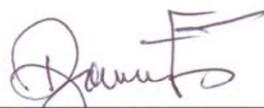


DESENVOLVIMENTO DE UM SISTEMA COMPUTACIONAL ORIENTADO A  
OBJETOS PARA SISTEMAS ELÉTRICOS DE POTÊNCIA: APLICAÇÃO A  
SIMULAÇÃO RÁPIDA E ANÁLISE DA ESTABILIDADE DE TENSÃO

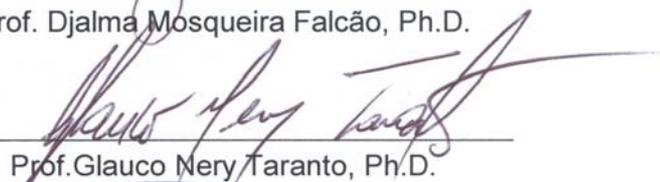
Alessandro Manzoni

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS  
PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE  
FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS  
PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA  
ELÉTRICA.

Aprovada por:



Prof. Djalma Mosqueira Falcão, Ph.D.



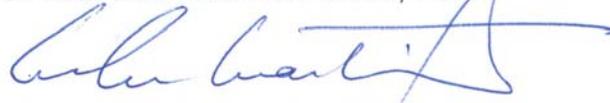
Prof. Glauco Nery Taranto, Ph.D.



Prof. Ildemar Cassana Decker, D.Sc.



Prof. José Luiz Rezende Pereira, Ph.D.



Dr. Nelson Martins, Ph.D.



Dr. Jorge Luiz de Araújo Jardim, Ph.D.

RIO DE JANEIRO, RJ – BRASIL

MARÇO DE 2005

MANZONI, ALESSANDRO

Desenvolvimento de um Sistema Computacional Orientado a Objetos para Sistemas Elétricos de Potência: Aplicação a Simulação Rápida e Análise da Estabilidade de Tensão [Rio de Janeiro] 2005.

XI, 165 p. 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia Elétrica, 2005)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1. Modelagem Orientada a Objetos
2. Estabilidade de Sistemas de Energia Elétrica

I. COPPE/UFRJ II. Título ( série )

Dedicada aos meus pais Nelsi e Alba  
e ao meu irmão Gustavo (*in memorium*).

A minha esposa Andréa e ao meu filho Filipe  
pelo apoio e paciência demonstrado  
em todas as horas da realização deste trabalho.

# Agradecimentos

Aos professores, orientadores e amigos Djalma M. Falcão e Glauco Nery Taranto pela orientação, dedicação, confiança e apoio demonstrados ao longo do desenvolvimento deste trabalho.

A todos os professores do Curso de Pós-Graduação em Engenharia Elétrica da UFRJ que, de uma ou de outra forma, contribuíram para a realização deste trabalho.

Um agradecimento especial ao professor Alquindar de Souza Pedroso pelas valiosas discussões e amizade demonstrada no período em que estive na UFRJ.

Ao colega e amigo Zulmar S. Machado Jr. pelo crescimento mútuo que tivemos com as discussões a respeito de nossos trabalhos.

A todos os funcionários, bolsistas, colegas e amigos do mestrado/doutorado pelas horas de convívio dentro e fora da universidade.

Ao CNPq pelo suporte financeiro.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

DESENVOLVIMENTO DE UM SISTEMA COMPUTACIONAL ORIENTADO A  
OBJETOS PARA SISTEMAS ELÉTRICOS DE POTÊNCIA: APLICAÇÃO A  
SIMULAÇÃO RÁPIDA E ANÁLISE DA ESTABILIDADE DE TENSÃO

Alessandro Manzoni

Março/2005

Orientadores: Djalma Mosqueira Falcão  
Glauco Nery Taranto

Programa: Engenharia Elétrica

Este trabalho tem por objetivo aplicar a Modelagem Orientada a Objetos para o desenvolvimento de uma base computacional capaz de integrar e dar suporte à construção de um amplo conjunto de ferramentas para simulação e análise de sistemas elétricos. O modelo proposto mostrou-se adequado para acomodar aplicativos tais como fluxo de potência, análise modal e simulação dinâmica completa. Aliado a isso, obteve-se um grau de generalização para os modelos e aplicativos que permitem que um novo equipamento (ou um novo modelo para este equipamento) seja adicionado ao sistema e assimilado automaticamente por todo o elenco de aplicativos. Em uma segunda etapa do trabalho, foi proposta uma modificação na metodologia de simulação rápida de médio e longo prazo que possibilitou a esta ferramenta um estudo mais amplo dos fenômenos que ocorrem em cenários de instabilidade de tensão. A retenção da dinâmica dos modelos na formulação do método possibilitou que técnicas de análise modal fossem utilizadas durante a simulação da trajetória do sistema. Adicionalmente uma metodologia de simulação combinada, que reúne o simulador rápido modificado e um simulador da dinâmica completa, permitiu ampliar ainda mais a faixa de fenômenos que o simulador rápido pode estudar.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.).

DEVELOPMENT OF AN OBJECT ORIENTED COMPUTATIONAL SYSTEM:  
APPLICATION TO FAST SIMULATION AND VOLTAGE STABILITY  
ANALYSIS

Alessandro Manzoni

March/2005

Advisors: Djalma Mosqueira Falcão  
Glauco Nery Taranto

Department: Electrical Engineering

The objective of this work is the application of Object Oriented Modeling to the development of a computational basis able to integrate and support the building of a large set of computational tools for simulation and analysis of electric power systems. Tests with the proposed model showed its adequacy for the implementation of applications like power flow, modal analysis, and full dynamic simulation. Moreover, the generalization achieved in model and applications allows that a device (or a new model for a device already included in the system) be automatically added to the system and assimilated by all applications. In a second stage of the work, it is proposed a modification in the methodology for mid- and long-term simulation that allows to this computational tool a more general study of phenomena present in voltage instability scenarios. The preservation of the full dynamic models in the method formulation allowed the use of modal analysis in points of the simulated system trajectory. Additionally, a combined simulation methodology, bringing together an enhanced fast dynamic simulator and a full dynamic simulator, allowed an even more enlarged range of dynamic phenomena that can be studied using the proposed methodology.

# Sumário

<b>CAPÍTULO 1 INTRODUÇÃO .....</b>	<b>1</b>
1.1 CONSIDERAÇÕES GERAIS .....	1
1.2 ESTRUTURA DO TRABALHO.....	2
<b>CAPÍTULO 2 MODELAGEM ORIENTADA A OBJETOS.....</b>	<b>4</b>
2.1 CONSIDERAÇÕES GERAIS .....	4
2.2 MUDANÇA NO ENFOQUE DE PROGRAMAÇÃO .....	5
2.2.1 <i>O Conceito de Objeto</i> .....	6
2.2.2 <i>Herança e Polimorfismo</i> .....	8
2.3 METODOLOGIAS DE PROJETO ORIENTADO A OBJETOS .....	8
2.3.1 <i>Método de Booch [2]</i> .....	9
2.3.2 <i>OMT [2][1]</i> .....	9
2.3.3 <i>Método OOSE [2]</i> .....	10
2.4 UNIFIED MODELING LANGUAGE (UML).....	11
2.4.1 <i>O Processo de Desenvolvimento da UML</i> .....	11
2.4.2 <i>A Representação dos Modelos na UML[2][3]</i> .....	13
2.4.3 <i>O Diagrama de Classes da UML e sua Notação Gráfica</i> .....	16
2.5 CONSIDERAÇÕES FINAIS.....	19
<b>CAPÍTULO 3 MODELAGEM ORIENTADA A OBJETOS APLICADA A SISTEMAS DE ENERGIA ELÉTRICA .....</b>	<b>20</b>
3.1 CONSIDERAÇÕES GERAIS .....	20
3.1.1 <i>Revisão Bibliográfica</i> .....	21
3.2 LEVANTAMENTO DOS REQUISITOS DO MODELO ORIENTADO A OBJETOS.....	26
3.3 DESCRIÇÃO TOPOLÓGICA DO SEE.....	29
3.3.1 <i>Descrição Física</i> .....	29
3.3.2 <i>Descrição Lógica</i> .....	35
3.3.3 <i>Coexistência das Descrições</i> .....	37
3.3.4 <i>A Configuração da Rede Elétrica</i> .....	40
3.4 FUNCIONALIDADES ESPECÍFICAS.....	41
3.4.1 <i>Estado de um Dispositivo</i> .....	43
3.4.2 <i>Modelo de um Dispositivo</i> .....	44
3.5 FERRAMENTAS MATEMÁTICAS .....	59
3.6 APLICATIVOS .....	61
3.6.1 <i>Um Aplicativo Exemplo</i> .....	64
3.7 CONSIDERAÇÕES FINAIS.....	67
<b>CAPÍTULO 4 METODOLOGIA DE SIMULAÇÃO RÁPIDA NO TEMPO .....</b>	<b>69</b>
4.1 CONSIDERAÇÕES GERAIS .....	69
4.2 ESTABILIDADE DE TENSÃO.....	70
4.2.1 <i>Definições em Estabilidade de Tensão</i> .....	70
4.2.2 <i>Escalas de Tempo</i> .....	72
4.2.3 <i>Métodos de Avaliação da Estabilidade de Tensão</i> .....	73
4.3 SIMULAÇÃO RÁPIDA .....	77
4.3.1 <i>Modelos Matemáticos</i> .....	78
4.4 SIMULAÇÃO RÁPIDA MODIFICADA.....	83

4.4.1	<i>Modelos Matemáticos</i> .....	83
4.4.2	<i>Considerações sobre os Modelos Matemáticos e suas Implicações</i> .....	85
4.5	COMPARAÇÃO ENTRE AS METODOLOGIAS .....	87
4.6	SIMULAÇÃO RÁPIDA E COMPLETA COMBINADAS.....	92
4.6.1	<i>Aspectos Computacionais</i> .....	93
4.7	CONSIDERAÇÕES FINAIS.....	95
<b>CAPÍTULO 5 RESULTADOS.....</b>		<b>97</b>
5.1	CONSIDERAÇÕES GERAIS .....	97
5.2	BASE DE APLICATIVOS IMPLEMENTADA.....	98
5.3	SIMULAÇÕES E AVALIAÇÃO DA FLEXIBILIDADE DO M.O.O. ....	99
5.3.1	<i>Sistema Exemplo</i> .....	100
5.3.2	<i>Fluxo de Potência</i> .....	100
5.3.3	<i>Análise de Estabilidade de Pequenos-Sinais</i> .....	103
5.3.4	<i>Simulação da Dinâmica Completa</i> .....	105
5.3.5	<i>Simulação Rápida</i> .....	109
5.3.6	<i>Simulações de Longo Prazo</i> .....	112
5.3.7	<i>Flexibilidade de Formulação</i> .....	117
5.3.8	<i>Flexibilidade de Modelagem</i> .....	118
5.3.9	<i>Comentários Gerais</i> .....	126
5.4	DESEMPENHO COMPUTACIONAL DO M.O.O.....	126
5.4.1	<i>Descrição dos Sistemas Utilizados</i> .....	127
5.4.2	<i>Desempenho Computacional do Fluxo de Potência</i> .....	128
5.4.3	<i>Desempenho Computacional da Simulação da Dinâmica Completa</i> .....	131
5.4.4	<i>Simulação de Longo Prazo para o Sistema Brasileiro</i> .....	137
5.4.5	<i>Simulação de Médio Prazo para o Sistema Brasileiro</i> .....	139
5.4.6	<i>Comentários Gerais</i> .....	141
5.5	CONSIDERAÇÕES FINAIS.....	141
<b>CAPÍTULO 6 CONCLUSÕES .....</b>		<b>143</b>
6.1	CONSIDERAÇÕES GERAIS .....	143
6.2	SUGESTÕES PARA FUTUROS TRABALHOS .....	146
<b>REFERÊNCIAS BIBLIOGRÁFICAS.....</b>		<b>147</b>

