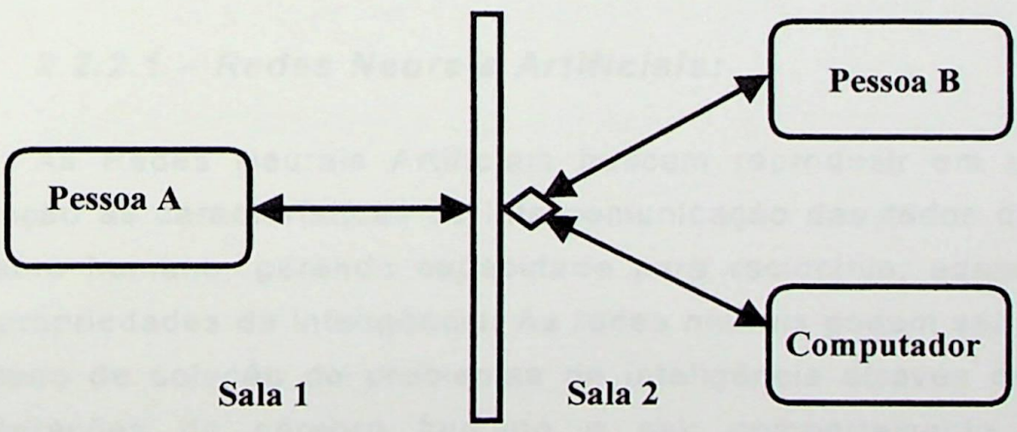


Figura 2.1: Teste de Turing

Esse é um teste clássico, e exige do computador não apenas a simulação de características de comunicação humanas, como dar respostas erradas, demorar a responder como se estivesse em dúvida e responder instantaneamente perguntas fáceis, mas requer também o entendimento do

significado das palavras e a formulação de expressões com sentido e, principalmente, dentro do contexto do diálogo. O computador precisa manter um diálogo inteligente com a pessoa A, ou, em outras palavras, precisa pensar.

A capacidade de raciocínio é crucial para o desenvolvimento de um sistema inteligente. Um programa inteligente deve, a partir das ocorrências percebidas, tomar as decisões corretas para a solução do problema. Não basta o armazenamento das possíveis soluções em um banco de dados. O sistema precisa compreender as ocorrências. O teste de Turing, trabalhando de maneira particular com linguagem e comunicação verbal, representa de forma completa essa realidade. Ele mostra que inteligência artificial não se resume a um algoritmo sofisticado, mas que é um campo que exige capacidade de estabelecimento de relações entre fatos, compreensão do contexto geral do problema e do ambiente e aprendizagem.

2.2.2 - CAMPOS DA IA

O estudo de IA se desdobrou por diversas vertentes com o passar do tempo, resultando na formação de vários campos de estudo e desenvolvimento. Entre outras áreas, a Inteligência Artificial apresenta os campos de Redes Neurais Artificiais, Lógica Fuzzy, Algoritmos Genéticos e Sistemas Especialistas.

2.2.2.1 – Redes Neurais Artificiais:

As Redes Neurais Artificiais buscam reproduzir em sistemas de computação as características de intercomunicação das redes de neurônios do cérebro humano, gerando capacidade para raciocínio, adaptabilidade e outras propriedades da inteligência. As redes neurais podem ser vistas como um método de solução de problemas de inteligência através da simulação das interações do cérebro humano e seu comportamento, com suas atividades de interpretação, descoberta de erros e aprendizado.

As aplicações de redes neurais artificiais são inúmeras. Entre outras, são encontradas redes neurais na análise de mercado financeiro, nos sistemas de reconhecimento de padrões, como no sistema OCR (Reconhecimento Óptico de Caracteres), em análise de voz, em detecção de

fraudes em cartões de crédito, análise e processamento de sinais e até em filtros de ruídos eletrônicos.

2.2.2.2 – Lógica Fuzzy:

A Lógica Fuzzy é utilizada em problemas onde estão envolvidos fenômenos de incerteza e precisão, apresentando fortes características de tomada de decisão. Em ambientes que apresentam relativa subjetividade, a lógica fuzzy auxilia no estabelecimento de padrões de localização e importância das partes componentes do problema. Em um problema, por exemplo, de estacionamento de um veículo, a lógica fuzzy ajuda a estabelecer o posicionamento e os deslocamentos angular e linear das rodas do veículo [11].

A lógica fuzzy se fundamenta na Teoria dos Conjuntos, onde um elemento pertence ou não pertence a um determinado intervalo. Nos Conjuntos Fuzzy (ou Conjuntos Difusos), ocorre uma generalização da Teoria dos Conjuntos, onde cada elemento tem uma medida de pertinência ao intervalo considerado. São atribuídos pesos às regiões do intervalo, e considera-se que certos pontos pertençam mais ao intervalo que outros. Mais que isso, permite a diferenciação de pontos próximos ao intervalo desejado e pontos distantes do intervalo desejado. Isso é importante porque em determinados tipos de problemas, na dificuldade de alcançar um resultado exato, surge a possibilidade de se encontrar aproximações válidas.

2.2.2.3 – Algoritmos Genéticos:

Os Algoritmos Genéticos se apóiam nos fundamentos da seleção natural e da genética. São constituídos de um algoritmo de otimização que explora busca paralela e aleatória, mas estruturada, cujo objetivo é encontrar pontos de convergência para o problema estudado.

Após cada iteração, o sistema sofre processos de seleção e reprodução, surgindo novas populações com conhecimento histórico das movimentações anteriores, que são chamadas gerações. Como nas definições biológicas, cada indivíduo tem registrado vetorialmente a presença ou a ausência de uma característica, o que constitui o seu genótipo. Os elementos desse vetor, combinados, fornecem o fenótipo do indivíduo, ou suas características gerais.

O Algoritmo prevê a existência de um critério de avaliação e seleção que conduza as características gerais da população de forma que, depois de algumas gerações, o conjunto inicial de indivíduos tenha dado origem a um conjunto mais apto. Esse grau de aptidão é estabelecido pelo próprio critério de avaliação, e a seleção prevê que uma porcentagem dos indivíduos mais aptos reproduza, de forma a melhorar o conjunto na média, mas mantendo um certo grau de heterogeneidade na população. Os indivíduos mais aptos, mantidos pela seleção, sofrem alterações em suas características através de cruzamentos, recombinação genética e mutações. O processo é repetido até que uma solução razoável seja alcançada.

Os sistemas baseados em algoritmos genéticos apresentam considerável carga computacional, mas representam um bom instrumento no tratamento de sistemas em ambientes dinâmicos, os quais requerem características de comportamento adaptativo. Podem ser usados nas áreas de escolha de rotas, configuração de sistemas complexos e problemas de otimização.

2.2.2.4 – Sistemas Especialistas:

Os Sistemas Especialistas consistem de sistemas inteligentes direcionados à resolução de problemas em área específica, contando com utilização de conhecimento de pessoas consideradas especialistas nessa área. Normalmente os sistemas especialistas são utilizados para a resolução de problemas que exigem a atuação de um especialista humano, fornecendo soluções inteligentes e disponibilizando a sua linha de raciocínio para o usuário. Na Figura 2.2 são representados a estrutura e o funcionamento de um sistema especialista.

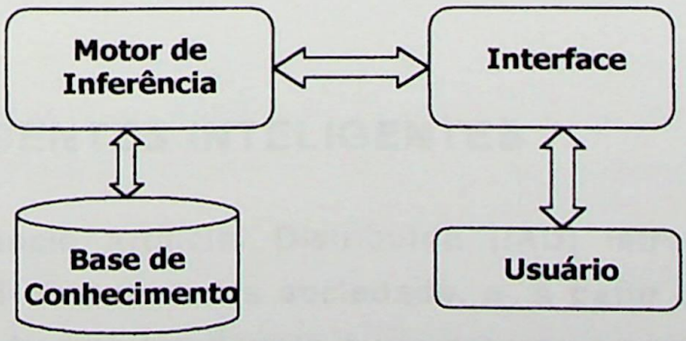


Figura 2.2 – Estrutura e funcionamento do SE

A estrutura de um sistema especialista deve possibilitar o armazenamento e processamento de conhecimento e disponibilizar uma interface compreensível com o usuário.

A Interface é responsável pela comunicação do sistema com o usuário, e através dela ocorre a entrada de dados referentes ao estado do mundo e a saída de resultados e conclusões alcançadas.

A Base de Conhecimento é composta pelos fatos e pelas regras armazenadas, necessárias para o funcionamento do dispositivo.

O Motor de Inferência acessa a base de conhecimento, consultando os fatos e regras armazenados, e, baseado nas informações fornecidas pela interface sobre o estado do mundo, infere sobre o conhecimento disponível gerando uma conclusão.

Com o crescimento da popularidade dos sistemas especialistas e o conseqüente aumento de suas aplicações, o porte dos problemas nos quais eles eram empregados começaram a crescer. O gerenciamento e criação das bases de conhecimento tornam-se, com o aumento do tamanho do problema analisado e conseqüente aumento do número de entradas e saídas, um problema de difícil solução.

A solução encontrada pelos estudiosos foi o uso do conceito de Inteligência Artificial Distribuída, que sugeria a princípio a divisão das bases de conhecimento em unidades menores, cada uma contando com seu próprio motor de inferência. Dentro de um problema distribuído, os sistemas especialistas trocam informações entre si, e, trabalhando com uma divisão hierárquica bem definida, conseguem um desempenho consideravelmente melhor, com custos computacional e de desenvolvimento substancialmente menores.

2.3 – AGENTES INTELIGENTES

A Inteligência Artificial Distribuída (IAD) introduz à Inteligência Artificial tradicional o conceito de sociedade, e, a partir dele, o conceito de Multi-Agentes. Um Agente Inteligente é um programa com núcleo inteligente

que se integra a uma sociedade modular, seguindo premissas padronizadas de comunicação e tendo função específica dentro dessa sociedade.

De acordo com Russell & Norvig [2], o agente pode ser visto como uma estrutura que percebe o ambiente através de sensores e atua nesse ambiente através de atuadores. Segundo Wooldridge & Jennings [12], o agente consiste de um hardware ou um sistema computacional baseado em software que possua as seguintes propriedades: autonomia, habilidade social, reatividade e iniciativa própria.

2.3.1 - PROPRIEDADES DO AGENTE

Dentre as características que um agente inteligente deve apresentar, se destacam:

- Autonomia;
- Racionalidade;
- Continuidade temporal;
- Reatividade;
- Habilidade social;
- Iniciativa própria.

2.3.1.1 – Autonomia

Ao contrário dos programas tradicionais, os agentes constituem entidades independentes, respondendo a situações não previstas e aprendendo a partir da observação do ambiente no qual está alocado. De fato, o comportamento do agente depende principalmente de sua seqüência de percepção. Através do histórico das variações do ambiente ele gera o seu comportamento. Assim, um agente particular pode ser descrito pela construção de uma tabela de ações que ele toma em resposta a uma dada seqüência de acontecimentos. À medida que aumentam as ocorrências, aumenta o tamanho da tabela, pois a cada entrada há a deflagração de uma etapa de estudo sobre a natureza da perturbação e sua influência no ambiente. Essa tabela é chamada de mapa de seqüência de percepção-ação.

A autonomia é uma das principais características que um agente inteligente deve possuir. Se o agente tem suas ações baseadas somente no conhecimento dado pelo programador, sem dedicar atenção a sua percepção, ele não tem autonomia. O comportamento de um agente pode ser baseado

no conhecimento dado na programação, mas deve ser também baseado no conhecimento adquirido.

O conhecimento adquirido pela experiência permite que o agente não fique sem saber como reagir a uma mudança brusca no ambiente [2]. Um sistema é autônomo a partir do momento em que seu comportamento é determinado por sua experiência adquirida. Enquanto um agente tem pouca ou nenhuma experiência, ele vai agir aleatoriamente até que o projetista, agindo como especialista humano na área de atuação do software, ensine o que fazer sob diversas circunstâncias. Essa assistência de especialista é fundamental, pois é a partir dela que o agente aprende como reagir às variações do ambiente de forma a cumprir seus objetivos. Assim, o agente deve ter, inicialmente, além de sua base de conhecimento, uma orientação para aprender a agir.

A autonomia é a característica que permite ao agente tomar iniciativas e operar sem a intervenção de elemento humano ou outro qualquer elemento computacional externo. Ela confere ao agente o poder de decisão e controle de suas ações na busca de suas metas.

De uma forma geral, a autonomia pode ser definida como a operação dos agentes sem intervenção de humanos ou outros, mantendo controle sobre suas ações e estados internos [12].

2.3.1.2 – Racionalidade

O agente deve estar preparado para tomar a ação correta, que conduza ao melhor resultado. Muitas vezes ele irá se deparar com uma variedade de possíveis soluções, algumas antagônicas. Poderá receber informações e solicitações também variadas e paradoxais, de fontes distintas. Como acontece com os agentes humanos.

Como encontrar a melhor solução? Como agir de forma a gerar a melhor condição para o sistema e as melhores respostas para as solicitações? As respostas para a questão do sucesso de um agente estarão na sua própria definição.

O que se espera desse agente? Se o esperado de um agente específico for precisão matemática, uma vez que ele calcula de forma correta as respostas desejadas, ele tem seu grau de sucesso garantido pelas respostas que gera. Mas se o esperado for, além da precisão nos cálculos, que ele pisque o olho esquerdo ao entregar o resultado, e ele não o faz, seu sucesso está comprometido.

Dessa forma, devem ser estabelecidos padrões de comportamento que sirvam de referência para as medidas de desempenho que determinarão o sucesso do agente. As medidas de desempenho devem analisar o como e o quando da atuação do agente. O como engloba as características que o agente deve possuir e o quando diz respeito à regularidade, à rapidez, etc.

O agente possui um conhecimento do ambiente em que irá atuar suficientemente grande para gerar respostas satisfatórias e desempenhar com sucesso suas tarefas. Porém, não é possível ao agente controlar todos os fatos que ocorrem ao seu redor, mesmo porque muitos dos fatos que podem vir a ocorrer não estarão figurando no seu algoritmo nem em sua base de conhecimento.

Um agente, para ser racional, deve tomar as decisões corretas nos momentos certos. Para tanto, ele necessita de quatro fatores:

- Uma medida de desempenho que define o grau de sucesso;
- Sua seqüência de percepção (o histórico de fatos que ele enxerga e entende);
- O que ele sabe sobre o ambiente;
- As ações que ele pode tomar.

Esses fatores são suficientes para se ter uma definição de agente ideal: "Para cada possível seqüência de percepção, um agente racional deveria agir de forma a maximizar sua medida de desempenho, baseado nas evidências registradas por sua seqüência de percepção e pela construção de conhecimento que ele possui" [2].

2.3.1.3 – Continuidade Temporal

Outra característica fundamental é a propriedade de continuidade temporal, pois o agente necessita estar em constante vigília. Isso vem da

própria definição de agente, que o descreve como sendo uma estrutura capaz de gerar ações baseadas no que ele percebe das modificações no ambiente. Assim, o agente se mantém em monitoramento contínuo do mundo, atento para todas as alterações que possam ocorrer [13].

2.3.1.4 – Reatividade

É a propriedade que garante uma resposta às modificações ocorridas no ambiente, em tempo hábil. O monitoramento previsto pela propriedade de continuidade temporal habilita a propriedade reativa quando da efetiva observação de uma ocorrência no mundo. A reatividade é responsável pela elaboração da atuação corretiva e sua literal aplicação.

2.3.1.5 – Habilidade Social

Agentes interagem com outros agentes e possivelmente com humanos, usando alguma linguagem de comunicação para agentes [12].

Os agentes precisam cooperar entre si, para que as metas comuns sejam alcançadas e para que o objetivo global da sociedade seja atingido. A interação social deve não somente ficar restrita dentro da sociedade, mas deve também vigorar entre sociedades diferentes, que necessitem de troca de informação e ajuda mútua para atingir seus objetivos.

Vale ressaltar que uma sociedade de agentes ideal se baseia em uma arquitetura aberta, que comportaria a entrada e saída de novos membros, o que destaca ainda mais a importância da habilidade social como propriedade de um agente inteligente.

2.3.1.6 – Iniciativa Própria

Um agente não simplesmente age em resposta às mudanças em seu ambiente, mas exibe comportamento baseado em seus propósitos, tomando iniciativa pertinente [12].

2.3.2 – ESTRUTURA DOS AGENTES INTELIGENTES

A estrutura de um agente depende do seu tipo e função, e do dispositivo computacional no qual será instalado e no qual irá trabalhar. O

programa deve ser compatível com a arquitetura do sistema computacional no qual será inserido. Essa arquitetura pode ser um computador pessoal, uma rede ou um hardware dedicado à aplicação. De uma forma geral, a arquitetura é responsável pelos sinais dos sensores ficarem disponíveis para o programa, é responsável por rodar o programa e alimentar os atuadores com as escolhas do agente.

De acordo com Russell & Norvig [2], a relação entre agente, programa e arquitetura é dada por:

- **Agente = Arquitetura + Programa**

Para a construção do programa, é necessário definir os limites de percepção e ação do agente, assim como ter conhecimento de suas metas e medidas de desempenho, além de conhecer o ambiente no qual ele irá operar. Na Tabela 2.1 são apresentados alguns exemplos.

Tabela 2.1 – Exemplos de Agentes

Tipo de Agente	Percepções	Ações	Metas	Ambiente
Mecânico de automóveis	Sintomas no veículo, explicações do proprietário	Perguntas, testes, substituição de peças, conserto	Funcionamento perfeito do veículo, baixo custo na manutenção	Oficina mecânica, carro.
Inspetor de qualidade em usinagem	Rugosidade no eixo, tolerâncias, acabamento	Medição, verificação visual, aceitação ou descarte	Entregar lotes com peças dentro de padrões desejados	Centro de usinagem, peças
Fisioterapeuta	Sintomas, explicações do paciente	Perguntas, testes, tratamento	Garantir melhora e o bem estar do paciente	Clínica, paciente
Relés em subestação	Variações de tensão, aquecimento em transformador	Monitoramento, sinalização de irregularidades	Mostrar variações indesejadas das grandezas elétricas	Subestação, linhas de transmissão

Uma vez definidos o ambiente, o tipo de agente e suas características, começa a se pensar nas suas metas e medidas de desempenho.

Essas medidas não fazem parte do corpo do agente. Elas são atribuições externas, definidas de acordo com as necessidades do ambiente. Uma maneira de se definir as medidas seria inserir uma análise de especialista no assunto da aplicação para se quantificar o sucesso das

decisões. A análise da atuação do agente face às modificações ambientais percebidas e a comparação com padrões de qualidade pré-estabelecidos mostra quão feliz ele conseguiu ser.

À medida que se torna mais complexo o ambiente, observa-se que o mapa de seqüência de percepção perde registros de acontecimentos. Não é possível a obtenção de uma tabela de entradas completa, salvo quando o ambiente é extremamente simples.

Normalmente, os problemas abordados pelos agentes apresentam um número muito grande de entradas, e à medida que aumenta esse número, redundâncias começam a aparecer. O programa precisa trabalhar com a seqüência de percepção armazenada, mesmo que essa não contenha todos os dados esperados. De qualquer forma, o monitoramento do ambiente é contínuo e a atualização do banco de dados da seqüência é constante (propriedade de continuidade temporal). Baseado na complexidade e nas necessidades intrínsecas do ambiente, o tipo de agente é escolhido.

2.3.2.1 – Tipos de Agente

De acordo com a sua estrutura interna, os agentes podem ser distribuídos em quatro grupos distintos [2]:

- **Agentes Reflexivos Simples**: O agente age de forma reflexiva à verificação de qualquer mudança no estado do ambiente. Uma vez detectada uma variação, ele busca a primeira regra associada de seu conjunto de regras que corresponda a essa descrição de estado para atuar no ambiente, como mostrado na Figura 2.3.

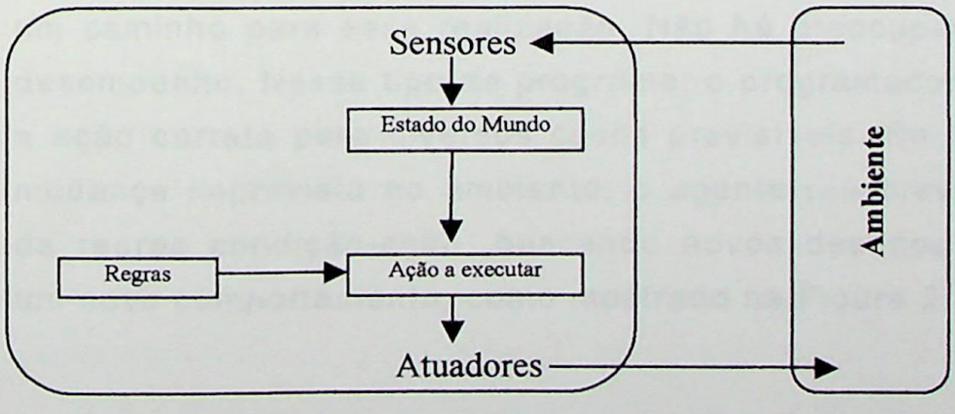


Figura 2.3: Agentes Reflexivos Simples

- **Agentes que Acompanham o Mundo**: Nesse tipo de agente, a percepção atual é combinada com o antigo estado interno, de forma a gerar uma descrição atualizada do estado corrente. O agente incorpora uma previsão sobre o que o fato percebido gera no ambiente, de acordo com estados anteriores guardados na memória. Baseado nessa visão e interpretação, ele escolhe a regra a ser utilizada, como mostrado na Figura 2.4.

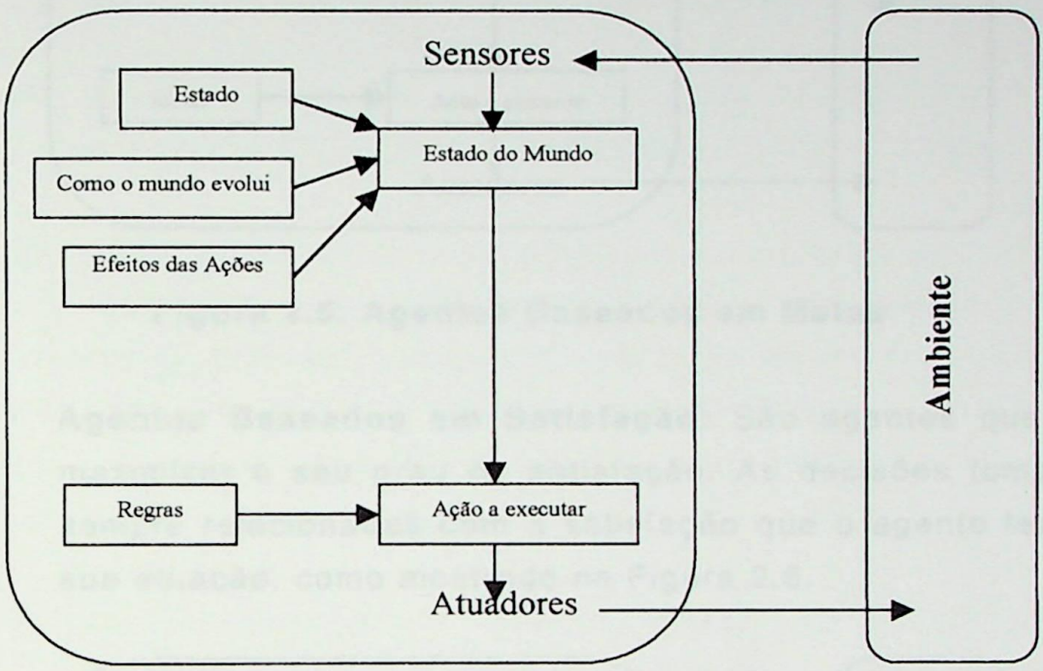


Figura 2.4: Agentes que Acompanham o Mundo

- **Agentes Baseados em Metas**: Esse tipo de Agente assume que apenas saber o estado atual do ambiente e os estados anteriores não é suficiente para decidir que ação tomar. Para ele, é preciso ter definida uma meta. Ele precisa ter uma noção do resultado que busca, precisa saber o que precisa fazer. Ele armazena algum tipo de meta na memória e, a partir dela, busca um caminho para essa realização. Não há preocupação com o desempenho. Nesse tipo de programa, o programador já embutiu a ação correta para diversos casos previsíveis. Se houver uma mudança imprevista no ambiente, o agente reescreve uma lista de regras condição-ação, buscando novos destinos e gerando um novo comportamento, como mostrado na Figura 2.5.

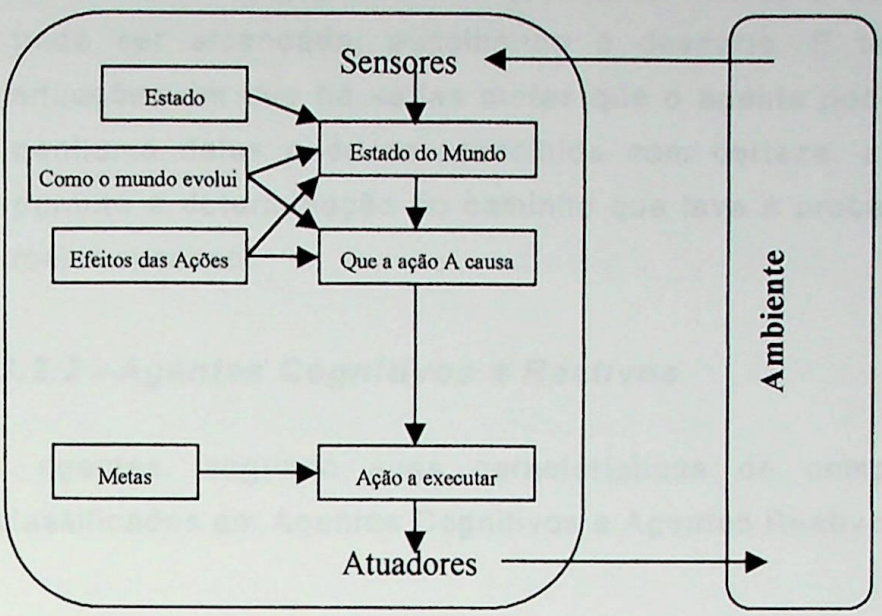


Figura 2.5: Agentes Baseados em Metas

- **Agentes Baseados em Satisfação**: São agentes que buscam maximizar o seu grau de satisfação. As decisões tomadas são sempre relacionadas com a satisfação que o agente terá com a sua atuação, como mostrado na Figura 2.6.

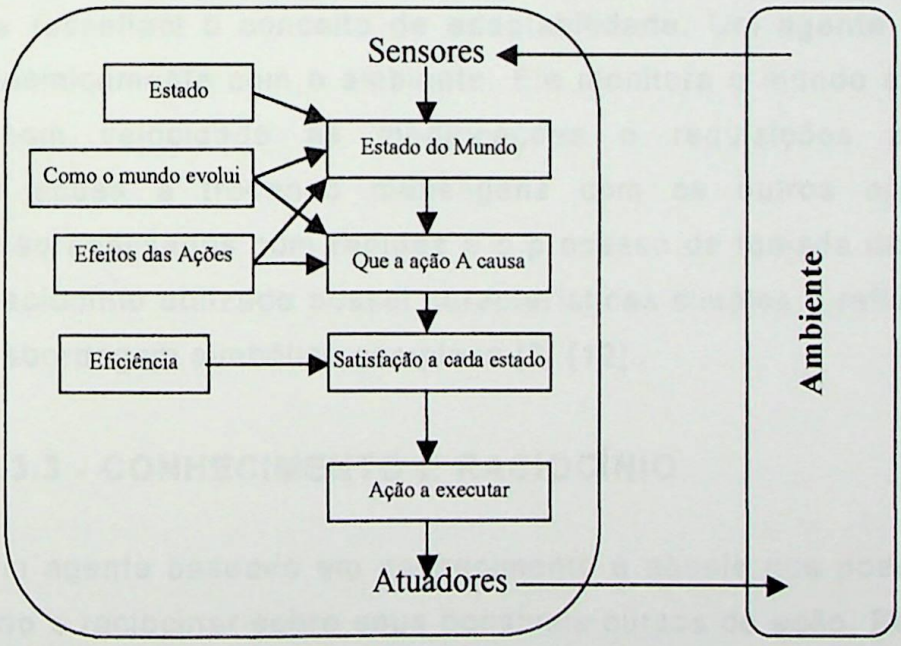


Figura 2.6: Agentes Baseados em Satisfação

Apenas metas não são suficientes para gerar comportamento de qualidade. Esse tipo de agente insere em um agente baseado em metas o sentido de eficiência. Ele busca o caminho de menor custo, o caminho mais rápido, etc. Permite

decisões em casos onde há conflito de metas e somente uma pode ser alcançada, escolhendo o descarte. E também, em situações em que há várias metas que o agente pode buscar, e nenhuma delas pode ser escolhida com certeza, a satisfação permite a determinação do caminho que leva à probabilidade de maior sucesso.

2.3.2.2 –Agentes Cognitivos e Reativos

Os agentes, segundo suas características de comportamento, podem ser classificados em Agentes Cognitivos e Agentes Reativos.

Os Agentes Inteligentes Cognitivos comportam-se de maneira racional, possuem uma concepção lógica simbólica do mundo e planejam suas estratégias de ação de acordo com um forte mecanismo de inferência lógica. Esse tipo de agente, além de utilizar técnicas de dedução e aprendizado, considera também aspectos característicos da vontade humana, como crença, desejo e intenção [12].

Os Agentes Inteligentes Reativos baseiam-se em comportamento reflexivo, e ressaltam o conceito de adaptabilidade. Um agente desse tipo interage dinamicamente com o ambiente. Ele monitora o mundo à sua volta, reagindo com velocidade às modificações e requisições percebidas, produzindo ações e trocando mensagens com os outros agentes. Os estímulos são analisados com rapidez e o processo de tomada de decisão é rápido. O raciocínio utilizado possui características simples e reflexivas, não possuindo abordagem simbólica complexa [2] [12].

2.3.3 - CONHECIMENTO E RACIOCÍNIO

Um agente baseado em conhecimento é aquele que pode conhecer o seu mundo e raciocinar sobre seus possíveis cursos de ação. Pode aceitar novas tarefas e metas, adquirir e atualizar conhecimento baseado nas mudanças do ambiente e se adaptar a essas mudanças. Esse agente precisa ter noções sobre o estado corrente do mundo, saber como inferir informações não visíveis do mundo a sua percepção, entender como o mundo se modifica com o decorrer do tempo, o que ele quer desenvolver e que ações deve tomar em circunstâncias variadas.

O agente baseado em conhecimento tem como núcleo uma Base de Conhecimento (BC), que é um conjunto de representações de fatos sobre o mundo. Cada representação é chamada de sentença. As sentenças são expressas na chamada linguagem de representação de conhecimento.

A aprendizagem se dá através do processo de atualização da BC.

O agente baseado em conhecimento funciona armazenando as representações do mundo em sua BC e utiliza um mecanismo para formular as novas sentenças, tendo como base essas representações. Esse mecanismo, chamado motor de inferência, além de formular as novas sentenças, é responsável por decidir a ação a ser tomada em resposta às variações do ambiente.

O Agente baseado em conhecimento é composto por uma BC e um motor de inferência, e pode ser descrito como uma estrutura composta por três níveis [2]:

- ***Nível de conhecimento:*** é uma representação do teor do conhecimento do agente. É o que o agente conhece, o que ele sabe;
- ***Nível lógico:*** é o nível no qual o conhecimento é codificado em sentenças. O significado de uma sentença é o que ela define sobre o estado do mundo;
- ***Nível de implementação:*** É o nível no qual fica abrigada a arquitetura do agente, onde há representação física das sentenças do nível lógico.

A linguagem de representação de conhecimento é definida por aspectos de sintaxe e semântica. A sintaxe descreve as possíveis configurações que podem constituir sentenças, e a semântica determina os fatos no mundo aos quais as sentenças se referem. Uma vez que uma sentença é uma interpretação da semântica, ela descreve o mundo sendo desse ou daquele jeito. Assim, ela pode ser verdadeira ou falsa. Uma sentença é verdadeira se o evento ocorrido é correspondente à sua representação [2].

Em um processo no qual conclusões são buscadas, a inferência e o raciocínio são os pilares fundamentais. A inferência lógica cuida da implementação da relação entre as sentenças. E o raciocínio, partindo do princípio de que as sentenças são configurações físicas de partes do agente, deve ser um processo de construção de partes novas a partir das antigas.

O raciocínio deve garantir que as novas configurações representem fatos que derivem das representações anteriores. Não há limite para a complexidade das sentenças que a inferência lógica manuseia. A inferência formal consegue derivar conclusões válidas até quando o computador não sabe a interpretação usada. O computador apenas gera conclusões válidas, independentemente de sua interpretação. E elas terão significado, pois seguem as premissas básicas de linguagem.

2.4 – SISTEMAS MULTI-AGENTES

Tradicionalmente, os sistemas de IA são projetados para cuidar de problemas pequenos, com domínios simples contendo poucas variáveis. Isso porque, à medida que aumenta o tamanho e a complexidade do problema, normalmente aumenta o tamanho e a complexidade do programa. Além do mais, numa abordagem tradicional, é impossível se imaginar uma arquitetura aberta como a que se busca construir com a sociedade de agentes.

Cada agente é um programa particular, contendo entradas, saídas e protocolo de comunicação padronizados, interagindo continuamente entre si. Dessa forma, um grande problema é fragmentado em um conjunto de pequenos problemas, e cada pequeno problema é estudado por um agente ou por uma pequena sociedade de agentes. Cada agente analisa as mudanças percebidas no ambiente, e a decisão é tomada a partir das sugestões dadas por cada um deles, seguindo hierarquia pré-determinada.

Os agentes podem representar um funcionário ou chefe em uma repartição ou um operário em uma obra. O agente-operário conhece a obra e tem nela tarefas a desenvolver em momentos específicos de seu desenrolar. Essa tarefa depende dele e de outros funcionários, com os quais ele tem que se comunicar pedindo e fornecendo informações, sugestões e resultados. Ele precisa observar o que acontece a cada instante para poder opinar e agir, de

forma a contribuir para o cumprimento do objetivo geral, que é a conclusão da obra.

2.4.1 – COMUNICAÇÃO ENTRE AGENTES

A Comunicação entre os agentes é alvo de muitas pesquisas e as linguagens comuns não apresentam características específicas para este fim, o que dificulta de forma significativa os processos de desenvolvimento de SMA. De uma forma geral, pode-se afirmar que dois agentes podem comunicar-se se possuírem uma linguagem comum ou usarem linguagens inter-traduzíveis.

O processo de comunicação é baseado em protocolos, que são um conjunto de regras que normaliza e controla a troca de mensagens entre as unidades. Para interagir com uma sociedade, o Agente deve conhecer o protocolo utilizado por ela.

A comunicação entre os agentes pode ser feita de maneiras diversas. Desde que eles falem a mesma linguagem, podem falar diretamente um com o outro. E podem se comunicar também fazendo uso de uma estrutura chamada Facilitador. O facilitador atua como um intérprete na comunicação entre agentes. Desde que o agente saiba como se comunicar com o facilitador, e o facilitador possa falar com o outro agente, a estrutura é válida no processo de comunicação [14].