# Índice de Figuras

FIGURA 1 – EXEMPLOS DE OBJETOS DE UMA APLICAÇÃO PARA GEOMETRIA PLANA.....	7
FIGURA 2 – EXEMPLO DE UM DIAGRAMA DE CLASSES DA UML.....	13
FIGURA 3 – EXEMPLO DE UM DIAGRAMA DE CASOS DE USO DA UML [2].....	14
FIGURA 4 – EXEMPLO DE UM DIAGRAMA DE SEQÜÊNCIA DA UML [2].....	14
FIGURA 5 – EXEMPLO DE UM DIAGRAMA DE COLABORAÇÃO DA UML [2] .....	15
FIGURA 6 – EXEMPLO DE UM DIAGRAMA DE ESTADO DA UML [2] .....	16
FIGURA 7 – REPRESENTAÇÃO DE CLASSES NA UML .....	17
FIGURA 8 – REPRESENTAÇÃO DE RELACIONAMENTOS ENTRE CLASSES NA UML.....	18
FIGURA 9 – REPRESENTAÇÃO DE PACOTES NA UML .....	18
FIGURA 10 – REPRESENTAÇÃO DE TEMPLATES NA UML.....	19
FIGURA 11 – RELAÇÃO ENTRE AS ABORDAGENS DO MOO .....	28
FIGURA 12 – ARRANJO TOPOLÓGICO DE UM SEE .....	30
FIGURA 13 – DIAGRAMA DE CLASSES DO SEE.....	31
FIGURA 14 – DIAGRAMA DE CLASSES DA SUBESTAÇÃO .....	32
FIGURA 15 – DIAGRAMA UNIFILAR DE UMA SUBESTAÇÃO HIPOTÉTICA .....	33
FIGURA 16 – PROCESSO DE CONFIGURAÇÃO DA SUBESTAÇÃO .....	36
FIGURA 17 – DIAGRAMA DE CLASSES DAS ILHAS ELÉTRICAS .....	36
FIGURA 18 – ESTRUTURA ORGANIZACIONAL DO MOO .....	38
FIGURA 19 – DIAGRAMA GERAL DE CLASSES.....	39
FIGURA 20 – ESTRUTURA GERAL DE UM DISPOSITIVO GENÉRICO .....	42
FIGURA 21 – ESTADO DOS DISPOSITIVOS.....	44
FIGURA 22 – DIAGRAMA DE BLOCOS REPRESENTATIVO DE UMA EQUAÇÃO.....	46
FIGURA 23 – ESTRUTURA DE CONEXÃO DE UM BLOCO CONSTRUTIVO ELEMENTAR.....	47
FIGURA 24 – DIAGRAMA DE CLASSES GERAL PARA O MODELO DE UM DISPOSITIVO.....	48
FIGURA 25 – MODELO DE UMA UNIDADE DE GERAÇÃO.....	49
FIGURA 26 – MECANISMO DE SOLUÇÃO E DERIVAÇÃO DO MODELO .....	51
FIGURA 27 – ESTRUTURA GERAL DOS BLOCOS COMPONENTES DO MODELO .....	53
FIGURA 28 – ESTRUTURA GERAL DE UM BLOCO DE SAÍDA DO TIPO <i>OSHUNT</i> .....	55
FIGURA 29 – MODELO DE CARGA ZIP .....	57
FIGURA 30 – DIAGRAMA DE CLASSES DAS FERRAMENTAS MATEMÁTICAS .....	60
FIGURA 31 – DIAGRAMA DE CLASSES DO SEE X APLICATIVOS .....	62
FIGURA 32 – ESTRUTURA GERAL DO MOO .....	63
FIGURA 33 – ESTRUTURA DE UTILIZAÇÃO .....	64
FIGURA 34 – DIAGRAMA DAS CLASSES UTILIZADAS NO APLICATIVO EXEMPLO .....	66
FIGURA 35 – SISTEMA DE REFERÊNCIA E MATRIZ DE TRANSFORMAÇÃO.....	78
FIGURA 36 – SISTEMA EXEMPLO DE 9 BARRAS .....	87
FIGURA 37 – TENSÃO TERMINAL DO GERADOR 1 EM AMBAS METODOLOGIAS.....	88
FIGURA 38 – TENSÃO TERMINAL DO GERADOR 3 EM AMBAS METODOLOGIAS.....	88
FIGURA 39 – POTÊNCIA REATIVA DO GERADOR 1 EM AMBAS METODOLOGIAS .....	89
FIGURA 40 – ANÁLISE DA ESTABILIDADE AO LONGO DA TRAJETÓRIA .....	89
FIGURA 41 – TENSÕES E ANÁLISE MODAL PARA UM DEGRAU DE CARGA DE 30% .....	90
FIGURA 42 – TENSÕES NAS BARRAS PARA UM DEGRAU DE CARGA DE 30%.....	91
FIGURA 43 – POTÊNCIAS REATIVAS GERADAS PARA UM DEGRAU DE CARGA DE 30% .....	91
FIGURA 44 – TENSÃO TERMINAL DO GERADOR 1 PARA UM DEGRAU DE CARGA.....	92
FIGURA 45 – TENSÃO TERMINAL DO GERADOR 3 PARA UM DEGRAU DE CARGA.....	93
FIGURA 46 – BLOCO DINÂMICO DO MODELO ORIENTADO A OBJETOS ( <i>ATRASO</i> ).....	94
FIGURA 47 – SISTEMA EXEMPLO DE 45 BARRAS .....	100
FIGURA 48 – TENSÕES DAS BARRAS ( <i>FLXPOT++</i> ) .....	101
FIGURA 49 – POTÊNCIA ATIVA E REATIVA DOS GERADORES ( <i>FLXPOT++</i> ) .....	101
FIGURA 50 – DIFERENÇA NAS TENSÕES ENTRE OS PROGRAMAS <i>FLXPOT++</i> E ANAREDE .....	102
FIGURA 51 – DIFERENÇA NAS GERAÇÕES ENTRE OS PROGRAMAS <i>FLXPOT++</i> E ANAREDE .....	102
FIGURA 52 – AUTOVALORES E FATORES DE PARTICIPAÇÃO / MODE SHAPE PARA UM AUTOVALOR SELECIONADO ( <i>AEPS++</i> ).....	103
FIGURA 53 – ÂNGULO DOS GERADORES ( <i>SIMSEES++</i> ) .....	106
FIGURA 54 – POTÊNCIA ELÉTRICA E CORRENTE DE CAMPO DOS GERADORES ( <i>SIMSEES++</i> ) .....	106
FIGURA 55 – ÂNGULO DOS GERADORES ( <i>SIMSEE4++</i> ).....	107

FIGURA 56 – POTÊNCIA ELÉTRICA E CORRENTE DE CAMPO DOS GERADORES ( <i>SIMSEE<sub>A++</sub></i> ) .....	107
FIGURA 57 – ÂNGULO DO GERADOR DA USINA DE ITAÚBA ( <i>SIMSEES<sub>++</sub>/SIMSEE<sub>A++</sub>/ANATEM</i> ) .....	108
FIGURA 58 – TENSÃO DE CAMPO DO GERADOR DA USINA DE ITAÚBA ( <i>SIMSEES<sub>++</sub>/SIMSEE<sub>A++</sub>/ANATEM</i> ) .....	108
FIGURA 59 – TENSÕES DAS BARRAS ( <i>FASTSIM<sub>++</sub></i> ) .....	109
FIGURA 60 – POTÊNCIA ATIVA E REATIVA DOS GERADORES ( <i>FASTSIM<sub>++</sub></i> ) .....	110
FIGURA 61 – OUTRAS GRANDEZAS ELÉTRICAS DOS GERADORES ( <i>FASTSIM<sub>++</sub></i> ) .....	110
FIGURA 62 – TRAJETÓRIA DOS AUTOVALORES PARA UMA RAMPA DE CARGA ( <i>FASTSIM<sub>++</sub></i> ) .....	111
FIGURA 63 – TENSÕES DAS BARRAS ( <i>FASTSIM<sub>++</sub></i> ) .....	112
FIGURA 64 – POTÊNCIA ATIVA E REATIVA DAS CARGAS ( <i>FASTSIM<sub>++</sub></i> ) .....	113
FIGURA 65 – GRANDEZAS ELÉTRICAS DOS GERADORES ( <i>FASTSIM<sub>++</sub></i> ) .....	113
FIGURA 66 – TENSÕES DAS BARRAS ( <i>SIMSEES<sub>++</sub></i> ) .....	114
FIGURA 67 – POTÊNCIA ATIVA E REATIVA DAS CARGAS ( <i>SIMSEES<sub>++</sub></i> ) .....	114
FIGURA 68 – GRANDEZAS ELÉTRICAS DOS GERADORES ( <i>SIMSEES<sub>++</sub></i> ) .....	115
FIGURA 69 – TENSÕES DAS BARRAS ( <i>FASTSIM<sub>++</sub></i> ) .....	117
FIGURA 70 – ÂNGULOS DOS GERADORES ( <i>FASTSIM<sub>++</sub></i> ) .....	117
FIGURA 71 – MODELO DO SVC INTRODUZIDO NA BARRA DE JOINVILLE .....	119
FIGURA 72 – TENSÕES DAS BARRAS SEM A INCLUSÃO DO SVC ( <i>FLXPOT<sub>++</sub></i> ) .....	120
FIGURA 73 – TENSÕES DAS BARRAS COM A INCLUSÃO DO SVC ( <i>FLXPOT<sub>++</sub></i> ) .....	120
FIGURA 74 – POTÊNCIAS GERADAS PELOS GERADORES E PELO SVC ( <i>FLXPOT<sub>++</sub></i> ) .....	121
FIGURA 75 – ESTRUTURA DA MATRIZ JACOBIANA COM A INCLUSÃO DO SVC ( <i>AEPS<sub>++</sub></i> ) .....	122
FIGURA 76 – AUTOVALORES DO SISTEMA COM A INCLUSÃO DO SVC ( <i>AEPS<sub>++</sub></i> ) .....	123
FIGURA 77 – TENSÕES DAS BARRAS E TENSÃO NA BARRA DO SVC ( <i>SIMSEES<sub>++</sub></i> ) .....	124
FIGURA 78 – POTÊNCIA REATIVA E SUSCEPTÂNCIA DO SVC ( <i>SIMSEES<sub>++</sub></i> ) .....	124
FIGURA 79 – TENSÕES DAS BARRAS ( <i>FASTSIM<sub>++</sub></i> ) .....	125
FIGURA 80 – POTÊNCIA REATIVA DOS GERADORES/SVC E SUSCEPTÂNCIA DO SVC ( <i>FASTSIM<sub>++</sub></i> ) .....	125
FIGURA 81 – COMPARAÇÃO DE DESEMPENHO DO PROGRAMA DE FLUXO DE POTÊNCIA GENERALIZADO COM O FLUXO DE POTÊNCIA CONVENCIONAL .....	129
FIGURA 82 – COMPARAÇÃO DE DESEMPENHO DE UMA ITERAÇÃO DO PROCESSO PARA O SISTEMA <i>S2000130</i>	
FIGURA 83 – COMPARAÇÃO DE DESEMPENHO DE UMA ITERAÇÃO DO PROCESSO PARA O SISTEMA <i>S2800130</i>	
FIGURA 84 – COMPARAÇÃO DE DESEMPENHO DOS PROGRAMAS DE SIMULAÇÃO DA DINÂMICA COMPLETA .....	132
FIGURA 85 – COMPOSIÇÃO DE UMA ITERAÇÃO DO PROCESSO DE CÁLCULO DO MÉTODO SIMULTÂNEO IMPLÍCITO ( <i>SIMSEES<sub>++</sub></i> ) .....	135
FIGURA 86 – DESEMPENHO DO SISTEMA <i>S0188</i> PARA VARIAÇÃO NO PASSO DE INTEGRAÇÃO ( <i>SIMSEES<sub>++</sub></i> ) .....	136
FIGURA 87 – ÂNGULOS DO GERADOR PARA $\Delta t = 1$ MSEG E $\Delta t = 10$ MSEG ( <i>SIMSEES<sub>++</sub></i> ) .....	137
FIGURA 88 – TENSÕES DAS PRINCIPAIS BARRAS DO SISTEMA ( <i>FASTSIM<sub>++</sub></i> ) .....	138
FIGURA 89 – GRANDEZAS ELÉTRICAS DE ALGUNS GERADORES DO SISTEMA ( <i>FASTSIM<sub>++</sub></i> ) .....	138
FIGURA 90 – TENSÃO DAS PRINCIPAIS BARRAS DO SISTEMA ( <i>FASTSIM<sub>++</sub></i> ) .....	140
FIGURA 91 – ÂNGULO E TENSÃO DE CAMPO DOS GERADORES ( <i>FASTSIM<sub>++</sub></i> ) .....	140