A linguagem de comunicação entre agentes tem dois níveis. O nível mais baixo é chamado de sintaxe, e representa a combinação de simbologia utilizada para compor as mensagens. O segundo nível é a semântica, que contém o significado da mensagem [2] [14].

Existem linguagens estruturadas para comunicação entre agentes, como ACL (Agent Communication Language) e sua derivação KQML (Knowledge Query and Manipulation Language), que são protocolos para trocar informações e conhecimento [15] [16].

Uma vez que um Sistema Multi-Agente possui características como processamento concorrente e unidades autônomas e independentes, uma mensagem deve conter informações sobre:

- Sua natureza, que a relaciona com o tipo de ação a ser tomada;
- Sua origem, que permite a identificação de quem enviou;
- Sua informação, que é o conteúdo [14].

2.4.2 – AMBIENTE

O ambiente modela o meio no qual o agente deve atuar. É um programa, assim como o agente, com premissas padronizadas de comunicação, e fornece ao agente dados referentes ao estado do meio. A atualização do ambiente é feita de acordo com a finalidade da aplicação, com referências a mudanças ocorridas com o passar do tempo e também com referências decorrentes da atuação dos agentes. De acordo com as características que possui, o ambiente pode ser [2]:

1 - **Acessível ou inacessível**: Quando os sensores detectam todas as alterações e aspectos do ambiente para escolher a ação a ser tomada, ele é dito acessível. Caso o ambiente não permita a detecção da seqüência completa, o agente não poderá agir com coerência. Portanto, o ambiente seria considerado inacessível.

2 - **Determinístico ou não determinístico:** Um ambiente no qual o próximo estado é determinado de forma completa e precisa pelo estado atual e pelas ações selecionadas pelo agente é dito determinístico. Mas à medida que o ambiente se torna complexo, ele tende a ser não determinístico. O número de dados de entrada tende a ficar maior à medida que o ambiente vai se tornando mais complexo, e a determinação do estado começa a perder informações. O ambiente passa a parecer não determinístico e inacessível, dificultando ao agente acompanhar o mundo. Mas as definições do agente devem prever as perdas de informações, e dessa forma, um ambiente definido rigorosamente como não determinístico, parece determinístico para ele. Assim, a definição mais satisfatória depende principalmente do ponto de vista do agente.

3 - **Episódico ou não episódico:** Episódico é o ambiente no qual a experiência do agente se divide em episódios. Cada

episódio é uma seqüência de percepção e a correspondente ação tomada. O novo estado vai depender da nova percepção, e não da ação anterior.

4 - **Estático ou dinâmico:** Se o ambiente pode mudar durante o processo de raciocínio do agente, ele é dinâmico. Se não pode, é estático. E se o ambiente não muda, mas a atuação do agente muda, o ambiente é dito semidinâmico. Os ambientes estáticos são mais simples porque o agente não precisa se ocupar do monitoramento do ambiente enquanto processa as informações da última percepção.

5 - **Discreto ou contínuo:** Se há um número limitado de percepções definidas de forma clara, e distintas, o ambiente é discreto. De forma contrária, o ambiente é contínuo.

Teoricamente, seria muito mais interessante, preciso e fácil trabalhar com ambientes acessíveis, determinísticos, episódicos, estáticos e discretos. Alcançar uma modelagem que represente essas características é, porém, algo muito difícil de se conseguir. As configurações dos sistemas modelados são, quase sempre, inacessíveis, não determinísticas, não episódicas, dinâmicas e contínuas. O que reflete em modelos com as mesmas características.

2.4.3 – APLICAÇÕES DE SISTEMAS MULTI-AGENTES

Os SMA podem ser aplicados em ambientes distribuídos, permitindo a resolução de problemas com o uso do conceito de comportamento cooperativo [13].

A utilização da tecnologia de orientação a objetos em conjunto com SMA permite a incorporação de características de modelagem distribuída e processos de encapsulamento, com o aproveitamento das propriedades de herança e agregação [17]. Objetos distribuídos que encapsulam núcleos com propriedades inteligentes podem ser considerados agentes inteligentes, desde que sigam os princípios gerais de SMA. Para isso é preciso que os núcleos tenham um dispositivo de monitoramento do ambiente, processem as informações recebidas acessando sua BC, efetuando processos de raciocínio e inferência, e que formulem novas ações para interagir com o meio [18].

No processo de construção de um SMA, o encapsulamento permite o uso de programas inteligentes e ferramentas matemáticas existentes, fornecendo a cada um desses softwares as características de agentes necessárias para a integração na sociedade [19].

Os SMA se encontram hoje atuando em diversas áreas de aplicação, tais como [13] [16] [17]:

- **Controle de Tráfego Aéreo**, como está sendo implantado no Aeroporto de Sydney, na Austrália;

- **Entretenimento**, aumentando o realismo de jogos e sistemas para a indústria de entrenimento em geral;

- **Aplicações para a Internet**, onde há aplicações diversas e em grande número, englobando sistemas para gerência de informação, sistemas de comércio eletrônico e sistemas de busca;

- **Simulação Social**, auxiliando estudos, através da representação de aspectos cognitivos e sociais de comunidades de pessoas;

- **Mercado Financeiro**, simulando comportamento de mercado e bolsas, auxiliando no estudo e compreensão da movimentação monetária para elaboração de estratégias de atuação;

- **Sistemas de Energia**, como em sistemas elétricos de potência, em unidades de geração ou subestações de energia, auxiliando no monitoramento e na melhoria de qualidade e continuidade de fornecimento.

2.5 – CONCLUSÕES

Este capítulo descreveu a estrutura dos SMA e os conceitos envolvidos em sua concepção.

Dentre as características da tecnologia de agentes, vale ressaltar as propriedades de topologia distribuída e de comportamento cooperativo, além da utilização de processamento concorrente, típica de ambientes multi-tarefas. Essas características conferem ao sistema a capacidade de segmentação das tarefas em partes menores, diminuindo a carga computacional e o custo de programação.

A arquitetura aberta insere flexibilidade ao sistema, permitindo a expansão da sociedade com a integração de novas unidades e a criação de novos níveis de ambiente.

3 – SUBESTAÇÕES ELÉTRICAS

3.1 – INTRODUÇÃO

São chamadas Subestações Elétricas (SE) as unidades componentes do Sistema Elétrico de Potência (SEP) responsáveis pela transformação, controle e distribuição de energia elétrica. As SE devem ter suas ações e comandos coordenadas a partir de programações e filosofias de proteção em conformidade com as informações coletadas pelos sistemas de medição e proteção. Além dos equipamentos principais, a SE possui equipamentos de manobra, medição e proteção.

A subestação deve oferecer segurança de serviço e operação a todas as partes componentes do SEP, além de garantir o fornecimento contínuo de energia elétrica de boa qualidade ao consumidor final. A restauração do abastecimento ou desligamento de partes defeituosas ou sob falta seguem rotinas de segurança e de restabelecimento baseadas na configuração física da planta, ou seja, pelo arranjo de barra e pela disposição dos equipamentos de manobra.

As ligações elétricas entre os componentes definem a topologia da subestação, e, juntamente com os equipamentos de manobra, definem suas características operativas. Cada arranjo possui suas vantagens e desvantagens, custos de construção e manutenção particulares, e a escolha do tipo utilizado é resultado das necessidades operativas do meio no qual a subestação estará trabalhando.

Os equipamentos de manobra de uma subestação são responsáveis pelo chaveamento entre seus componentes. Esses equipamentos são os disjuntores e chaves seccionadoras [20].

A importância desses equipamentos se evidencia também quando da necessidade de ações corretivas na situação de falta.

Os dispositivos de medida e proteção permitem a observação e o registro de eventos, fornecendo elementos necessários à operação do SEP, sendo de vital importância na análise de ocorrências e planejamento de restabelecimento do sistema, além de zelarem pela segurança dos equipamentos [21].

3.2 – EQUIPAMENTOS DE MANOBRA

3.2.1 - CHAVES

Chaves são dispositivos de manobra utilizados para funções diversas dentro de uma SE. De acordo com as funções que desempenham, podem ser classificadas em:

3.2.1.1 – Seccionadoras

- Podem operar somente quando houver uma variação de tensão insignificante entre seus terminais, ou nos casos de restabelecimento ou interrupção de correntes insignificantes;
- Contornar equipamentos , como disjuntores e capacitores série, para a execução de manutenção ou por necessidade operativa;
- Manobrar circuitos entre os barramentos de uma subestação;
- Isolar equipamentos, como disjuntores, barramentos, transformadores, reatores, geradores ou linhas, para a execução de manutenção.

3.2.1.2 – Chaves de operação em carga

- Abrir e/ou fechar determinados circuitos em carga;
- Manobrar bancos de reatores e de capacitores.

3.2.1.3 – Chaves de aterramento rápido

- Aterrar determinados componentes energizados, normalmente com o objetivo de provocar uma falta intencional na rede, de forma a sensibilizar os sistemas de proteção;
- Necessitam de tempos de fechamento extremamente rápidos, exigindo quase sempre acionamento por explosivos [20] [22].

3.2.1.4 – Chaves de terra

- Aterrar componentes do sistema em manutenção;
- Aterrar linhas de transmissão, barramentos ou bancos de capacitores em derivação;
- Não é necessariamente prevista para conduzir correntes em condições normais do circuito.

3.2.2 – DISJUNTORES

Os disjuntores são os elementos ativos dos instrumentos de manobra, por atuarem na presença de carga. Eles atuam na interrupção de correntes de falta rapidamente, efetuando uma minimização nos danos causados aos equipamentos na ocasião de ocorrência de curto-circuitos. Os disjuntores atuam também na interrupção de correntes de magnetização de transformadores e reatores, de correntes normais de carga e de correntes capacitivas de bancos de capacitores e de linhas em vazio.

Os disjuntores são, em geral, chamados a mudar de uma condição para outra ocasionalmente, e devem ser capazes de prover o fechamento de circuitos elétricos durante condições normais de carga e durante a ocorrência de curto-circuitos. Além disso, as funções mais freqüentemente desempenhadas pelos disjuntores são a condução de correntes de carga na posição fechada, e o isolamento entre duas partes de um sistema elétrico [20] [22].

3.3 – SISTEMA DE MEDIÇÃO

A aplicação de equipamentos de medição em sistemas elétricos tem como objetivo permitir a observação e o registro das grandezas elétricas e não-elétricas, fornecendo elementos necessários à operação do SEP. E

ainda, possibilita a atuação preventiva e corretiva a fim de garantir o fornecimento de energia aos consumidores com a qualidade adequada [23].

As medidas elétricas em uma subestação devem, portanto, existir em qualidade e quantidade suficientes às necessidades da subestação a ser supervisionada, viabilizando o acesso aos dados necessários à sua operação e à correção de suas falhas.

Apesar do sistema de medição supervisionar várias grandezas, as principais grandezas observadas para a operação de uma subestação são tensão, corrente, fator de potência e freqüência.

1 – Funções da medição de tensão:

- Permitir a operação de equipamentos e instalações;
- Observar e respeitar as restrições impostas pelos equipamentos da subestação em relação ao isolamento;
- Fornecer dados para o ajuste e atuação dos equipamentos de controle de tensão e compensação de reativo.

2 – Funções da medição de corrente:

- Permitir o controle do carregamento de equipamentos, de acordo com suas restrições operativas;
- Permitir a operação de equipamentos e instalações.

3 – Funções da medição de fator de potência:

- Avaliar as condições operativas do SEP;
- Identificar a necessidade de compensação de potência reativa;
- Permitir o gerenciamento da carga.

4 – Funções da medição de freqüência:

- Permitir o restabelecimento de partes do SEP desligadas pelo sistema de alívio de cargas;
- Fornecer dados para o ajuste e atuação dos equipamentos.

Os medidores e relés de proteção das subestações são atuados por tensões e correntes supridas por transformadores de potencial (TP) e de corrente (TC). A função dos TP e TC é transformar as correntes e tensões do sistema de potência em magnitudes menores, e fornecer isolação galvânica entre o sistema de potência e os relés de proteção e instrumentos de medição.

3.4 - SISTEMA DE PROTEÇÃO

O sistema de proteção visa detectar e minimizar os efeitos de perturbações e anomalias no comportamento do SEP, gerando maior qualidade e continuidade no serviço de abastecimento, além de proteger os equipamentos da área afetada. Os aspectos mais importantes para análise de proteção de um sistema elétrico são:

- Operação normal do sistema;
- Prevenção contra falhas elétricas;
- Limitação dos defeitos devidos às falhas.

A proteção por meio de relés vem contribuir com o terceiro aspecto, e tem como função principal promover uma rápida retirada de serviço de um elemento do sistema quando esse sofre um curto-circuito, ou quando ele começa a operar de modo anormal que possa causar danos ou, de outro modo, interferir com a correta operação do resto do sistema. E ainda, tem como função secundária promover a indicação da localização e do tipo de defeito.

As principais características funcionais dos relés de proteção são:

- **Velocidade:** rapidez de atuação que garante a diminuição da extensão do dano ocorrido e auxilia na manutenção da estabilidade do SEP, assegurando a manutenção das condições normais de operação nas partes sãs do sistema;
- **Confiabilidade:** probabilidade do sistema de proteção satisfazer a função prevista, após longa inatividade, seguida de operação em condições difíceis, o que exige do equipamento de proteção simplicidade e robustez;
- **Sensibilidade:** capacidade de responder às anormalidades nas condições de operação e aos curtos-circuitos para os quais foi projetado;

- **Seletividade:** habilidade em reconhecer e selecionar entre as várias condições, aquelas para as quais uma operação imediata é requerida, e aquelas para as quais nenhuma operação ou um retardo de atuação é exigido [24] [25].

Os relés de proteção são constituídos por um elemento sensor, um elemento comparador e um elemento de controle. São dispositivos através dos quais um equipamento elétrico é operado quando se produzem variações nas condições deste equipamento ou do circuito em que ele está ligado, ou em outro equipamento ou circuito associado.

Esses relés supervisionam constantemente grandezas dos sistemas elétricos, tais como tensão, corrente, freqüência, etc., e grandezas inerentes aos próprios equipamentos, temperatura, por exemplo. Eles monitoram estas grandezas do SEP através de transformadores de corrente (TC) e de potencial (TP).

Neste estudo, o foco é direcionado aos relés de proteção que acionam disjuntores, direta ou indiretamente. Na Tabela 3.1 são apresentados os principais relés encontrados em subestações.

Tabela 3.1 - Principais Relés de Proteção Encontrados em Subestações

ASA	DESCRIÇÃO
21	Relé de distância
25	Relé de conferência de sincronismo
26	Relé de temperatura do óleo
27	Relé de subtensão
32	Relé direcional de potência
49	Relé de temperatura do enrolamento
50	Relé de sobrecorrente instantâneo
51	Relé de sobrecorrente temporizado
59	Relé de sobretensão
62	Relé de interrupção ou abertura temporizada
63	Relé de gás
64	Relé de proteção de terra
66	Relé de intercalaçãp
67	Relé direcional de sobrecorrente CA
74	Relé de alarme
78	Relé de proteção contra falta de sincronismo
79	Relé de religamento CA
81	Relé de freqüência
83	Relé de seleção de controle ou transferência automática
87	Relé de proteção diferencial

3.4.1 - PROTEÇÃO DAS LINHAS

Para as linhas são usados basicamente os relés de proteção de sobrecorrente e de distância, e realizadas medições de correntes e tensões nas 3 fases, as potências ativa e reativa trifásicas e a energia ativa trifásica, Figura 3.1.

Além dos defeitos causados por curtos-circuitos, os defeitos mais comuns nas linhas são devidos às sobrecargas. A proteção contra sobrecarga possibilita a utilização da linha em sua capacidade máxima, evitando danos causados por aquecimento excessivo. Essa proteção gera um sinal para que sejam tomadas medidas quando a temperatura máxima admissível for atingida, evitando o desligamento.

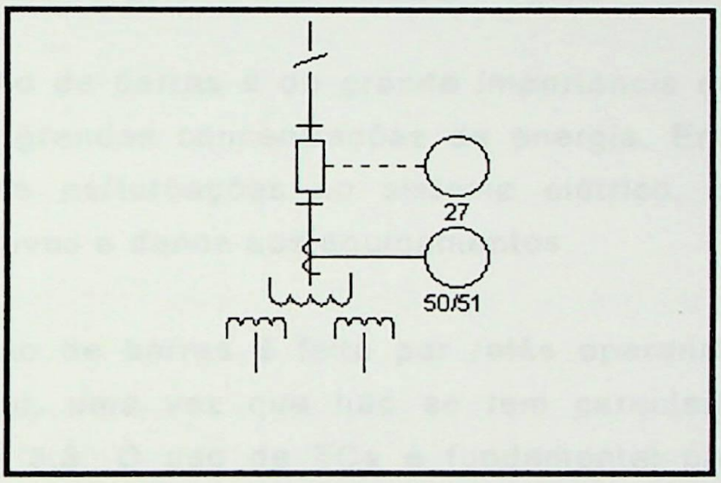


Figura 3.1 – Proteção de Linha da Subestação Modelo

3.4.2 - PROTEÇÃO DE TRANSFORMADORES

Nos transformadores são utilizadas basicamente as proteções de Buchholz, diferencial, sobrecorrente e térmica, Figura 3.2.

Pelo lado primário, usualmente são medidas as três correntes. Pelo lado secundário, as medições usuais são as três correntes e tensões, as potências ativa e reativa, e a energia ativa para faturamento.

As principais fontes de defeito nos isolamentos do transformador são as sobretensões de origem atmosférica e o aquecimento dos enrolamentos devido a sobrecargas permanentes ou temporárias repetitivas.

Esses fenômenos desgastam o isolante dos enrolamentos, diminuindo sua vida útil.

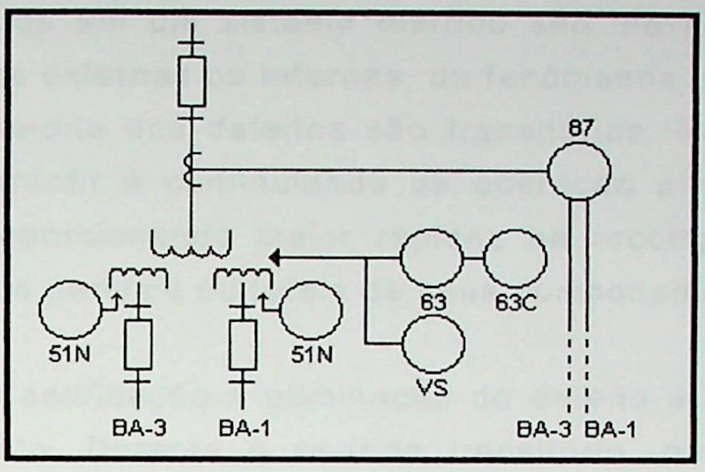


Figura 3.2 – Proteção de Transformadores da Subestação Modelo

3.4.3 - PROTEÇÃO DE BARRAS

A proteção de barras é de grande importância devido à presença, nesses locais, de grandes concentrações de energia. Em caso de defeito, pode haver graves perturbações no sistema elétrico, o que acarretaria prejuízos significativos e danos aos equipamentos.

A proteção de barras é feita por relés operando no princípio de corrente diferencial, uma vez que não se tem características de defeito peculiares, Figura 3.3. O uso de TCs é fundamental para a proteção de barras, uma vez que a proteção diferencial depende da soma das correntes no secundário do TC ser zero quando a soma das correntes primárias entrando e saindo da barra é nula.

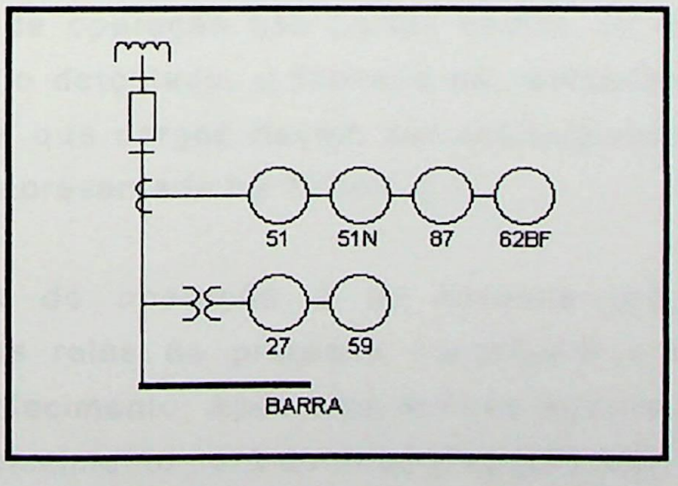


Figura 3.3 – Proteção de Barras da Subestação Modelo

3.5 – RESTABELECIMENTO

Os defeitos em um sistema elétrico são inevitáveis, podendo ser oriundos de causas externas ou internas, de fenômenos elétricos, ambientais ou humanos. A maioria dos defeitos são transitórios. Procura-se minimizar seus efeitos e garantir a continuidade da operação através de técnicas e equipamentos, proporcionando maior rapidez na recomposição do sistema após desligamentos parciais ou totais de seus componentes.

Após a identificação e eliminação do defeito é iniciado o processo de restabelecimento. Durante o período transitório, os relés de proteção percebem alguma anormalidade nas características básicas de funcionamento do sistema e acionam os equipamentos de manobra para, seletivamente, isolarem o componente defeituoso ou envolvido diretamente no defeito.

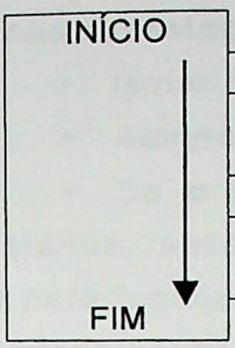
Em situação de defeito transitório, uma vez decorrido o tempo típico de eliminação ou extinção da falta, inicia-se a reenergização seqüencial dos componentes pelo sistema de restabelecimento.

Em caso de defeito permanente, as chaves seccionadoras devem ser manobradas de forma a isolar a parcela do sistema que realmente contém o defeito, mantendo-a desenergizada aguardando a manutenção. As partes sadias, inicialmente desligadas devido às restrições impostas pelo sistema de proteção, devem ser reenergizadas.

A velocidade da atuação dos relés de proteção é importante para diminuir a extensão dos danos ocorridos, assegurando a manutenção das condições normais de operação nas partes sadias do sistema elétrico. De acordo com o defeito detectado, a filosofia de restabelecimento define que partes do sistema e que cargas devem ser reenergizadas. A hierarquia de restabelecimento é apresentada na Tabela 3.2.

A filosofia de operação e os estados pré e pós falta dos equipamentos e dos relés de proteção constituem a base de dados do processo de restabelecimento. Após uma análise desses dados, a seqüência de ações de restabelecimento será definida [26] [27] [28].

Tabela 3.2 - Hierarquia de Restabelecimento



Fontes	LT
Transformação	TRANSFORMADORES
	AUTOTRANSFORMADORES
Carga	LT, ALIMENTADORES
Controle	BANCO DE CAPACITORES
	REATORES E SÍNCRONOS

3.5.1 - LINHAS DE TRANSMISSÃO E ALIMENTADORES DE DISTRIBUIÇÃO

O sistema de restabelecimento analisa a seqüência de eventos ocorridos e a atuação das proteções para planejar as medidas a serem tomadas no caso de desligamentos forçados de Linhas de Transmissão (LT) e de Alimentadores de Distribuição. Esta análise deve ser estruturada de forma a estabelecer uma resposta contendo uma seqüência de manobras, que possua um elevado grau de confiança na eficiência de sua aplicação [29].

Durante a execução dos testes de religamento previstos, as variações das grandezas de interesse (V, I, f e θ) nos circuitos e nas barras da subestação devem ser monitoradas, visando identificar e localizar o defeito.

Para os alimentadores, somente deverá ser permitida a seqüência de religamentos automáticos típica de seus religadores.

Caso se caracterize um defeito permanente pela análise das proteções operadas e das grandezas monitoradas, após uma tentativa insatisfatória de restabelecimento devem ser acionadas as equipes de manutenção.

Havendo a constatação de falta de fase na linha, pela falta de corrente ou de tensão em uma das fases, deve-se adotar os seguintes procedimentos:

- Certificar-se que o defeito é interno à linha, verificando o valor da tensão nas três fases nos terminais da LT;

- Identificar o trecho defeituoso, verificando a tensão nas três fases nas subestações intermediárias, quando for o caso;
- Isolar o trecho defeituoso;
- Acionar a equipe de manutenção;
- Se o religamento da linha representar risco para terceiros e proprietários, essa deverá ser desligada imediatamente. Nesse caso, os testes para localização de defeito devem ser executados após a equipe de manutenção verificar a ausência de anormalidades nos trechos da linha que atravessam áreas urbanas.

No restabelecimento de LT de sistemas em anel, tem-se sempre presente a possível necessidade de se verificar o sincronismo em um dos terminais. No caso de LT radiais, nas quais, em condições normais de operação, só pode haver fluxo de energia em um único sentido, somente é necessário supervisionar a drenagem de carga capacitiva em linhas longas ou paralelas.

Se houver outra LT energizada com o terminal remoto aberto ou a vazio ou se não existirem as condições mínimas de pré-energização, a LT não deve ser energizada. O sentido de energização pode depender da natureza da ocorrência ou possuir sentido único [16]. Devem ser observadas as seguintes condições de pré-energização:

- Tensão máxima de pré-energização no barramento;
- Garantia da estabilidade do SEP, através de restrições referentes aos equipamentos e preservação das LT em serviço.

3.5.1.1 - Critérios de Restabelecimento

- Antes de se energizar uma linha deve-se verificar se não há tensão de retorno;
- É indispensável a verificação das condições de paralelo ou anel em toda a operação de fechamento de disjuntor;
- Desarmado o disjuntor de uma linha em apenas um de seus terminais, esse poderá ser fechado imediatamente se as condições de anel ou paralelo permitirem;
- Desarmados ambos os terminais de uma linha, devem ser observados os sentidos de energização estabelecidos pelas instruções operativas da malha regional a que pertença;

- Na execução dos testes de energização das linhas, o comportamento dos amperímetros e voltímetros de linha deve ser observado;
- Sempre que necessário, o disjuntor de transferência pode e deve ser utilizado para execução de teste de restabelecimento;
- Durante a energização de uma determinada linha, deve-se aguardar o fechamento do seu terminal remoto e uma tomada de carga, para depois se iniciar o processo de energização de outra linha [29].

3.5.2 - TRANSFORMADORES E AUTOTRANSFORMADORES

O restabelecimento de transformadores e/ou autotransformadores só deverá ser feito se não houver a possibilidade da presença de defeito interno.

Quando na subestação houver qualquer perturbação causada por defeito interno em transformadores e/ou autotransformadores, o restabelecimento deste equipamento deverá ser feito manualmente, de acordo com instruções operativas específicas da subestação, após terem sido feitos os testes pertinentes [30] [31] [32] [33].

3.5.3 - EQUIPAMENTOS DE CONTROLE DE TENSÃO

O sistema elétrico deverá operar, após o restabelecimento oriundo de uma contingência simples, considerando apenas a atuação dos reguladores de tensão, LTC automáticos ou de outros equipamentos de controle de tensão, dentro dos seguintes limites com relação a sua tensão nominal (V_n):

- Tensão mínima: 90% V_n
- Tensão máxima: 110% V_n

A tensão deverá ser restabelecida aos valores normais através de ações sobre controladores disponíveis como tensão de excitação de unidades geradoras ou compensadores síncronos, taps de transformadores e manobra em equipamentos de compensação de potência reativa, à medida que a carga for sendo retomada.

Assim, os equipamentos de compensação de potência reativa tais como banco de capacitores, reatores e síncronos, caso existam, não deverão ser chaveados durante o processo de restabelecimento [30] [31] [32] [33].

3.5.4 – CARGAS

As cargas devem ser restabelecidas preferencialmente em blocos iguais ou menores a 3% da carga total dentro de uma seqüência definida e de forma rápida, na medida que não haja variações bruscas de freqüência e tensão [34].

O início da restauração da carga deverá ocorrer quando a freqüência estiver acima da faixa de 59,7 a 60 Hz, durante um tempo de retardo inicial.

A temporização da restauração da carga em blocos graduais permite que o sistema de restabelecimento tenha condições de distinguir se o estado operativo do sistema elétrico é estável ou oscilatório sob as novas condições operativas, permitindo um equilíbrio estável entre geração e carga nos estágios intermediários de restabelecimento. A definição de valores de temporização parte da função da taxa de recuperação de freqüência ou da adoção de um valor de tempo pré-definido a partir da experiência operativa.

A queda de freqüência associada a cada bloco ligado não deverá exceder a 0,25 Hz sob a pior condição operativa, e ainda, a restauração deve ser interrompida quando a freqüência cair abaixo de 59 Hz [30] [31] [32] [33].

3.6 - FILOSOFIA DE RESTABELECIMENTO DE SUBESTAÇÕES

3.6.1 - BASES FILOSÓFICAS

O objetivo de um sistema de restabelecimento é recompor a subestação de forma eficiente e rápida, reintegrando-a ao sistema elétrico o mais próximo possível da configuração em que se encontrava antes da perturbação, excluindo apenas os componentes defeituosos ou afetados por faltas permanentes, e aqueles que poderiam levar novamente a subestação, ou o sistema elétrico, a uma nova contingência.

As influências nas condições operativas do sistema elétrico ao longo do processo de recomposição da subestação são avaliadas a cada

manobra de restabelecimento de forma a validar o que já foi feito e liberar a execução das ações subseqüentes.

3.6.2 - HIERARQUIA E PRIORIDADES DE RESTABELECIMENTO

3.6.2.1 - Falta de Tensão

Verificada a falta de tensão na subestação, por um tempo superior a um valor pré-fixado, o sistema de restabelecimento inicia a abertura de todos os disjuntores para que nenhum equipamento seja ou venha a ser energizado diretamente das subestações remotas. Caso a subestação possua o relé de subtensão de barra (27) essa ação já é automática.

3.6.2.2 - Atuação de Proteção

A reenergização de componentes após a atuação da proteção deve ser criteriosa e fundamentada em análise dos relés de proteção que atuaram.

A principal função de um sistema de proteção é detectar uma perturbação da grandeza monitorada, que caracterize um comportamento anormal do sistema elétrico ou que venha a comprometer a vida útil do equipamento. Baseado nessa detecção, ele deve comandar a abertura do disjuntor de modo a interromper a anormalidade e isolar o equipamento ou a parte do sistema elétrico afetado pela anomalia, impedindo que a perturbação danifique equipamentos, comprometa a operação ou propague-se para outras partes que não tenham sido atingidas até então.

A função secundária é promover uma indicação da localização e do tipo do defeito, sinalizando ao operador o maior número de características presentes na condição anormal detectada, de forma a permitir uma eficiente ação de restabelecimento através da análise de suas sinalizações.

Assim, pela análise dos relés de proteção que atuaram, pode-se identificar o agente da ocorrência, caracterizar a falta (quanto ao tipo e fases envolvidas) e a sua localização, e em seguida, estabelecer ou eliminar prioridades no processo de restabelecimento.

A seguir, são apresentadas as orientações básicas de restabelecimento associadas a atuação de cada um dos principais relés de proteção presentes em uma subestação [30] [31] [32] [33]:

1 - Por atuação do relé de subtensão (27) da linha de transmissão

- Sinalizar e permitir o religamento, no caso do terminal ser seguidor;
- Sinalizar e bloquear o religamento, no caso do terminal ser iniciador.

2 - Por atuação do relé de subtensão (27) da barra

- Abrir ou verificar aberto os disjuntores dos equipamentos de controle de tensão, ou acionar seus bloqueios no caso de um colapso total de tensão na subestação.

3 - Por atuação do relé de sobrecorrente instantâneo (50) da linha de transmissão

- Fechar o disjuntor monitorando os amperímetros de linha;
- Caso o restabelecimento seja não satisfatório e os amperímetros de linha tenham atingido fim de escala, isolar a linha de transmissão.

4 - Por atuação do relé de sobrecorrente temporizado (51) da linha de transmissão

- Fechar o disjuntor monitorando os amperímetros de linha;
- Caso o restabelecimento seja não satisfatório e os amperímetros de linha tenham atingido fim de escala, isolar a linha de transmissão.

5 - Por atuação do relé de sobretensão (59)

- **Se atuou o primeiro estágio:**
- Acionar um alarme;

- Monitorar as correntes no referido equipamento;

- **Se atuou o segundo estágio:**
- Abrir ou verificar aberto os disjuntores dos equipamentos de controle de tensão;
- Bloquear o religamento dos disjuntores dos equipamentos de controle de tensão.

6 - Por atuação do relé de tempo (62)

- Normalmente o relé de tempo é aplicado agregando a temporização a outra função, vale nesse caso, a orientação básica de restabelecimento da função principal.

7 - Por atuação do relé de gás (63)

- **Se atuou o primeiro estágio:**
- Acionar um alarme;
- Monitorar as correntes no referido equipamento;

- **Se atuou o segundo estágio:**
- Abrir ou verificar aberto os disjuntores da alta e da baixa;
- Abrir as chaves seccionadoras da alta e da baixa, promovendo o completo isolamento do transformador;
- Abrir a chave de aterramento, caso exista;
- Bloquear o religamento dos disjuntores da alta e da baixa.

8 - Por atuação do relé de religamento (79)

- Prover o religamento do disjuntor aberto por alguma variação temporária do sistema. Após a abertura do disjuntor, o relé tenta por três vezes fecha-lo.

9 - Por atuação do relé diferencial (87)

- Abrir ou verificar aberto os disjuntores da alta (138kV) e da baixa (13,8kV);

- Abrir as chaves seccionadoras da alta e da baixa, promovendo o completo isolamento do transformador;
- Abrir a chave de aterramento, caso exista.

3.6.2.3 - Falha de Disjuntor

Ocorrendo falha de algum disjuntor durante a abertura automática do mesmo, o sistema de restabelecimento bloqueia todos os outros ligados ao barramento do disjuntor defeituoso. Em seguida, passa a rotina que verifica se uma nova configuração pode ou não ser realizada, em função das restrições operativas existentes e face à nova restrição imposta pela falha do disjuntor. Posteriormente, decide as ações de recomposição da subestação ou o fim do restabelecimento automático.

No caso de defeito no disjuntor e o mesmo bloquear fechado, o isolamento poderá ser feito através da abertura das chaves seccionadoras, independentemente do tipo de defeito no mesmo, desde que a corrente passante seja suficientemente pequena [30] [31] [32] [33].

3.6.3 - RESTABELECIMENTO DA SE

No restabelecimento, a segurança precede a agilidade. Portanto, o fechamento de qualquer disjuntor somente poderá ser efetivado após a análise conclusiva da perturbação, efetuada em função das proteções operadas e valores das grandezas medidas, e sendo identificada a falta e suas características [34].

Assim sendo, isolado o componente defeituoso nas faltas permanentes, ou eliminada a falta transitória, as condições estipuladas de pré-energização dos equipamentos deverão ser rigorosamente observadas.

Caso a análise da perturbação não permita uma conclusão, devem ser usadas regras heurísticas, em rotina paralela de restabelecimento, na qual quaisquer dificuldades devem interromper e bloquear o processo.