# Índice de Tabelas

TABELA 1 – DERIVADAS DO MODELO ZIP DE CARGA .....	59
TABELA 2 – DESEMPENHO COMPUTACIONAL DOS PROGRAMAS PARA AS SIMULAÇÕES DE LONGO PRAZO .....	115
TABELA 3 – CONVERGÊNCIA DO FLUXO DE POTÊNCIA .....	118
TABELA 4 – DIMENSÕES DOS SISTEMAS UTILIZADOS .....	127
TABELA 5 – DESEMPENHO COMPUTACIONAL DOS PROGRAMAS DE FLUXO DE POTÊNCIA .....	128
TABELA 6 – COMPARAÇÃO DE DESEMPENHO DOS PROGRAMAS DE SIMULAÇÃO DA DINÂMICA COMPLETA .....	132
TABELA 7 – TIPOS DE MODELOS UTILIZADOS.....	134
TABELA 8 – DESEMPENHO DO SISTEMA S0188 PARA VARIAÇÃO NO PASSO DE INTEGRAÇÃO .....	136
TABELA 9 – DESEMPENHO DO SISTEMA BRASILEIRO COM MODELOS REAIS.....	139

---

# Capítulo 1

## Introdução

### 1.1 Considerações Gerais

No atual cenário do setor elétrico é cada vez mais crescente a necessidade de programas computacionais para auxiliar nas atividades de operação e planejamento dos sistemas de energia elétrica. Isto tem levado a um crescimento substancial da quantidade de *softwares* disponíveis para o setor elétrico. No entanto, apesar do sucesso e qualidade destes *softwares*, sua grande maioria ainda está baseada na primeira geração das linguagens de alto nível (FORTRAN em sua grande maioria). É reconhecido que programas escritos em linguagens convencionais apresentam problemas quanto a manutenção e atualizações, sobretudo quando aumentam as dimensões e a complexidade destes programas. Outras características desta geração de programas podem ainda ser citadas:

- Estruturas de dados vulneráveis à manutenções;
- Baixo nível de integração entre diferentes aplicativos;
- Requerem elevados investimentos para manutenções e atualizações;
- Pouca ou nenhuma reusabilidade de códigos já escritos;
- Dificuldades no gerenciamento de módulos desenvolvidos por equipes distintas.

Em geral, o elenco de *software* atualmente disponível para o setor elétrico possui um baixo nível de integração e são usados isoladamente, diminuindo a produtividade dos estudos envolvidos. O novo cenário do setor elétrico, baseado em um ambiente desregulamentado e competitivo, onde as necessidades de mercado

---

levam a utilização máxima do sistema de transmissão, resulta na operação em condições de reduzidas margens de segurança. Neste cenário, é fundamental que os engenheiros de sistemas de potência possam contar com o apoio de ferramentas computacionais ágeis e extremamente flexíveis, de utilização amigável, e, principalmente, um conjunto amplo e integrado de ferramentas para simulação e análise de sistemas elétricos. A Modelagem Orientada a Objetos (MOO), como metodologia de desenvolvimento de *softwares*, apresenta-se como uma opção promissora para enfrentar os novos desafios da produção de ferramentas computacionais para a indústria de energia elétrica. A MOO visa a construção de uma estrutura de dados sólida e consistente, a partir da qual são implementados *softwares* flexíveis, com alto grau de reutilização dos códigos e com facilidades para manutenções e atualizações.

O presente trabalho tem por objetivo aplicar a Modelagem Orientada a Objetos para o desenvolvimento de uma base computacional capaz de integrar e dar suporte à construção de um amplo conjunto de ferramentas para simulação e análise de sistemas elétricos de grande porte. O modelo orientado a objetos deve possuir um grau de generalização tal que os aplicativos possam assimilar rapidamente modificações e alterações decorrentes do surgimento de novas metodologias de análise ou de novos equipamentos (ou o aperfeiçoamento dos já existentes).

Em uma segunda etapa do trabalho, e utilizando-se do ferramental de *software* já desenvolvido, desenvolveu-se uma nova geração de simuladores rápidos da dinâmica de médio e longo prazo para sistemas elétricos de potência. Estudou-se ainda a viabilidade da utilização das informações obtidas com a análise modal ao longo da simulação para o controle e avaliação da estabilidade de tensão do sistema.

## **1.2 Estrutura do Trabalho**

O texto deste documento está organizado em seis capítulos, da seguinte forma:

O Capítulo 1 apresenta uma breve introdução do assunto e os objetivos gerais do trabalho.

No Capítulo 2 são apresentados os conceitos e idéias básicas da Modelagem Orientada a Objetos, onde são descritas algumas metodologias de projeto disponíveis na literatura. A linguagem UML (*Unified Modeling Language*) é apresentada como

---

linguagem padrão de modelagem de objetos, e sua notação gráfica para descrever os modelos orientados a objetos é descrita. A compreensão dos conceitos básicos da modelagem orientada a objetos e da notação gráfica utilizada é indispensável para o entendimento da estrutura orientada a objetos proposta.

O Capítulo 3 apresenta o modelo computacional orientado a objetos proposto para aplicações gerais em sistemas de energia elétrica. São levantados os requisitos deste modelo e identificadas funcionalidades comuns que podem ser utilizadas em uma grande diversidade de aplicativos da área. Todos os modelos computacionais apresentados são descritos na forma de diagramas de classes utilizando-se a notação gráfica da linguagem UML. Um pacote para suporte e apoio matemático também é proposto e apresentado, com funções de gerenciamento de matrizes esparsas e solução de sistemas lineares de grande porte. Por fim, é apresentada uma estrutura computacional base para a construção de aplicativos utilizando como suporte os modelos computacionais desenvolvidos.

No Capítulo 4 é descrita a metodologia de simulação rápida da dinâmica de médio e longo prazo. Neste capítulo é proposta uma modificação na proposta original [38], que permite a utilização de técnicas de análise modal para o controle e avaliação da estabilidade de tensão do sistema ao longo da simulação, possibilitando estudar uma faixa mais ampla de fenômenos que a formulação original.

No Capítulo 5 são apresentados resultados obtidos com o modelo computacional desenvolvido. Os resultados mostram a flexibilidade obtida com o modelo proposto bem como algumas análises comparativas referente a performance computacional da estrutura proposta.

O Capítulo 6 apresenta as conclusões deste trabalho e propõe sugestões para trabalhos futuros.

---

## Capítulo 2

# Modelagem Orientada a Objetos

### 2.1 Considerações Gerais

A indústria da informática vêm oferecendo soluções que buscam facilitar a tarefa de programação no desenvolvimento de sistemas computacionais complexos. De tempos em tempos surge um novo paradigma de programação que rompe com antigos conceitos e apresenta uma forma inteiramente nova de abordar a tarefa de programação. Isto aconteceu com o surgimento das linguagens de alto nível, que substituíram as antigas linguagens de máquina; com as linguagens estruturadas, que substituíram as linguagens lineares; e, mais recentemente, com as linguagens orientadas a objeto, que se propõem a substituir as linguagens estruturadas ou procedurais. Outros paradigmas de programação recentes, muito embora restritos a aplicações específicas, são a programação para processamento paralelo e a programação visual.

O surgimento de um novo paradigma de programação normalmente impõe mudanças profundas na maneira de abordar a tarefa da programação. Neste sentido, a programação orientada a objetos (POO) e a modelagem orientada a objetos (MOO), mudam significativamente o enfoque da programação, antes centrado fundamentalmente nas funcionalidades de um programa e agora passando a priorizar os elementos conceituais do domínio do problema. Além do projeto e implementação de programas computacionais, os modelos baseados em objetos são úteis ainda para a compreensão de problemas, para a comunicação entre peritos de várias especialidades na elaboração das aplicações, para modelar empresas, preparar documentação, projetar bancos de dados, etc [1].

---

Este capítulo introduz os conceitos e idéias básicas da modelagem orientada a objetos, apresenta a UML (*Unified Modeling Language*) como linguagem padrão de modelagem de objetos, e sua notação gráfica para descrever os modelos orientados a objetos. A compreensão dos conceitos básicos da modelagem orientada a objetos e da notação gráfica utilizada é indispensável para o entendimento dos próximos capítulos.

## **2.2 Mudança no Enfoque de Programação**

O enfoque tradicional de modelagem para a construção de sistemas computacionais baseia-se na compreensão desse sistema como um conjunto de funções que executam processos sobre dados. Nesta abordagem os elementos centrais do programa são as funções que executam tarefas sobre os dados, ou seja, no contexto do programa, os dados são sempre elementos passivos que sofrem algum tipo de processamento.

No modelo orientado a objetos os dados (ou objetos) são os elementos centrais do programa. Modelando-se desta forma o programa passa a ser visto como uma coletânea de objetos que relacionam-se e interagem entre si, onde cada objeto representa um conceito real e bem definido do domínio do problema. Desta forma, um modelo orientado a objetos é mais próximo do problema real e, por esta razão, mais fácil de ser entendido e gerenciado, afinal a abordagem trata com elementos e conceitos que são familiares as pessoas envolvidas no projeto. Além disso, modelos fundamentados na essência do problema e não em funções que devem ser executadas são mais estáveis, mais robustos, suportam melhor o tempo, sua implementação e manutenção são mais baratas, e geralmente servem de instrumentos para a compreensão de aspectos do problema das quais não se tinha conhecimento inicialmente.

A mudança de enfoque é justificada pelo fato de que os objetos existem na natureza muito antes de haver qualquer tipo de aplicação deles em sistemas computacionais. Esta abordagem oferece como principais benefícios [2]:

- Mantém a modelagem do sistema o mais próximo possível de uma visão conceitual do mundo real;

- 
- Prioriza os dados na modelagem do sistema, que são os elementos mais estáveis entre todos aqueles que compõem um sistema computacional;
  - Oferece maior transparência na passagem da fase de modelagem para a de implementação, onde são apenas introduzidos detalhes no sistema, não requerendo uma reorganização do modelo.

O desenvolvimento de sistemas baseado em objetos concentra a maior parte dos esforços nas etapas de modelagem e entendimento do problema, somente quando os conceitos relativos a aplicação são identificados, organizados e compreendidos é que os detalhes de implementação são tratados. A vantagem advém do fato que as eventuais falhas são tratadas ainda na fase de projeto quando tem correção menos dispendiosa do que àquelas encontradas durante a fase de implementação. Na abordagem tradicional a ênfase repousa mais na implementação do que na análise ou projeto. O enfoque precoce dos problemas de implementação restringe as opções de projeto e muitas vezes conduz a um produto menos flexível.

### **2.2.1 O Conceito de Objeto**

O objeto é a entidade fundamental do modelo orientado a objetos e representa um conceito no domínio do problema, algo com limites nítidos e significado bem definido com relação ao problema em questão. Um objeto consiste de um conjunto de atributos (dados) que definem a sua descrição interna e o seu estado atual, e um conjunto de métodos (funções) que executam ações pertinentes ao conceito que o objeto representa. Um objeto pode ser entendido como algo real dentro do programa, uma estrutura com identidade própria. Ou seja, dois objetos são distintos mesmo que todos os valores de seus atributos sejam idênticos. Tomando como exemplo uma aplicação qualquer para geometria plana, a Figura 1 mostra três possíveis objetos pertencentes ao domínio da aplicação, seus atributos e funções. A escolha dos objetos pode parecer óbvia para este exemplo simples, mas representa uma das tarefas mais difíceis na modelagem de sistemas complexos e não triviais.