Os sentidos de energização das linhas de transmissão devem ser observados. Se houver sobrecarga inadmissível em equipamentos o restabelecimento deve ser interrompido. Em caso de perda de equipamentos na subestação deve-se adotar os seguintes procedimentos:

- Se a perda ocorrer simultaneamente com a perturbação total, o equipamento deverá ser isolado de imediato;
- Se a perda ocorrer durante o restabelecimento da subestação, o equipamento deverá ser isolado após a recomposição das linhas.

3.6.3.1 - Após um Colapso Total

Uma perturbação total é caracterizada pela falta de tensão na subestação, normalmente causada por desarme dos disjuntores ou falta de tensão nas linhas de transmissão fonte (LT-fonte). A seguir, é apresentada a seqüência de restabelecimento, no caso de desligamento das linhas de transmissão fonte [34].

- **Preparação da SE:**
- Abrir ou verificar abertos todos os disjuntores das linhas de alimentação (LT-fonte), dos transformadores e dos equipamentos de controle de tensão;
- Caso algum disjuntor não aceite o comando de abertura, abrir as chaves seccionadoras do mesmo, promovendo o seu isolamento e acionando um sinal de alarme;
- Fechar ou verificar fechado os disjuntores ou chaves seccionadoras dos circuitos ou linhas de carga;
- Abrir ou isolar as chaves de aterramento rápido que foram acionadas durante o processo de interrupção da contingência, caso existam.

- **Restabelecimento:**
- Aguardar tensão por uma LT-fonte;
- Verificar a chegada de tensão pelo TP de linha de uma LT-fonte;
- Fechar o disjuntor de alimentação dos transformadores;
- Restabelecer as cargas dos transformadores, através da energização dos alimentadores das redes de distribuição local (RDU e RDR);
- Se for necessário, energizar os bancos de capacitores;
- Aguardar ou enviar tensão pelas demais LT-fonte;
- Fechar seus respectivos disjuntores em anel, verificando as condições de sincronismo, ou aguardar o fechamento dos disjuntores do terminal remoto;

- Energizar as LT-carga;
- Monitorar a corrente durante a reenergização de um equipamento que tiver seu disjuntor aberto por acionamento do relé de proteção.

Sendo satisfatória a seqüência de restabelecimento o defeito foi transitório. Caso contrário, o defeito é permanente e deve ser localizado. Para isso, deve-se abrir seccionadoras e/ou disjuntores do circuito indicado pela análise das proteções e realizar teste de energização. Se o teste de energização for satisfatório, então o defeito foi identificado e está no circuito isolado, caso contrário o defeito deve ser procurado em outro circuito, considerando-se as probabilidades associadas aos valores pré e pós falta, e aos monitorados durante o restabelecimento [34].

O restabelecimento das cargas deverá ser feito dentro de uma seqüência ótima e atendendo aos critérios pós falta do esquema de alívio de carga, caso exista.

3.6.3.2 - Após uma Perturbação no Sistema que envolva a Subestação

Uma perturbação parcial é uma ocorrência em que não há perda total de tensão na subestação, e que a eliminação da contingência envolve apenas alguns dos seus componentes. Apesar da filosofia de restabelecimento ser a mesma, a sua seqüência é variável e dependente do arranjo remanescente após o término da ocorrência (pós falta), e restabelecido o regime permanente [34].

Para esses casos vale o seguinte roteiro de restabelecimento:

- **Desarmando um disjuntor da alta (138kV)**
- Verificar se o terminal é seguidor, iniciador ou iniciador/seguidor, pelos sentidos de energização das linhas e as características da perturbação;
- Se o terminal for iniciador/seguidor, havendo ou não tensão de retorno, fechar o disjuntor enviando tensão se iniciador, ou fechar o disjuntor em anel se seguidor;

- Se o terminal for iniciador, havendo tensão de retorno, enviar sinal de comando de abertura do disjuntor do terminal remoto e aguardar a abertura do disjuntor do terminal remoto, não tendo tensão de retorno fechar o disjuntor enviando tensão;
- Se o terminal for seguidor, havendo tensão de retorno, fechar o disjuntor em anel, caso contrário enviar sinal de comando de fechamento do disjuntor do terminal remoto, aguardar o fechamento do disjuntor do terminal remoto e recebendo tensão de retorno, fechar o disjuntor em anel ou paralelo.

- **Desarmando um disjuntor de um transformador ou de um equipamento de controle de tensão**
- Seguir orientação estabelecida nos itens 3.5.2 e 3.6.3. No caso de caracterização ou de suspeita de um defeito interno, o restabelecimento deve ser bloqueado, e a equipe de manutenção deve ser acionada.

- **Desarmando um disjuntor (religador) da baixa (13,8kV)**
- Iniciar o ciclo normal de religamento automático do alimentador, normalmente nenhuma ação global é tomada associada ao restabelecimento de um alimentador.

3.6.4 - INFLUÊNCIA DAS CONDIÇÕES OPERATIVAS DO SISTEMA

Ao longo de todo o processo de restabelecimento, o sistema de restabelecimento deverá monitorar as variações das grandezas características de interesse [34] [35]:

- Tensão da barra e nos circuitos (V_n);
- Corrente de fase e residual nos circuitos (I_i);
- Freqüência da barra (f_n);
- Ângulo entre a tensão e a corrente (θn_i).

Tal acompanhamento permite identificar qualquer possibilidade de defeito ou oscilação que possa vir a comprometer o restabelecimento. Os valores amostrados podem validar uma etapa, interromper o processo ou mesmo modificar as ações e a seqüência de restabelecimento [34].

3.7 – SUBESTAÇÃO MODELO

Como implementação prática, é utilizada nos testes do programa de decisão uma subestação de distribuição, apresentando dois bays de alimentação por circuitos independentes de 138 kV. Os bays apresentam, cada um, um transformador com tap de 3 enrolamentos 138/13,8/13,8, delta/estrela aterrado, com potência de 40/60 MVA, Figura 3.4.

Esses transformadores (TR-1 e TR-2) possuem como dispositivos de proteção relés de proteção de tensão e/ou de gás (63), relé diferencial (87) e relé de sobrecorrente temporizado no neutro (51N).

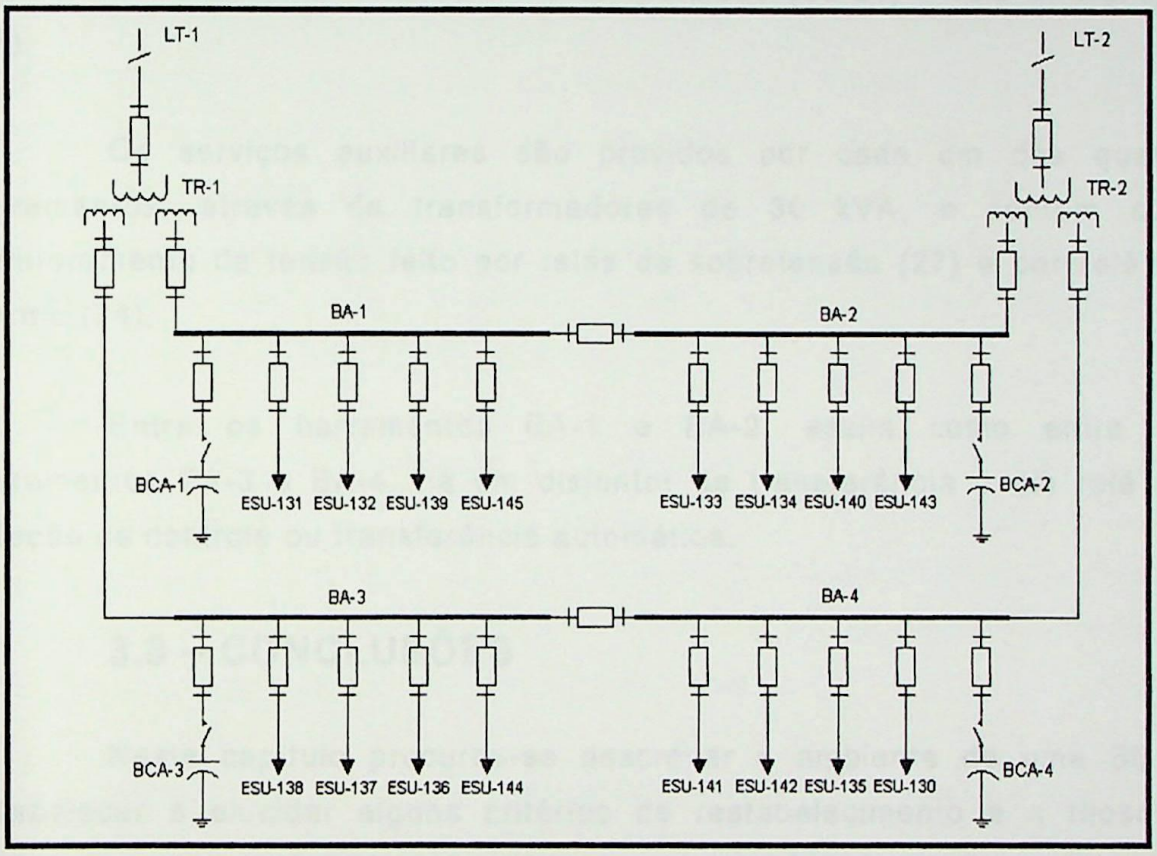


Figura 3.4 – Subestação Modelo

Cada transformador alimenta, através do secundário e do terciário, um barramento independente. Dessa forma, a subestação apresenta quatro barramentos de baixa, sendo eles BA-1, BA-2, BA-3 e BA-4.

Cada um desses barramentos apresenta como instrumentos de medição 3 voltímetros, 3 amperímetros e 3 medidores de Wh digitais. As proteções encontradas nos barramentos são os relés de subtensão (27), sobretensão (59), diferencial (87), relé de sobrecorrente temporizado (51) e relé de sobrecorrente temporizado no neutro (51N).

Cada barramento de baixa apresenta acoplados quatro alimentadores e um banco de capacitores, sendo estes BCA-1, BCA-2, BCA-3 e BCA-4. Os bancos apresentam como instrumentos de medidas 3 amperímetros cada, e como proteção estão associados relés sobrecorrente instantâneo e temporizado (50/51).

Os Alimentadores são 16 no total, de nomes ESU-130, ESU-131, até ESU 145. Cada alimentador possui como dispositivos de proteção relés de religamento (79) e de sobrecorrente instantâneo e temporizado (50/51).

As linhas que alimentam a subestação são LT -1 e LT -2, possuindo relés de sobrecorrente instantâneo e temporizado (50/51) e relé diferencial (87).

Os serviços auxiliares são providos por cada um dos quatro barramentos, através de transformadores de 30 kVA, e contam com monitoramento de tensão feito por relés de sobretensão (27) e por relé de alarme (74).

Entre os barramentos BA-1 e BA-2, assim como entre os barramentos BA-3 e BA-4, há um disjuntor de transferência e um relé de seleção de controle ou transferência automática.

3.8 – CONCLUSÕES

Neste capítulo procurou-se descrever o ambiente de uma SE e estabelecer e elucidar alguns critérios de restabelecimento e a filosofia básica de restabelecimento automático de subestações. Um sistema de apoio à decisão para ajudar na tarefa de restabelecimento deve possuir as características descritas para ser aplicável no auxílio à operação da subestação.

4 – MODELAGEM ORIENTADA A OBJETO

4.1 – INTRODUÇÃO

A Programação Orientada a Objetos (POO) é uma forma de programação que busca modelar as características físicas da topologia do sistema no mundo real. Uma vez que os ambientes do mundo real são na maioria naturalmente distribuídos, a modelagem desse ambiente deve, conseqüentemente, ser distribuída. No paradigma de orientação a objetos, o mundo é visto como uma coleção de objetos interagindo com outros para alcançar um comportamento significativo [36].

O método de orientação a objeto é, dessa forma, um método que ajuda programadores a produzir sistemas cujas estruturas são simples e descentralizadas. Além disso, a filosofia de programação orientada a objetos possibilita a construção de programas que ofereçam ao usuário benefícios como rapidez, segurança e facilidade de manuseio.

Esses benefícios estão relacionados com o conceito de qualidade de software, que se preocupa tanto com o funcionamento do programa como com a facilidade de programação. As características de um software de qualidade que são apresentadas por um sistema orientado a objeto são [37]:

- Habilidade para cumprir as tarefas de forma correta, como estiverem definidas em suas especificações;
- Habilidade para reagir de forma apropriada quando se encontrar sob condições anormais;
- Facilidade de adaptação a mudanças de especificação;
- Capacidade de reaproveitamento para aplicações em diversos processos de desenvolvimento;

- Compatibilidade, que é a facilidade de combinar elementos de software com outros, e que requer padronização de formato de arquivos, de estrutura de dados e de interface;
- Eficiência, que é a habilidade de gerenciamento dos recursos do sistema, como os dispositivos de comunicação e o espaço ocupado na memória;
- Facilidade de transferência de produtos de software para vários ambientes de software e hardware;
- Facilidade de uso, para pessoas de diversas formações e capacidades terem acesso ao uso do software para resolverem problemas;
- Habilidade para realizar testes e detectar falhas;
- Habilidade de proteger seus componentes, como programas e dados, contra acesso não autorizado e modificações.

Um modelo orientado apresenta propriedades distribuídas e comportamento cooperativo dinâmico. É constituído de um conjunto de métodos de modelagem e implementação através dos quais um grupo de softwares distintos se organiza como uma coleção cooperativa de objetos, incorporando ao mesmo tempo a estrutura e o comportamento dos dados.

A POO enfatiza a preocupação com a estrutura dos objetos em si, e não com seus detalhes de utilização. O uso de um objeto depende dos detalhes da aplicação, e a aplicação muda com o decorrer do tempo. Enquanto os requisitos mudam, as características do objeto permanecem mais estáveis que o modo como ele é utilizado, caracterizando um comportamento estável.

As características de qualidade de software desejadas são alcançadas, no modelo de orientação a objeto, graças a dois princípios:

1. **Continuidade:** Esse princípio diz respeito à padronização de modelos. A POO possui um documento chamado Diagrama de Classes, que constitui um padrão a ser utilizado pela equipe de desenvolvimento, mostrando um conjunto de elementos declarativos de modelo, como classes, tipos, conteúdos e relações.
2. **Reusabilidade:** Esse princípio determina a possibilidade de reaproveitamento de classes prontas do próprio sistema, ou

mesmo de classes definidas em outro sistema. Em casos onde a classe existente não reflita de maneira completa as necessidades do ambiente, novas classes podem ser criadas a partir das já existentes, através de processos de herança, especialização e generalização. A reusabilidade permite economia de tempo no desenvolvimento do software, pois componentes construídos e testados serão reaproveitados, garantindo confiabilidade e eficiência ao novo sistema. A construção de softwares reusáveis é um meio de preservar o know-how das equipes de desenvolvimento, transformando uma espécie de recurso frágil em patrimônio permanente [37].

De fato, uma vez que novos sistemas são criados a partir de classes testadas e com funcionamento garantido, com código de fácil compreensão e relações bem estabelecidas, o processo de desenvolvimento terá como características produtividade, rapidez e segurança.

4.2 – CLASSES E OBJETOS

O uso da orientação a objetos muda os paradigmas da computação tradicional, pois troca o enfoque a funções por uma modelagem de classes e seus objetos relacionados.

Apesar de ser chamada programação orientada a objetos, o conceito central dessa tecnologia não é o objeto em si, mas sim a classe. Uma classe é um tipo de dado abstrato equipado com uma implementação parcial possível [37].

As classes de um sistema normalmente apresentam derivações de outras classes e interações entre si, podendo ser definidas como um conjunto estruturado de padrões a partir dos quais os objetos são criados. Elas especificam o comportamento e o raio de ação comum ao objeto, descrevendo sua estrutura e apresentando suas declarações de atributos e métodos.

A classe pode ser vista como um mecanismo de geração de instâncias, definindo atributos e operações particulares, à medida que essas

forem necessárias. Cada objeto derivado de uma classe é chamado uma instância dessa classe.

A classe representa um conjunto de coisas reais ou abstratas, reconhecidas como de mesmo tipo por possuírem características comuns relativas a atributos, relações, operações e semântica. De uma forma geral, a classe pode ser vista como um conjunto de objetos que possuem estrutura e comportamento comuns, ou até mesmo ser definida como um conjunto de objetos idênticos.

A possibilidade de reutilização e a funcionalidade são os fatores mais importantes a serem considerados no momento de definição de uma classe.

Um objeto é a modelagem individualizada de um conceito ou uma estrutura física qualquer da realidade. O objeto pode ser concreto como os sinais de entrada em uma placa de aquisição de dados, ou conceitual, como a interpretação que um software de análise retira desses sinais.

Cada objeto tem sua própria identidade, que é uma propriedade que o distingue de todos os demais objetos, mesmo que todos os seus atributos sejam os mesmos. A identidade é preservada mesmo quando o estado do objeto muda totalmente, sob qualquer tipo de anormalidade ou situação.

Os objetos de uma classe possuem os mesmos padrões de comportamento, compartilhando objetivo semântico e requisitos de atributos comuns. O que normalmente diferencia os objetos de uma classe são as diferenças entre os valores de seus atributos e o relacionamento de cada um com os outros objetos do meio [38].

O trabalho de definição de um modelo orientado a objeto e sua subdivisão em classes depende da abstração empregada, que é responsável pela identificação das classes.

A abstração pode ser definida como o mecanismo de análise de um ambiente real, que prioriza a observação da realidade descrita pelo estado inicial e pelos processos de ação-reação dos fatos, retirando informações

sobre as entidades envolvidas e os fenômenos essenciais encontrados, de forma a permitir a exclusão de aspectos secundários ou de pouca importância. Em outras palavras, o processo de abstração seleciona os aspectos fundamentais do domínio do problema para modelagem, desconsiderando os aspectos adicionais irrelevantes.

Ao invés da busca da reprodução de todos os detalhes de um sistema real, o processo de abstração em um sistema orientado a objeto busca a seleção das características realmente indispensáveis do objeto para uma modelagem parcial, mas perfeitamente delineada a nível comportamental. Esse modelo possui capacidade de generalização para casos semelhantes. As definições comuns são armazenadas como características de classe, assim como as operações. Dessa forma, os objetos da classe conseguem se beneficiar da reutilização do código escrito.

4.3 – DIAGRAMA DE CLASSE

A classe é representada através de um diagrama contendo uma coleção de elementos definindo um modelo. Constam nesse diagrama um campo para o nome da classe, um campo para a definição dos atributos e um terceiro campo reservado para as definições de operações e métodos, Figura 4.1.

O campo reservado para a adoção de um nome serve para diferenciar a classe em questão de uma outra qualquer.

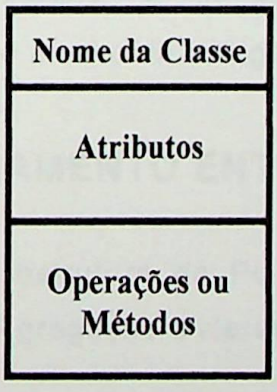


Figura 4.1 – Diagrama de Classe

Os atributos definidos no segundo campo do diagrama possuem nome e um valor associado. Os valores são limitados a uma faixa

determinada e definem o comportamento estático das instâncias, podendo ser fixos, variáveis, ou padrão.

Atributo é um valor de dado referente a um objeto de uma classe. Diferentes instâncias de objetos podem ter valores iguais ou valores diferentes para um dado atributo. O atributo não possui identidade, uma vez que não representa um objeto. Ele representa alguma propriedade da estrutura a ser modelada, compartilhada por todos os objetos da classe.

As operações ou métodos, que ocupam o terceiro campo do diagrama, são funções ou transformações a serem aplicadas a objetos ou por objetos a classes. Os métodos são serviços à disposição dos objetos da classe para uma eventual necessidade de mudança de comportamento, e são compartilhadas por todos os objetos de uma classe. Eles definem o comportamento dinâmico da classe, e são responsáveis pelo acesso aos atributos. Podem ser divididos em métodos públicos ou privados. São ditos públicos quando sua atuação se dá no processo de definição de serviços, e privados quando constituem rotinas de uso interno. Além disso, um método pode ser aplicado a diversas classes, caracterizando o polimorfismo.

O polimorfismo é um conceito utilizado para descrever a característica da POO que permite a utilização de um mesmo nome de método para tarefas similares em classes diferentes. De acordo com o polimorfismo, uma operação pode tomar formas diferentes em classes diferentes. Uma classe pode ter definida uma operação que pode ser executada de maneiras diversas, e nesse caso cada método será implementado por um diferente trecho de código [38].

4.4 – RELACIONAMENTO ENTRE CLASSES

As classes de uma estrutura de POO se relacionam através de processos como Associação, Agregação e Herança, entre outros.

4.4.1 – ASSOCIAÇÃO

A associação é uma relação que descreve um conjunto de ligações com semântica e estrutura comuns, do mesmo modo que uma classe descreve um grupo de objetos com características comuns. As associações

apresentam características bidirecionais, mas não necessitam ser implementadas em ambas direções.

Uma associação ocorre quando uma classe apresenta interdependência consigo própria ou com outra classe, se possui uma ou mais instâncias originadas ou associadas a uma ou mais instâncias da outra.

Com relação ao número de relacionamentos entre as instâncias de classe, aparece o conceito de multiplicidade. A multiplicidade define o número de instâncias de uma classe relacionadas a uma única instância de uma classe associada. A multiplicidade, de uma forma geral, pode ser definida como multiplicidade um para um, um para muitos ou muitos para muitos. Nos diagramas de classes, pode ser colocado um ponto cheio para indicar a multiplicidade igual ou superior a dois ou um ponto vazio para indicar multiplicidade um.

O diagrama apresentado na Figura 4.2 representa uma associação de multiplicidade um para um. Cada associação no diagrama de classes é correspondente a um conjunto de ligações no diagrama de instâncias. A ligação é representada por uma linha entre os objetos.

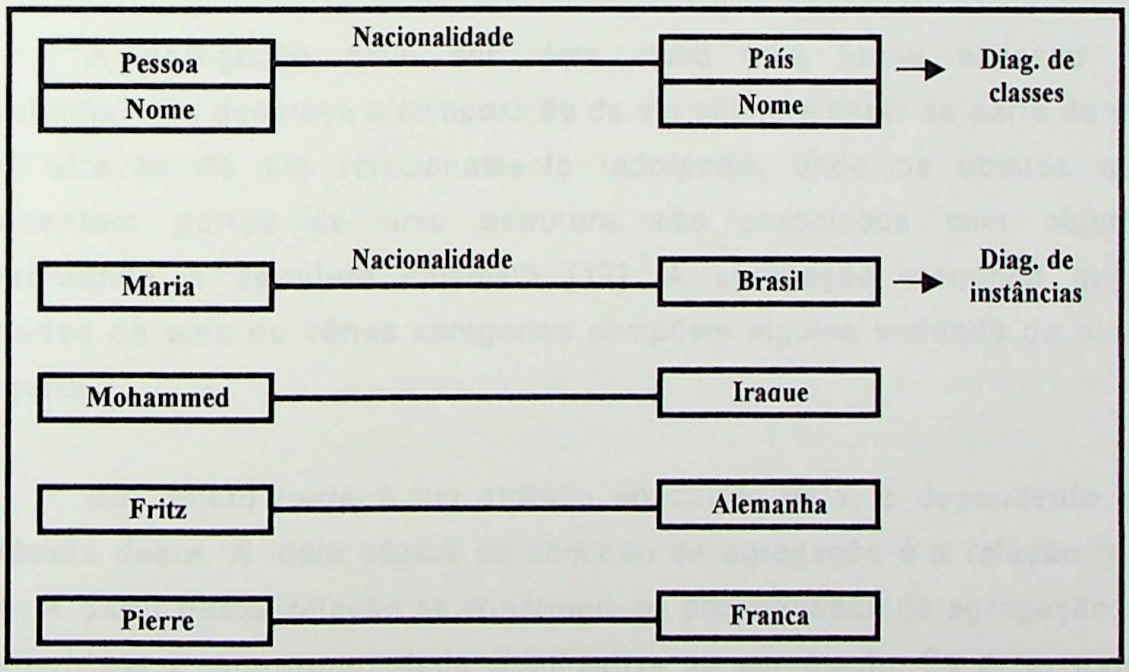


Figura 4.2 – Multiplicidade Um para Um

Já o diagrama apresentado na Figura 4.3 representa uma associação muitos para muitos, e nas linhas que representam as ligações podem ser colocadas referências à multiplicidade de cada ligação [38].

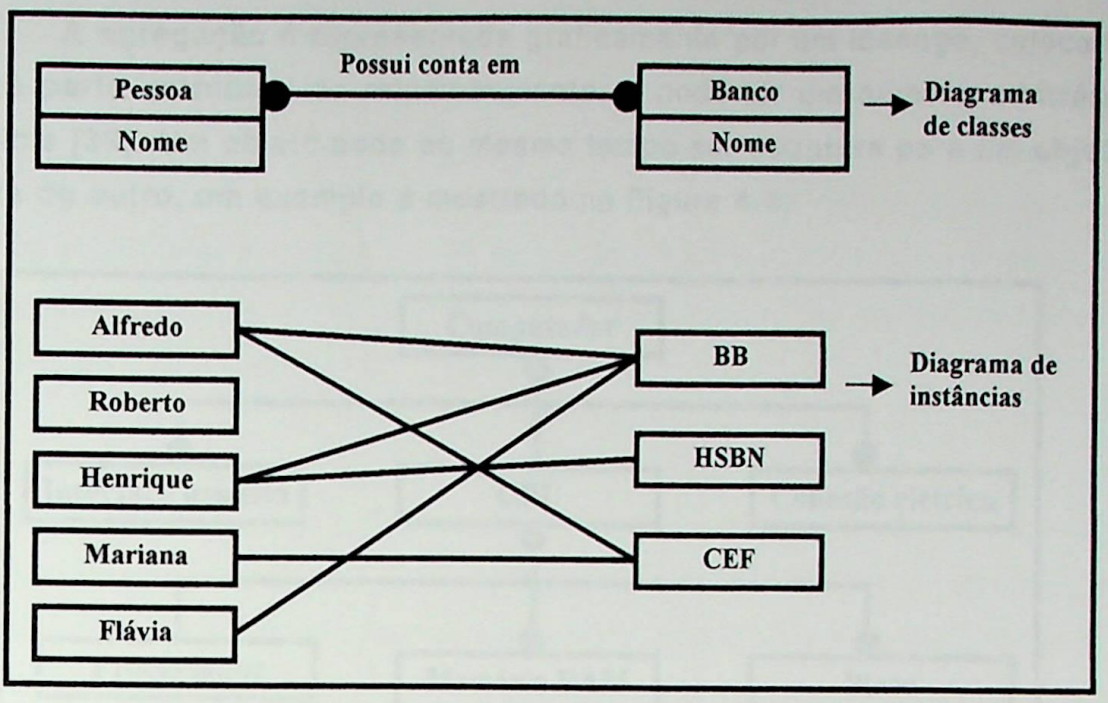


Figura 4.3 – Multiplicidade Muitos para Muitos

4.4.2 – AGREGAÇÃO

A agregação é uma forma de relacionamento entre elementos de um modelo, que serve para expressar que cada objeto de um determinado tipo é uma combinação de outros objetos [37].

A agregação pode ser vista como uma forma especial de associação, que descreve a composição de um objeto a partir de parte de um outro através de um relacionamento todo/parte, onde os objetos que representam partes de uma estrutura são associados com objetos apresentando a estrutura completa [17]. A agregação identifica quais entidades de uma ou várias categorias compõem alguma entidade de outra categoria.

Um objeto parte é um atributo do objeto todo, e dependente da existência deste. A idéia básica do conceito de agregação é a relação faz-parte. A partir dessa relação se constroem as propriedades de agregação. A transitividade é uma propriedade significativa da agregação. Se X faz parte de Y e Y faz parte de Z, logo X faz parte de Z. Um relacionamento de agregação é definido como o relacionamento de uma classe estrutural a uma classe de um componente. Uma estrutura com muitos tipos de componentes corresponde a muitos relacionamentos de agregação [38].

A agregação é representada graficamente por um losango, colocado junto à parte estrutural do relacionamento, e pode ter um número arbitrário de níveis [38]. Um objeto pode ao mesmo tempo ser estrutura para um objeto e parte de outro, um exemplo é mostrado na Figura 4.4.

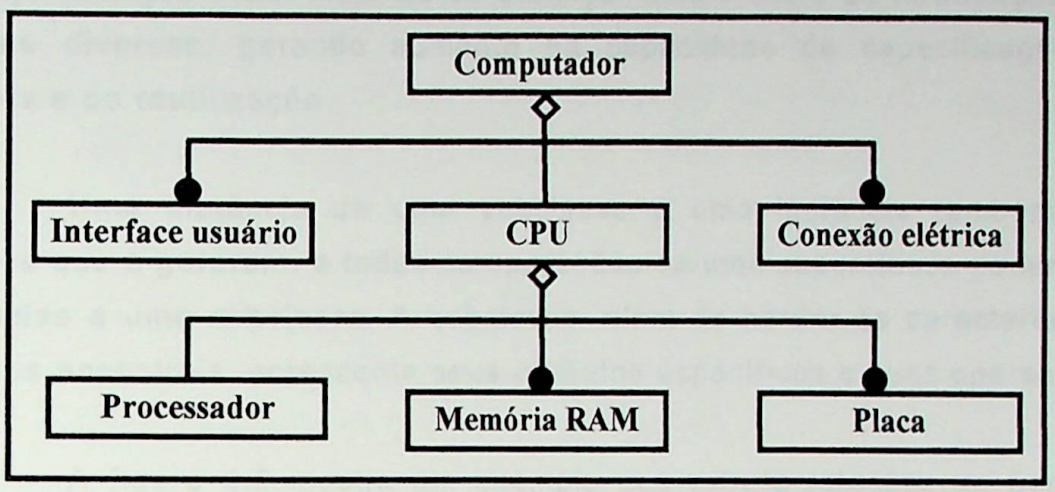


Figura 4.4 – Agregação

4.4.3 – GENERALIZAÇÃO E HERANÇA

A generalização se traduz pela análise do relacionamento entre uma classe e seus refinamentos, identificando as características comuns a todas e definindo uma versão mais genérica. A classe que passa pelo refinamento é conhecida como superclasse (classe pai ou classe base), e as versões especializadas são chamadas subclasse (classe filha ou classe derivada). A generalização tem como idéia básica a relação "é um", pois as instâncias da subclasse são também instâncias da superclasse. A notação utilizada é um triângulo interligando a superclasse a suas subclasses.

A herança é um mecanismo típico da POO segundo o qual a subclasse herda os atributos e métodos da superclasse, potencializando a consolidação do código. As subclasses herdam todas as características da superclasse, inclusive a dependência de relacionamento que esta poderia ter com outras classes [17].

Em relação à melhoria da qualidade de software, a herança representa um método de construção de elementos reutilizáveis, simplificando a definição de classes com características semelhantes através do reaproveitamento das similaridades. Sem a herança, cada módulo novo precisaria de uma definição completa de suas características [37].

Quando a herança se restringe ao reaproveitamento de características de uma única superclasse na formação da subclasse, esta é chamada herança simples. Quando os atributos e métodos utilizados têm como origem superclasses diversas, ocorre a chamada herança múltipla. A herança múltipla é um meio de se alcançar uma mescla de informações de origens diversas, gerando aumento na capacidade de especificação de classes e de reutilização.

Uma instância de uma subclasse é uma instância também das classes que a geraram, e todas as operações de uma superclasse podem ser aplicadas a uma subclasse. A subclasse, além de herdar as características de seus ancestrais, acrescenta seus atributos específicos e suas operações.

A figura 4.5 mostra um exemplo que retrata relações de herança simples no qual as classes Transmissor de Imagem e Transmissor de Som herdam as características da Classe Transmissor. O mesmo acontece quando Monitor de Vídeo herda características de Transmissor de Imagem, e quando Rádio herda características de Transmissor de Som. Ocorre herança múltipla quando a classe Televisão herda características das classes Transmissor de Imagem e Transmissão de Som simultaneamente.

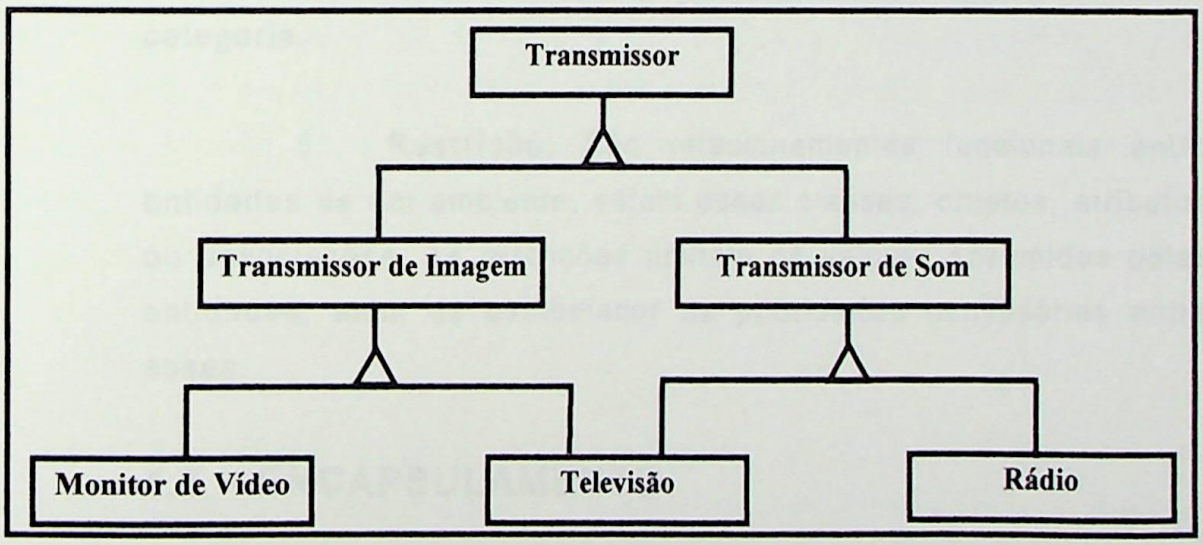


Figura 4.5 – Herança

4.4.4 – OUTRAS RELAÇÕES

Além das relações de associação, agregação e herança, utilizadas na modelagem realizada nesse trabalho, são também de grande importância

na modelagem de sistemas orientados a objeto as relações de especialização, instanciação, classificação, decomposição e restrição:

1. **Especialização:** É o processo de criação de subcategorias a partir de uma categoria existente, onde o grau de especialização da categoria filha supera a da categoria pai. Esse resultado é atingido através da adição de atributos, redefinições dos métodos e aplicação de restrições necessárias.

2. **Instanciação:** É uma operação que cuida da análise das categorias de um ambiente, identificando as características comuns existentes entre duas ou mais delas, de forma a possibilitar a definição de uma categoria mais genérica.

3. **Classificação:** Trata-se de uma operação que identifica, dentro de um número limitado de entidades, a existência de características que sejam comuns a todas essas entidades.

4. **Decomposição:** Consiste em uma operação através da qual é feita a identificação das entidades pertencentes a uma categoria que apresentam em sua composição entidades de outra categoria.

5. **Restrição:** São relacionamentos funcionais entre entidades de um ambiente, sejam essas classes, objetos, atributos ou associações. As restrições limitam os valores assumidos pelas entidades, além de estabelecer as prioridades necessárias entre essas.

4.5 – ENCAPSULAMENTO

Muitas vezes, durante o trabalho em ambiente computacional, faz-se necessário o uso de ferramentas matemáticas ou mesmo o aproveitamento de algum programa como ferramenta para se alcançar os resultados buscados. Na programação tradicional, normalmente desenvolvem-se rotinas para suprir essas necessidades.

O processo de encapsulamento utilizado em POO tem como princípio o processo de ocultar informações. Dessa forma, uma entidade esconde informações, que sejam necessárias apenas para seu próprio funcionamento, mas permanece acessível através da interface oficial [37]. Um objeto deve ser visto como uma entidade completa com o qual outros objetos se comunicam através de seus métodos, sem que seja necessário haver conhecimento de sua estrutura interna.

O encapsulamento separa os aspectos externos do objeto, acessíveis por outros objetos, dos detalhes internos de implementação, deixando-os ocultos e possibilitando mudanças de implementação sem com isso afetar os objetos que o utilizam [38].

Observando o resultado do encapsulamento sob um prisma externo, o processo permite um tratamento dos objetos como "caixas pretas", onde a comunicação se faz possível desde que sejam conhecidas a natureza e finalidade de cada objeto e o formato de seus dados de entrada e saída.

Um grande benefício gerado pelo encapsulamento consiste no reaproveitamento de programas e ferramentas matemáticas existentes, no desenvolvimento de um projeto orientado a objeto. Desde que o formato e o significado dos dados de entrada e saída dessas "caixas pretas" sejam conhecidos, é possível fazer a partir deles o encapsulamento e reaproveitá-las como objetos. Para isso se fazem necessárias interfaces adaptativas às formas de entrada e saída do núcleo reaproveitado.

4.6 – JAVA

A linguagem Java é orientada a objetos, possuindo fortes características de portabilidade e exibindo um conjunto de bibliotecas que facilitam o desenvolvimento de aplicações. As características de portabilidade e arquitetura neutra permitem aos programas a execução em qualquer local da rede, independentemente do tipo de plataforma.

A maneira como os programas Java são executados oferece uma série de atrativos, que tornam a linguagem uma opção interessante para o desenvolvimento de SMA.

A princípio, os programas desenvolvidos em Java podem rodar em plataformas diversas, enquanto as demais linguagens orientadas exigem um código específico para cada tipo de plataforma. Isso ocorre porque o programa Java não tem contato com o computador real. Ele roda dentro de uma máquina virtual, a chamada JVM (Java Virtual Machine – Máquina Virtual Java), que faz o papel de um computador abstrato. A JVM age como uma barreira entre o computador e o programa. O programa não tem acesso aos dispositivos de entrada e saída, ao sistema de arquivos nem à memória do computador. Quem acessa esses recursos é a JVM.

O compilador Java não gera código de máquina de acordo com o hardware em que foi desenvolvido, mas sim um código independente para a JVM, os bytecodes Java. Um fluxo de bytecodes representa uma seqüência de instruções. Cada instrução é composta de um byte opcode e de operandos. O opcode informa à JVM que ação tomar e os operandos oferecem as informações requeridas para a realização dessa ação.

A execução de programas Java como linguagem interpretada via JVM torna a velocidade de execução razoável. Para superar essa limitação as JVM utilizam compiladores "just in time", que compilam bytecodes para código nativo durante a execução, melhorando o desempenho.

Java possui um sistema de processamento multitarefa, o multithreading. Esse sistema consiste de um meio de semi-simultaneidade, onde entradas múltiplas são recebidas e processadas concorrentemente, e o tempo de execução é otimizado. A biblioteca Java fornece uma classe Thread contendo métodos para iniciar, executar, parar e checar o estado de uma tarefa.

Portanto, esses são os recursos oferecidos por Java que tornam a linguagem uma opção bastante interessante para o desenvolvimento de SMA. Suas características de orientação a objeto favorecem a abstração de distribuição topológica. O processamento multitarefa permite o convívio dinâmico de entidades independentes. Por fim, a independência de plataforma facilita a distribuição das unidades em rede tornando o software compatível com qualquer hardware e plataforma.

4.7 – MODELAGEM DE SUBESTAÇÕES VIA ORIENTAÇÃO A OBJETO

O SEP constitui um ambiente complexo e exigente. Os softwares utilizados no tratamento das unidades do SEP apresentam, portanto, características complexas. As dificuldades de desenvolvimento de softwares para esse tipo de ambiente, que exige precisão e rapidez, derivam dos seguintes fatos [36]:

- Os sistemas a serem modelados são relativamente grandes;
- O banco de dados envolvido é, geralmente, grande;
- A interface com o usuário deve ser gráfica (GUI – Grafical User Interface), clara, objetiva e explicativa.

O ambiente constituído pelo SEP é evolucionário, ou seja, as especificações mudam com o tempo. As aplicações desenvolvidas para esse meio precisam acompanhar a evolução do sistema. Isso exige uma estratégia de desenvolvimento com características modulares e adaptativas. As linguagens orientadas a objeto possuem características que permitem atingir esses objetivos com o desenvolvimento de sistemas hierárquicos baseados nas relações de herança, agregação e polimorfismo. Através da modelagem orientada a objeto se faz possível a construção de uma plataforma estável com características evolutivas. As alterações em classes, que poderiam levar a falhas, são protegidas pelo encapsulamento de dados e pelas definições de dados públicos e privados [39] [40] [41].

4.7.1 – O MODELO DE SUBESTAÇÃO ELÉTRICA

A modelagem orientada a objeto busca a visualização topológica detalhada do sistema para reproduzir, em ambiente computacional, as interações de rotina que ocorrem no mundo real. Para tal, no caso da SE, a criação das classes com sua distribuição de objetos se dá baseada nas propriedades de cada componente da planta, levando em conta suas similaridades funcionais e operativas.

Os benefícios gerados pela modelagem orientada a objeto de sistemas de potência estão relacionados com o estabelecimento um patamar de modularização no qual as macrofunções são reunidas em blocos

funcionais independentes, possibilitando um relacionamento dinâmico entre as partes, e gerando vantagens como flexibilidade e adaptabilidade.

A modelagem, no paradigma de orientação a objetos, se aproveita dos benefícios que uma abordagem hierárquica pode oferecer. Nele, a SE é vista como um sistema complexo composto de subsistemas inter-relacionados que possuem dentro de si outros subsistemas. Dessa forma, a abstração busca isolar os mais elementares componentes do sistema [39] [42].

A abordagem hierárquica prevê também que os sistemas são compostos de apenas alguns tipos de subsistemas em várias combinações e arranjos isso possibilita a redução de padrões [39].

No processo de modelagem de uma SE, além da abordagem hierárquica, a abstração sugere o uso de três tipos de relações entre objetos, sendo eles associação, generalização e agregação.

A qualidade do processo de abstração é fator crucial para o perfeito funcionamento da estrutura computacional, pois a POO enfoca a preocupação com a estrutura dos objetos. Apesar dos sistemas elétricos de potência caracterizarem um ambiente dinâmico, a estrutura dos objetos permanece estável enquanto mudam os detalhes de utilização e os requisitos. Dessa forma, o modelo permanece preservado, atuando e aguardando nova atualização do estado do mundo.

A generalização ou herança identifica as similaridades entre uma classe e as versões mais refinadas dela. A classe da qual se parte é chamada classe base ou pai, e as classes refinadas a partir dela são chamadas subclasses ou classes filhas. Os atributos e métodos da classe base são herdados pela subclasse [17].

O modelo orientado a objeto para SE, desenvolvido para este trabalho, está representado na Figura 4.6. Esse modelo busca representar um padrão geral para subestações de 138 KV, representando as estruturas essenciais para o monitoramento e para a proteção do sistema.

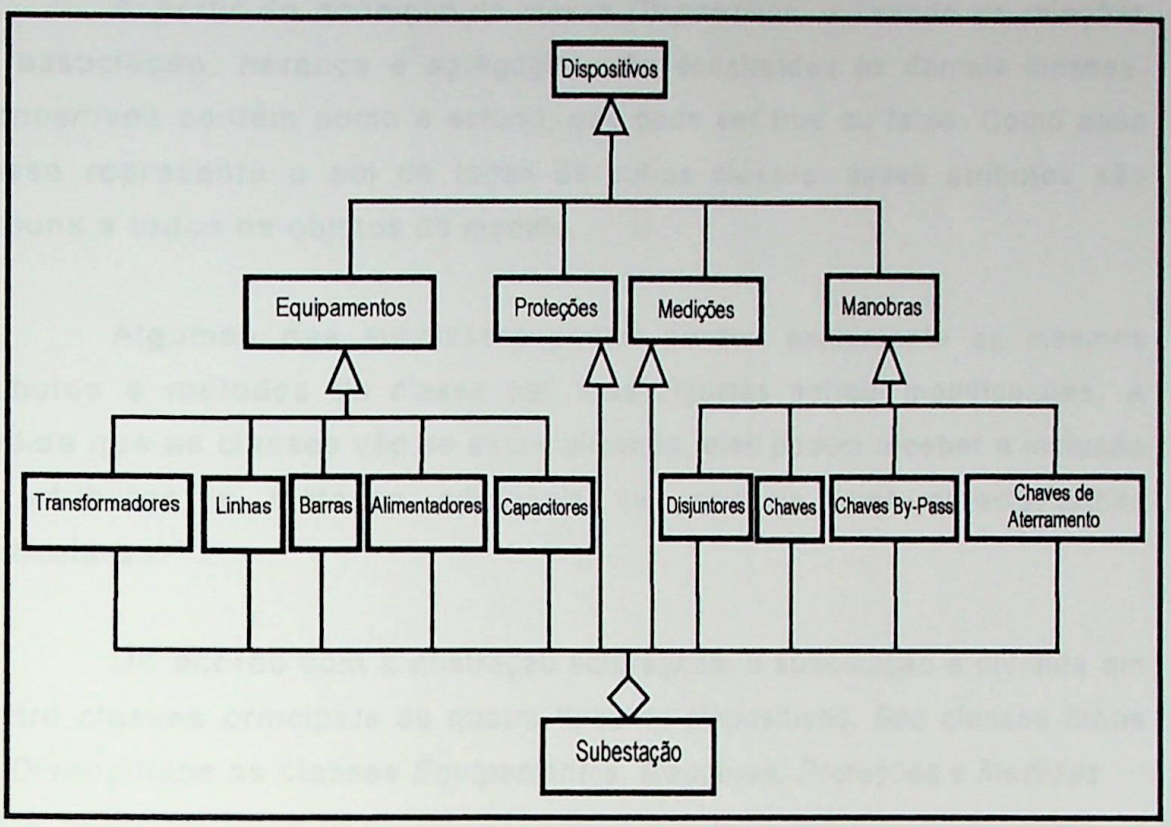


Figura 4.6 – Modelo de Subestação via Orientação a Objeto

A SE, do ponto de vista orientado a objeto, pode ser vista como um conjunto de dispositivos interligados, com relações definidas a partir da função e da natureza de cada entidade em si.

A classe *Dispositivos* é o ponto de partida para a divisão de classes do sistema, Figura 4.7. Ela define que todos os elementos fundamentais para o monitoramento e para a operação da subestação são representados por dispositivos no modelo computacional.

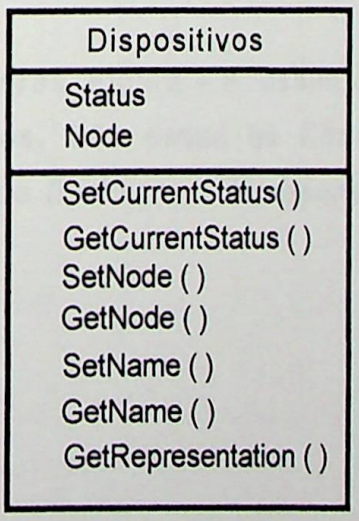


Figura 4.7 – Classe *Dispositivos*

A partir da definição da classe *Dispositivos*, utilizando as relações de associação, herança e agregação, são construídas as demais classes. *Dispositivos* contém nome e estado, que pode ser true ou false. Como essa classe representa o pai de todas as outras classes, esses atributos são comuns a todos os objetos do modelo.

Algumas das subclasses podem possuir exatamente os mesmos atributos e métodos da classe pai, mas algumas sofrem modificações. À medida que as classes vão se especializando, elas podem receber a inclusão de atributos e métodos adicionais que definem suas características particulares.

De acordo com a abstração empregada, a subestação é dividida em quatro classes principais ou quatro tipos de dispositivos. São classes filhas de *Dispositivos* as classes *Equipamentos, Manobras, Proteções* e *Medidas.*

Os dispositivos relacionados pela classe *Equipamentos* são os barramentos, transformadores, bancos de capacitores, linhas e alimentadores. Conforme representado na Figura 4.6, cada um desses dispositivos está associado a uma classe. Essas classes herdam os atributos da classe *Dispositivos*, uma vez que são suas filhas. São chamadas, respectivamente, de *Barras, Transformadores, Capacitores, Linhas* e *Alimentadores.* Essas classes representam os vãos da subestação, e contêm sistemas de proteção e medição acoplados em suas estruturas físicas.

A classe *Equipamentos* e suas subclasses apresentam os mesmos atributos e métodos que a classe *Dispositivos.*

A classe *Manobras*, Figura 4.8, assim como a classe *Equipamentos,* tem várias classes filhas. São essas as *Chaves, Disjuntores, Chaves de Aterramento* e *Chaves de By-Pass*, e representam os elementos de atuação do sistema.

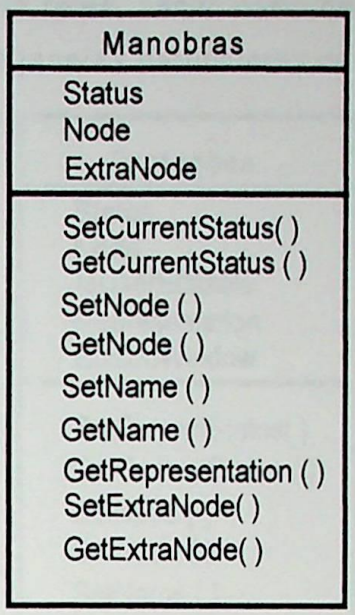


Figura 4.8 – Classe *Manobras*

A classe *Disjuntores* representa os elementos ativos do sistema de proteção da subestação, Figura 4.9. Os disjuntores são editáveis e possuem representação e janela particular de edição.

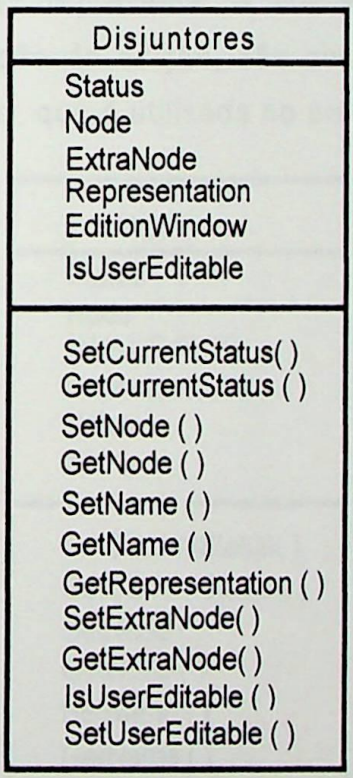


Figura 4.9 – Classe *Disjuntores*

As classes *Proteções* e *Medidas* não têm filhos, e englobam os objetos responsáveis por medir, monitorar e proteger o sistema. A classe *Proteções* representa os relés que monitoram e desarmam os disjuntores da

subestação, Figura 4.10. Os relés, assim como os disjuntores, são editáveis, possuindo representação e janelas particulares de edição.

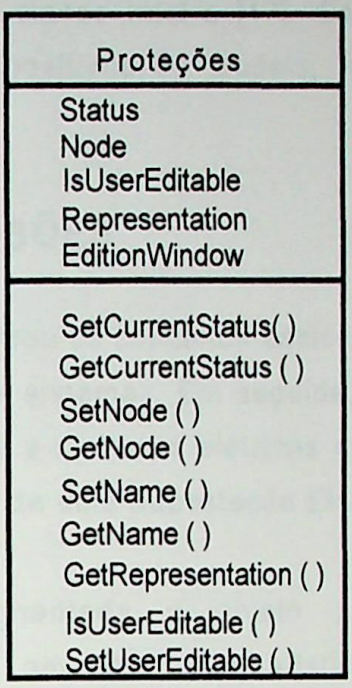


Figura 4.10 – Classe *Proteções*

A classe *Medidas* representa os voltímetros e amperímetros, e, apesar de apresentar função de edição, não possui janela editável, Figura 4.11. Possui representação, que é utilizada no processo de edição.

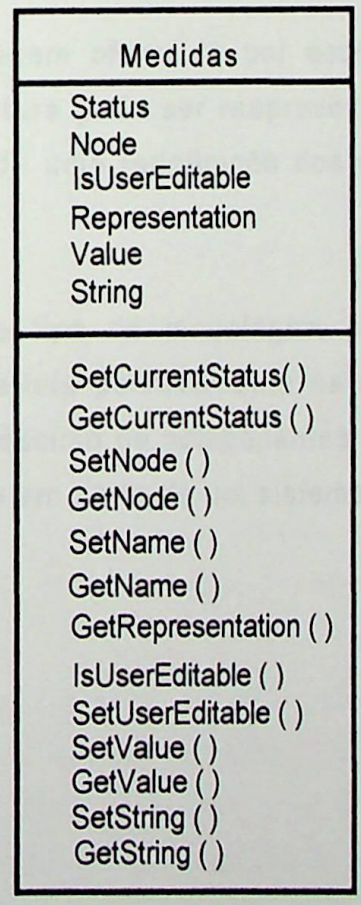


Figura 4.11 – Classe *Medidas*

A relação de Agregação permite a introdução do sentido parte-todo, permitindo aos componentes do sistema que se associem com objetos que representam a reunião da classe inteira [17]. Como mostra a Figura 4.6, *Subestação* agrega os dispositivos do modelo, representando o ambiente como um todo.

4.8 – CONCLUSÕES

Este capítulo mostrou os conceitos básicos de POO e suas relações aplicáveis à modelagem de sistemas. Em seguida, apresentou a viabilidade de aplicação dessa técnica a sistemas elétricos de potência, e, por fim, foi desenvolvida a modelagem de uma Subestação Elétrica genérica de 138 KV.

A modelagem orientada a objeto permite a representação distribuída do sistema, gerando características de flexibilidade. A representação distribuída é interessante na abordagem de sistemas de potência porque as propriedades naturais desses sistemas apresentam topologia distribuída. E a flexibilidade com relação a mudanças no sistema é importante porque o modelo deve conseguir reproduzir as alterações ocorridas no ambiente real.

Uma outra vantagem oferecida por esse tipo de modelagem é o reaproveitamento. A estrutura pode ser reaproveitada para a modelagem de outra subestação, bastando uma redefinição dos dispositivos e o acréscimo de regras.

Finalmente, esse tipo de modelagem permite a criação de uma arquitetura aberta. Isso reflete positivamente na qualidade do modelo, pois se tornam possíveis o acréscimo de componentes na planta e a expansão do sistema, transformando-se em parte de um sistema maior.

5 – IMPLEMENTAÇÃO

5.1 – INTRODUÇÃO

O sistema elétrico brasileiro encontra-se mergulhado em uma crise sem precedentes na história do país. Essa crise estabelece um cenário que incentiva o desenvolvimento de inovações técnicas que ofereçam ao sistema melhorias na qualidade, no rendimento e na segurança.

Para suprir as necessidades de demanda, além de uma campanha nacional de conscientização sobre o uso racional de energia e a aplicação de uma política de racionamento, as preocupações se voltam para a maximização do aproveitamento dos recursos do sistema e para a busca de meios alternativos de produção de energia.

O desenvolvimento de ferramentas computacionais para controle de sistemas elétricos de potência e restabelecimento de subestações elétricas está relacionado ao aproveitamento dos recursos do sistema.

A restauração da configuração normal de uma SE após uma falta ou até um desligamento intencional, é feita através da atuação de um operador humano. Considerando a crescente complexidade nos arranjos de subestações e a probabilidade de falha humana, o tempo gasto na execução das ações de restauração é grande e deve ser otimizado [17].

Para tal, diversos sistemas de automação e restabelecimento são desenvolvidos, sendo empregadas diversas técnicas e várias soluções envolvendo inteligência artificial, como a aplicação de sistemas especialistas [1] e planejamento inteligente utilizando MFM (Multilevel Flow Modeling) [43].

O projeto desenvolvido neste trabalho consiste de um módulo offline e passo a passo de um sistema inteligente de apoio à decisão para restabelecimento de subestações elétricas, que pode ser utilizado para estudar o comportamento do sistema e para o treinamento de operadores. Esse módulo analisa as ocorrências registradas em um banco de dados representando a subestação, e auxilia o operador, gerando um plano diretor de ações de restabelecimento.

A alternativa computacional aplicada é a metodologia de SMA, utilizando modelagem orientada a objeto, sendo a programação feita em Java. A expansão desse projeto para um módulo online permitiria a aplicação no apoio à decisão em restabelecimento de subestações elétricas, gerando uma ferramenta que otimizaria e facilitaria o processo de restabelecimento, sob os prismas econômico e operacional.

A aplicação de modelagem orientada a objeto em sistemas de potência mostrou ser apropriada devido às suas propriedades de reusabilidade e abstração [44] [39] [41]. Além disso, a abordagem multi-agente vem sendo aplicada em problemas do SEP e bons resultados têm sido alcançados, de forma que o interesse pela área tem aumentado, nos últimos dois anos [45] [46].

Independentemente da ferramenta computacional utilizada e do tipo de sistema inteligente implantado, um sistema de suporte a decisão que ajuda na restauração de uma SE deve apresentar meios para:

- Avaliar o agente da ocorrência;
- Definir a área defeituosa isolada pela atuação da proteção;
- Caracterizar se o defeito é permanente ou transitório;
- Identificar os componentes envolvidos e os afetados;
- Gerar um plano de restabelecimento [47].

5.2 – MODELO MULTI-AGENTE PARA SUBESTAÇÃO ELÉTRICA

Um SMA apresenta algumas características de modelagem como processamento assíncrono e concorrente, estrutura descentralizada e arquitetura aberta. A arquitetura do modelo multi-agente construído respeita essas características, apresentando uma estrutura segmentada composta por

nove agentes com funções e comportamento distintos, trabalhando em conjunto na tarefa de monitoramento e restabelecimento da subestação após ocorrência de uma falta.

Os agentes componentes do modelo são, Figura 5.1:

- Agente Comunicação;
- Agente Interface;
- Agente Modelo;
- Agente Planejador;
- Agente Identificador de Eventos;
- Agente Medidas;
- Agente Proteção;
- Agente Chaves;
- Agente Equipamento.

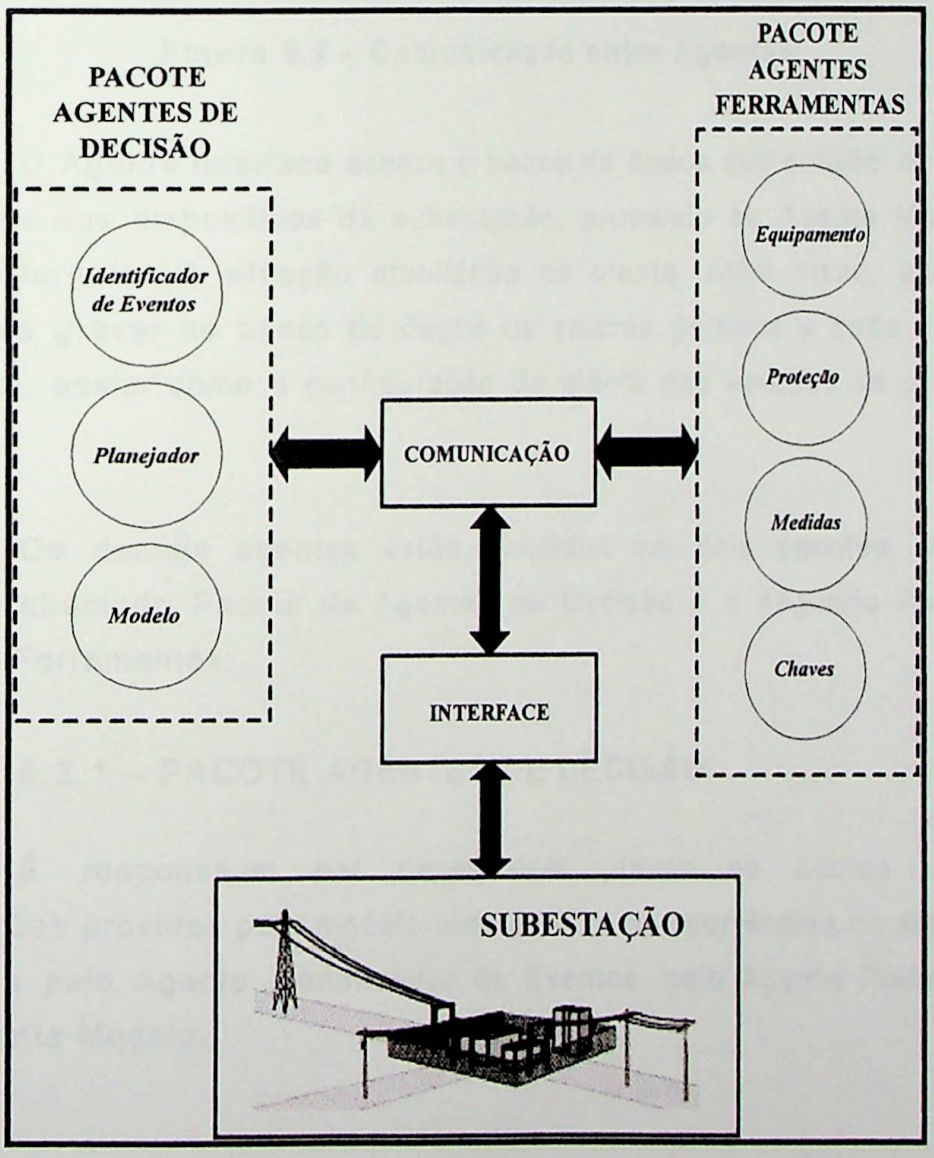


Figura 5.1 – Modelo Multi-Agente para Subestação Elétrica

O Agente Comunicação é responsável pela interconexão entre os agentes. A comunicação entre os agentes acontece por seu intermédio, ele gerencia a troca de mensagens entre as unidades do sistema, e atua como ouvinte global do sistema, recebendo as mensagens de todos os agentes do ambiente. Não há troca de mensagens entre os outros agentes, pois essas devem ser roteadas pelo Agente Comunicação, Figura 5.2.

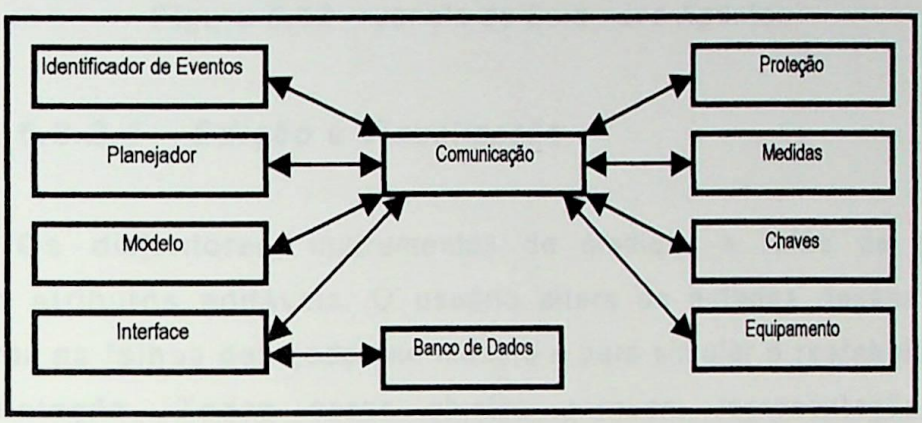


Figura 5.2 – Comunicação entre Agentes

O Agente Interface acessa o banco de dados que contém os valores e estados dos dispositivos da subestação, provendo ao Agente Modelo os dados referentes à situação atualizada da planta. Além disso, ele tem a função de gravar no banco de dados os valores gerados a cada etapa do programa, assim como a configuração da planta nos estados de pré e pós falta.

Os demais agentes estão divididos em dois pacotes, sendo o primeiro chamado Pacote de Agentes de Decisão e o segundo Pacote de Agentes Ferramentas.

5.2.1 – PACOTE AGENTES DE DECISÃO

É responsável por desenvolver planos de acordo com as informações providas pelo modelo virtual sobre as ocorrências no sistema. É composto pelo Agente Identificador de Eventos, pelo Agente Planejador e pelo Agente Modelo.

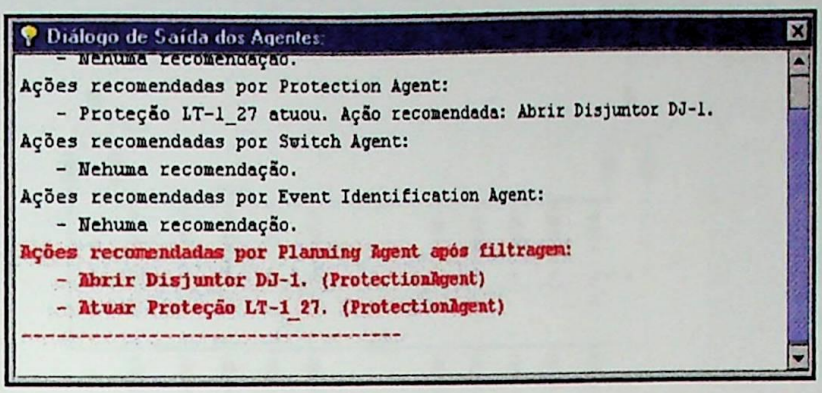


Figura 5.12 – Janela de Saída dos Agentes

5.3.2.4 – Edição e Visualização

Os disjuntores, instrumentos de medidas e relés de proteção possuem atributos editáveis. O usuário altera os estados desses objetos para gerar as falhas desejadas no modelo e para simular o restabelecimento da subestação. Todos esses objetos possuem representação, e os disjuntores e proteções possuem janela de edição.

Para visualizar os dados das medições da subestação, o usuário deve:

1 - Acessar o menu "Medições";

2 - Selecionar o tipo de equipamento a verificar;

3 - Escolher o equipamento específico pelo código, Figura 5.13.

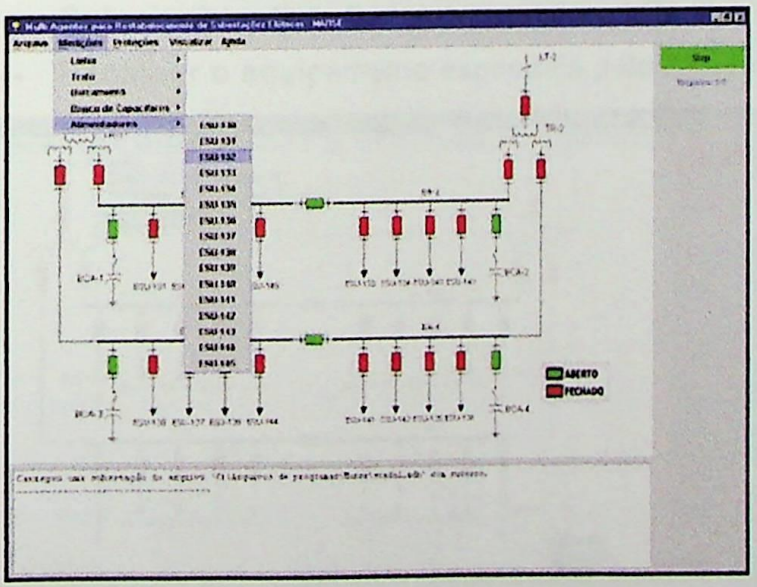


Figura 5.13 – Escolhendo o Equipamento para Visualizar as Medições

Os dados serão apresentados na área de medições, Figura 5.14.

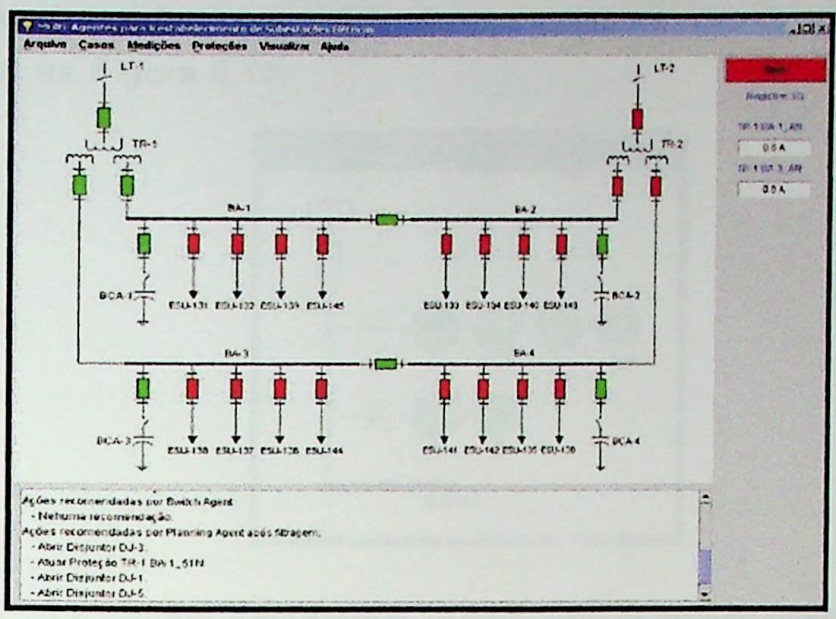


Figura 5.14 – Visualizando Dados de Medições

O MARSE apresenta três tipos de medição: Corrente (A), Corrente de neutro (AN) e tensão (V). O procedimento para alterar os valores das medições é:

1 - Selecionar o campo desejado;

2 - Digitar o valor escolhido;

3 - Apertar a tecla Enter.

Para a vizualização dos dados referentes às proteções, o procedimento é, Figura 5.15:

1 - Acessar o menu "Proteções";

2 - Selecionar o tipo de equipamento a verificar;

3 - Escolher o equipamento específico pelo código.