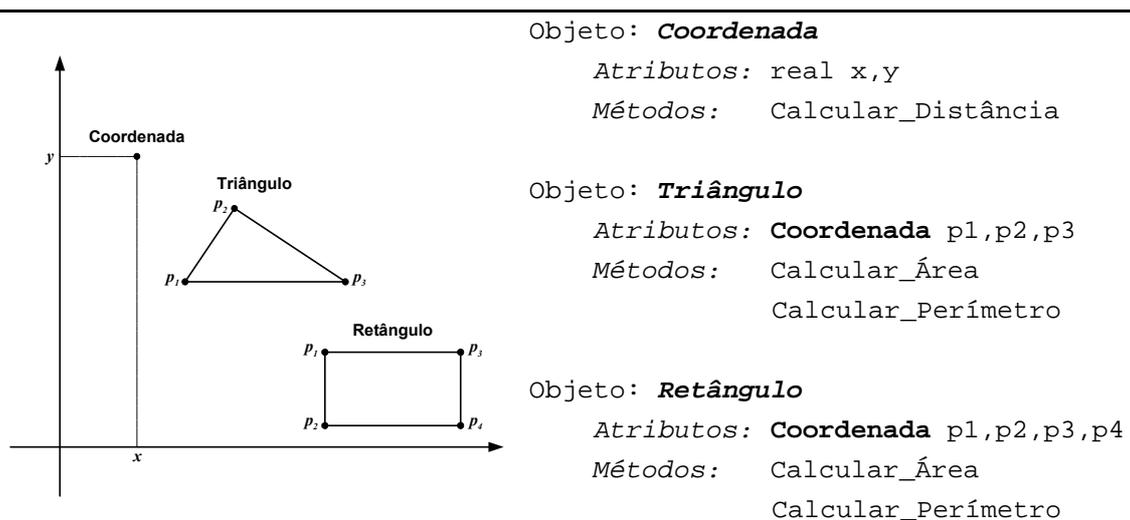


Figura 1 – Exemplos de Objetos de uma Aplicação para Geometria Plana

Os atributos e métodos de um objeto devem concentrar os aspectos essenciais da entidade que representa e ignorar propriedades acidentais, evitando atribuir-lhe características que não pertencem ao conceito que o objeto representa simplesmente para facilitar algum aspecto da implementação. Isto significa concentrar-se no que um objeto “é” e “faz”, antes de decidir “como” ele deve ser implementado, preservando assim a liberdade de tomar decisões e evitando o comprometimento prematuro com detalhes.

Os objetos comunicam-se entre si e com o mundo exterior através de mensagens. Cada mensagem ativa um ou um conjunto de métodos do objeto, requisitando uma determinada ação que o objeto compreende e sabe como executar. Os métodos definem então todas as tarefas que o objeto está habilitado a realizar, ou seja, são os elementos ativos dentro de um objeto. Durante a execução de uma ação os objetos normalmente fazem uso de seus atributos internos e de seu alcance na hierarquia do problema, requisitando, se necessário, a ação de outros objetos para a execução de uma operação em particular. Assim, um programa orientado a objetos é essencialmente o resultado de uma colaboração organizada entre ações individuais de objetos.

A declaração da estrutura computacional que descreve um tipo específico de objeto denomina-se *Classe*. Objetos com a mesma estrutura de dados (atributos) e o mesmo comportamento (métodos) pertencem a mesma classe. Cada objeto é dito ser instância de sua classe, porém, como já foi dito, possui identidade própria.

---

## 2.2.2 Herança e Polimorfismo

Existe alguma discordância entre os autores sobre quais são exatamente as características exigidas pela abordagem baseada em objetos, mas elas geralmente incluem quatro aspectos essenciais: Identidade e Classificação (apresentados no item 2.2.1) e Polimorfismo e Herança [1][2] que serão discutidos a seguir.

- **Polimorfismo:** representa a capacidade de vários objetos diferentes possuírem métodos com o mesmo nome (mensagens idênticas), geralmente associados a uma tarefa semelhante. Isto permite uma padronização das mensagens com a mesma finalidade para objetos distintos, facilitando a legibilidade do código. Um exemplo claro disto é a definição do método para o cálculo da área de uma figura geométrica do exemplo apresentado na Figura 1. Tanto para o objeto *Triângulo*, quanto para o objeto *Retângulo*, a mensagem será igual (Calcular Área), porém cada objeto terá uma implementação diferente para este método.
- **Herança:** é a capacidade do objeto herdar todas as características de um objeto antecessor (que lhe deu origem). A herança torna possível a especialização de objetos descendentes através da inclusão de novas características a este objeto. O processo de herança define uma hierarquia de objetos, todos descendendo e herdando as características do seu objeto antecessor. Isto permite uma eficiente organização de funcionalidades entre os objetos, onde os objetos dos níveis superiores da hierarquia possuem funcionalidades mais gerais e os objetos dos níveis inferiores possuem funcionalidades mais específicas e direcionadas a uma determinada aplicação. Além disso, esta capacidade reduz substancialmente as repetições nos projetos e programas.

## 2.3 Metodologias de Projeto Orientado a Objetos

Várias metodologias para modelagem e projeto baseados em objetos foram propostas ao longo dos últimos anos [2]. Embora existam interpretações particulares em cada metodologia, todas elas apresentam a programação orientada a objetos como um novo paradigma, ou seja uma nova maneira de pensar, e não simplesmente uma técnica de programação. A seguir são apresentadas resumidamente as metodologias mais relevantes no campo da orientação a objetos.

---

### 2.3.1 Método de Booch [2]

Grady Booch propôs um método que consiste no emprego de técnicas de desenho orientado a objetos, apesar de ter sido estendido para contemplar também a análise orientada a objetos. Booch descreve um objeto como sendo um modelo do mundo real que consiste de dados e habilidades para o tratamento desses dados. Uma vez tendo sido localizados e definidos os objetos, esses passam a servir de base para os módulos do sistema ou são considerados módulos relacionados. Segundo Booch o desenho estruturado funciona bem como linguagem de programação estruturada, mas o enfoque estruturado direciona a concentração de esforços na lógica do módulo, com pouca ênfase no uso dos dados. O resultado é a geração de código adicional e menos dados, pois estruturas são derivadas dos processos e esses necessitam interagir com os dados. A estreita relação existente entre o desenho orientado a objetos, proposto por Booch, e programação orientada a objeto permite aos projetistas tomar decisões antes da criação do código, contribuindo para a melhoria do sistema gerado.

### 2.3.2 OMT [2][1]

A Técnica de Modelagem de Objetos (OMT – Object Modeling Technique) foi desenvolvida por Rumbaugh, e é baseada na modelagem semântica de dados. O método OMT tornou-se um enfoque testado e maduro, cobrindo as diversas fases do desenvolvimento orientado a objetos. A notação empregada por Rumbaugh é semelhante a dos métodos estruturados e utiliza a notação de modelo de objeto que suporta conceitos de modelagem de dados (atributos e relacionamentos), modelagem de objetos (composição e agregação) e herança. O método OMT utiliza três modelos ortogonais e interligados para a descrição de um sistema completo, são eles:

- **Modelagem de Objetos:** descreve a estrutura estática dos objetos de um sistema e seus relacionamentos. Utiliza para isso os *Diagramas de Objetos*, que são gráficos cujos nós são classes de objetos e cujos arcos são os relacionamentos entre as classes.
- **Modelagem Dinâmica:** descreve os aspectos de um sistema que se modificam com o tempo, sendo usado para especificar e implementar os aspectos de controle do sistema. O modelo dinâmico contém *Diagramas de Estados*, que são gráficos cujos nós são estados e cujos arcos são transições entre estados causados por eventos.

- 
- **Modelagem Funcional:** descreve as transformações dos valores dos dados de um sistema. Este modelo contém *Diagramas de Fluxo de Dados*, que são gráficos cujos nós são processos e cujos arcos são fluxos de dados.

O ponto forte do método OMT é a notação utilizada e o enfoque relativamente conservador do uso da teoria de objetos. Por outro lado, um problema apresentado é a falta de notação específica para representar a passagem de mensagem de um objeto para outro.

### 2.3.3 Método OOSE [2]

Jacobson criou as bases para o método *Object-Oriented Software Engineering* (OOSE). O que diferencia a OOSE de outros métodos orientados a objeto é o seu foco em *Casos de Uso* e a categorização de pessoas e equipamentos dependendo de seu papel no sistema global. A análise no método de Jacobson é baseada em modelos de requerimentos e análise, que consistem de um conjunto de *Casos de Uso*, de um modelo do domínio do problema, e de uma descrição da interface do sistema. Seguindo a criação do modelo de análise, são gerados diagramas baseados no modelo que também descrevem a comunicação entre blocos. O modelo de blocos é então implementado utilizando-se linguagens apropriadas e ferramentas orientadas a objeto. Um dos pontos fracos do enfoque original da OOSE é a notação simplista usada para objetos de domínio (objetos são simplesmente representados como círculos). Por outro lado, o método OOSE tem sido adaptado para a engenharia de negócio, onde as idéias são usadas para modelar e melhorar processos.

Várias outras metodologias para modelagem e projetos baseados em objetos foram propostas [2], destacam-se os métodos de SHLAER/MELLOR, COAD/YOURDON, MARTIN/ODELL, WIRFS-BROCK (CRC), EMBLEY/KURTZ e FUSION. Porém, o método de Booch, a OMT, e o método OOSE foram as metodologias de modelagem para objetos que mais se destacaram, obtendo reconhecimento pela comunidade usuária como métodos de classe mundial. Seus autores juntaram forças, então, e unificaram seus métodos criando uma linguagem de modelagem unificada, a UML (*Unified Modeling Language*). Esta linguagem tem sido aceita como linguagem de modelagem padrão atualmente, e será vista em detalhes no próximo item.

---

## 2.4 Unified Modeling Language (UML)

A UML [3] é uma tentativa de padronizar a modelagem orientada a objetos de forma que qualquer sistema possa ser modelado corretamente e com consistência, interagindo facilmente com outras aplicações, e de simples atualização e compreensão. No entanto, a linguagem de modelagem criada por Booch, Rumbaugh e Jacobson vai além de uma simples padronização em busca de uma notação unificada, uma vez que contém conceitos novos que não são encontrados em outros métodos orientados a objetos.

Atualmente, a UML é aceita como linguagem padrão (aprovada em novembro de 1997 pela *OMG – Object Management Group* como linguagem padrão de desenvolvimento) para especificar, visualizar, documentar e construir sistemas computacionais orientados a objeto, podendo ser utilizada em todos os processos ao longo do ciclo de desenvolvimento e através de diferentes tecnologias de implementação.

A UML é composta por um *Processo para o Desenvolvimento de Sistemas*, e por uma linguagem gráfica para *Representação dos Modelos* (Diagramas da UML). Cada um dos componentes da UML possui uma enorme riqueza de detalhes e notações que permitem modelar uma grande variedade de sistemas e situações de projeto. Não é propósito deste trabalho apresentar em detalhes todos os aspectos e notações da UML, serão apresentados tão somente os aspectos necessários ao entendimento dos modelos propostos nos capítulos seguintes deste trabalho. Uma descrição mais detalhada da UML pode ser encontrada em [3].