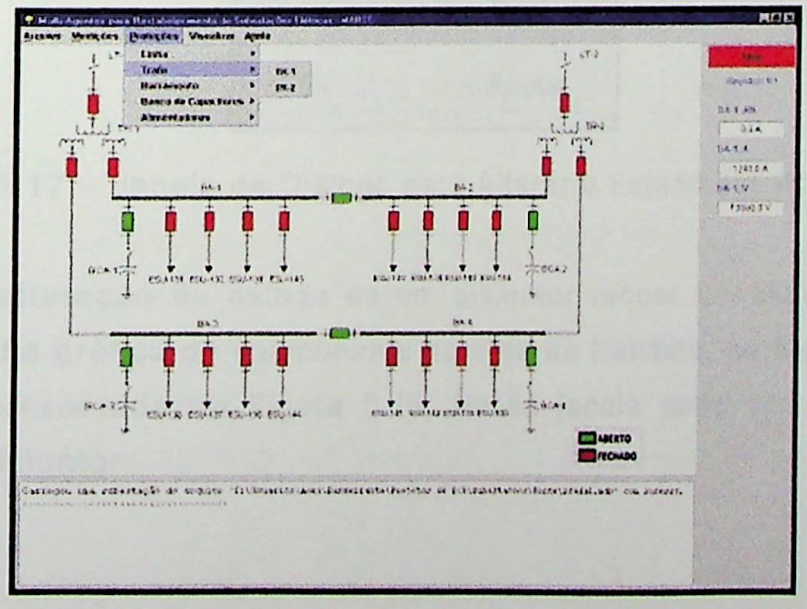


Figura 5.15 – Escolhendo o Equipamento para Visualizar as Proteções

As alterações desejadas são realizadas em uma janela flutuante, apresentada na Figura 5.16.

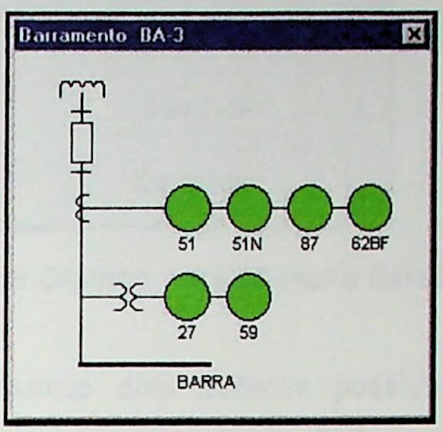


Figura 5.16 – Janela Representativa de Proteções do Barramento BA-3

A representação das proteções possui formato redondo, sendo utilizadas cores para mostrar seus estados. A cor verde representa o estado não atuado da proteção, e a cor vermelha representa a proteção atuada. A alteração do estado de uma proteção requer dois passos:

1 - Com um clique duplo na proteção desejada aparecerá a janela apresentada pela Figura 5.17;

2 - Nessa janela, deve-se selecionar o estado desejado para a proteção e clicar no botão *Salvar*. Se for desejado o cancelamento da operação, deve-se clicar na opção *Cancelar*.

Figura 5.17 – Janela de Diálogo para Alterar o Estado das Proteções

A alteração de estado de um disjuntor requer um clique duplo na representação gráfica do componente na área de trabalho, para que apareça a janela apresentada na Figura 5.18. Nessa janela pode ser escolhido o estado do disjuntor.

Figura 5.18 – Janela de Diálogo para Alterar o Estado dos Disjuntores

O MARSE apresenta dois estados possíveis para o disjuntor: aberto, representado pela cor verde, e fechado, representado pela cor vermelho. Caso seja desejado o cancelamento da ação, basta clicar no botão "Cancelar".

5.3.2.5 – Salvando Casos

A opção "Salvar", no menu "Arquivo", abre a janela apresentada na Figura 5.19.

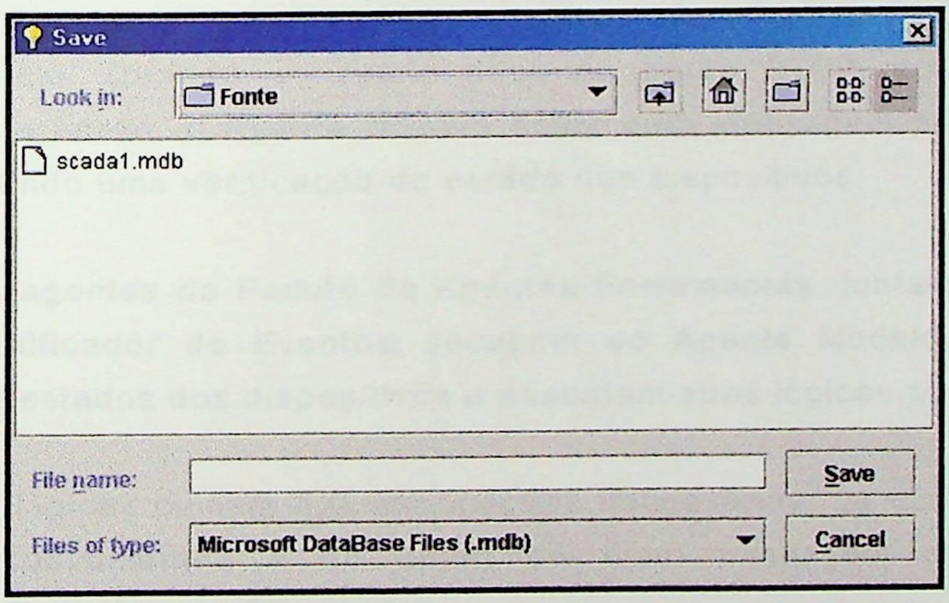


Figura 5.19 – Salvando um Caso Existente

Nessa janela, deve-se digitar o nome do arquivo no referido campo e clicar no botão "Salvar". Caso seja digitado um nome já existente, o arquivo antigo será sobre-escrito com os dados do caso existente.

5.3.2.6 – Funcionamento Passo a Passo

O MARSE é um programa projetado para ser executado no modo Passo a Passo, de forma a possibilitar ao usuário a visão detalhada dos eventos na subestação, seus efeitos e a reação do sistema. O programa é interativo, tornando possível o acompanhamento das variações das medições, dos alarmes e as atuações dos disjuntores da planta.

Uma vez inicializado o programa e instanciados os objetos e os agentes, o usuário entra com os valores e estados referentes à falha que deseja simular, nos campos de medida e proteção. Ao ser pressionado o botão STEP, o programa roda toda a lógica implementada, simulando a reação do sistema à perturbação inserida, e apresentando as sugestões parciais dadas pelos agentes e pelos dispositivos. O usuário entra novamente com os dados necessários tantas vezes seja preciso, até que se caracterize a falta escolhida.

O controle dos passos do programa é função desempenhada pelo Agente Modelo. Quando o usuário clica no botão STEP, aplicando uma requisição de passo, o Agente Modelo envia uma mensagem para todos os agentes pedindo uma verificação do estado dos dispositivos.

Os agentes do Pacote de Agentes Ferramentas, juntamente com o Agente Identificador de Eventos, recebem do Agente Modelo as tabelas contendo os estados dos dispositivos e executam suas lógicas particulares.

As lógicas cuidam das verificações necessárias, da elaboração das sugestões convenientes e do envio de uma mensagem contendo as sugestões para o Agente Planejador.

O Agente Planejador executa sua lógica, que cuida da filtragem das sugestões recebidas dos outros agentes e da elaboração da sugestão final que será enviada ao Agente Modelo.

Se a mensagem enviada pelo Agente Planejador ao Agente Modelo não contiver sugestões, o estado atual dos dispositivos é salvo no banco de dados e um novo registro é criado.

Uma vez caracterizado algum evento, o Agente Modelo recebe sugestões do Agente Planejador, como mostra a Figura 5.20. Ele salva o estado atual dos dispositivos no banco de dados e realiza as alterações sugeridas. Em seguida, salva o novo estado gerado com as alterações realizadas. Por fim, ele cria um novo registro no banco de dados à espera de novas alterações nos dispositivos, na seqüência de geração da falta. Uma vez pressionado o botão STEP a lógica, representada por "run( )" na Figura 5.21, é executada novamente.

```
**********************************************************************
* Término da falha. Iniciando restabelecimento. Ajustar medições. *
**********************************************************************
-----------------------------------
Ações recomendadas por Planning Agent após filtragem:
   - Recomenda-se restabelecer o Disjuntor DJ-3 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-1 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-5 e ajustar suas medições.
-----------------------------------
Ações recomendadas por Planning Agent após filtragem:
   - Recomenda-se restabelecer o Disjuntor DJ-22 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-21 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-16 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-15 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-14 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-13 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-24 e ajustar suas medições.
   - Recomenda-se restabelecer o Disjuntor DJ-23 e ajustar suas medições.
-----------------------------------
```

Figura 5.20 – Sugestões do Planejador

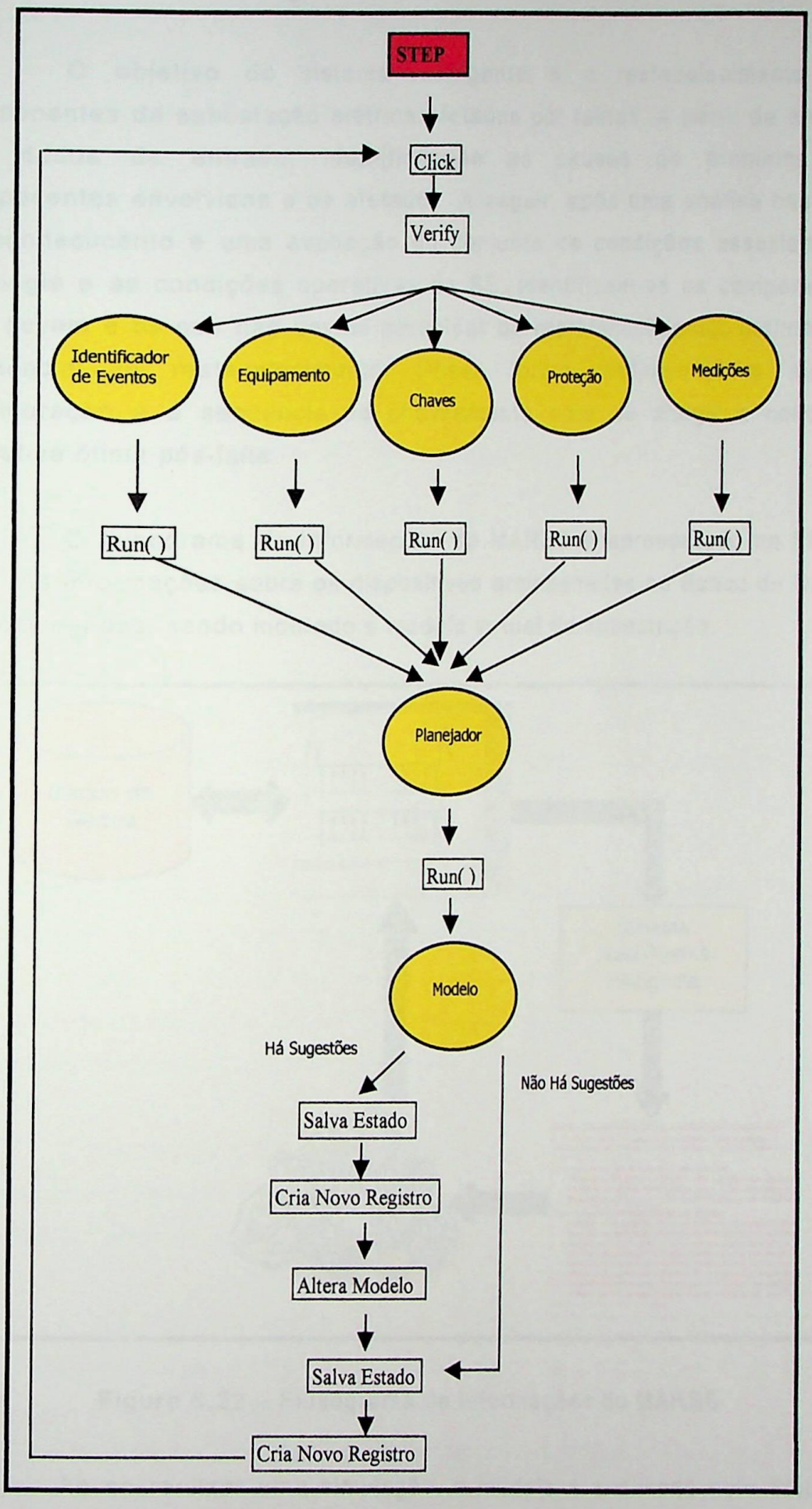


Figura 5.21 – Processamento Passo a Passo do MARSE

5.3.2.7 – Fluxograma de Informações

O objetivo do sistema inteligente é o restabelecimento dos componentes da subestação elétrica afetados por falhas. A partir da análise dos dados de entrada, identificam-se as causas do problema, os componentes envolvidos e os afetados. A seguir, após uma análise baseada no conhecimento e uma avaliação do conjunto de condições associadas à topologia e às condições operativas da SE, identificam-se os componentes que devem e os que não devem participar do restabelecimento, definindo a seqüência de restabelecimento. Dessa forma, definem-se a melhor configuração e a seqüência de chaveamento para se atingir a condição operativa ótima pós-falta.

O fluxograma de informações do MARSE é apresentado na Figura 5.22. As informações sobre os dispositivos armazenadas no Banco de Dados são carregadas, sendo montado o modelo virtual da subestação.

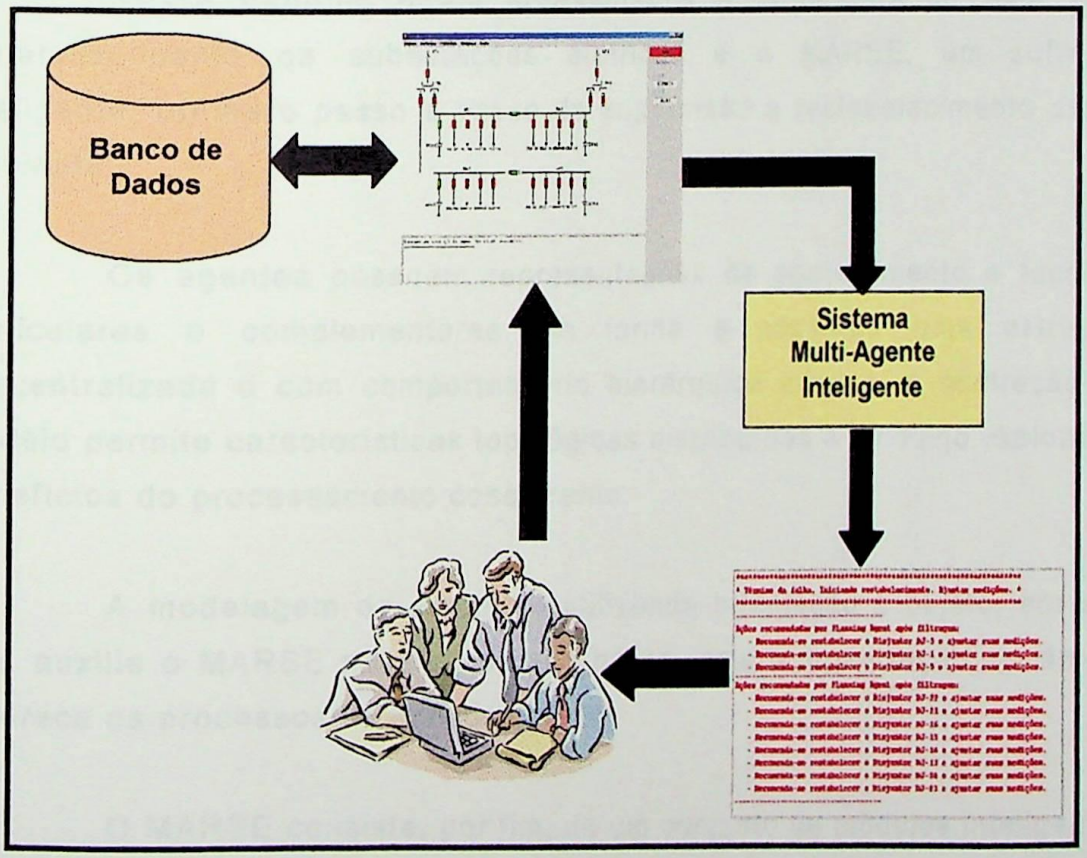


Figura 5.22 – Fluxograma de Informações do MARSE

Ao se realizar uma simulação, o modelo é analisado pelo Sistema Multi-Agente Inteligente. Este é responsável por detectar o evento, encontrar

as soluções possíveis e apresentar uma seqüência de sugestões para o usuário através da Janela do Planejador.

Essa seqüência consta de uma lista de ações necessárias e suficientes para normalizar operativamente a SE após desligamentos parciais ou totais de seus componentes, reintegrando-a, de forma estabilizada, ao SEP.

De posse do plano inicial de restabelecimento o usuário deverá iniciar as ações de restabelecimento na ordem apresentada pelo planejador. O Planejador acompanha as ações do usuário, e, ao detectar seqüências de ações que resultem em falhas, analisa o sistema e sugere nova rotina de correção.

5.4 - CONCLUSÕES

Nesse capítulo foram apresentados a arquitetura do SMA para restabelecimento de subestações elétricas e o MARSE, um software inteligente, offline e passo a passo de supervisão e restabelecimento desse ambiente.

Os agentes possuem representações de conhecimento e funções particulares e complementares, de forma a constituir uma estrutura descentralizada e com comportamento hierárquico mínimo. A abstração do modelo permite características topológicas distribuídas e o código explora os benefícios do processamento concorrente.

A modelagem do ambiente utilizando orientação a objeto, por sua vez, auxilia o MARSE no processo de busca, pois a organização em árvore favorece os processos de encadeamento.

O MARSE consiste, por fim, de um conjunto de módulos inteligentes com características particulares, trabalhando em conjunto na tarefa de supervisionar o funcionamento de uma SE e auxiliar na restauração do sistema em caso de ocorrências. Ele caracteriza um sistema inteligente, pois apresenta propriedades de busca, representação de conhecimento e raciocínio. E caracteriza um SMA, pois, além dessas propriedades, constitui

um mecanismo de ação-reação que utiliza processamento concorrente e possui topologia distribuída.

6 - TESTES E RESULTADOS

6.1 – INTRODUÇÃO

Esse capítulo é dedicado à realização de testes e discussão de resultados. Foram escolhidos três casos de falhas comuns em subestações elétricas, de forma a abranger um grande número de dispositivos afetados. As simulações realizadas constam do estudo de uma falha na barra BA-1 e de uma falha no transformador TR-1, sendo realizada a comparação do resultado do MARSE com a ação de um especialista. Finalmente simula-se um colapso total na subestação, ocasionado por falhas nas linhas de transmissão LT-1 e LT-2, comparando o resultado do MARSE com resultados de um sistema especialista (ESRASE) [34] e de um planejador inteligente (PIRSE) [50].

As respostas encontradas e as sugestões dos agentes encontram-se listadas abaixo, acompanhadas das referentes interfaces visuais.

6.2 – FALHA NA BARRA BA-1

6.2.1 – SIMULAÇÃO

Foi simulada uma sobrecorrente transitória no barramento BA-1, com a entrada do valor de 2200 A via Amperímetro BA-1_A. O relé BA-1_51 atuou, gerando a seqüência de eventos descrita abaixo.

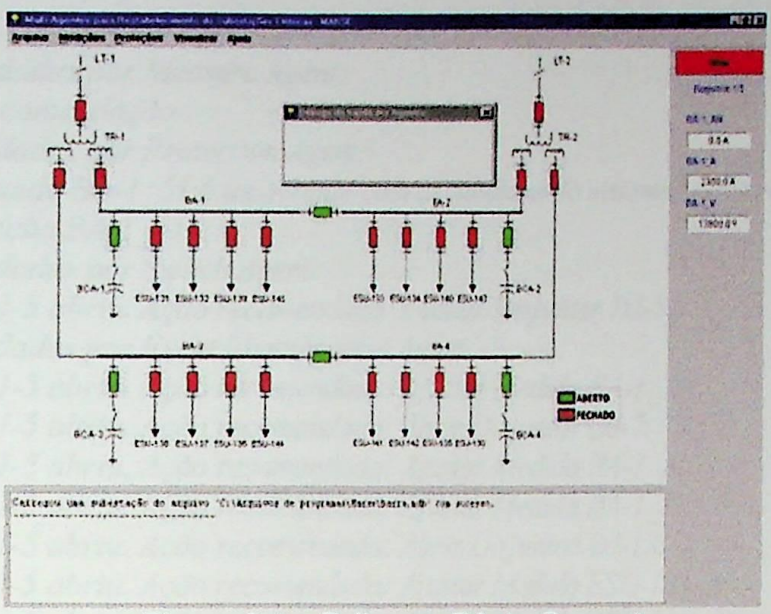


Figura 6.1 – Falha em BA-1 - Passo 1

Histórico dos Avisos dos Agentes
Banco de Dados: C:\Arquivos de programas\Marse\barra.mdb

Ações recomendadas por Measure Agent:
- Nenhuma recomendação.
Ações recomendadas por Switch Agent:
- Nenhuma recomendação.
Ações recomendadas por Event Identification Agent:
- Nenhuma recomendação.
Ações recomendadas por Protection Agent:
- Proteção BA-1_51 atuou. Ação recomendada: Abrir Disjuntor DJ-5.
- Proteção BA-1_51 atuou. Ação recomendada: Abrir Disjuntor DJ-7.
- Proteção BA-1_51 atuou. Ação recomendada: Ajustar Medida BA-1_A.
Ações recomendadas por Planning Agent após filtragem:
- Abrir Disjuntor DJ-5. (ProtectionAgent)
- Atuar Proteção BA-1_51. (ProtectionAgent)

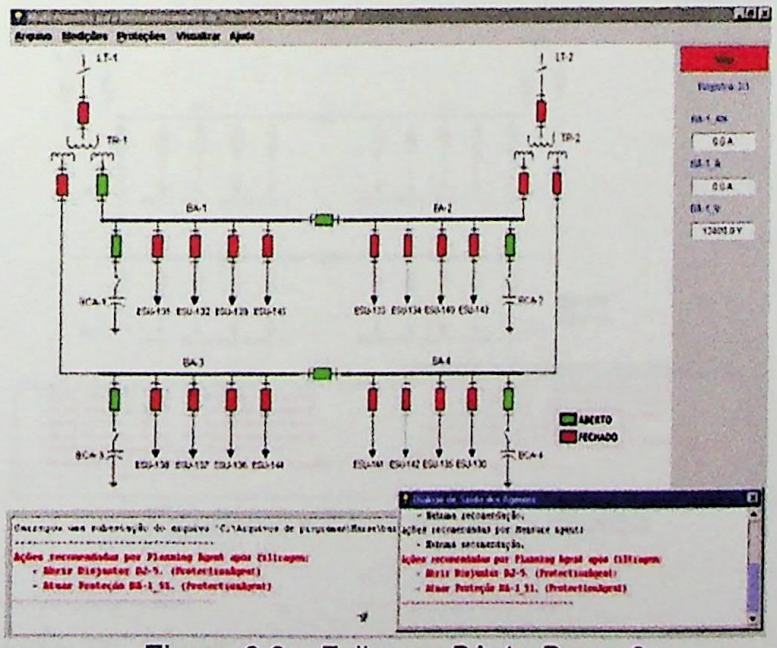


Figura 6.2 – Falha em BA-1 - Passo 2

Ações recomendadas por Measure Agent:

- Nenhuma recomendação.

Ações recomendadas por Protection Agent:

- Proteção atuada BA-1_51 é incoerente com as medidas do sistema. Ação recomendada: Desarmar proteção BA-1_51.

Ações recomendadas por Switch Agent:

- Disjuntor DJ-5 abriu. Ação recomendada: Fechar Disjuntor DJ-5.

Ações recomendadas por Event Identification Agent:

- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-7.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_V.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-13.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-131_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-131_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-14.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-132_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-132_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-15.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-139_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-139_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-16.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-145_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-145_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-9.

Ações recomendadas por Planning Agent após filtragem:

- Abrir Disjuntor DJ-16. (EventIdentificationAgent)
- Abrir Disjuntor DJ-15. (EventIdentificationAgent)
- Abrir Disjuntor DJ-14. (EventIdentificationAgent)
- Abrir Disjuntor DJ-13. (EventIdentificationAgent)
- Desatuar Proteção BA-1_51. (ProtectionAgent)

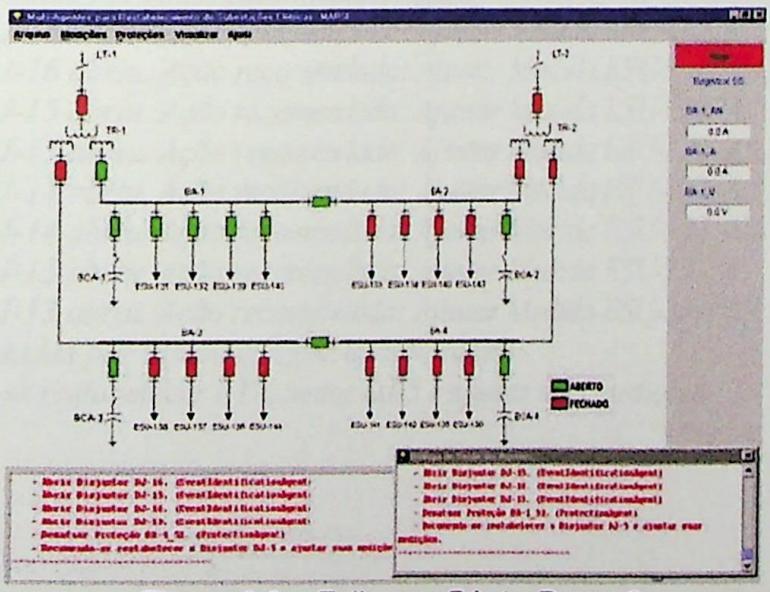


Figura 6.3 – Falha em BA-1 - Passo 3

Ações recomendadas por Measure Agent:

- Nenhuma recomendação.

Ações recomendadas por Protection Agent:
- Nenhuma recomendação.
Ações recomendadas por Switch Agent:
- Nenhuma recomendação.
Ações recomendadas por Event Identification Agent:
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-7.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_V.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-13.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-131_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-131_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-14.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-132_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-132_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-15.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-139_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-139_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-16.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-145_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-145_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-9.

****
** Término da falha. Iniciando restabelecimento. **
****
Ações recomendadas por Measure Agent:
- Nenhuma recomendação.
Ações recomendadas por Protection Agent:
- Nenhuma recomendação.
Ações recomendadas por Switch Agent:
- Disjuntor DJ-5 abriu. Ação recomendada: Fechar Disjuntor DJ-5.
Ações recomendadas por Event Identification Agent:
- Disjuntor DJ-16 abriu. Ação recomendada: Ajustar Medida ESU-145_A.
- Disjuntor DJ-16 abriu. Ação recomendada: Ajustar Medida ESU-145_AN.
- Disjuntor DJ-15 abriu. Ação recomendada: Ajustar Medida ESU-139_A.
- Disjuntor DJ-15 abriu. Ação recomendada: Ajustar Medida ESU-139_AN.
- Disjuntor DJ-14 abriu. Ação recomendada: Ajustar Medida ESU-132_A.
- Disjuntor DJ-14 abriu. Ação recomendada: Ajustar Medida ESU-132_AN.
- Disjuntor DJ-13 abriu. Ação recomendada: Ajustar Medida ESU-131_A.
- Disjuntor DJ-13 abriu. Ação recomendada: Ajustar Medida ESU-131_AN.
Ações recomendadas por Planning Agent após filtragem:
- Recomenda-se restabelecer o Disjuntor DJ-5 e ajustar suas medições.

Histórico dos Avisos dos Agentes
Banco de Dados: C:\Arquivos de programas\Marse\Colapso total.mdb
---

Ações recomendadas por Measure Agent:
- Nenhuma recomendação.
Ações recomendadas por Protection Agent:
- Proteção LT-1_27 atuou. Ação recomendada: Abrir Disjuntor DJ-1.
- Proteção LT-2_27 atuou. Ação recomendada: Abrir Disjuntor DJ-2.
Ações recomendadas por Switch Agent:
- Nenhuma recomendação.
Ações recomendadas por Event Identification Agent:
- Nenhuma recomendação.
Ações recomendadas por Planning Agent após filtragem:
- Abrir Disjuntor DJ-2. (ProtectionAgent)
- Abrir Disjuntor DJ-1. (ProtectionAgent)
- Atuar Proteção LT-2_27. (ProtectionAgent)
- Atuar Proteção LT-1_27. (ProtectionAgent)

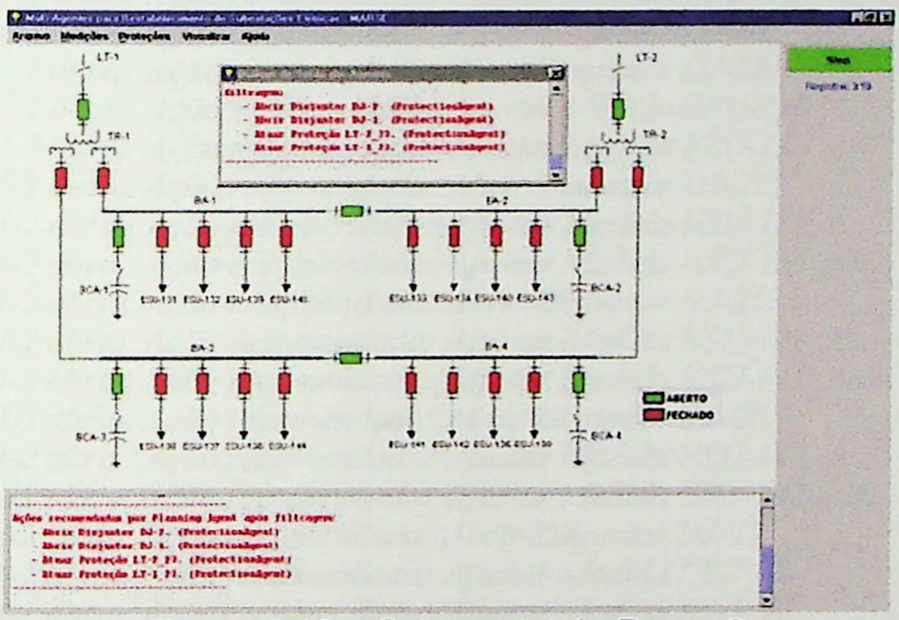


Figura 6.12 – Colapso total - Passo 2

Ações recomendadas por Measure Agent:
- Nenhuma recomendação.
Ações recomendadas por Protection Agent:
- Proteção LT-1_27 atuou. Ação recomendada: Abrir Disjuntor DJ-1.
- Proteção LT-2_27 atuou. Ação recomendada: Abrir Disjuntor DJ-2.
Ações recomendadas por Switch Agent:
- Disjuntor DJ-2 abriu. Ação recomendada: Fechar Disjuntor DJ-2.
- Disjuntor DJ-1 abriu. Ação recomendada: Fechar Disjuntor DJ-1.
Ações recomendadas por Event Identification Agent:
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida LT-2_V.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida LT-2_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-4.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-6.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida TR-2 BA-2_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida TR-2 BA-4_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida BA-2_AN.

- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-7.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida BA-2_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida BA-2_V.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-8.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida BA-4_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida BA-4_V.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida BA-4_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-17.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-133_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-133_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-18.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-134_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-134_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-19.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-140_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-140_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-20.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-143_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-143_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-11.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-28.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-130_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-130_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-27.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-135_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-135_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-25.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-141_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-141_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-26.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-142_A.
- Disjuntor DJ-2 abriu. Ação recomendada: Ajustar Medida ESU-142_AN.
- Disjuntor DJ-2 abriu. Ação recomendada: Abrir Disjuntor DJ-12.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida LT-1_V.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida LT-1_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-3.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-5.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida TR-1 BA-1_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida TR-1 BA-3_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida BA-1_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida BA-1_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida BA-1_V.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida BA-3_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida BA-3_V.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida BA-3_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-13.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-131_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-131_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-14.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-132_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-132_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-15.

- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-139_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-139_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-16.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-145_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-145_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-9.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-23.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-136_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-136_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-22.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-137_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-137_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-21.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-138_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-138_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-24.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-144_A.
- Disjuntor DJ-1 abriu. Ação recomendada: Ajustar Medida ESU-144_AN.
- Disjuntor DJ-1 abriu. Ação recomendada: Abrir Disjuntor DJ-10.

Ações recomendadas por Planning Agent após filtragem:

- Abrir Disjuntor DJ-28. (EventIdentificationAgent)
- Abrir Disjuntor DJ-27. (EventIdentificationAgent)
- Abrir Disjuntor DJ-26. (EventIdentificationAgent)
- Abrir Disjuntor DJ-25. (EventIdentificationAgent)
- Abrir Disjuntor DJ-24. (EventIdentificationAgent)
- Abrir Disjuntor DJ-23. (EventIdentificationAgent)
- Abrir Disjuntor DJ-22. (EventIdentificationAgent)
- Abrir Disjuntor DJ-21. (EventIdentificationAgent)
- Abrir Disjuntor DJ-20. (EventIdentificationAgent)
- Abrir Disjuntor DJ-19. (EventIdentificationAgent)
- Abrir Disjuntor DJ-18. (EventIdentificationAgent)
- Abrir Disjuntor DJ-17. (EventIdentificationAgent)
- Abrir Disjuntor DJ-16. (EventIdentificationAgent)
- Abrir Disjuntor DJ-15. (EventIdentificationAgent)
- Abrir Disjuntor DJ-14. (EventIdentificationAgent)
- Abrir Disjuntor DJ-13. (EventIdentificationAgent)
- Abrir Disjuntor DJ-6. (EventIdentificationAgent)
- Abrir Disjuntor DJ-5. (EventIdentificationAgent)
- Abrir Disjuntor DJ-4. (EventIdentificationAgent)
- Abrir Disjuntor DJ-3. (EventIdentificationAgent)

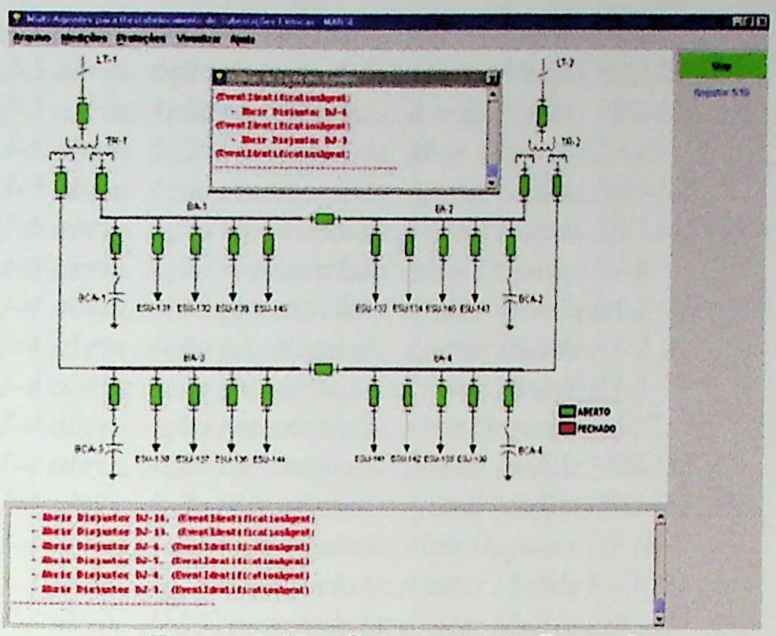


Figura 6.13 – Colapso total - Passo 3

Ações recomendadas por Measure Agent:
- Nenhuma recomendação.
Ações recomendadas por Protection Agent:
- Nenhuma recomendação.
Ações recomendadas por Switch Agent:
- Nenhuma recomendação.
Ações recomendadas por Event Identification Agent:
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-8.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida BA-4_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida BA-4_V.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida BA-4_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-28.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-130_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-130_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-27.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-135_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-135_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-25.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-141_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-141_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-26.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-142_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-142_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-12.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-7.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_V.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-13.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-131_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-131_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-14.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-132_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-132_AN.

- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-15.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-139_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-139_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-16.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-145_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-145_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-9.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida BA-2_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida BA-2_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida BA-2_V.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-17.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-133_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-133_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-18.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-134_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-134_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-19.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-140_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-140_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-20.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-143_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-143_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-11.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida BA-3_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida BA-3_V.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida BA-3_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-23.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-136_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-136_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-22.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-137_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-137_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-21.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-138_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-138_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-24.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-144_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-144_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-10.

****

** Término da falha. Iniciando restabelecimento. Ajustar medições. **

****

O usuário ajusta as medições apresentadas na Tabela 6.6.

Tabela 6.6 – Ajuste de Medidas Colapso Total Passo 3

Medidor	Medida
LT-1_V	138000 A
LT-1_A	240 A
LT-2_V	138000 A
LT-2_A	240 A

Ações recomendadas por Measure Agent:

- Nenhuma recomendação.

Ações recomendadas por Protection Agent:

- Nenhuma recomendação.

Ações recomendadas por Switch Agent:

- Disjuntor DJ-6 abriu. Ação recomendada: Fechar Disjuntor DJ-6.
- Disjuntor DJ-5 abriu. Ação recomendada: Fechar Disjuntor DJ-5.
- Disjuntor DJ-4 abriu. Ação recomendada: Fechar Disjuntor DJ-4.
- Disjuntor DJ-3 abriu. Ação recomendada: Fechar Disjuntor DJ-3.
- Disjuntor DJ-2 abriu. Ação recomendada: Fechar Disjuntor DJ-2.
- Disjuntor DJ-1 abriu. Ação recomendada: Fechar Disjuntor DJ-1.

Ações recomendadas por Event Identification Agent:

- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-8.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida BA-4_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida BA-4_V.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida BA-4_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-28.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-130_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-130_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-27.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-135_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-135_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-25.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-141_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-141_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-26.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-142_A.
- Disjuntor DJ-6 abriu. Ação recomendada: Ajustar Medida ESU-142_AN.
- Disjuntor DJ-6 abriu. Ação recomendada: Abrir Disjuntor DJ-12.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-7.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida BA-1_V.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-13.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-131_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-131_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-14.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-132_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-132_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-15.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-139_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-139_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-16.

- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-145_A.
- Disjuntor DJ-5 abriu. Ação recomendada: Ajustar Medida ESU-145_AN.
- Disjuntor DJ-5 abriu. Ação recomendada: Abrir Disjuntor DJ-9.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida BA-2_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida BA-2_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida BA-2_V.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-17.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-133_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-133_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-18.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-134_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-134_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-19.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-140_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-140_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-20.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-143_A.
- Disjuntor DJ-4 abriu. Ação recomendada: Ajustar Medida ESU-143_AN.
- Disjuntor DJ-4 abriu. Ação recomendada: Abrir Disjuntor DJ-11.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida BA-3_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida BA-3_V.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida BA-3_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-23.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-136_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-136_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-22.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-137_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-137_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-21.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-138_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-138_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-24.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-144_A.
- Disjuntor DJ-3 abriu. Ação recomendada: Ajustar Medida ESU-144_AN.
- Disjuntor DJ-3 abriu. Ação recomendada: Abrir Disjuntor DJ-10.

Ações recomendadas por Planning Agent após filtragem:

- Recomenda-se restabelecer o Disjuntor DJ-3 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-2 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-1 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-6 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-5 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-4 e ajustar suas medições.

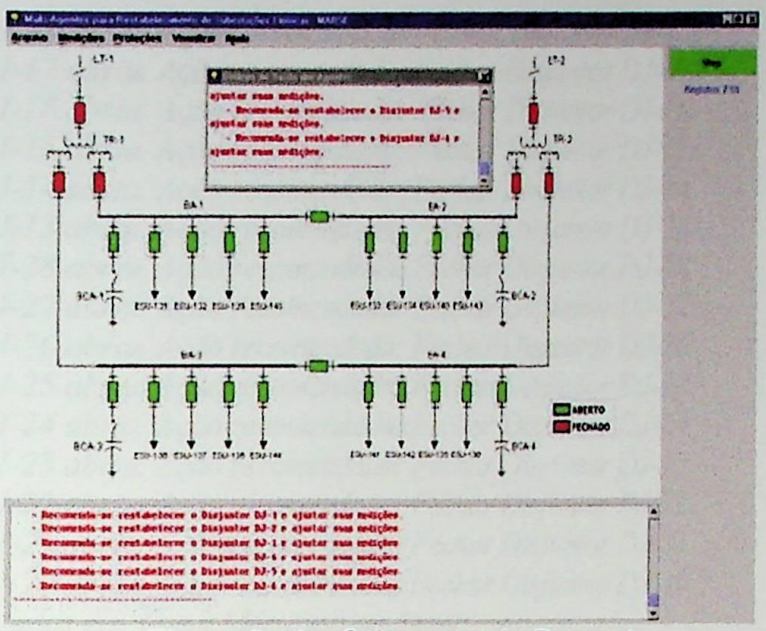


Figura 6.14 – Colapso total - Passo 4

O usuário restabelece nesse passo os disjuntores DJ-1, DJ-2, DJ-3, DJ-4, DJ-5 e DJ6 e ajusta as medições. Os valores ajustados pelo usuário estão listados na Tabela 6.7.

Tabela 6.7 – Ajuste de Medidas Colapso Total Passo 4

Medidor	Medida
TR-1.BA-1_AN	0 A
TR-1.BA-3_AN	0 A
TR-2.BA-2_AN	0 A
TR-2.BA-4_AN	0 A
BA-1_AN	0 A
BA-1_A	1200 A
BA-1_V	13800 V
BA-2_AN	0 A
BA-2_A	1200 A
BA-2_V	13800 V
BA-3_AN	0 A
BA-3_A	1200 A
BA-3_V	13800 V
BA-4_AN	0 A
BA-4_A	1200 A
BA-4_V	13800 V

Ações recomendadas por Measure Agent:

- Nenhuma recomendação.

Ações recomendadas por Protection Agent:

- Nenhuma recomendação.

Ações recomendadas por Switch Agent:

- Disjuntor DJ-19 abriu. Ação recomendada: Fechar Disjuntor DJ-19.

- Disjuntor DJ-18 abriu. Ação recomendada: Fechar Disjuntor DJ-18.
- Disjuntor DJ-17 abriu. Ação recomendada: Fechar Disjuntor DJ-17.
- Disjuntor DJ-16 abriu. Ação recomendada: Fechar Disjuntor DJ-16.
- Disjuntor DJ-15 abriu. Ação recomendada: Fechar Disjuntor DJ-15.
- Disjuntor DJ-14 abriu. Ação recomendada: Fechar Disjuntor DJ-14.
- Disjuntor DJ-13 abriu. Ação recomendada: Fechar Disjuntor DJ-13.
- Disjuntor DJ-28 abriu. Ação recomendada: Fechar Disjuntor DJ-28.
- Disjuntor DJ-27 abriu. Ação recomendada: Fechar Disjuntor DJ-27.
- Disjuntor DJ-26 abriu. Ação recomendada: Fechar Disjuntor DJ-26.
- Disjuntor DJ-25 abriu. Ação recomendada: Fechar Disjuntor DJ-25.
- Disjuntor DJ-24 abriu. Ação recomendada: Fechar Disjuntor DJ-24.
- Disjuntor DJ-23 abriu. Ação recomendada: Fechar Disjuntor DJ-23.
- Disjuntor DJ-22 abriu. Ação recomendada: Fechar Disjuntor DJ-22.
- Disjuntor DJ-21 abriu. Ação recomendada: Fechar Disjuntor DJ-21.
- Disjuntor DJ-20 abriu. Ação recomendada: Fechar Disjuntor DJ-20.

Ações recomendadas por Event Identification Agent:

- Nenhuma recomendação.

Ações recomendadas por Planning Agent após filtragem:

- Recomenda-se restabelecer o Disjuntor DJ-21 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-19 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-20 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-18 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-17 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-16 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-15 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-14 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-13 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-28 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-27 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-26 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-25 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-24 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-23 e ajustar suas medições.
- Recomenda-se restabelecer o Disjuntor DJ-22 e ajustar suas medições.

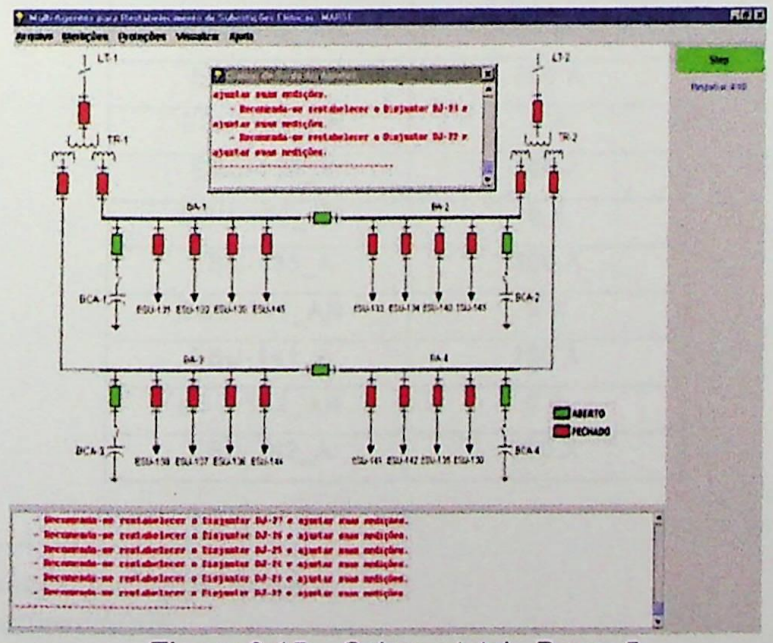


Figura 6.15 – Colapso total - Passo 5

O usuário restabelece nesse passo os disjuntores DJ-13, DJ-14, DJ-15, DJ-16, DJ-17, DJ-18, DJ-19, DJ-20, DJ-21, DJ-22, DJ-23, DJ-24, DJ-25, DJ-26, DJ-27 e DJ-28 e ajusta as medições referentes a esses circuitos. Os valores ajustados pelo usuário estão listados na Tabela 6.8.

Tabela 6.8 – Ajuste de Medidas Colapso Total Passo 5

Medidor	Medida
ESU-131_AN	0 A
ESU-131_A	300 A
ESU-132_AN	0 A
ESU-132_A	300 A
ESU-139_AN	0 A
ESU-139_A	300 A
ESU-145_AN	0 A
ESU-145_A	300 A
ESU-133_AN	0 A
ESU-133_A	300 A
ESU-134_AN	0 A
ESU-134_A	300 A
ESU-140_AN	0 A
ESU-140_A	300 A
ESU-143_AN	0 A
ESU-143_A	300 A
ESU-136_AN	0 A
ESU-136_A	300 A
ESU-137_AN	0 A
ESU-137_A	300 A
ESU-138_AN	0 A
ESU-138_A	300 A
ESU-144_AN	0 A
ESU-144_A	300 A
ESU-130_AN	0 A
ESU-130_A	300 A
ESU-135_AN	0 A
ESU-135_A	300 A
ESU-141_AN	0 A
ESU-141_A	300 A
ESU-142_AN	0 A
ESU-142_A	300 A

Ações recomendadas por Measure Agent:

- Nenhuma recomendação.

Ações recomendadas por Protection Agent:

- Nenhuma recomendação.

Ações recomendadas por Switch Agent:
- Nenhuma recomendação.
Ações recomendadas por Event Identification Agent:
- Nenhuma recomendação.
Ações recomendadas por Planning Agent após filtragem:
- Nenhuma recomendação.

6.4.2 – DISCUSSÃO DOS RESULTADOS

PROBLEMA: Houve uma falha transitória envolvendo simultaneamente as linhas de transmissão LT-1 e LT-2, que alimentam a subestação. Os relés de subtensão LT-1_27 e LT-2_27 percebem baixos níveis de tensão e dão trip nos disjuntores DJ-1 e DJ-2, provocando um colapso total. Não há tensão de retorno em nenhuma das linhas e todos os circuitos da subestação foram desenergizados.

SISTEMA ESPECIALISTA: Nessa situação, o Sistema Especialista, verificando a ocorrência de um colapso, procede como especificado em 3.6.3.1. Inicialmente ele prepara a SE para a reenergização, estabelecendo uma configuração definida previamente por estudos de engenharia. A seguir, restabelece cada um dos circuitos dentro de uma ordem e seqüência ótimos, assumindo carga até completar a reenergização de toda a SE.

PLANEJADOR INTELIGENTE: O Planejador não contempla os procedimentos de preparação da subestação, mas desde que essas ações sejam colocadas como entradas do programa, assim como os dados da falha, o plano gerado estará de acordo com os critérios e filosofias estabelecidos.

MARSE: O processo de caracterização de falta, o monitoramento do sistema e a elaboração do plano de restabelecimento pelo MARSE estão representados pela seqüência mostrada em 6.4.1.

DISCUSSÃO: A ocorrência foi identificada corretamente pelo MARSE. Todos os circuitos da subestação foram atingidos, sendo caracterizada uma situação de colapso total. O processo de restabelecimento elaborado está em conformidade com os critérios e filosofias estabelecidos, totalizando vinte e oito manobras.

Conforme está mostrado em 6.4.1, o MARSE analisa concomitantemente os circuitos envolvidos, de acordo com a hierarquia empregada. As verificações dos problemas ocorridos nas linhas de transmissão LT-1 e LT-2 são feitas simultaneamente. Depois são realizadas as verificações simultâneas dos problemas ocorridos nas barras BA-1, BA-2, BA-3 e BA-4. O mesmo ocorre para os alimentadores.

A Preparação da Subestação é tratada de forma conveniente pelo MARSE, e a resposta do programa tem detalhamento superior à resposta do Planejador, porém é menos detalhada que a resposta do Sistema Especialista.

6.5 – CONCLUSÕES

Nesse capítulo foram apresentados os resultados de testes efetuados com o programa MARSE e a comparação desses resultados com a ação de um especialista em restabelecimento de subestações elétricas e também com resultados obtidos pelos sistemas inteligentes PIRSE [50] e ESRASE [34].

Os resultados obtidos nas simulações foram bons, após as devidas análises e comparações. O nível de detalhamento das respostas foi razoável, e a interface do programa se mostrou amigável e de fácil compreensão.

Alguns desenvolvimentos futuros podem ser visualizados, como a implementação das lógicas para atuação automática dos bancos de capacitores e para os disjuntores de by-pass DJ-7 e DJ-8, que caracterizam transfers na planta, completando o processo de programação previsto pela modelagem. O programa apresentado, contudo, mostra resultados de qualidade suficiente para validar a abstração empregada e a aplicação da técnica de sistemas multi-agentes no restabelecimento de subestações elétricas.

7 - CONCLUSÕES

O resultado final desse trabalho é a construção de mais uma ferramenta de apoio às atividades na operação de sistemas elétricos de potência, utilizando técnicas de Inteligência Artificial. O objetivo principal é a busca constante de otimização das ações e procedimentos de supervisão e controle de subestações elétricas.

O modelo orientado a objeto da subestação foi desenvolvido visando um bom grau de abstração, de forma a permitir o reaproveitamento em outras topologias. O SMA mostrou-se adequado para trabalhar o modelo orientado a objeto, dadas as características de distribuição de ambas as tecnologias.

A modelagem orientada a objeto oferece ao sistema benefícios como:

- Representação distribuída, permitindo ao modelo virtual a reprodução das propriedades distribuídas do sistema;
- Flexibilidade, dando ao modelo virtual adaptabilidade às alterações que podem ocorrer como representação de alterações no mundo real;
- Reaproveitamento, utilizando o sistema desenvolvido para a modelagem de uma nova subestação, através da redefinição dos dispositivos e acréscimo ou mudança de regras;
- Arquitetura aberta, permitindo acréscimo de componentes e a expansão do sistema como parte de um sistema maior.

As propriedades de distribuição da modelagem orientada a objeto permitem o uso desses benefícios para expandir o sistema desenvolvido para subestações elétricas em um sistema de distribuição, bastando para

isso o acréscimo de dispositivos, como religadores e controladores de tensão, ao modelo.

O SMA, por sua vez, oferece benefícios como:

- Comportamento cooperativo, caracterizando dinamismo na troca de informações entre entidades e possibilitando segmentação de tarefas, além de permitir a redução das características hierárquicas do processo de decisão;
- Processamento concorrente, possibilitando melhor uso dos recursos de hardware e diminuição da carga computacional, melhorando a velocidade de execução;
- Topologia distribuída, possibilitando no processo de abstração, a divisão do sistema em partes menores limitadas por características funcionais;
- Arquitetura aberta, permitindo acréscimo de componentes e a expansão do sistema como parte de um sistema maior, com a criação de novos níveis de ambiente.

O programa MARSE foi concebido, estruturado e implementado de uma forma modular, de forma a garantir flexibilidade quanto à expansão e implementação de modificações operacionais. A abordagem modular, característica dos sistemas multi-agentes, permite a inclusão de novos módulos e a expansão do sistema como parte de um sistema maior, mantendo as características de distribuição da técnica.

O programa foi desenvolvido para operação "off-line" com a finalidade de testar a nova abordagem proposta, principal contribuição dessa dissertação. Além disso, pode ser utilizado para treinamento de operadores de subestações, pois o funcionamento passo a passo evidencia as seqüências de eventos de forma ordenada, através de uma interface fácil e amigável. E ainda, este tipo de ferramenta pode ser utilizado conjuntamente com um sistema automatizado de supervisão e controle de subestações oferecendo grandes benefícios à realização do monitoramento e das atividades de manobras.

O MARSE utilizou uma série de algoritmos, com particulares lógicas referenciadas aos agentes desenvolvidos. A divisão de classes em termos das propriedades operacionais dos dispositivos de uma planta genérica

permite a flexibilidade para adaptação do modelo ao monitoramento de novas plantas. Para tal, é necessária a reestruturação da árvore da subestação e a redefinição das normas e critérios de restabelecimento. O reaproveitamento fica evidente, valorizando de forma considerável a aplicação da tecnologia.

O sistema desenvolvido dispõe de recursos para trabalhar com algumas informações quantitativas, como medidas de tensão e corrente. No entanto, não possui ainda recursos para trabalhar com fator de potência, necessário para a implementação da lógica dos bancos de capacitores.

Os principais benefícios do Programa MARSE são:

- Auxilia na tomada de decisões do operador de subestação, aumentando a sua eficiência na execução de tarefas;
- Rapidez na definição e implementação das ações de restabelecimento;
- Viabiliza o processo de automação completa da subestação, com o uso de um sistema de aquisição de dados;
- Contribui para a diminuição do tempo de interrupção do fornecimento de energia.

Os testes realizados mostraram bons resultados, que comprovam os benefícios visualizados e descritos acima.

Como resultado dos estudos desenvolvidos foram escritos um capítulo de livro e dois artigos técnicos:

- O capítulo *"Multi-Agent Model for Power System Simulation"* foi publicado no livro *"Mathematics and Simulation with Biological, Economical and Musicoacoustical Applications"*, de C. E. D'Attellis, V. V. Kluev e N. E. Mastorakis [13].
- O primeiro artigo, *"Multi-Agent Model for Power Substation Restoration"*, foi apresentado na Fifth IASTED International Conference on Power and Energy System, em Tampa, Florida, USA [17].
- O segundo, *"Multi-Agent Simulation and Educational Tool for Power System Operation"*, aprovado para a VII International Conference on Engeneering and Technology Education (INTERTECH 2002), Santos, Brasil [19].

7.1 - TRABALHOS FUTUROS

A análise dos resultados obtidos já possibilita a visualização de desenvolvimentos futuros para o programa MARSE:

- Módulo de operação ON-LINE em conjunto com um sistema automatizado de controle e supervisão de subestação;
- Módulo específico para treinamento de operadores;
- Procedimentos de aprendizado a serem incorporados no algoritmo de planejamento;
- Melhoria do detalhamento das respostas;
- Simulação e teste em ambiente real de operação;

Esta nova abordagem dá margens para se desenvolver novos sistemas de restabelecimento, não somente para subestações, mas também para redes de distribuição, englobando o sistema elétrico como um todo.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Lambert-Torres, G.; Ribeiro, G.M.; Costa, C.I.A.; Alves da Silva, A.P.; Quintana, V.H. Knowledge Engineering Tool for Training Power-Substation Operators, IEEE Transactions on Power Systems, Vol. 12 (2), pp. 694-699, April 1997.

[2] Russell, S; Norvig, P. *Artificial Intelligence – A Modern Approach,* Prentice Hall Series in Artificial Intelligence, Prentice Hall, New Jersey 07458, 1995.

[3] Haugeland, J. *Artificial Intelligence: The Very Idea.* MIT Press, Cambridge, Massachussets.

[4] Bellman, R. E. An Introduction to Artificial Intelligence: Can Computers Think? Boyd & Fraser Publishing Company, San Francisco, 1978.

[5] Charniak, E e McDermott, D. Introduction to Artificial Inteligence. Addison-Wesley, Reading, Massachussetts.

[6] Winston, P. H. Artificial Intelligence. Addison Wesley Publisher Co., 1992.

[7] Kurzweil, R. The Age of Intelligent Machines. MIT Press, Cambridge, Massachussets, 1990.

[8] Rich, Elaine e Knight Kevin Inteligência Artificial. Makron Books, 2ª Edição 1991.

[9] Shalkoff, R. J. Artificial Intelligence: Na Engineering Approach. McGraw-Hill, New Your, 1990.

[10] Luber, G. F. and Stubblefield, W. A. Artificial Intelligence: Structures and Strategies for Complex Problem Solving. Benjamin/Cummings, Redwood City, California, 2nd Edition, 1993.

[11] Lambert-Torres, G., Carvalho, M. A., Silva, L. E. B., Pinto, J. O. P. *"Fitting Fuzzy Membership Functions using Genetic Algorithms"*, 2000 IEEE International Conference on Systems, Man & Cybernetics, Nashville, Tennessee, USA, October 8-11, 2000.

[12] Wooldridge, M., Jennings, N. R. *"Intelligent Agents: Theory and Practice"*, Knowledge Engineering Review, Cambridge University Press, Vol. 10 (2), June 1995.

[13] Esmin, A.A.A.; Aoki, A.R.; Lopes Jr., C.R.; Lambert-Torres, G. "*Multi-Agent Model for Power System Simulation"*, in "Mathematics and Simulation with Biological, Economical and Musicoacoustical Applications", ISBN 960-8052-46-7, WSES Press, pp. 29-33.

[14] Bigus, J. P. e Bigus, J., *Constructing Intelligent Agents with Java: A Programmer's Guide to Smarter Applications.* Wiley Computer Publishing 1998.

[15] Finin, T.; Fritzson, R.; McKay D. and McEntrie, R. "*KQML as an Agent Communication Language"*, Third International Conference on Information and Knowledge Management (CIJM´94), ACM Press, November 1994.

[16] Bordini, R.H.; Vieira, R.; Moreira, A.F. *"Fundamentos de Sistemas Multiagentes"* Anais do XXI Congresso da Sociedade Brasileira de Computação, Fortaleza, CE, BR, 30/07 a 03/08/2001.

[17] Lopes Jr.; C.R., Esmin, A.A.A.; Aoki, A.R.; Lambert-Torres, G. *"Multi-Agent Model for Power Substation Restoration"*, Fifth IASTED International Conference on Power and Energy Systems, Tampa, Florida, USA, November 19 - 22, 2001.

[18] A. M. P. Rezende, Uma Abordagem Tabular para o Desenvolvimento de Agentes Inteligentes Reativos, Tese de Mestrado, Instituto Tecnológico da Aeronáutica, Novembro 1999.

[19] Esmin, A.A.A.; Aoki, A.R.; Lopes Jr.; C.R., Lambert-Torres, G. "*Multi-Agent Simulation and Educational Tool for Power System Operation"*,

VII International Conference on Engineering and Technology Education (INTERTECH 2002), Santos, Brazil, March 17 – 20, 2002.

[20] Carvalho, A.C.C; *et al. Disjuntores e Chaves - Aplicação em Sistemas de Potência.* Niterói: Editora da Universidade Federal Fluminense, 1995.

[21] Horowitz, S.H.; Phadke, A.G. *Power System Relaying - Second Edition.* Research Studies Press, 1995.

[22] D'Ajuz, A. *Equipamentos Elétricos - Especificação e Aplicação em Subestações de Alta Tensão.* Rio de Janeiro: FURNAS Centrais elétricas 1985.

[23] CEMIG. *Aplicação de Medição em Subestações - Requisitos Básicos.* Belo Horizonte: CEMIG 1987.

[24] ELETROBRÁS. *Proteção de Sistemas Aéreos de Distribuição.* Rio de Janeiro: Editora Campus 1982.

[25] Caminha, A.C. *Introdução à Proteção dos Sistemas Elétricos.* São Paulo: Editora Edgard Blücher 1977.

[26] Adibi, M.M.; Fink, L.H. "Power System Restoration Planning", *IEEE Trans. Power Systems*, vol. 9 (1), February 1994, pp. 22-28.

[27] Adibi, M.M.; *et al.* "Power System Restoration - A Task Force Report", *IEEE Transactions on Power Systems*, Vol. PWRS-2 (2), May 1987, pp. 271-277.

[28] Nielsen, K.E. *et al.* "System Operations Challenges", *IEEE Transactions on Power Systems*, Vol. 3 (1), February 1988, pp. 118-126.

[29] CEMIG. Manual de controle do sistema, critérios para o restabelecimento de linhas de transmissão e alimentadores de distribuição do sistema CEMIG. Belo Horizonte: CEMIG, 1991 (NO-OP/OS2-03.002).

[30] CEMIG. Manual de controle do sistema, critérios para o restabelecimento da malha regional oeste - 138 kV. Belo Horizonte: CEMIG, 1988 (IO-OP/OS2-02.006i).

[31] CEMIG. Manual de controle do sistema, critérios para o restabelecimento da SE Jaguara 500 kV. Belo Horizonte: CEMIG, 1990 (IO-OP/OS2-03.028g).

[32] CEMIG. Manual de controle do sistema, critérios para o restabelecimento da SE Jaguara 345 kV. Belo Horizonte: CEMIG, 1991 (IO-OP/OS2-03.029e).

[33] CEMIG. Manual de controle do sistema, critérios para o restabelecimento da SE Barbacena 2. Belo Horizonte: CEMIG, 1989 (IO-OP/OS2-03.038d).

[34] G.M. Ribeiro, *Sistemas Especialistas para Restabelecimento Automático de Subestações*. Tese de Mestrado, Escola Federal de Engenharia de Itajubá, Maio 1993.

[35] Jardini, J. A., Sistemas Digitais para Automação da Geração, Transmissão e Distribuição de Energia Elétrica. São Paulo, s.ed. 1996.

[36] S. Pandit, S.A. Soman, S.A. Khaparde *Object-Oriented Design for Power System Applications*. IEEE Computer Applications in Power, Volume 13 (4) October 2000, pp. 43-47.

[37] Meyer, B. *Object Oriented Software Constructio, 2nd Ed,* Prentice Hall, New Jersey 07458

[38] Rumbaugh, J., Blaha, M., Premerlani, W. Eddy, F., Lorensen, W. *Modelagem e Projetos Baseados em Objetos.* Ed. Campus, 1994.

[39] S. Pandit, S.A. Soman, S.A. Khaparde. Object-Oriented Network Topology Processor. IEEE Computer Applications in Power, Volume 14 (2) April 2001, pp. 42-46.

[40] D. Becker, H. Falk, J. Gillerman, S. Mauser, R. Podmore, L. Schneberger. *Standards-Based Approach Integrates Utility Applications*. IEEE Computer Applications in Power, Volume 13 (4) October 2000, pp. 13-20.

[41] S.K. Abidi, A.K. David. *An Object-Oriented Intelligent Approach to System Restoration*. Proceedings of the International Conference on Intelligent System Application to Power Systems (ISAP'99). April 1999, Rio de Janeiro – Brazil, pp. 61-65.

[42] G. Booch. *Object-Oriented Analysis and Design with Applications,* 2nd Edition. Reading, MA: Addison Wesley. 1994.

[43] Aoki, A. Rasi., Lambert-Torres, G., de Souza, L. E., Moraes, C. H. V., *Solving Power Substation Restoration using Intelligent Planning.* Proceedings of the IASTED International Conference, Power and Energy Systems, November 8-10, 1999 – Las Vegas – USA.

[44] D. Becker, H. Falk, J. Gillerman, S. Mauser, R. Podmore, L. Schneberger. *Standards-Based Approach Integrates Utility Applications*. IEEE Computer Applications in Power, Volume 13 (4) October 2000, pp. 13-20.

[45] M. Amin. *Toward Self-Healing Energy Infrastructure Systems*. IEEE Computer Applications in Power, Volume 14 (1) January 2001, pp. 20-28.

[46] M. S. Tsai. *Conceptual Design of Distributed Rule Based Expert System for Distribution Automation.* Proceedings of the International Conference on Intelligent System Application to Power Systems (ISAP'99). April 1999, Rio de Janeiro – Brazil, pp. 402-406.

[47] A.R. Aoki; G. Lambert-Torres, L.E. de Souza, C.H.V. de Moraes. *Intelligent Planner for Electrical Substations Restoration*, Progress in Simulation, Modeling, Analysis and Synthesis of Modern Electrical and Electronic Devices and Systems, by N.E. Mastorakis, World Scientific and Engineering Society Press, ISBN 960-8052-08-4, pp. 94-99, 1999.

[48] Pereira, R. M., *Agentes de Software e Java: Uma Poderosa Combinação*, Developer's Magazine, Axcel Books do Brasil Ed. Ltda, Ano 3, Volume 34, pp 12-14, Junho 1999.

[49] Jaworski, J., *Java 1.2 Unleashed,* SAMS, 1999.

[50] A. R. Aoki, Planejador Inteligente para o Restabelecimento de Subestações Elétricas, Tese de Mestrado, Escola Federal de Engenharia de Itajubá, Abril de 1999.