### 2.4.1 O Processo de Desenvolvimento da UML

O processo de desenvolvimento da UML (denominado *Rational Objectory Process*) é um processo interativo centrado nos conceitos de Casos de Uso e métodos de projeto orientados a objetos. O *Objectory* é composto por quatro componentes principais que são descritos em termos de atividades e representam o aspecto estático do processo. Os componentes principais do processo são descritos a seguir:

- **Levantamento dos Requisitos:** A meta do levantamento dos requisitos consiste em descrever “o *quê o sistema deve fazer*”, permitindo aos desenvolvedores e usuários concordarem sobre uma descrição comum do problema. O sistema é delimitado, sendo definido seus contornos e comportamento a partir das

---

necessidades potenciais modeladas nos Casos de Uso. O diagrama de Casos de Uso é a principal técnica utilizada na fase de levantamento dos requisitos, muito embora um diagrama de classes de alto nível possa ser especificado.

- **Análise e Projeto:** A meta da análise é mostrar como o sistema será materializado na fase de implementação. Busca-se construir um sistema estruturado e robusto que execute, em um ambiente de implementação específico, as funções e tarefas descritas nos Casos de Uso. O propósito básico do modelo de análise é prover uma compreensão do sistema e facilitar a comunicação entre técnicos e usuários, não voltando-se para soluções técnicas ou detalhes de código dos programas. Por outro lado, o modelo de projeto serve como uma abstração do código fonte, atuando como um esboço de sua estrutura. O projeto também resulta em descrições da visão interna e realizações de Casos de Uso, detalhando como são percebidos em termos de objetos e classes participantes.
- **Implementação:** A fase de implementação ocorre no instante em que as classes necessitam ser programadas tomando-se por base o modelo do projeto. O sistema é percebido através da implementação, produzindo-se os arquivos de código fonte que resultarão em um sistema executável. Se o projeto foi elaborado corretamente e com detalhes suficientes, a tarefa de codificação é mecânica. O trabalho é suportado por regras de programação que buscam padronizar o código desenvolvido por programadores diferentes e prevenir construções inadequadas.
- **Testes:** O objetivo desta fase é identificar erros no código através de um número de casos de teste que avaliem diferentes aspectos de cada módulo. Primeiramente testa-se cada Caso de Uso separadamente para avaliar se as classes participantes trabalham corretamente e, em seguida, realizam-se os testes do sistema como um todo empregando as descrições dos Casos de Uso como entrada para o teste. O processo de testes dirigido pelos Casos de Uso é um processo quase automatizado, uma vez que os teste já estão definidos desde as etapas iniciais do projeto (quando são definidos os Casos de Uso).

Cada componente do processo da UML é subdivido em ciclos ou fases, representando o aspecto dinâmico e interativo do processo. Assim, o trabalho é realizado através de interações, sendo que o produto final é obtido de maneira incremental, muito embora o produto em termos de programa executável somente estará disponível nas etapas finais do processo.

## 2.4.2 A Representação dos Modelos na UML[2][3]

O modo para descrever os vários aspectos da modelagem pela UML é através da notação definida pelos vários tipos de diagramas. Cada diagrama representa uma perspectiva do modelo (*views*), mostrando aspectos particulares do sistema e dando enfoque a ângulos e níveis de abstrações diferentes para que uma figura completa do sistema seja construída. Um diagrama é uma apresentação gráfica de uma coleção de elementos de modelo, freqüentemente mostrado como um gráfico conectado de arcos e vértices. Os diagramas principais da UML são:

- **Diagramas de Classes:** Gráfico bidimensional de elementos de modelagem que pode conter tipos, pacotes, relacionamentos, instâncias, objetos e vínculos (conexões entre dois objetos). Um diagrama de classes denota a estrutura estática de um sistema e as classes representam coisas que são manipuladas por esse sistema. O diagrama é considerado estático pois a estrutura descrita é sempre válida em qualquer ponto no ciclo de vida do sistema. Um diagrama de objetos é uma variação do diagrama de classe e utiliza quase a mesma notação.

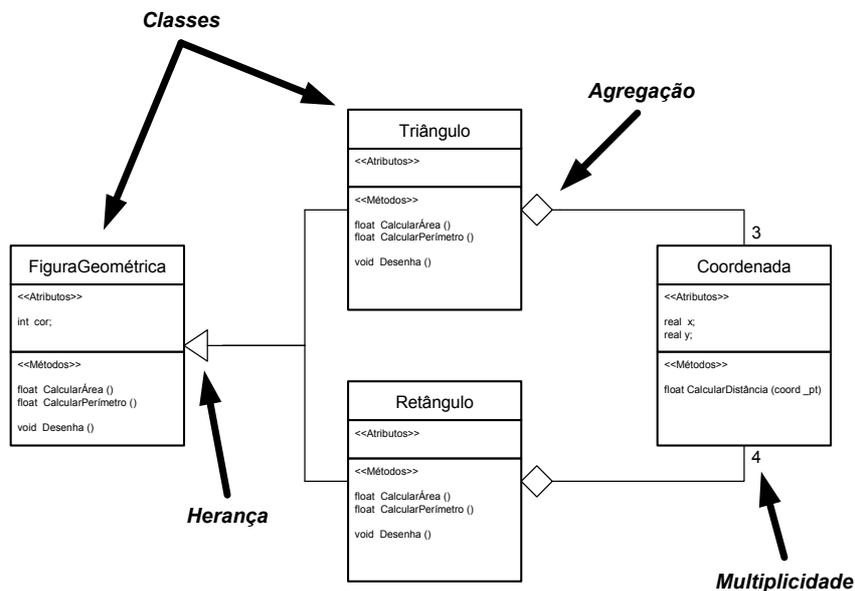


Figura 2 – Exemplo de um Diagrama de Classes da UML

- **Diagramas de Casos de Uso:** Os Casos de Uso descrevem a funcionalidade do sistema percebida por atores externos (usuários). Um ator interage com o sistema podendo ser um usuário, dispositivo ou outro sistema. Um Caso de Uso descreve as operações que o sistema deve cumprir para cada usuário, formalizando as

funções que o sistema deve cumprir. No entanto, os Casos de Uso utilizam sempre uma linguagem informal (de fácil entendimento pelo usuário) para descrever as operações a serem cumpridas.

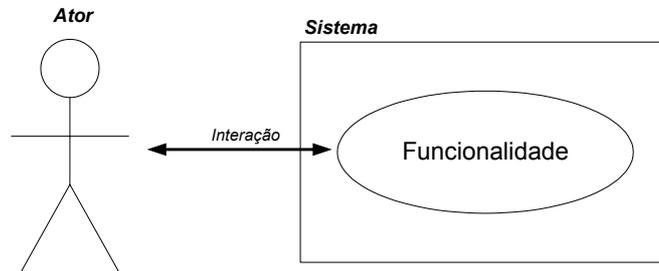


Figura 3 – Exemplo de um Diagrama de Casos de Uso da UML [2]

- **Diagramas de Interação – Diagramas de Seqüência:** Apresenta a interação e a seqüência de tempo dos objetos que participam de uma ação. As duas dimensões de um diagrama de seqüência consistem na dimensão vertical (tempo) e na direção horizontal (objetos diferentes). O diagrama de seqüência mostra a colaboração dinâmica entre um número de objetos, sendo o aspecto principal desse diagrama mostrar a seqüência de mensagens enviadas entre objetos.

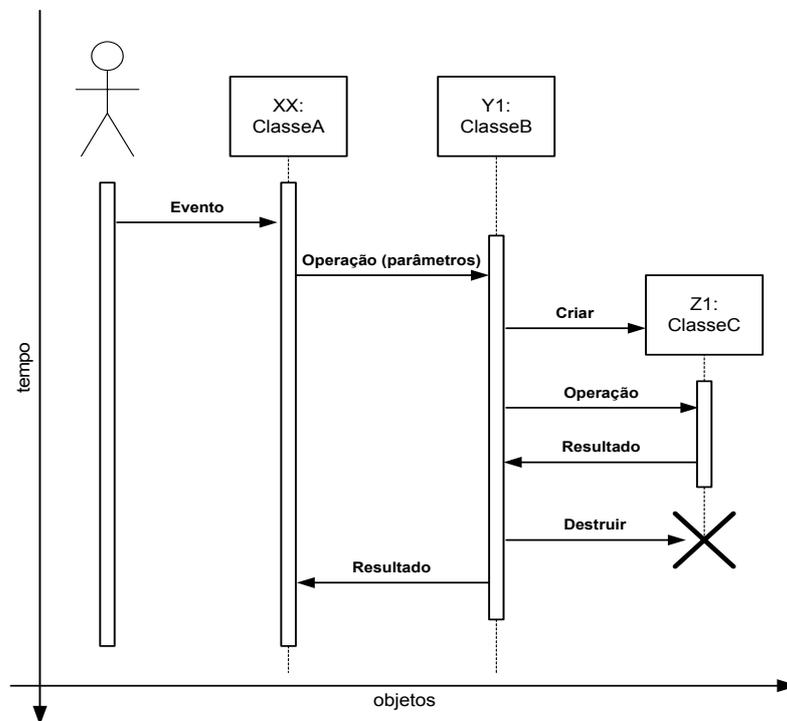


Figura 4 – Exemplo de um Diagrama de Seqüência da UML [2]

- 
- **Diagramas de Interação – Diagramas de Colaboração:** Mostra uma interação de um Caso de Uso organizado em torno de objetos e seus vínculos mútuos, de maneira que são usados números de seqüência para evidenciar a seqüência de mensagens. É desenhado como um diagrama de objeto onde um número de objetos é mostrado com seus relacionamentos.

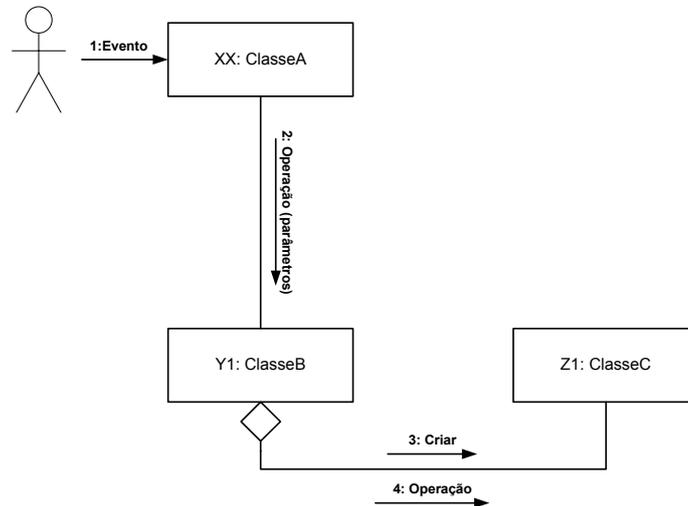


Figura 5 – Exemplo de um Diagrama de Colaboração da UML [2]

- **Diagramas de Estado:** Mostra as seqüências de estados que um sistema assume em resposta a estímulos recebidos, juntamente com suas respostas e ações. Um diagrama de estado é tipicamente o complemento de uma classe e relaciona os possíveis estados que objetos da classe podem ter e quais eventos podem causar a mudança de estado. Um tipo de diagrama de estado especial são os *Diagramas de Atividade*, onde a maioria dos estados são estados de ação e a maioria das transições é ativada por conclusão das ações. Seu propósito é estudar os fluxos dirigidos por processos internos, descrevendo as atividades desempenhadas em uma operação.

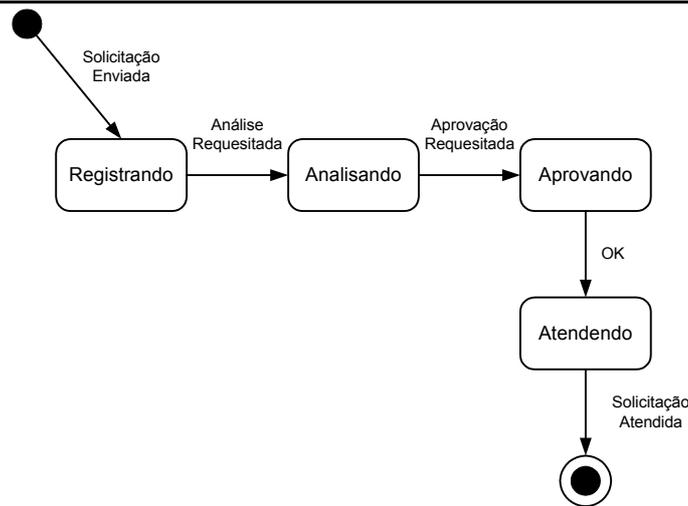


Figura 6 – Exemplo de um Diagrama de Estado da UML [2]

Existem ainda outros diagramas da UML de menor importância, que não serão descritos neste trabalho. São eles: Diagramas de Componente e Diagramas de Implantação [3].