ANEXO 1 – BANCO DE DADOS DO MODELO COMPLETO DA SUBESTAÇÃO

Td deviceID	deviceName	deviceType	deviceVoltage	deviceLimit	deviceArgs	deviceLinkage
1	LT-1	Line	88/138			219
2	LT-2	Line	88/138			219
3	TR-1	Transformer	13.8			1
4	TR-2	Transformer	13.8			2
5	BA-1	Bus	13.8			3
6	BA-2	Bus	13.8			4
7	BA-3	Bus	13.8			3
8	BA-4	Bus	13.8			4
9	ESU-131	Feeder	13.8			5
10	ESU-132	Feeder	13.8			5
11	ESU-139	Feeder	13.8			5
12	ESU-145	Feeder	13.8			5
13	ESU-133	Feeder	13.8			6
14	ESU-134	Feeder	13.8			6
15	ESU-140	Feeder	13.8			6
16	ESU-143	Feeder	13.8			6
17	ESU-136	Feeder	13.8			7
18	ESU-137	Feeder	13.8			7
19	ESU-138	Feeder	13.8			7
20	ESU-144	Feeder	13.8			7
21	ESU-130	Feeder	13.8			8
22	ESU-135	Feeder	13.8			8
23	ESU-141	Feeder	13.8			8
24	ESU-142	Feeder	13.8			8
25	BCA-1	Capacitor	13.8			5
26	BCA-2	Capacitor	13.8			6
27	BCA-3	Capacitor	13.8			7
28	BCA-4	Capacitor	13.8			8
29	TR-1_87	ProtectedRelay			248,74	3
30	TR-1 BA-1_51N	ProtectedRelay			126,143	3
31	TR-1 BA-3_51N	ProtectedRelay			16,143	3
32	TR-1_63	ProtectedRelay			181,117	3
33	TR-1_63c	ProtectedRelay			219,117	3
34	TR-1_vs	ProtectedRelay			181,159	3
35	TR-2_87	ProtectedRelay			248,74	4
36	TR-2 BA-2_51N	ProtectedRelay			126,143	4
37	TR-2 BA-4_51N	ProtectedRelay			16,143	4
38	TR-2_63	ProtectedRelay			181,117	4
39	TR-2_63c	ProtectedRelay			219,117	4
40	TR-2_vs	ProtectedRelay			181,159	4
41	LT-1_27	ProtectedRelay			102,67	1
42	LT-1_50/51	ProtectedRelay			102,110	1
43	LT-2_27	ProtectedRelay			102,67	2
44	LT-2_50/51	ProtectedRelay			102,110	2
45	BA-1_27	ProtectedRelay			90,135	5
46	BA-1_51	ProtectedRelay			90,81	5
47	BA-1_51N	ProtectedRelay			129,81	5
48	BA-1_62_BF	ProtectedRelay			207,81	5

Td deviceID	deviceName	deviceType	deviceVoltage	deviceLimit	deviceArgs	deviceLinkage
49	BA-1_87	ProtectedRelay			168,81	5
50	BA-1_59	ProtectedRelay			129,135	5
51	BA-2_27	ProtectedRelay			90,135	6
52	BA-2_51	ProtectedRelay			90,81	6
53	BA-2_51N	ProtectedRelay			129,81	6
54	BA-2_62_BF	ProtectedRelay			207,81	6
55	BA-2_87	ProtectedRelay			168,81	6
56	BA-2_59	ProtectedRelay			129,135	6
57	BA-3_27	ProtectedRelay			90,135	7
58	BA-3_51	ProtectedRelay			90,81	7
59	BA-3_51N	ProtectedRelay			129,81	7
60	BA-3_62_BF	ProtectedRelay			207,81	7
61	BA-3_87	ProtectedRelay			168,81	7
62	BA-3_59	ProtectedRelay			129,135	7
63	BA-4_27	ProtectedRelay			90,135	8
64	BA-4_51	ProtectedRelay			90,81	8
65	BA-4_51N	ProtectedRelay			129,81	8
66	BA-4_62_BF	ProtectedRelay			207,81	8
67	BA-4_87	ProtectedRelay			168,81	8
68	BA-4_59	ProtectedRelay			129,135	8
69	ESU-131_50/51	ProtectedRelay			142,82	9
70	ESU-131_50/51N	ProtectedRelay			181,82	9
71	ESU-131_79	ProtectedRelay			142,37	9
72	ESU-132_50/51	ProtectedRelay			142,82	10
73	ESU-132_50/51N	ProtectedRelay			181,82	10
74	ESU-132_79	ProtectedRelay			142,37	10
75	ESU-139_50/51	ProtectedRelay			142,82	11
76	ESU-139_50/51N	ProtectedRelay			181,82	11
77	ESU-139_79	ProtectedRelay			142,37	11
78	ESU-145_50/51	ProtectedRelay			142,82	12
79	ESU-145_50/51N	ProtectedRelay			181,82	12
80	ESU-145_79	ProtectedRelay			142,37	12
81	ESU-133_50/51	ProtectedRelay			142,82	13
82	ESU-133_50/51N	ProtectedRelay			181,82	13
83	ESU-133_79	ProtectedRelay			142,37	13
84	ESU-134_50/51	ProtectedRelay			142,82	14
85	ESU-134_50/51N	ProtectedRelay			181,82	14
86	ESU-134_79	ProtectedRelay			142,37	14
87	ESU-140_50/51	ProtectedRelay			142,82	15
88	ESU-140_50/51N	ProtectedRelay			181,82	15
89	ESU-140_79	ProtectedRelay			142,37	15
90	ESU-143_50/51	ProtectedRelay			142,82	16
91	ESU-143_50/51N	ProtectedRelay			181,82	16
92	ESU-143_79	ProtectedRelay			142,37	16
93	ESU-136_50/51	ProtectedRelay			142,82	17
94	ESU-136_50/51N	ProtectedRelay			181,82	17
95	ESU-136_79	ProtectedRelay			142,37	17
96	ESU-137_50/51	ProtectedRelay			142,82	18
97	ESU-137_50/51N	ProtectedRelay			181,82	18
98	ESU-137_79	ProtectedRelay			142,37	18
99	ESU-138_50/51	ProtectedRelay			142,82	19
100	ESU-138_50/51N	ProtectedRelay			181,82	19

Td deviceID	deviceName	deviceType	deviceVoltage	deviceLimit	deviceArgs	deviceLinkage
101	ESU-138_79	ProtectedRelay			142,37	19
102	ESU-144_50/51	ProtectedRelay			142,82	20
103	ESU-144_50/51N	ProtectedRelay			181,82	20
104	ESU-144_79	ProtectedRelay			142,37	20
105	ESU-130_50/51	ProtectedRelay			142,82	21
106	ESU-130_50/51N	ProtectedRelay			181,82	21
107	ESU-130_79	ProtectedRelay			142,37	21
108	ESU-135_50/51	ProtectedRelay			142,82	22
109	ESU-135_50/51N	ProtectedRelay			181,82	22
110	ESU-135_79	ProtectedRelay			142,37	22
111	ESU-141_50/51	ProtectedRelay			142,82	23
112	ESU-141_50/51N	ProtectedRelay			181,82	23
113	ESU-141_79	ProtectedRelay			142,37	23
114	ESU-142_50/51	ProtectedRelay			142,82	24
115	ESU-142_50/51N	ProtectedRelay			181,82	24
116	ESU-142_79	ProtectedRelay			142,37	24
117	BCA-1_50/51	ProtectedRelay			126,85	25
118	LT-1_V	MeasuringDevice			V,false	1
119	BCA-1_66	ProtectedRelay			126,38	25
120	BCA-2_50/51	ProtectedRelay			126,85	26
121	LT-2_V	MeasuringDevice			V,false	2
122	BCA-2_66	ProtectedRelay			126,38	26
123	BCA-3_50/51	ProtectedRelay			126,85	27
124	BA-1_AN	MeasuringDevice			A,false	5
125	BCA-3_66	ProtectedRelay			126,38	27
126	BCA-4_50/51	ProtectedRelay			126,85	28
127	BA-2_AN	MeasuringDevice			A,false	6
128	BCA-4_66	ProtectedRelay			126,38	28
129	DJ-1	CircuitBreaker	88/138		80,67,45,69,false	3,1
130	DJ-2	CircuitBreaker	88/138		692,67,45,69,false	4,2
131	DJ-3	CircuitBreaker	13.8		56,151,55,162,false	3,7
132	DJ-4	CircuitBreaker	13.8		668,151,103,162,false	4,6
133	DJ-5	CircuitBreaker	13.8		104,151,103,162,false	3,5
134	DJ-6	CircuitBreaker	13.8		716,151,55,162,false	4,8
135	DJ-7	CircuitBreaker	13.8		382,198,0,0,true	5,6
136	DJ-8	CircuitBreaker	13.8		382,378,0,0,true	7,8
137	DJ-9	CircuitBreaker	13.8		122,223,69,40,false	25
138	DJ-10	CircuitBreaker	13.8		650,223,69,40,false	27
139	DJ-11	CircuitBreaker	13.8		122,403,69,40,false	26
140	DJ-12	CircuitBreaker	13.8		650,403,69,40,false	28
141	DJ-13	CircuitBreaker	13.8		173,223,85,39,false	9
142	DJ-14	CircuitBreaker	13.8		220,223,85,39,false	10
143	DJ-15	CircuitBreaker	13.8		268,223,85,39,false	11
144	DJ-16	CircuitBreaker	13.8		314,223,85,39,false	12
145	DJ-17	CircuitBreaker	13.8		458,223,85,39,false	13
146	DJ-18	CircuitBreaker	13.8		505,223,85,39,false	14
147	DJ-19	CircuitBreaker	13.8		553,223,85,39,false	15
148	DJ-20	CircuitBreaker	13.8		599,223,85,39,false	16
149	DJ-21	CircuitBreaker	13.8		173,403,85,39,false	19
150	DJ-22	CircuitBreaker	13.8		220,403,85,39,false	18
151	DJ-23	CircuitBreaker	13.8		268,403,85,39,false	17
152	DJ-24	CircuitBreaker	13.8		314,403,85,39,false	20

Td deviceID	deviceName	deviceType	deviceVoltage	deviceLimit	deviceArgs	deviceLinkage
153	DJ-25	CircuitBreaker	13.8		458,403,85,39,false	23
154	DJ-26	CircuitBreaker	13.8		505,403,85,39,false	24
155	DJ-27	CircuitBreaker	13.8		553,403,85,39,false	22
156	DJ-28	CircuitBreaker	13.8		599,403,85,39,false	21
157	SC-1	Switch	88/138			1
158	SC-2	Switch	88/138			2
159	SC-3	Switch	13.8			25
160	SC-4	Switch	13.8			27
161	SC-5	Switch	13.8			26
162	SC-6	Switch	13.8			28
163	LT-1_A	MeasuringDevice			A,True	1
164	LT-2_A	MeasuringDevice			A,false	2
166	TR-1 BA-1_AN	MeasuringDevice			A,false	3
167	TR-1 BA-3_AN	MeasuringDevice			A,true	3
169	TR-2 BA-2_AN	MeasuringDevice			A,false	4
170	TR-2 BA-4_AN	MeasuringDevice			A,false	4
171	BA-1_A	MeasuringDevice			A,false	5
173	BA-1_V	MeasuringDevice			V,false	5
174	BA-2_A	MeasuringDevice			A,false	6
176	BA-2_V	MeasuringDevice			V,false	6
177	BA-3_A	MeasuringDevice			A,false	7
179	BA-3_V	MeasuringDevice			V,false	7
180	BA-4_A	MeasuringDevice			A,false	8
182	BA-4_V	MeasuringDevice			kV,false	8
183	ESU-131_A	MeasuringDevice			A,false	9
184	ESU-131_AN	MeasuringDevice			A,true	9
185	ESU-132_A	MeasuringDevice			A,false	10
186	ESU-132_AN	MeasuringDevice			A,false	10
187	ESU-139_A	MeasuringDevice			A,false	11
188	ESU-139_AN	MeasuringDevice			A,false	11
189	ESU-145_A	MeasuringDevice			A,false	12
190	ESU-145_AN	MeasuringDevice			A,false	12
191	ESU-136_A	MeasuringDevice			A,true	17
192	ESU-136_AN	MeasuringDevice			A,false	17
193	ESU-137_A	MeasuringDevice			A,false	18
194	ESU-137_AN	MeasuringDevice			A,false	18
195	ESU-138_A	MeasuringDevice			A,false	19
196	ESU-138_AN	MeasuringDevice			A,false	19
197	ESU-144_A	MeasuringDevice			A,true	20
198	ESU-144_AN	MeasuringDevice			A,false	20
199	ESU-133_A	MeasuringDevice			A,false	13
200	ESU-133_AN	MeasuringDevice			A,false	13
201	ESU-134_A	MeasuringDevice			A,false	14
202	ESU-134_AN	MeasuringDevice			A,true	14
203	ESU-140_A	MeasuringDevice			A,false	15
204	ESU-140_AN	MeasuringDevice			A,true	15
205	ESU-143_A	MeasuringDevice			A,false	16
206	ESU-143_AN	MeasuringDevice			A,false	16
207	ESU-130_A	MeasuringDevice			A,false	21
208	ESU-130_AN	MeasuringDevice			A,false	21
209	ESU-135_A	MeasuringDevice			A,false	22
210	ESU-135_AN	MeasuringDevice			A,false	22

Td deviceID	deviceName	deviceType	deviceVoltage	deviceLimit	deviceArgs	deviceLinkage
211	ESU-141_A	MeasuringDevice			A,false	23
212	ESU-141_AN	MeasuringDevice			A,false	23
213	ESU-142_A	MeasuringDevice			A,false	24
214	ESU-142_AN	MeasuringDevice			A,false	24
215	BCA-1_A	MeasuringDevice			A,false	25
216	BCA-2_A	MeasuringDevice			A,false	26
217	BCA-3_A	MeasuringDevice			A,false	27
218	BCA-4_A	MeasuringDevice			A,false	28
219	Substation 1	Substation				0
220	BA-3_AN	MeasuringDevice			A,false	7
221	BA-4_AN	MeasuringDevice			A,false	8

ANEXO 2 – CÓDIGO DO MODELO DA SUBESTAÇÃO

Device.java

```
public class Device {
  // nome do Device
  protected String name;
  // estado atual do Device
  protected boolean status; //true => ligado, false => desligado
  // node representatando sua posição na hierarquia;
  protected DefaultMutableTreeNode node;

  public Device(boolean status, String name) {
    this.name = name;
    this.status = status;
  }

  public void setNode(DefaultMutableTreeNode node) { this.node = node; }
  public DefaultMutableTreeNode getNode() { return node; }

  public Component getRepresentation() { return null; };
  public Component getDefaultRepresentation() { return null; };

  public boolean getCurrentStatus() { return status; }
  public void setCurrentStatus(boolean status) { this.status = status; }

  public String getName() { return name; }
  public void setName(String name) { this.name = name; }
  public String toString() {
    return this.name + " - " + super.toString();
  }
  public static String getClassName() { return "Device"; }

}
```

EquipmentDevice.Java

```
public class EquipmentDevice extends Device {

  public EquipmentDevice(boolean status, String name) {
    super(status, name);
  }
}
```

SwitchingDevice.Java

```
public class SwitchingDevice extends Device {
  protected DefaultMutableTreeNode extraNode;
  private boolean isLinkageDevice = false;

  public SwitchingDevice(boolean status, String name) {
    super(status, name);
  }
  protected void close() { super.status = true; }
  protected void open() { super.status = false; }

  public boolean isLinkageDevice() { return this.isLinkageDevice; }
  public void setLinkageDevice(boolean isLinkageDevice) { this.isLinkageDevice =
                                                          isLinkageDevice; }
  public void setExtraNode(DefaultMutableTreeNode extraNode) {
    this.extraNode = extraNode;
    this.isLinkageDevice = true;
  }
  public DefaultMutableTreeNode getExtraNode() { return this.extraNode; }
}
```

MeasuringDevice.Java

```
public class MeasuringDevice extends Device {
  private JLabel nameRepresentation;
  private JTextField valueRepresentation;
  private Object value;
```

```
  private boolean isDefault, isUserEditable;
  private String unit;

  public MeasuringDevice(boolean status, String name, Object value, String unit, boolean
                         isDefault) {
    super(status, name);
    this.unit = unit;
    this.isDefault = isDefault;

    nameRepresentation = new JLabel();
    setName(name);
    // opções de vizualização
    nameRepresentation.setPreferredSize(new Dimension(90, 21));
    nameRepresentation.setMaximumSize(new Dimension(100, 30));
    // opções de vizualização
    valueRepresentation = new JTextField();
    setValue(value);
    valueRepresentation.addFocusListener(new FocusAdapter() {
      public void focusLost(FocusEvent e) { refreshRepresentation(); }
    });
    valueRepresentation.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
        try {
          setValue(new Double(((JTextField)e.getSource()).getText()));
        } catch (java.lang.NumberFormatException e1) {
          JOptionPane.showMessageDialog(valueRepresentation, "O valor entrado é inválido!",
              "Erro ao alterar valor:", JOptionPane.ERROR_MESSAGE);
        }
      }
    });

    valueRepresentation.setBorder(BorderFactory.createLoweredBevelBorder());
    valueRepresentation.setPreferredSize(new Dimension(90, 21));
    valueRepresentation.setMaximumSize(new Dimension(100, 30));
    valueRepresentation.setHorizontalAlignment(SwingConstants.CENTER);
  }

  public void setUserEditable(boolean isUserEditable) {
    this.isUserEditable = isUserEditable;
    valueRepresentation.setEditable(isUserEditable);
    refreshRepresentation();
  }

  public void refreshRepresentation() {
    if (nameRepresentation != null)
      nameRepresentation.setText(name);
    if (valueRepresentation != null)
      valueRepresentation.setText(value.toString()+ " " + unit);
  }

  public boolean isUserEditable() { return isUserEditable; }
  public boolean isDefault() { return isDefault; }
  public void setName(String name) {
    this.name = name;
    refreshRepresentation();
  }

  public Object getValue() { return value; }
  public void setValue(Object value) {
    this.value = value;
    refreshRepresentation();
  }

  public JLabel getNameRepresentation() { return nameRepresentation; }
  public JTextField getValueRepresentation() { return valueRepresentation; }
}
```

Bus.Java

```
public class Bus extends EquipmentDevice {

  public Bus(boolean status, String name) {
    super(status, name);
  }
  public static String  getClassName() { return "Barramento"; }
}
```

ByPassSwitch.Java

```
public class ByPassSwitch extends SwitchingDevice {

  public ByPassSwitch(boolean status, String name) {
    super(status, name);
  }
  public static String  getClassName() { return "Chave de ByPass"; }
}
```

Capacitor.Java

```
public class Capacitor extends EquipmentDevice {

  public Capacitor(boolean status, String name) {
    super(status, name);
  }
  public static String getClassName() { return "Banco de Capacitores"; }
}
```

CircuitBreaker.Java

```
public class CircuitBreaker extends SwitchingDevice {
  private JLabel defaultRepresentation = new JLabel();
  private JLabel representation = new JLabel();
  private boolean isUserEditable;

  public CircuitBreaker(boolean status, String name) {
    super(status, name);
  }

  public void setUserEditable(boolean isUserEditable) {
    this.isUserEditable = isUserEditable;
  }

  public boolean isUserEditable() { return isUserEditable; }
  // instancia a representação do CircuitBreaker
  public void setDefaultRepresentationLocation(int posX, int posY) {
    defaultRepresentation.setLocation(posX, posY);
  }

  public void setRepresentation(int posX, int posY, int defaultPosX, int defaultPosY, boolean
      isHorizontal) {
    representation.setBorder(BorderFactory.createLineBorder(Color.black,1));
    defaultRepresentation.setBorder(BorderFactory.createLineBorder(Color.black,1));
    if (isHorizontal) {
      representation.setSize(25,13);
      defaultRepresentation.setSize(25,13);
    } else {
      representation.setSize(13,25);
      defaultRepresentation.setSize(13,25);
    }
    representation.setToolTipText(name);
    defaultRepresentation.setToolTipText(name);
    representation.setOpaque(true);
    defaultRepresentation.setOpaque(true);
    representation.setLocation(posX, posY);
    defaultRepresentation.setLocation(defaultPosX, defaultPosY);

    representation.addMouseListener(new java.awt.event.MouseAdapter() {
      public void mouseClicked(MouseEvent e) { this_mouseClicked(e); }
    });

    this.refreshRepresentation();
  }

  // metódo ao clicar o mouse sobre a representação
  void this_mouseClicked(MouseEvent e) {
    if (e.getClickCount() >= 2 && this.isUserEditable) {
      CircuitBreakerDialog dlg = new CircuitBreakerDialog(this.getClassName() + "  " +
          this.name);
      dlg.show();
      if (dlg.getValue() != null)
        this.setCurrentStatus(dlg.getValue().booleanValue());
    }
  }
```

```
  // atualiza a representação
  void refreshRepresentation() {
    if (status) {
      representation.setBackground(Color.red);
      defaultRepresentation.setBackground(Color.red);
    } else {
      representation.setBackground(Color.green);
      defaultRepresentation.setBackground(Color.green);
    }
  }
  public Component getRepresentation() {
    return representation;
  }
  public Component getDefaultRepresentation() {
    return defaultRepresentation;
  }

  public void setCurrentStatus(boolean status) {
    this.status = status;
    this.refreshRepresentation();
  }
  public void setName(String name) {
    this.name = name;
    representation.setToolTipText(name);
    defaultRepresentation.setToolTipText(name);
  }

  public void close() {
    super.close();
    refreshRepresentation();
  }
  public void open() {
    super.open();
    refreshRepresentation();
  }
  public static String getClassName() { return "Disjuntor"; }
}
```

CircuitBreakerDialog.Java

```
public class CircuitBreakerDialog extends JDialog {
  JButton jButton1 = new JButton("Abrir");
  JButton jButton2 = new JButton("Fechar");
  JButton jButton3 = new JButton("Cancelar");
  FlowLayout flowLayout1 = new FlowLayout();

  Boolean value;

  public CircuitBreakerDialog(String title) {
    super();
    this.setTitle(title);
    try {
      jbInit();
    }
    catch(Exception e) {
      e.printStackTrace();
    }
  }
  private void jbInit() throws Exception {
    this.setModal(true);
    this.setResizable(false);
    this.setSize(new Dimension(150, 100 + 25));
    this.setLocation(utils.MyTools.getWindowAtCenter(this.getSize()));
    jButton1.setPreferredSize(new Dimension(85,27));
    jButton1.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(ActionEvent e) {
        jButton1_actionPerformed(e);
      }
    });
    jButton2.setPreferredSize(new Dimension(85,27));
    jButton2.addActionListener(new java.awt.event.ActionListener() {
      public void actionPerformed(ActionEvent e) {
        jButton2_actionPerformed(e);
      }
    });
    jButton3.setPreferredSize(new Dimension(85,27));
    jButton3.addActionListener(new java.awt.event.ActionListener() {
```

```
      public void actionPerformed(ActionEvent e) {
        jButton3_actionPerformed(e);
      }
    });
    this.getContentPane().setLayout(flowLayout1);
    this.getContentPane().add(jButton1);
    this.getContentPane().add(jButton2);
    this.getContentPane().add(jButton3);
  }
  public Boolean getValue() { return value; }

  void jButton1_actionPerformed(ActionEvent e) {
    value = new Boolean(false);
    this.setVisible(false);
  }

  void jButton2_actionPerformed(ActionEvent e) {
    value = new Boolean(true);
    this.setVisible(false);
  }

  void jButton3_actionPerformed(ActionEvent e) {
    value = null;
    this.setVisible(false);
  }
}
```

Feeder.Java

```
public class Feeder extends EquipmentDevice {

  public Feeder(boolean status, String name) {
    super(status, name);
  }
  public static String getClassName() { return "Alimentador"; }
}
```

GroundingSwitch.java

```
public class GroundingSwitch extends SwitchingDevice {

  public GroundingSwitch(boolean status, String name) {
    super(status, name);
  }
  public static String  getClassName() { return "Chave de Aterramento"; }
}
```

EquipmentDevice.java

```
public class Line extends EquipmentDevice {

  public Line(boolean status, String name) {
    super(status, name);
  }
  public static String getClassName() { return "Linha"; }
}
```

MeasuresPanel.Java

```
public class MeasuresPanel extends JPanel {
  private Vector measures = new Vector();
  FlowLayout flowLayout = new FlowLayout(FlowLayout.CENTER);

  public MeasuresPanel(Enumeration measures) {
    super();
    //this.setBackground(Color.blue);
    this.setMinimumSize(new Dimension(60, 20));
    this.setMinimumSize(new Dimension(75, 50));
    this.setMaximumSize(new Dimension(80, 2000));
    this.setLayout(flowLayout);
    if (measures != null) {
      while (measures.hasMoreElements())
        this.measures.add(measures.nextElement());
    }
```

```
  }
  public void placeMeasures(DefaultMutableTreeNode device) {
    this.removeAll();
    this.setMinimumSize(new Dimension(60, 20));
    this.setMinimumSize(new Dimension(75, 50));
    this.setMaximumSize(new Dimension(80, 60));
    this.setLayout(flowLayout);
    if (!device.isLeaf()) {
      Enumeration children = device.children();
      while (children.hasMoreElements()) {
        Device tmp_device =
(Device)((DefaultMutableTreeNode)children.nextElement()).getUserObject();
        if (tmp_device instanceof MeasuringDevice) {
          MeasuringDevice tmp_measure = (MeasuringDevice)tmp_device;
          if (tmp_measure.getNameRepresentation() != null &&
            tmp_measure.getValueRepresentation() != null) {
            this.add(tmp_measure.getNameRepresentation());
            this.add(tmp_measure.getValueRepresentation());
          }
        }
      }
      this.validate();
      this.repaint();
    }
  }
}
```

ProtectedRelay.Java

```
public class ProtectedRelay extends Device {
  private ImageIcon img_true = new ImageIcon("protectedrelay_true.gif");
  private ImageIcon img_false = new ImageIcon("protectedrelay_false.gif");
  private JLabel representation = new JLabel();
  private boolean isUserEditable = false;

  public ProtectedRelay(boolean status, String name) {
    super(status, name);
  }
  public void setUserEditable(boolean isUserEditable) { this.isUserEditable = isUserEditable;
}
  public boolean isUserEditable() { return isUserEditable; }

  public static String getClassName() { return "Proteção"; }
  public void setRepresentation(int posX, int posY) {
    representation.setToolTipText(name);
    representation.setLocation(posX, posY);
    representation.setOpaque(true);
    representation.setSize(img_true.getIconWidth(), img_true.getIconHeight());

    representation.addMouseListener(new java.awt.event.MouseAdapter() {
      public void mouseClicked(MouseEvent e) { this_mouseClicked(e); }
    });

    this.refreshRepresentation();
  }
  // metódo ao clicar o mouse sobre a representação
  void this_mouseClicked(MouseEvent e) {
    if (e.getClickCount() >= 2 && isUserEditable) {
      ProtectedRelayDialog dlg = new
          ProtectedRelayDialog(((ProtectedRelay)this).getClassName() + "  " + this.name,
          this.getCurrentStatus());
      dlg.show();
      if (dlg.getValue() != null)
        this.setCurrentStatus(dlg.getValue().booleanValue());
    }
  }
  // atualiza a representação
  void refreshRepresentation() {
    if (status)
      representation.setIcon(img_true);
    else
      representation.setIcon(img_false);
    representation.validate();
  }
  public void setName(String name) {
    super.setName(name);
    representation.setToolTipText(name);
  }
```

```
  public void setCurrentStatus(boolean value) {
    super.setCurrentStatus(value);
    refreshRepresentation();
  }
  public Component getDefaultRepresentation() {
    return representation;
  }
}
```

Susbstation.Java

```
public class Substation  {
  private Hashtable devices;
  private String name;

  public Substation(String name, Hashtable devices) {
    this.name = name;
    this.devices = devices;
  }
  public String getName() { return name; }
}
```

Switch.Java

```
public class Switch extends SwitchingDevice {

  public Switch(boolean status, String name) {
    super(status, name);
  }
  public static String  getClassName() { return "Seccionadora"; }
}
```

Transformer.Java

```
public class Transformer extends EquipmentDevice {

  public Transformer(boolean status, String name) {
    super(status, name);
  }
  public static String getClassName() { return "Transformador"; }
}
```

ANEXO 3 – CÓDIGO DOS AGENTES

InterfaceAgent.Java

```
public InterfaceAgent(String name, String file) {
  this.name = name;
  this.jLabel1.setMaximumSize(new Dimension(60,30));
  this.jLabel1.setHorizontalAlignment(JLabel.CENTER);
  try {
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    connection = java.sql.DriverManager.getConnection("jdbc:odbc:Driver={MicroSoft Access
                 Driver (*.mdb)};DBQ=" + file, "", "");
  } catch (java.sql.SQLException e) {
    JOptionPane.showMessageDialog(null, "Ocorreu um erro ao carregar o Banco de
       Dados.\nCertifique-se de que este Banco de Dados é válido.", "Atenção!",
       JOptionPane.WARNING_MESSAGE);
    invalid = true;
  } catch (java.lang.ClassNotFoundException e) {
    e.printStackTrace();
    invalid = true;
  }
}

public JLabel getLabelStep() { return this.jLabel1; }

public void setProtectionsHistoryTable() throws java.sql.SQLException {
  try {
    Statement statement = connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
        ResultSet.CONCUR_READ_ONLY);
    protectionsSet = statement.executeQuery("SELECT * FROM Messages INNER JOIN
        ProtectionsHistory ON Messages.msgID = ProtectionsHistory.msgID ORDER BY
        Messages.msgID");
  }
  catch (java.sql.SQLException e) {
    invalid = true;
    e.printStackTrace();
    throw new java.sql.SQLException();
  }

public void setSwitchesHistoryTable() throws java.sql.SQLException {
  try {
    Statement statement = connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
        ResultSet.CONCUR_READ_ONLY);
    switchesSet = statement.executeQuery("SELECT * FROM Messages INNER JOIN SwitchesHistory ON
        Messages.msgID = SwitchesHistory.msgID ORDER BY Messages.msgID");
  }
  catch (java.sql.SQLException e) {
    invalid = true;
    e.printStackTrace();
    throw new java.sql.SQLException();
  }
}

public void setEquipmentHistoryTable() throws java.sql.SQLException {
  try {
    Statement statement = connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
        ResultSet.CONCUR_READ_ONLY);
    equipmentSet = statement.executeQuery("SELECT * FROM Messages INNER JOIN EquipmentHistory
        ON Messages.msgID = EquipmentHistory.msgID ORDER BY Messages.msgID");
    equipmentSet.last();
    totalRecords = equipmentSet.getRow();
  }
  catch (java.sql.SQLException e) {
    invalid = true;
    e.printStackTrace();
    throw new java.sql.SQLException();
  }
}

public void setMeasuresHistoryTable() throws java.sql.SQLException {
  try {
    Statement statement = connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
        ResultSet.CONCUR_READ_ONLY);
    measuresSet = statement.executeQuery("SELECT * FROM Messages INNER JOIN MeasuresHistory ON
        Messages.msgID = MeasuresHistory.msgID ORDER BY Messages.msgID");
  }
```

```
    catch (java.sql.SQLException e) {
      invalid = true;
      e.printStackTrace();
      throw new java.sql.SQLException();
    }
  }

  public void iaEventFired(IAgentEvent e) {
    trace("<" + Now.getCurrentTime() + "> " + name + ": got message from \"" +
        ((IAgent)e.getSource()).getName() + "\". Message type=\"" + e.getAction() + "\"") ;
    if (e.getAction().equals("query_measures") || e.getAction().equals("query_protections") ||
        e.getAction().equals("query_switches") || e.getAction().equals("query_equipment")) {
        if (e.getAction().equals("query_measures"))
            this.measures = MyTools.EnumerationToVector((Enumeration)e.getAttach());
        else if (e.getAction().equals("query_protections"))
            this.protections = MyTools.EnumerationToVector((Enumeration)e.getAttach());
        else if (e.getAction().equals("query_switches"))
            this.switches = MyTools.EnumerationToVector((Enumeration)e.getAttach());
        else if (e.getAction().equals("query_equipment"))
            this.equipment = MyTools.EnumerationToVector((Enumeration)e.getAttach());
        if (protections != null && switches != null && measures != null && equipment != null)
          start();
    }
    else if (e.getAction().equals("save_history")) {
        saveHistory((Hashtable)e.getAttach());
    }
    else if (e.getAction().equals("insert_history")) {
        insertHistory((Hashtable)e.getAttach());
    }
    else if (e.getAction().equals("step")) {
        step();
    }
  }

    public void loadDevicesTable() throws java.sql.SQLException {
      try {
        setProtectionsHistoryTable();
        setSwitchesHistoryTable();
        setMeasuresHistoryTable();
        setEquipmentHistoryTable();

        Statement statement = connection.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
            ResultSet.CONCUR_READ_ONLY);
        ResultSet devResultSet = statement.executeQuery("SELECT * FROM Devices");

        invalid = false;
        trace("<" + Now.getCurrentTime() + "> " + name + ": agent is not INVALID to RUN!") ;
        this.notifyIAgentEventListeners(new IAgentEvent(this, "load_model", devResultSet));

      } catch (java.sql.SQLException e) {
        JOptionPane.showMessageDialog(null, "Ocorreu um erro ao carregar a tabela 'Devices' do
            Banco de Dados.\nCertifique-se de que este Banco de Dados é válido.", "Atenção!",
            JOptionPane.WARNING_MESSAGE);
        invalid = true;
        throw new java.sql.SQLException();
      }
    }
    public boolean isInvalid() { return invalid; }
    public String getCurrentRow() {
      try {
      if (equipmentSet != null)
        if (equipmentSet.getRow() == 0)
          return this.totalRecords + "/" + this.totalRecords;
        else
          return equipmentSet.getRow() + "/" + this.totalRecords;
      else
        return "not loaded";
      } catch (java.sql.SQLException e) {
        return "erro";
      }
    }
    public void step() {
      System.out.println("Step()");
      synchronized (this) { notify(); }
    }
    public void run() {
      try {
        Hashtable protections_output = new Hashtable();
        Hashtable measures_output = new Hashtable();
        Hashtable switches_output = new Hashtable();
```

```
      Hashtable equipment_output = new Hashtable();

      protectionsSet.first();
      switchesSet.first();
      equipmentSet.first();
      measuresSet.first();
      while (!stopped) {
        if (!(switchesSet.isAfterLast() && protectionsSet.isAfterLast() &&
            measuresSet.isAfterLast() && equipmentSet.isAfterLast())) {
          if (protectionsSet != null && !protectionsSet.isAfterLast()) {
            protections_output.clear();
            for (int i=0; i < protections.size(); i++) {
              protections_output.put(protections.get(i), new
                  Boolean(protectionsSet.getBoolean((String)protections.get(i))));
            }
            this.notifyIAgentEventListeners(new IAgentEvent(this, "update_protections",
                  protections_output));
          }
          if (measuresSet != null && !measuresSet.isAfterLast()) {
            measures_output.clear();
            for (int i=0; i < measures.size(); i++) {
              measures_output.put(measures.get(i), new
                  Double(measuresSet.getDouble((String)measures.get(i))));
            }
            this.notifyIAgentEventListeners(new IAgentEvent(this, "update_measures",
                measures_output));
          }
          if (switchesSet != null && !switchesSet.isAfterLast()) {
            switches_output.clear();
            for (int i=0; i < switches.size(); i++) {
              switches_output.put(switches.get(i), new
                  Boolean(switchesSet.getBoolean((String)switches.get(i))));
            }
            this.notifyIAgentEventListeners(new IAgentEvent(this, "update_switches",
                switches_output));
          }
          if (equipmentSet != null && !equipmentSet.isAfterLast()) {
            equipment_output.clear();
            for (int i=0; i < equipment.size(); i++) {
              equipment_output.put(equipment.get(i), new
                  Boolean(equipmentSet.getBoolean((String)equipment.get(i))));
            }
            this.notifyIAgentEventListeners(new IAgentEvent(this, "update_equipment",
                switches_output));
          }
        }
        else {
        JOptionPane.showMessageDialog(getFramework(), name + ": não há mais linhas para ler.",
            "Atenção", JOptionPane.INFORMATION_MESSAGE);
        }
        this.jLabel1.setText("Registro: " + this.getCurrentRow());
        synchronized(this) { this.wait(); }
        if (!protectionsSet.isAfterLast()) protectionsSet.next();
        if (!measuresSet.isAfterLast()) measuresSet.next();
        if (!equipmentSet.isAfterLast()) equipmentSet.next();
        if (!switchesSet.isAfterLast()) switchesSet.next();
      }
      protectionsSet.close();
      switchesSet.close();
      equipmentSet.close();
      measuresSet.close();
      connection.close();
      System.out.println("Stopped reading the DataBank");
    } catch (java.sql.SQLException e) {
      e.printStackTrace();
    } catch (java.lang.InterruptedException e) {
      e.printStackTrace();
    }

  }
  //fim Thread
  public void saveHistory (Hashtable all_devices) {
    try {
      int currentRow, currentMsgID;
      if (!equipmentSet.isAfterLast() && !equipmentSet.isBeforeFirst()) {
        currentMsgID = equipmentSet.getInt("msgID");
        currentRow = equipmentSet.getRow();
      } else {
        currentMsgID = 0;
        currentRow = 0;
```

```
      }

      Statement statement = connection.createStatement();

      Hashtable tmp_devices = (Hashtable)all_devices.get("measures");
      //make QUERY
      String query = "UPDATE [MeasuresHistory] SET ";
      for (int i=0; i < measures.size(); i++) {
        if (i!=0)
          query = query + ",";
        query = query + "[" + measures.get(i) + "]=" + tmp_devices.get(measures.get(i));
      }
      query = query + " WHERE [msgID]=" + currentMsgID;
      if (statement.executeUpdate(query) == 1) {
        measuresSet.refreshRow();
      }

      tmp_devices = (Hashtable)all_devices.get("protections");
      //make QUERY
      query = "UPDATE [ProtectionsHistory] SET ";
      for (int i=0; i < protections.size(); i++) {
        if (i!=0)
          query = query + ",";
        query = query + "[" + protections.get(i) + "]=" + tmp_devices.get(protections.get(i));
      }
      query = query + " WHERE [msgID]=" + currentMsgID;
      if (statement.executeUpdate(query) == 1) {
        protectionsSet.refreshRow();
      }

      tmp_devices = (Hashtable)all_devices.get("equipment");
      //make QUERY
      query = "UPDATE [EquipmentHistory] SET ";
      for (int i=0; i < equipment.size(); i++) {
        if (i!=0)
          query = query + ",";
        query = query + "[" + equipment.get(i) + "]=" + tmp_devices.get(equipment.get(i));
      }
      query = query + " WHERE [msgID]=" + currentMsgID;
      if (statement.executeUpdate(query) == 1) {
        equipmentSet.refreshRow();
      }

      tmp_devices = (Hashtable)all_devices.get("switches");
      //make QUERY
      query = "UPDATE [SwitchesHistory] SET ";
      for (int i=0; i < switches.size(); i++) {
        if (i!=0)
          query = query + ",";
        query = query + "[" + switches.get(i) + "]=" + tmp_devices.get(switches.get(i));
      }
      query = query + " WHERE [msgID]=" + currentMsgID;
      if (statement.executeUpdate(query) == 1) {
        switchesSet.refreshRow();
      }

      this.jLabel1.setText("Registro: " + this.getCurrentRow());
      System.runFinalization();
    } catch (java.sql.SQLException e) {
      e.printStackTrace();
    }
  }
  public void insertHistory(Hashtable all_devices) {
    try {
      if (equipmentSet.isLast() || equipmentSet.isAfterLast()) {
        int currentMsgID;
        if (equipmentSet.isAfterLast())
          equipmentSet.last();
        currentMsgID = equipmentSet.getInt("msgID");

        ResultSet resultSet;
        Statement statement = connection.createStatement();
        String query;
        query = "INSERT INTO [Messages] ( [msgTime] ) VALUES ( '" + Now.getNow() + "' )";
        statement.executeUpdate(query);

        resultSet = statement.executeQuery("SELECT * FROM [Messages]");
        while (resultSet.next())
        currentMsgID = resultSet.getInt("msgID");
        resultSet.close();
```

```
        statement = connection.createStatement();
        query = "INSERT INTO [EquipmentHistory] ( [msgID] ) VALUES ( " + currentMsgID + " )";
        if (statement.executeUpdate(query) == 1) {
          equipmentSet.close();
          setEquipmentHistoryTable();
          equipmentSet.last();
        }
        query = "INSERT INTO [MeasuresHistory] ( [msgID] ) VALUES ( " + currentMsgID + ")";
        if (statement.executeUpdate(query) == 1) {
          measuresSet.close();
          setMeasuresHistoryTable();
          measuresSet.last();
        }
        query = "INSERT INTO [ProtectionsHistory] ( [msgID] ) VALUES ( " + currentMsgID + ")";
        if (statement.executeUpdate(query) == 1) {
          protectionsSet.close();
          setProtectionsHistoryTable();
          protectionsSet.last();
        }
        query = "INSERT INTO [SwitchesHistory] ( [msgID] ) VALUES ( " + currentMsgID + ")";
        if (statement.executeUpdate(query) == 1) {
          switchesSet.close();
          setSwitchesHistoryTable();
          switchesSet.last();
        }
        this.saveHistory(all_devices);
        if (all_devices.get("modifications") != null)
          this.notifyIAgentEventListeners(new IAgentEvent(this, "update_devices",
all_devices.get("modifications")));
      } else
      JOptionPane.showMessageDialog(getFramework(), "Você não pode inserir no meio do banco de
          dados. Vá até o fim apertando 'Step' e depois aperte o botão inserir.", "Atenção",
          JOptionPane.INFORMATION_MESSAGE);
    } catch (java.sql.SQLException e) {
      e.printStackTrace();
    }
  }
}
```