### 2.4.3 O Diagrama de Classes da UML e sua Notação Gráfica

Os conceitos usados nos diagramas são chamados de modelos de elementos. Um modelo de elemento é definido formalmente na UML com o exato significado do que ele representa, sem definições duvidosas ou ambíguas. A UML também define uma representação gráfica para cada modelo de elemento, que é mostrada nos diagramas. Um elemento pode existir em diversos tipos de diagramas, mas existem regras que definem que elementos podem ser mostrados em que tipos de diagramas. Alguns exemplos de modelos de elementos são as classes, objetos, estados, pacotes e componentes. A seguir serão definidos e exemplificados alguns modelos de elementos da UML, juntamente com sua representação gráfica.

- **Classe:** Uma classe é a descrição de um tipo de objeto. Todos os objetos são instâncias de classes, onde a classe descreve as propriedades e comportamentos daquele objeto. Em UML as classes são representadas por um retângulo dividido em três compartimentos: o compartimento de nome, que conterá apenas o nome da classe modelada, o de atributos, que possuirá a relação de atributos que a classe possui em sua estrutura interna, e o compartimento de operações, que serão os métodos de manipulação de dados e de comunicação de uma classe com outras do sistema. A sintaxe usada em cada um destes compartimentos é independente de qualquer linguagem de programação. A UML aceita uma forma

---

simplificada de representação das classes, normalmente quando não é necessário uma maior riqueza de detalhes, onde apenas o retângulo com o nome da classe aparece.

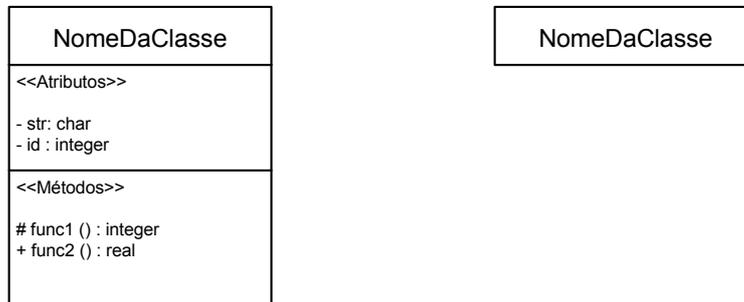


Figura 7 – Representação de Classes na UML

- **Relacionamentos entre Classes:** Os relacionamentos ligam as classes/objetos entre si criando relações lógicas entre estas entidades. Os relacionamentos podem ser dos seguintes tipos:

*Associação:* são ligações físicas ou conceituais entre os objetos, uma associação normalmente determina interações entre objetos. Uma associação é uma conexão entre classes, isto significa que é também uma conexão entre objetos daquelas classes.

*Dependência:* indica um relacionamento entre duas ou mais classes, onde uma classe cliente é dependente de outra, porém não existe uma ligação física ou estrutural entre os objetos. Uma modificação em um elemento independente afetará diretamente elementos dependentes da anterior.

*Herança:* permite que uma classe herde todas as características de outra classe (superclasse). Através deste mecanismo é possível reunir características comuns entre objetos correlatos, definindo uma estrutura hierárquica onde os níveis mais altos são elementos mais gerais e os níveis mais baixos elementos mais especializados.

*Agregação:* define um relacionamento “parte-de”, ou seja um objeto é formado de vários outros objetos componentes. A agregação é um caso particular da associação. As palavras chaves usadas para identificar uma agregação são: “consiste em”, “contém”, “é parte de”.

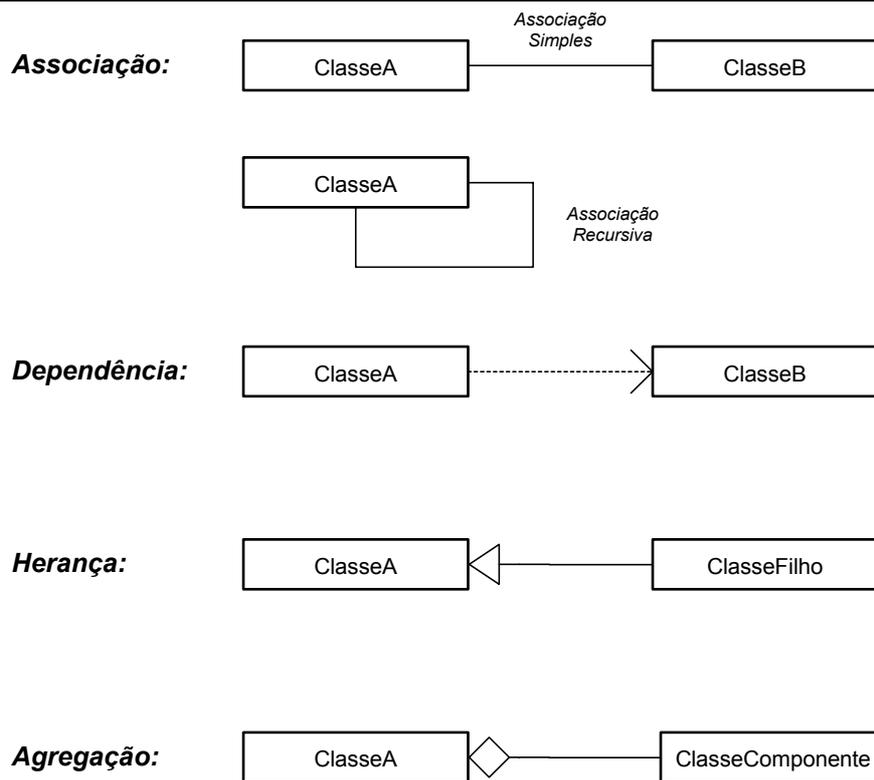


Figura 8 – Representação de Relacionamentos entre Classes na UML

- **Pacotes:** São macroentidades de propósito geral que agregam vários objetos e seus relacionamentos definindo um conceito mais amplo e geral. Pacote é um mecanismo de agrupamento, onde todos os modelos de elementos podem ser agrupados.



Figura 9 – Representação de Pacotes na UML

- **Templates:** Representam um tipo especial de classe parametrizada onde o tipo base dos atributos é passado como parâmetro. Esta técnica permite definir uma classe sem especificar todos os tipos que ela utiliza. Os tipos não especificados são fornecidos como parâmetro no momento da utilização.

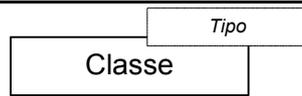


Figura 10 – Representação de Templates na UML

## 2.5 Considerações Finais

A organização da modelagem orientada a objetos em diagramas, especificando características estáticas e dinâmicas do modelo, permite que qualquer tipo de comportamento do sistema possa ser facilmente visualizado. Com a UML os modelos possuem um padrão simples e extremamente robusto para especificar e descrever a grande maioria das funções, relacionamentos e técnicas de desenvolvimento orientado a objetos. Desta forma, a UML facilita a comunicação e o aproveitamento dos modelos desenvolvidos pelos analistas envolvidos no processo de produção de *software*, já que a linguagem que será utilizada por todos será a mesma, acabando assim com qualquer problema de interpretação e mal-entendimento de modelos criados por outros desenvolvedores. Além disso, os modelos criados poderão ser facilmente analisados por futuras gerações de desenvolvedores acabando com a diversidade de tipos e nomenclaturas de modelos.

No entanto, se por um lado a *Representação de Modelos* da UML é extremamente útil e recomendável para o desenvolvimento de *softwares* na área de sistemas de energia elétrica, o *Processo de Desenvolvimento* da UML não se aplica com a mesma eficácia. Programas para sistemas de energia normalmente possuem características bastante peculiares, onde não se conhece claramente a forma do produto final. Inúmeras modificações e alterações se fazem necessárias durante o ciclo de desenvolvimento do *software*, sobretudo se novas metodologias e algoritmos de engenharia estão sendo desenvolvidos e experimentados. Assim, processos que utilizam como base *Casos de Uso* (base da metodologia de desenvolvimento recomendado pela UML) possuem dificuldades no desenvolvimento desta classe de programas, uma vez que não se conhece exatamente as características do produto final. A metodologia de desenvolvimento para programas deste tipo deve ser criada pelos próprios desenvolvedores conforme as peculiaridades e característica do programa em desenvolvimento.

---

## Capítulo 3

# Modelagem Orientada a Objetos Aplicada a Sistemas de Energia Elétrica

### 3.1 Considerações Gerais

A primeira geração de ferramentas computacionais para sistemas de energia elétrica constituíam-se de programas aplicativos direcionados para um número limitado de funções (em sua grande maioria escritas em linguagem FORTRAN). Muito pouca ou nenhuma importância era dada à reutilização do código para futuros aplicativos que se fizessem necessários. Assim, cada vez que uma nova ferramenta era necessária um programa totalmente novo era implementado. Contrastando com esta realidade, o setor elétrico sempre exigiu ferramentas com características bastante dinâmicas, onde o surgimento de novos equipamentos (e por conseqüência novos modelos computacionais), o aperfeiçoamento dos modelos já existentes e o surgimento de novas metodologias de análise possam ser rapidamente assimiladas pelos programas aplicativos. Sendo assim, os *softwares* para a área de energia elétrica devem ser ágeis e extremamente flexíveis face a mudanças. Um projeto que não levar em consideração a possibilidade de mudanças ao longo de sua vida está sujeito ao risco de uma grande reformulação no futuro.

Atualmente duas classes de ferramentas orientadas a objetos tem surgido para agilizar a produção de programas aplicativos, são os **frameworks** e os **toolkits** de classes [4]. Um *framework* constitui-se de um conjunto de classes cooperantes que constroem um projeto reutilizável para uma específica classe de aplicativos (aplicativos para sistemas de energia elétrica, por exemplo). Um *framework* é customizado para uma aplicação específica através da criação de subclasses

---

direcionadas para a aplicação, derivadas das classes abstratas do *framework*. Assim, um *framework* dita a arquitetura da aplicação, define sua estrutura geral, sua divisão em classes e como estas colaboram, de maneira que o projetista ou o implementador da aplicação pode se concentrar nos aspectos específicos da sua aplicação. Como resultado, as decisões de projeto são reduzidas permitindo não somente construir aplicações mais rapidamente como também aplicações com estrutura similar, mais consistentes e fáceis de manter. O projeto de um *framework* aposta que uma arquitetura funcionará para todas as aplicações do domínio, portanto é imperativo projetar o *framework* de maneira que ele seja tão flexível e extensível quanto possível. **Toolkits**, por sua vez, são conjuntos de classes relacionadas e reutilizáveis, projetadas para fornecer uma funcionalidade útil e de finalidade geral (interfaces gráficas, por exemplo). *Toolkits* devem funcionar em muitas aplicações para serem úteis, independentemente de quais serão as aplicações e necessidades especiais.

Este capítulo apresenta um *framework* de classes (base computacional) para aplicações gerais em sistemas de energia elétrica. São levantados os requisitos de um modelo orientado a objetos base para a área de sistemas elétricos e identificadas funcionalidades comuns que podem ser utilizadas em uma grande diversidade de aplicativos da área. Um *toolkit* para suporte e apoio matemático também é proposto e apresentado, com funções de gerenciamento de matrizes esparsas e solução de sistemas lineares de grande porte. Por fim, é apresentada uma estrutura computacional base para a construção de aplicativos utilizando como suporte o *framework* e o *toolkit* desenvolvidos.