ModelAgent.Java

```
public void iaEventFired(IAgentEvent e) {
    trace("<" + Now.getCurrentTime() + "> " + name + ": got message from \"" +
        ((IAgent)e.getSource()).getName() + "\". Message type=\"" + e.getAction() + "\"") ;

    if (e.getAction().equals("update_switches")) {
        Hashtable attach = (Hashtable) e.getAttach();
        Enumeration tmp_switches = attach.keys();
        while (tmp_switches.hasMoreElements()) {
          String key = (String)tmp_switches.nextElement();

((SwitchingDevice)((DefaultMutableTreeNode)all_DevicesNodes.get(key)).getUserObject()).setCurr
entStatus(((Boolean)attach.get(key)).booleanValue());
        }
    } else if (e.getAction().equals("update_measures")) {
        Hashtable attach = (Hashtable) e.getAttach();
        Enumeration tmp_measures = attach.keys();
        while (tmp_measures.hasMoreElements()) {
          String key = (String)tmp_measures.nextElement();
((MeasuringDevice)((DefaultMutableTreeNode)all_DevicesNodes.get(key)).getUserObject()).setValu
e((Double)attach.get(key));
        }
    } else if (e.getAction().equals("update_protections")) {
        Hashtable attach = (Hashtable) e.getAttach();
        Enumeration tmp_protections = attach.keys();
        while (tmp_protections.hasMoreElements()) {
          String key = (String)tmp_protections.nextElement();
((ProtectedRelay)((DefaultMutableTreeNode)all_DevicesNodes.get(key)).getUserObject()).setCurre
ntStatus(((Boolean)attach.get(key)).booleanValue());
        }
    } else if (e.getAction().equals("update_devices")) {
        updateDevices((Hashtable)e.getAttach());
    } else if (e.getAction().equals("load_model")) {
        loadDevices((ResultSet)e.getAttach());
    } else if (e.getAction().equals("step_sugestions")) {
        if (e.getAttach() instanceof Boolean) {
```

```
            if (((Boolean)e.getAttach()).booleanValue() == false) {
                if (this.isUserEditable) {
                    this.saveModelState();
                    this.insertModelState(null);
                }
                else {
                    this.notifyIAgentEventListeners(new IAgentEvent(this, "step", null));
                }
            }
        } else if (e.getAttach() instanceof Hashtable) {
            if (this.isUserEditable) {
                this.saveModelState();
                this.insertModelState((Hashtable)e.getAttach());
                this.insertModelState(null);
            } else {
                this.notifyIAgentEventListeners(new IAgentEvent(this, "step", null));
            }
        }
        isWaintingAnswer = false;
    }
}
public void loadDevices(ResultSet devicesTable) {
  try {
    substations = new Hashtable();
    all_DevicesNodes = new Hashtable();
    Hashtable all_devices = new Hashtable();
    Hashtable linkageTable = new Hashtable();

    DefaultMutableTreeNode tmp_node;
    Device tmp_dev;
    String dev_type, dev_name, dev_param;
    StringTokenizer dev_params = null, dev_linkage = null;

    //cria instancias das Subestações (Substation)
    devicesTable.first();
    while (!devicesTable.isAfterLast()) {
      dev_type = devicesTable.getString("deviceType");
      dev_name = devicesTable.getString("deviceName");
      dev_linkage = new StringTokenizer(devicesTable.getString("deviceLinkage"), ",");
      Substation tmp_subs;
      if (dev_type.equals("Substation")) {
          tmp_subs = new Substation(dev_name, null);
      }
      else
          tmp_subs = null;

      if (tmp_subs != null) {
          tmp_node = new DefaultMutableTreeNode(tmp_subs);
          substations.put(tmp_subs.getName(), tmp_node);
          linkageTable.put(new Integer(devicesTable.getInt("deviceID")), tmp_node);
      }
      devicesTable.next();
    }

    //cria instancias das LINHAS (Line)
    devicesTable.first();
    while (!devicesTable.isAfterLast()) {
      dev_type = devicesTable.getString("deviceType");
      dev_name = devicesTable.getString("deviceName");
      dev_linkage = new StringTokenizer(devicesTable.getString("deviceLinkage"), ",");

      if (dev_type.equals("Line")) {
          tmp_dev = new Line(false, dev_name);
      } else {
          tmp_dev = null;
      }

      if (tmp_dev != null) {
          tmp_node = new DefaultMutableTreeNode(tmp_dev);
          tmp_dev.setNode(tmp_node);
          ((DefaultMutableTreeNode)linkageTable.get(new
               Integer(dev_linkage.nextToken()))).add(tmp_node);
          linkageTable.put(new Integer(devicesTable.getInt("deviceID")), tmp_node);
          equipment_hash.put(tmp_dev.getName(), tmp_dev);
          all_DevicesNodes.put(tmp_dev.getName(), tmp_node);
          all_devices.put(tmp_dev.getName(), tmp_dev);
      }
      devicesTable.next();
    }
```

```
//cria instancias dos TRAFOS (Transformer)
devicesTable.first();
while (!devicesTable.isAfterLast()) {
  dev_type = devicesTable.getString("deviceType");
  dev_name = devicesTable.getString("deviceName");
  dev_linkage = new StringTokenizer(devicesTable.getString("deviceLinkage"), ",");

  if (dev_type.equals("Transformer"))
      tmp_dev = new Transformer(false, dev_name);
  else
      tmp_dev = null;

  if (tmp_dev != null) {
      tmp_node = new DefaultMutableTreeNode(tmp_dev);
      tmp_dev.setNode(tmp_node);
      ((DefaultMutableTreeNode)linkageTable.get(new
          Integer(dev_linkage.nextToken()))).add(tmp_node);
      linkageTable.put(new Integer(devicesTable.getInt("deviceID")), tmp_node);
      equipment_hash.put(tmp_dev.getName(), tmp_dev);
      all_DevicesNodes.put(tmp_dev.getName(), tmp_node);
      all_devices.put(tmp_dev.getName(), tmp_dev);
  }
  devicesTable.next();
}

//cria instancias das BARRAS (Bus)
devicesTable.first();
while (!devicesTable.isAfterLast()) {
  dev_type = devicesTable.getString("deviceType");
  dev_name = devicesTable.getString("deviceName");
  dev_linkage = new StringTokenizer(devicesTable.getString("deviceLinkage"), ",");

  if (dev_type.equals("Bus"))
      tmp_dev = new Bus(false, dev_name);
  else
      tmp_dev = null;

  if (tmp_dev != null) {
      tmp_node = new DefaultMutableTreeNode(tmp_dev);
      tmp_dev.setNode(tmp_node);
      ((DefaultMutableTreeNode)linkageTable.get(new
          Integer(dev_linkage.nextToken()))).add(tmp_node);
      linkageTable.put(new Integer(devicesTable.getInt("deviceID")), tmp_node);
      equipment_hash.put(tmp_dev.getName(), tmp_dev);
      all_DevicesNodes.put(tmp_dev.getName(), tmp_node);
      all_devices.put(tmp_dev.getName(), tmp_dev);
  }
  devicesTable.next();
}

//cria instancias dos BANCOS DE CAPACITORES (Capacitor) e ALIMENTADORES (Feeder)
devicesTable.first();
while (!devicesTable.isAfterLast()) {
  dev_type = devicesTable.getString("deviceType");
  dev_name = devicesTable.getString("deviceName");
  dev_linkage = new StringTokenizer(devicesTable.getString("deviceLinkage"));

  if (dev_type.equals("Capacitor"))
      tmp_dev = new Capacitor(false, dev_name);
  else if (dev_type.equals("Feeder"))
      tmp_dev = new Feeder(false, dev_name);
  else
      tmp_dev = null;

  if (tmp_dev != null) {
      tmp_node = new DefaultMutableTreeNode(tmp_dev);
      tmp_dev.setNode(tmp_node);
      ((DefaultMutableTreeNode)linkageTable.get(new
           Integer(dev_linkage.nextToken()))).add(tmp_node);
      linkageTable.put(new Integer(devicesTable.getInt("deviceID")), tmp_node);
      equipment_hash.put(tmp_dev.getName(), tmp_dev);
      all_DevicesNodes.put(tmp_dev.getName(), tmp_node);
      all_devices.put(tmp_dev.getName(), tmp_dev);
  }
  devicesTable.next();
}

// cria instancias das CHAVES (SwitchingDevice), das MEDIDAS (MeasuringDevice)
// e PROTEÇÕES (ProtectedRelays).
devicesTable.first();
```

```
        while (!devicesTable.isAfterLast()) {
          dev_type = devicesTable.getString("deviceType");
          dev_name = devicesTable.getString("deviceName");
          dev_param = devicesTable.getString("deviceArgs");
          if (dev_param != null)
            dev_params = new StringTokenizer(dev_param, ",");
          dev_linkage = new StringTokenizer(devicesTable.getString("deviceLinkage"), ",");

          if (dev_type.equals("Switch")) {
              tmp_dev = new Switch(false, dev_name);
              switches_hash.put(tmp_dev.getName(), tmp_dev);
          }
          else if (dev_type.equals("CircuitBreaker")) {
              tmp_dev = new CircuitBreaker(false, dev_name);
              ((CircuitBreaker)tmp_dev).setUserEditable(this.isUserEditable);

((CircuitBreaker)tmp_dev).setRepresentation(Integer.parseInt(dev_params.nextToken()),
Integer.parseInt(dev_params.nextToken()), Integer.parseInt(dev_params.nextToken()),
Integer.parseInt(dev_params.nextToken()), (new
     Boolean(dev_params.nextToken())).booleanValue());
              switches_hash.put(tmp_dev.getName(), tmp_dev);
          }
          else if (dev_type.equals("GroundingSwitch")) {
              tmp_dev = new GroundingSwitch(false, dev_name);
              switches_hash.put(tmp_dev.getName(), tmp_dev);
          }
          else if (dev_type.equals("ByPassSwitch")) {
              tmp_dev= new ByPassSwitch(false, dev_name);
              switches_hash.put(tmp_dev.getName(), tmp_dev);
          }
          else if (dev_type.equals("MeasuringDevice")) {
              tmp_dev = new MeasuringDevice(false, dev_name, new Double(0),
                dev_params.nextToken(), Boolean.valueOf(dev_params.nextToken()).booleanValue());
              ((MeasuringDevice)tmp_dev).setUserEditable(this.isUserEditable);
              measures_hash.put(tmp_dev.getName(), tmp_dev);
          }
          else if (dev_type.equals("ProtectedRelay")) {
              tmp_dev =  new ProtectedRelay(false, dev_name);

((ProtectedRelay)tmp_dev).setRepresentation(Integer.parseInt(dev_params.nextToken()),
Integer.parseInt(dev_params.nextToken()));
              ((ProtectedRelay)tmp_dev).setUserEditable(this.isUserEditable);
              protections_hash.put(tmp_dev.getName(), tmp_dev);
          }
          else {
              tmp_dev= null;
          }

          if (tmp_dev!= null) {
            tmp_node = new DefaultMutableTreeNode(tmp_dev, false);

            ((DefaultMutableTreeNode)linkageTable.get(new
                Integer(dev_linkage.nextToken()))).add(tmp_node);
            tmp_dev.setNode(tmp_node);
            while (dev_linkage.hasMoreTokens())
                if (tmp_dev instanceof SwitchingDevice) {
                    DefaultMutableTreeNode tmp_ExtraNode = new DefaultMutableTreeNode(tmp_dev,
                        false);
                    ((SwitchingDevice)tmp_dev).setExtraNode(tmp_ExtraNode);
                    ((DefaultMutableTreeNode)linkageTable.get(new
                        Integer(dev_linkage.nextToken()))).add(tmp_ExtraNode);
                }

            linkageTable.put(new Integer(devicesTable.getInt("deviceID")), tmp_node);

            all_DevicesNodes.put(tmp_dev.getName(), tmp_node);
            all_devices.put(tmp_dev.getName(), tmp_dev);
          }
          devicesTable.next();
        }
        trace("<" + Now.getCurrentTime() + "> " + this.name + ": agent loaded all Devices") ;

        this.notifyIAgentEventListeners(new IAgentEvent(this, "load_measures", measures_hash));
        this.notifyIAgentEventListeners(new IAgentEvent(this, "load_protections",
            protections_hash));
        this.notifyIAgentEventListeners(new IAgentEvent(this, "load_switches", switches_hash));
        this.notifyIAgentEventListeners(new IAgentEvent(this, "load_equipment",
            equipment_hash));
        this.notifyIAgentEventListeners(new IAgentEvent(this, "load_devices", all_devices));
```

```
        this.notifyIAgentEventListeners(new IAgentEvent(this, "query_equipment",
            equipment_hash.keys()));
        this.notifyIAgentEventListeners(new IAgentEvent(this, "query_measures",
            measures_hash.keys()));
        this.notifyIAgentEventListeners(new IAgentEvent(this, "query_switches",
            switches_hash.keys()));
        this.notifyIAgentEventListeners(new IAgentEvent(this, "query_protections",
            protections_hash.keys()));

        switchesPanel = new SwitchesPanel(switches_hash.elements(), "circuit.gif");
        measuresPanel = new MeasuresPanel(measures_hash.elements());
        if (substations.size() > 1) {
          tmp_node = new DefaultMutableTreeNode("Loaded Substations");
          Enumeration tmp_substations = substations.elements();
          while (tmp_substations.hasMoreElements()) {
            tmp_node.add((DefaultMutableTreeNode)tmp_substations.nextElement());
          }
          this.schematicsDialog = new SubstationSchematics(getFramework(), tmp_node);
        } else if (substations.size() > 0) {
          this.schematicsDialog = new SubstationSchematics(getFramework(),
              (DefaultMutableTreeNode)substations.elements().nextElement());
        }
        ((SubstationDialog)getFramework()).refresh();

      } catch (java.sql.SQLException e) {
        e.printStackTrace();
      }
    }
    public Hashtable getMeasuresState() {
      Hashtable measures = new Hashtable();
      Enumeration tmp_devs = measures_hash.elements();
      Device tmp_dev;
      while (tmp_devs.hasMoreElements()) {
        tmp_dev = (Device)tmp_devs.nextElement();
        measures.put(tmp_dev.getName(), ((MeasuringDevice)tmp_dev).getValue());
      }
      return measures;
    }
    public Hashtable getProtectionsState() {
      Hashtable protections = new Hashtable();
      Enumeration tmp_devs = protections_hash.elements();
      Device tmp_dev;
      while (tmp_devs.hasMoreElements()) {
        tmp_dev = (Device)tmp_devs.nextElement();
        protections.put(tmp_dev.getName(), new
Boolean(((ProtectedRelay)tmp_dev).getCurrentStatus()));
      }
      return protections;
    }
    public Hashtable getSwitchesState() {
      Hashtable switches = new Hashtable();
      Enumeration tmp_devs = switches_hash.elements();
      Device tmp_dev;
      while (tmp_devs.hasMoreElements()) {
        tmp_dev = (Device)tmp_devs.nextElement();
        switches.put(tmp_dev.getName(), new
Boolean(((SwitchingDevice)tmp_dev).getCurrentStatus()));
      }
      return switches;
    }
    public Hashtable getEquipmentState() {
      Hashtable equipment = new Hashtable();
      Enumeration tmp_devs = equipment_hash.elements();
      Device tmp_dev;
      while (tmp_devs.hasMoreElements()) {
        tmp_dev = (Device)tmp_devs.nextElement();
        equipment.put(tmp_dev.getName(), new
            Boolean(((EquipmentDevice)tmp_dev).getCurrentStatus()));
      }
      return equipment;
    }
    public void saveModelState() {
      Hashtable msg = new Hashtable();
      msg.put("measures", this.getMeasuresState());
      msg.put("protections", this.getProtectionsState());
      msg.put("switches", this.getSwitchesState());
      msg.put("equipment", this.getEquipmentState());

      this.notifyIAgentEventListeners(new IAgentEvent(this, "save_history", msg));
    }
```

```
  public void insertModelState(Hashtable modifications) {
    Hashtable msg = new Hashtable();
    msg.put("measures", this.getMeasuresState());
    msg.put("protections", this.getProtectionsState());
    msg.put("switches", this.getSwitchesState());
    msg.put("equipment", this.getEquipmentState());
    if (modifications != null) msg.put("modifications", modifications);
    this.notifyIAgentEventListeners(new IAgentEvent(this, "insert_history", msg));
  }
  public void notifyToolAgents() {
    if (!isWaintingAnswer) {
      isWaintingAnswer = true;
      this.notifyIAgentEventListeners(new IAgentEvent(this, "verify", null));
    }
  }
  public void updateDevices(Hashtable devices) {
    if (devices != null) {
      Enumeration tmp_devices = devices.keys();
      String key;
      while (tmp_devices.hasMoreElements()) {
        key = (String)tmp_devices.nextElement();
        if (((DefaultMutableTreeNode)all_DevicesNodes.get(key)).getUserObject() instanceof
            MeasuringDevice)

((MeasuringDevice)((DefaultMutableTreeNode)all_DevicesNodes.get(key)).getUserObject()).setValu
e(devices.get(key));
        else

((Device)((DefaultMutableTreeNode)all_DevicesNodes.get(key)).getUserObject()).setCurrentStatus
(((Boolean)devices.get(key)).booleanValue());
      }
    }
  }
```

PlanningAgent.Java

```
  public void iaEventFired(IAgentEvent e) {
    trace("<" + Now.getCurrentTime() + "> " + name + ": got message from \"" +
        ((IAgent)e.getSource()).getName() + "\". Message type=\"" + e.getAction() + "\"
       attach=" + e.getAttach()) ;
      if (e.getAction().equals("load_devices")) {
          if (e.getAttach() instanceof Hashtable)
              all_devices = (Hashtable)e.getAttach();
      } else if (e.getAction().equals("switches_sugestions")) {
          if (e.getAttach() instanceof Hashtable)
              SwitchAgentSugestions = (Hashtable)e.getAttach();
          else if (e.getAttach() instanceof Boolean)
              if (((Boolean)e.getAttach()).booleanValue())
                  clear();
              else
                  SwitchAgentSugestions = new Hashtable();
          verifyMessages();
      } else if (e.getAction().equals("protections_sugestions")) {
          if (e.getAttach() instanceof Hashtable)
              ProtectionAgentSugestions = (Hashtable)e.getAttach();
          else if (e.getAttach() instanceof Boolean)
              if (((Boolean)e.getAttach()).booleanValue())
                  clear();
              else
                  ProtectionAgentSugestions = new Hashtable();
          verifyMessages();
      } else if (e.getAction().equals("equipment_sugestions")) {
          if (e.getAttach() instanceof Hashtable)
              EquipmentAgentSugestions = (Hashtable)e.getAttach();
          else if (e.getAttach() instanceof Boolean)
              if (((Boolean)e.getAttach()).booleanValue())
                  clear();
              else
                  EquipmentAgentSugestions = new Hashtable();
          verifyMessages();
      } else if (e.getAction().equals("measures_sugestions")) {
          if (e.getAttach() instanceof Hashtable)
              MeasureAgentSugestions = (Hashtable)e.getAttach();
          else if (e.getAttach() instanceof Boolean)
              if (((Boolean)e.getAttach()).booleanValue())
                  clear();
              else
                  MeasureAgentSugestions = new Hashtable();
```

```
            verifyMessages();
        } else if (e.getAction().equals("event_sugestions")) {
            if (e.getAttach() instanceof Hashtable)
                EventIdentificationAgentSugestions = (Hashtable)e.getAttach();
            else if (e.getAttach() instanceof Boolean)
                if (((Boolean)e.getAttach()).booleanValue())
                    clear();
                else
                    EventIdentificationAgentSugestions = new Hashtable();
            verifyMessages();
        }
    }
    public void run() {
        Vector possible_sugestions = new Vector();
        Hashtable final_sugestions = new Hashtable();
        Vector messages = new Vector();

        messages.add("Ações recomendadas por " + name + " após filtragem:");

        // primeiro passo: procurar por sugestões iguais de agentes diferentes
        String device1, device2;
        Object value1, value2;
        int priority;

        //verifica por sugestões do agente de MEDIDAS
        Vector sugested_measures = MyTools.EnumerationToVector(MeasureAgentSugestions.keys());
        for (int i=0; i < sugested_measures.size(); i++) {
            device1 = (String)sugested_measures.get(i);
            value1 = MeasureAgentSugestions.get(device1);
            priority = 3;

            //verifica por sugestões do agente de EVENTOS
            Vector sugested_events =
                MyTools.EnumerationToVector(EventIdentificationAgentSugestions.keys());
            for (int j=0; j < sugested_events.size(); j++) {
                device2 = (String)sugested_events.get(j);
                value2 = EventIdentificationAgentSugestions.get(device2);
                if (device1.equals(device2) && value1.equals(value2)) {
                    priority = 2;
                    EventIdentificationAgentSugestions.remove(device2);
                }
            }

            //verifica por sugestões do agente de PROTEÇÕES
            Vector sugested_protections =
                MyTools.EnumerationToVector(ProtectionAgentSugestions.keys());
            for (int j=0; j < sugested_protections.size(); j++) {
                device2 = (String)sugested_protections.get(j);
                value2 = ProtectionAgentSugestions.get(device2);
                if (device1.equals(device2) && value1.equals(value2)) {
                    priority = 1;
                    ProtectionAgentSugestions.remove(device2);
                }
            }

            //verifica por sugestões do agente de CHAVES
            Vector sugested_switches =
                MyTools.EnumerationToVector(SwitchAgentSugestions.keys());
            for (int j=0; j < sugested_switches.size(); j++) {
                device2 = (String)sugested_switches.get(j);
                value2 = SwitchAgentSugestions.get(device2);
                if (device1.equals(device2) && value1.equals(value2)) {
                    SwitchAgentSugestions.remove(device2);
                }
            }

            possible_sugestions.add(new Sugestion(device1, value1, priority));
            MeasureAgentSugestions.remove(device1);
        }

        //verifica por sugestões do agente de EVENTOS
        Vector sugested_events =
            MyTools.EnumerationToVector(EventIdentificationAgentSugestions.keys());
        for (int i=0; i < sugested_events.size(); i++) {
            device1 = (String)sugested_events.get(i);
            value1 = EventIdentificationAgentSugestions.get(device1);
            priority = 2;

            //verifica por sugestões do agente de PROTEÇÕES
            Vector sugested_protections =
```

```
            MyTools.EnumerationToVector(ProtectionAgentSugestions.keys());
        for (int j=0; j < sugested_protections.size(); j++) {
            device2 = (String)sugested_protections.get(j);
            value2 = ProtectionAgentSugestions.get(device2);
            if (device1.equals(device2) && value1.equals(value2)) {
                priority = 1;
                ProtectionAgentSugestions.remove(device2);
            }
        }

        //verifica por sugestões do agente de CHAVES
        Vector sugested_switches =
            MyTools.EnumerationToVector(SwitchAgentSugestions.keys());
        for (int j=0; j < sugested_switches.size(); j++) {
            device2 = (String)sugested_switches.get(j);
            value2 = SwitchAgentSugestions.get(device2);
            if (device1.equals(device2) && value1.equals(value2)) {
                SwitchAgentSugestions.remove(device2);
            }
        }
        possible_sugestions.add(new Sugestion(device1, value1, priority));
        MeasureAgentSugestions.remove(device1);
    }

    //verifica por sugestões do agente de PROTEÇÕES
    Vector sugested_protections =
        MyTools.EnumerationToVector(ProtectionAgentSugestions.keys());
    for (int i=0; i < sugested_protections.size(); i++) {
        device1 = (String)sugested_protections.get(i);
        value1 = ProtectionAgentSugestions.get(device1);
        priority = 1;

        //verifica por sugestões do agente de CHAVES
        Vector sugested_switches =
            MyTools.EnumerationToVector(SwitchAgentSugestions.keys());
        for (int j=0; j < sugested_switches.size(); j++) {
            device2 = (String)sugested_switches.get(j);
            value2 = SwitchAgentSugestions.get(device2);
            if (device1.equals(device2) && value1.equals(value2)) {
                SwitchAgentSugestions.remove(device2);
            }
        }
        possible_sugestions.add(new Sugestion(device1, value1, priority));
        ProtectionAgentSugestions.remove(device1);
    }

    //verifica por sugestões do agente de CHAVES
    Vector sugested_switches = MyTools.EnumerationToVector(SwitchAgentSugestions.keys());
    for (int i=0; i < sugested_switches.size(); i++) {
        device1 = (String)sugested_switches.get(i);
        value1 = SwitchAgentSugestions.get(device1);
        priority = 3;

        possible_sugestions.add(new Sugestion(device1, value1, priority));
        SwitchAgentSugestions.remove(device1);
    }
    // fim do primeiro passo!

    // segundo passo: olhar por sugestões antagônicas
    Sugestion sug1, sug2;
    Enumeration possible_sugestions1 = possible_sugestions.elements();
    int priority1, priority2;
    while (possible_sugestions1.hasMoreElements()) {
        sug1 = (Sugestion)possible_sugestions1.nextElement();
        Enumeration possible_sugestions2 = possible_sugestions.elements();
        while (possible_sugestions2.hasMoreElements()) {
            sug2 = (Sugestion)possible_sugestions2.nextElement();
            // verifica se há repetições e dá as priodidades
            if (sug1.getName().equals(sug2.getName())) {
                if (sug1.getPriority() < sug2.getPriority()) {
                    possible_sugestions.remove(sug2);
                } else if (sug1.getPriority() < sug2.getPriority()) {
                    possible_sugestions.remove(sug1);
                }
            }
        }
    }
    // fim do segundo passo

    // terceiro passo: mandar as sugestões filtradas
```

```
        String message;
        Sugestion sug;
        boolean hasMessages = false;
        for (int i=0; i < possible_sugestions.size(); i++) {
            sug = (Sugestion)possible_sugestions.get(i);
            if (sug.getState() instanceof Boolean) {
                if (MyTools.getDeviceType(sug.getName()).equals("DJ") && sug.getPriority() == 3
                     && ((Boolean)sug.getState()).booleanValue()) {
                     message = "Recomenda-se restabelecer o Disjuntor " + sug.getName() + " e
                     ajustar suas medições.";
                    hasMessages = true;
                } else if (((Device)all_devices.get(sug.getName())).getCurrentStatus() !=
                    ((Boolean)sug.getState()).booleanValue()) {
                    message = MyTools.getDeviceMessage(sug.getName(), sug.getState()) + " (" +
                    MyTools.getAgent(sug.getPriority()) + ")";
                    final_sugestions.put(sug.getName(), sug.getState());
                } else
                    message = "";
            }
            else {
                if
(!((MeasuringDevice)all_devices.get(sug.getName())).getValue().equals(sug.getState())) {
                    message = MyTools.getDeviceMessage(sug.getName(), sug.getState()) + " (" +
                        MyTools.getAgent(sug.getPriority()) + ")";
                    final_sugestions.put(sug.getName(), sug.getState());
                } else
                    message = "";
            }

            if (!message.equals(" (" + MyTools.getAgent(sug.getPriority()) + ")") &&
                !message.equals(""))
              messages.add("   - " + message);
        }
        if (final_sugestions.isEmpty()) {
            //sem sugestões e sem erros
            if (hasFailure && !hasMessages) {
                messages.clear();

messages.add("**************************************************************************");
            messages.add("* Término da falha. Iniciando restabelecimento. Ajustar medições. *");

messages.add("**************************************************************************");
                hasFailure = false;
            } else if (!hasMessages) {
                messages.add("   - Nehuma recomendação.");
            }
            messages.add("-----------------------------------");
            this.notifyIAgentEventListeners(new IAgentEvent(this, "step_sugestions", new
                Boolean(false)));
        } else {
            hasFailure = true;
            messages.add("-----------------------------------");
            this.notifyIAgentEventListeners(new IAgentEvent(this, "step_sugestions",
                final_sugestions));

        }

        //---- imprime sugestões do agente ----
        this.agentOutput.append(messages.elements(), OutputTextArea.STYLE_BOLD_RED);
        ((SubstationDialog)getFramework()).output.append(messages.elements(),
            OutputTextArea.STYLE_BOLD_RED);
        //---- ----- ----- -----

        // fim terceiro passo
        clear();
    }
    synchronized void verifyMessages() {
        if (SwitchAgentSugestions != null && ProtectionAgentSugestions != null &&
            MeasureAgentSugestions != null && EquipmentAgentSugestions != null &&
            EventIdentificationAgentSugestions != null) {
            start();
        }
    }
```

EventIdentificationAgent.Java

```
    public void iaEventFired(IAgentEvent e) {
```

```
    trace("<" + Now.getCurrentTime() + "> " + name + ": got message from \"" +
((IAgent)e.getSource()).getName() + "\". Message type=\"" + e.getAction() + "\" attach=" +
e.getAttach()) ;
    if (e.getAttach() instanceof Hashtable) {
      if (e.getAction().equals("load_switches")) {
        switches = (Hashtable)e.getAttach();
        saveState();
      }
    } else if (e.getAttach() == null) {
      if (e.getAction().equals("verify"))
        start();
    }
  }
  void saveState() {
    switchesLastState.clear();
    Enumeration switchesNames = switches.keys();
    String key;
    while (switchesNames.hasMoreElements()) {
      key = (String)switchesNames.nextElement();
      switchesLastState.put(key, new
          Boolean(((SwitchingDevice)switches.get(key)).getCurrentStatus()));
    }
  }
  public void run() {
      Hashtable sugestions = new Hashtable();
      Vector messages = new Vector();
      boolean hasError = false;
      messages.add("Ações recomendadas por " + name + ":");
      if (switches != null) {
          Device switchingDevice;
          // verifica a coerência de todos os disjuntores
          Enumeration allSwitches = switches.elements();
          while (allSwitches.hasMoreElements()) {
              switchingDevice = (Device)allSwitches.nextElement();
              if (switchingDevice instanceof CircuitBreaker) {
                  if (!switchingDevice.getCurrentStatus() &&
                      !MyTools.isBarToBarDevice((SwitchingDevice)switchingDevice) ) {
                      if (!downSwitches.contains(switchingDevice.getName())) {
                          downSwitches.add(switchingDevice.getName());
                          Enumeration children;
                          if (((SwitchingDevice)switchingDevice).isLinkageDevice())
                            children =
((DefaultMutableTreeNode)((SwitchingDevice)switchingDevice).getExtraNode().getParent()).breadt
hFirstEnumeration();
                          else
                            children =
((DefaultMutableTreeNode)((SwitchingDevice)switchingDevice).getNode().getParent()).breadthFirs
tEnumeration();
                          Device childDevice=null;
                          while (children.hasMoreElements()) {
                              childDevice =
(Device)((DefaultMutableTreeNode)children.nextElement()).getUserObject();
                              if (childDevice instanceof CircuitBreaker) {
                                  if (!switchingDevice.equals(childDevice)) {
                                    if (!sugestions.containsKey(childDevice.getName())) {
                                        sugestions.put(childDevice.getName(), new
                                                       Boolean(false));
                                        messages.add("   - " + "Disjuntor " +
                                         switchingDevice.getName() + " abriu. Ação
                                         recomendada: Abrir Disjuntor " +
                                         childDevice.getName() + ".");
                                    }
                                  }
                              } else if (childDevice instanceof MeasuringDevice &&
                                   !(MyTools.getParent(childDevice) instanceof Capacitor)) {
                                  if (!sugestions.containsKey(childDevice.getName())) {
                                    sugestions.put(childDevice.getName(), new Double(0));
                                    messages.add("   - " + "Disjuntor " +
                                    switchingDevice.getName() + " abriu. Ação recomendada:
                                    Ajustar Medida " + childDevice.getName() + ".");
                                  }
                              }
                          }
                      }
                  } else if (switchingDevice.getCurrentStatus()) {
                      Enumeration children = MyTools.getSiblingMeasures(switchingDevice, "A");
                      Device childDevice=null;
                      while (children.hasMoreElements()) {
                          childDevice = (Device)children.nextElement();
                          if (childDevice instanceof MeasuringDevice) {
```

```
                                if
(((Double)((MeasuringDevice)childDevice).getValue()).doubleValue() == 0 &&
!(MyTools.getParent(switchingDevice) instanceof Capacitor) &&
!MyTools.isBarToBarDevice((CircuitBreaker)switchingDevice)) {
                                    if (sugestions.containsKey(childDevice.getName()))
                                        sugestions.remove(childDevice.getName());
                                    sugestions.put(switchingDevice.getName(), new
                                        Boolean(false));
                                    messages.add("   - " + "Medida " + childDevice.getName() + "
                                        está em zero. Correção recomendada: Abrir Disjuntor " +
                                        switchingDevice.getName() + ".");
                                }
                            }
                        }

                    }
                }
                if (!switchingDevice.getCurrentStatus() &&

                    ((Boolean)switchesLastState.get(switchingDevice.getName())).booleanValue())
{
                    if (downSwitches.contains(switchingDevice.getName()))
                        downSwitches.remove(switchingDevice.getName());
                }
            }
        }
        if (hasError || sugestions.isEmpty())
          this.notifyIAgentEventListeners(new IAgentEvent(this, "event_sugestions", new
              Boolean(hasError)));
        else
          this.notifyIAgentEventListeners(new IAgentEvent(this, "event_sugestions",
              sugestions));

        //---- imprime sugestões do agente ----
        if (messages.size() == 1)
            messages.add("   - Nehuma recomendação.");
        this.agentOutput.append(messages.elements());
        //---- ----- ----- -----
        hasError = false;
        saveState();
    }
```