### 3.1.1 Revisão Bibliográfica

O número de trabalhos encontrados na literatura onde foram utilizadas técnicas de modelagem orientada a objetos ainda é pequeno, contudo há uma tendência de alteração significativa deste cenário. Inicialmente os trabalhos relacionados ao tema concentraram-se quase que exclusivamente nas áreas de interface gráfica [9][10][11], gerenciamento de base de dados [6][7][8][16][29] e na solução de fluxo de potência [12][14][16][19][26]. Desenvolvimentos mais recentes utilizando a orientação a objetos propõem a implementação de uma plataforma base orientada a objetos para aplicações gerais em SEE [16][21][26][27][28][29][31][32][33][34]. Outras aplicações da metodologia orientada a objetos encontram-se nas áreas de restauração de sistemas após a ocorrência de contingências [15][20], planejamento da transmissão [18], distribuição de energia elétrica [17][21][32], simulação da dinâmica de SEE

---

[13][23][24], simulação de faltas em subestações [22] e configuração de redes elétricas [10][28][32]. A seguir apresenta-se uma revisão bibliográfica relacionada ao tema.

Em 1990, Neyer et alii [5] aplicaram técnicas de programação orientada a objetos ao problema do fluxo de potência e avaliaram as potencialidades, na época, de aplicação da POO em SEE. Os autores propõem uma estrutura orientada a objetos que suporta tanto a configuração da rede elétrica quanto o problema do fluxo de potência, utilizando para isso o conceito de elementos físicos e conceituais. Os elementos físicos são estáticos na estrutura e os elementos conceituais dinâmicos, determinando a estrutura topológica atualizada do sistema conforme o estado operativo dos dispositivos de manobra (chaves, disjuntores, etc.). Foram relatados desempenhos 2 a 3 vezes inferiores aos de implementações equivalentes em FORTRAN. Especificamente, 30 a 50% do tempo de processamento foram identificados como tempos despendidos em *overhead* associado ao uso da POO.

Flinn e Dugan em [7] propõem uma base de dados orientada a objetos capaz de, segundo os autores, suportar diversas aplicações, tais como análise harmônica, fluxo de potência, curto-circuito e estabilidade transitória. Relatam dificuldades em usar banco de dados relacional para modelar estruturas complexas como as encontradas em sistemas de energia elétrica. Os autores destacam ainda a necessidade de um banco de dados único para sistemas de energia capaz de suportar as mais diversas aplicações.

Foley et alii em [9] apresentam uma interface gráfica homem-máquina para SEE utilizando a POO. Neste trabalho é proposta uma estrutura hierárquica de classes direcionada para o desenvolvimento de interfaces gráficas. Posteriormente, Foley e Bose [10] utilizam conceitos semelhantes aos elementos físicos e conceituais de [5] para abordar o problema da configuração de redes elétricas. A estrutura computacional agrupa os dispositivos em Subestações, imitando a estrutura física real de um SEE. Um aplicativo de Fluxo de Potência pelo método de Gauss-Seidel é também implementado sobre a estrutura orientada a objetos proposta, segundo os autores esta metodologia é mais coerente com a metodologia orientada a objetos.

Hakavik e Holen [12] propuseram uma nova estrutura hierárquica de classes para a modelagem computacional dos elementos de SEE e avaliaram a viabilidade da utilização da POO para manipulações de matrizes esparsas. Os autores utilizam uma estrutura de classes direcionada para o problema de Fluxo de Potência, e uma hierarquia para os aplicativos descendente da estrutura que descreve o SEE. São

---

descritos problemas de performance computacional em função de uma interpretação purista da filosofia da POO. Os autores também mostraram que a POO não causa necessariamente aumento do tempo computacional.

Zhou [14] aplicou a POO para a implementação de um programa de fluxo de potência, utilizando para isso uma estrutura orientada a objetos semelhante a proposta em [12]. Neste trabalho é relatado um bom desempenho computacional da POO, embora as razões não tenham sido claramente expostas. Foi apresentado também a aplicação dos conceitos da POO para manipulações de matrizes esparsas e solução de sistemas lineares de grande porte.

Gaing et alii [15] utilizaram a POO para implementar um pacote computacional para auxílio a restauração de SEE após a ocorrência de contingências, integrando interface gráfica e programas aplicativos. Os autores seguem criteriosamente todos os passos da metodologia de desenvolvimento da OMT [1], sendo o primeiro trabalho que utiliza uma notação formal para representar a estrutura orientada a objetos proposta. São apresentados *Diagramas de Objetos*, de *Fluxo de Dados* e de *Estados* para descrever o processo de restauração.

Phillips et alii inovam em [16] ao proporem uma base computacional orientada a objetos para aplicações gerais em sistemas de energia elétrica (*framework*). A estrutura é centrada sobre um banco de dados orientado a objetos, e utiliza o fluxo de potência como aplicativo exemplo. Os autores relatam problemas de desempenho computacional do fluxo de potência quando comparado a um programa tradicional escrito em C, porém o desempenho do banco de dados orientado a objetos apresenta desempenho de 36% a 173% superior aos bancos de dados relacionais tradicionais.

Zhu e Lubkeman em [17] propõem uma base orientada a objetos para sistemas de distribuição. Os autores acreditam que, uma vez que a estrutura computacional é modelada utilizando como base os conceitos reais do sistema esta será capaz de suportar uma grande quantidade de aplicativos. Para isso, propõem um modelo para descrever o SEE independente de qualquer aplicativo. Zhu e Lubkeman utilizaram uma estrutura hierárquica, derivada de um elemento base denominado *Device*, dividida em três grandes grupos: elementos *Shunt*, elementos *Branch* e *Protective*. Todos os elementos (*Devices*) são agrupados em classes que representam os alimentadores e subestações de um sistema de distribuição. O trabalho mostra *Diagramas de Classes* (OMT) detalhados da estrutura utilizada.

---

Handschin et alii [18] utilizaram a modelagem orientada a objetos para planejamento da transmissão em ambiente desregulamentado. Os autores classificam os dispositivos do SEE de acordo com o número de pólos e representam um modelo para o mercado de energia.

Fuerte-Esquivel et alii [19] realizaram uma aplicação da POO para a implementação de um programa de fluxo de potência considerando a modelagem de dispositivos FACTS. Neste trabalho também é relatado um desempenho computacional comparável com implementações equivalentes em FORTRAN. As razões para esses resultados são atribuídas a boas práticas de programação e a não utilização purista da filosofia da POO na resolução de sistemas lineares de grande porte.

Manzoni, Silva e Decker, em [24], desenvolveram um novo modelo hierárquico de classes para a representação de SEE. Esta estrutura foi aplicada especificamente para o problema da simulação da dinâmica. No mesmo trabalho são propostos uma classe *Matriz* e técnicas para a solução eficiente de sistemas lineares esparsos de grande porte, envolvendo matrizes com elementos complexos. Foram realizados testes com sistemas elétricos de grande porte e obtidos índices de desempenho comparáveis aos do programa de Análise de Transitórios Eletromecânicos (ANATEM), desenvolvido pelo CEPEL. Detalhes relativos a modelagem dos elementos do SEE, incluindo-se controladores, caldeiras e outros, podem ser encontrados em [23]. O bom desempenho computacional é atribuído a práticas de programação que contemplem os requisitos de eficiência computacional para a resolução de sistemas lineares de grande porte. Os resultados deste trabalho mostram que a POO é uma alternativa para o desenvolvimento de *softwares* de grande porte em SEE, mesmo quando são críticos os requisitos de desempenho computacional.

Zhu e Jossman em [21] propõem a utilização de *Padrões de Projeto (Design Patterns* [4]) para a modelagem de certos aspectos dos sistemas elétricos. Os autores destacam a necessidade do desenvolvimento de uma base comum orientada a objetos para suportar uma grande variedade de aplicativos para SEE de forma integrada, porém salientam a dificuldade de se chegar a uma estrutura ótima.

Bradley et alii [22] utilizam modelos baseado em objetos para o desenvolvimento de um aplicativo de simulação de faltas em SEE. Os autores utilizam um modelo semelhante ao conceito de elementos físicos e conceituais de [5] com sucessíveis níveis de agregação para representar a estrutura física e conceitual do

---

SEE. Assim, dispositivos são agregados em módulos, módulos são agregados em subestações, e estas, por sua vez, reúnem-se para compor o SEE. Paralelamente a esta estrutura, elementos de manobra são agrupados em barras, barras são agregadas em ilhas elétricas, e estas compõem o SEE. Infelizmente, Bradley et alii não utilizam uma notação formal para descrever o modelo orientado a objetos adotado.

Pandit et alii [27][28] propõem um plataforma de classes para aplicações em SEE. Os autores focalizaram sua plataforma para problemas de regime permanente (fluxo de potência, curto-circuito, processamento de topologia, fluxo de potência ótimo, análise de observabilidade e estimação de estados), argumentando que domínios de aplicações distintos requerem ferramentas de modelagem matemática distintas. A estrutura é hierarquizada com base em três classes principais: *Apparatus*, *Substation* e *Graph*. Um processador de topologia é utilizado para formar a rede elétrica (classe *Network*) que será posteriormente utilizada pelos aplicativos. Os trabalhos utilizam uma estrutura genérica para gerenciar matrizes esparsas e sistemas lineares de grande porte, que é apresentada em detalhes em [30].

Araujo e Pereira [25] desenvolveram uma estrutura orientada a objetos para representar matrizes esparsas e sistemas lineares de grande porte, obtendo excelentes resultados em relação ao desempenho computacional para matrizes com dimensão na ordem de 53000.

Agostini et alii em [33][34] propõem um base orientada a objetos para aplicações gerais em SEE (*framework*). Os autores utilizam uma estrutura hierárquica baseada nos conceitos de elementos estruturais e elementos de composição. Os elementos estruturais representam os dispositivos físicos do sistema, são derivados de uma classe base denominada *C\_Structural* e dividem-se em três grandes grupos: *C\_Shunt*, *C\_Branch* e *C\_Bar*. Os elementos de composição são derivados de uma classe denominada *C\_Composition* e representam o conjunto de equipamentos que compõem um elemento físico (para uma unidade de geração, por exemplo, os elementos de composição compreendem a máquina síncrona e seu conjunto de controladores). Neste trabalho é proposta também uma estratégia para troca de funcionalidade entre os elementos estruturais em tempo de execução através de um *Padrão de Projeto* denominado *Adapter* [4]. Os aplicativos aparecem como um conjunto hierárquico de classes associadas ao pacote (macroentidade) que representa o SEE. Em um trabalho posterior [35], o modelo base proposto por Agostini et alii é utilizado para o desenvolvimento de um segundo conjunto de aplicativos que incluem

---

fluxo de potência, avaliação da segurança dinâmica e seleção/classificação de contingências críticas, comprovando as facilidades propiciadas pelo modelo orientado a objetos para a incorporação de novas metodologias de análise de sistemas de energia elétrica.

Araujo et alii [32] propõem uma metodologia baseada em objetos para análise de sistemas de distribuição trifásicos desbalanceados. Os autores utilizam uma estrutura hierárquica, derivada de um elemento base denominado *CComponente*, dividida em três grandes grupos: *CElemento*, *CBarra* e *CChave*. A classe *CElemento* dá origem ainda a duas outras classes denominadas *CPassivo*, para representar elementos que possuem sempre o mesmo comportamento, e *CAtivo*, para representar elementos cujos parâmetros podem variar em função do estado do sistema. Um processador de topologia é utilizado para formar a rede elétrica do SEE, gerando uma estrutura denominada *CRede*, onde então são conectados os aplicativos.

Dentre as estruturas orientadas a objetos propostas para representar SEE, sem dúvida alguma, a mais sofisticada é a proposta pelo EPRI (*Electric Power Research Institute*) para integração de *softwares* em centros de operação e controle de SEE [29]. O CIM (*Common Information Model*) é um *framework* de classes proposto para padronizar o fluxo de informações em um centro de controle, sua estrutura permite representar uma ampla faixa de componentes de um SEE. O CIM padroniza nomes de classes, seus atributos e relacionamentos criando uma base de dados comum que facilita a integração entre aplicativos nos centros de controle. Apesar de impor um padrão a base CIM ainda permanece aberta para aceitar definições próprias. A estrutura proposta no CIM pode servir como base para a construção de bancos de dados únicos para o SEE, relacionais ou orientados a objetos.

### **3.2 Levantamento dos Requisitos do Modelo Orientado a Objetos**

A definição de uma estrutura computacional genérica para propósitos gerais em SEE (*framework*) deve ser sólida, inequívoca e expansível. Além disto, deve fornecer os meios para que os mais diversos aplicativos possam ser acomodados sob sua estrutura, provendo mecanismos que permitam a troca de informações e resultados entre estes aplicativos.

---

Em sua maioria os trabalhos anteriores propuseram modelos centrados na solução de um problema específico [5][9][10][11][12][14][15][18][19][20][22]. Ainda que estes trabalhos tenham sido pioneiros e contribuíram para difundir a MOO na área, seus modelos foram direcionados para o aplicativo utilizado como exemplo (fluxo de potência normalmente). A escolha de um aplicativo específico para conduzir o desenvolvimento pode levar a modelos que tornarão inviáveis a sua reutilização em outras aplicações. Esta abordagem pode resultar em um forte acoplamento entre o MOO adotado e aspectos específicos do aplicativo alvo, uma vez que detalhes da solução do problema são normalmente adicionados aos objetos desenvolvidos.

O sucesso de um MOO com o propósito de prover uma base computacional sólida para SEE reside na identificação das principais características e funcionalidades de um SEE, e no agrupamento destas em funcionalidades gerais, comuns a todos os aplicativos, e funcionalidades específicas, direcionadas a um determinado aplicativo. O MOO deve então implementar uma estrutura base fixa, composta das características mais gerais do SEE, e prover mecanismos para que características específicas possam ser adicionadas, utilizadas e removidas com segurança e consistência conforme o aplicativo em uso.

Com base no exposto acima, o modelo orientado a objetos adotado neste trabalho delimita muito claramente três aspectos fundamentais do problema:

- **Descrição Topológica:** representa a estrutura base do MOO, descrevendo aspectos topológicos e estruturais da rede elétrica. A descrição topológica define o arranjo estrutural de um SEE (áreas, subestações, etc), seus dispositivos e equipamentos componentes (geradores, cargas, LTs, etc) bem como os relacionamentos e conexões entre estes componentes. O modelo estrutural base deve descrever o sistema como ele é fisicamente, por esta razão é fixo (muda somente por ocasião da instalação de novos equipamentos) e independente de qualquer aplicativo;
- **Funcionalidades Específicas:** permite que características e funcionalidades específicas de uma determinada aplicação sejam incluídas no MOO (um gerador, por exemplo, possui dados e funcionalidades diferentes para o fluxo de potência e para a simulação dinâmica). As características específicas devem ser adicionadas ao MOO, executadas durante o tempo de vida da aplicação, e então removidas de forma segura e consistente;

- 
- **Ferramentas Matemáticas:** em aplicações voltadas a SEE é necessário ainda dispor de ferramentas matemáticas para suporte e apoio (operações matriciais, solução de sistemas lineares, etc). Embora estas ferramentas conceitualmente não pertençam ao domínio do SEE, sua utilização é fundamental para a formulação e solução dos aplicativos.

Estas três características devem implementar interfaces de acoplamento bem definidas, dando consistência ao MOO e provendo a base e os recursos para a construção de qualquer aplicativo. A Figura 11 mostra como as abordagens do problema se complementam definindo a base computacional para a implementação de qualquer aplicativo em SEE. Nesta estrutura, o conjunto de *Funcionalidades Específicas* adicionadas ao modelo deve ser adequada ao tipo de aplicativo sendo implementado.

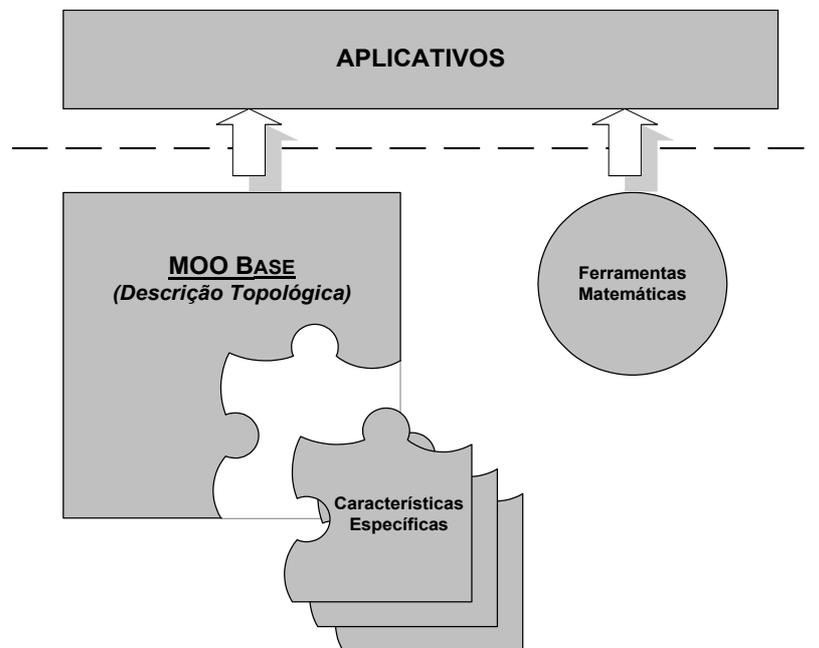


Figura 11 – Relação entre as abordagens do MOO

A filosofia da MOO recomenda ainda que os objetos sejam direcionados para representar conceitos do mundo real, mantendo o modelo computacional suficientemente próximo da estrutura real do problema. Para isso é necessário definir classes que representem fielmente os conceitos do SEE tal qual sua estrutura física real, abstraindo completamente qualquer aplicativo alvo. Isto se justifica pelo fato de que os objetos (geradores, cargas, subestações, LTs, etc) existem no SEE e se relacionam entre si independentemente de qualquer tipo de aplicativo. Assumindo que

---

o MOO implementado é uma cópia bastante próxima do SEE real, é razoável esperar que as aplicações se acomodem naturalmente sobre o modelo computacional.

Uma discussão mais detalhada sobre cada um dos aspectos, bem como sua implementação computacional, é apresentada a seguir.

### 3.3 Descrição Topológica do SEE

A primeira e mais fundamental funcionalidade do MOO é descrever a estrutura topológica geral do SEE. Todas as estruturas, dispositivos, arranjos e interconexões do SEE devem ser adequadamente descritos e gerenciados através desta funcionalidade. Em termos de sua descrição estrutural e topológica um SEE pode ser visto de duas formas distintas:

- **Descrição Física (barramento-disjuntor):** descreve o SEE como ele é fisicamente. Todos os equipamentos e dispositivos das subestações (geradores, disjuntores, chaves seccionadoras, cargas, etc.) são representados e descritos em seu arranjo topológico. As subestações são agrupadas em Áreas e/ou Sub-Áreas e interligadas através das Linhas de Transmissão, definindo assim o arranjo estrutural do SEE.
- **Descrição Lógica (barra-injeção):** fornece a descrição topológica resultante do processo de Configuração da Rede Elétrica [36], sendo esta direcionada especialmente para os aplicativos. Nesta descrição os dispositivos lógicos (seccionadoras, disjuntores, etc.) não são representados. O SEE é reduzido a *Barras Elétricas* e dispositivos efetivamente conectados a estas barras, determinando a configuração operativa atual do SEE.

A seguir serão apresentados detalhes dos modelos computacionais adotados para cada uma das descrições.

#### 3.3.1 Descrição Física

Um SEE de grande porte é, normalmente, subdividido em Áreas ou Regiões de interesse. As Áreas podem representar uma região geográfica do SEE ou uma região de abrangência de uma empresa de energia elétrica, e delimitam um conjunto de Subestações e Linhas de Transmissão. A Figura 12 mostra um exemplo real do sistema elétrico do Rio Grande do Sul, onde observa-se a grande área RS e três áreas

internas (Norte, Centro-Oeste e Sul), cada área com seu conjunto de subestações e linhas de transmissão. O detalhe na figura traz o diagrama unifilar de uma das subestações deste sistema.

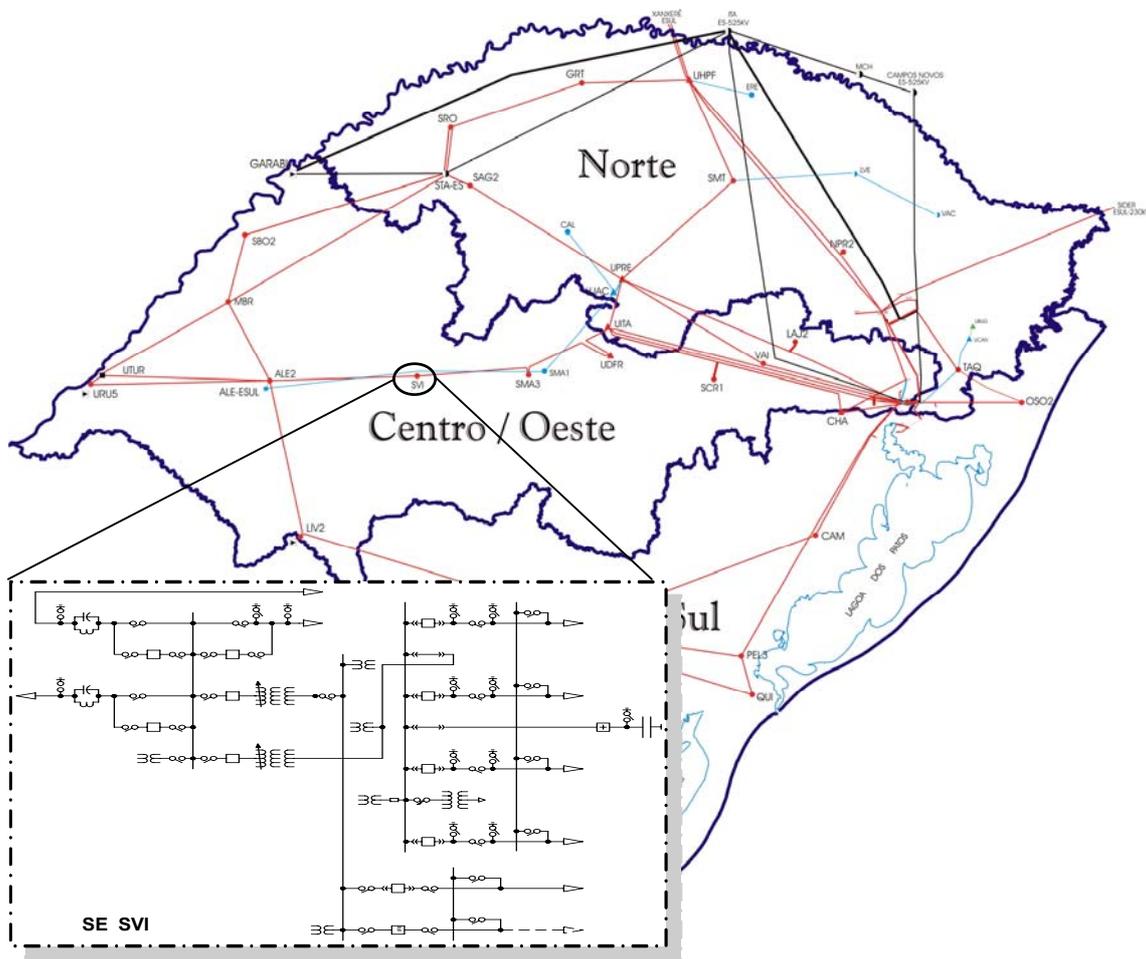


Figura 12 – Arranjo Topológico de um SEE

O exemplo acima mostra que um SEE pode ser descrito em dois níveis de profundidade. O primeiro contendo as áreas, subestações e linhas de transmissão, e o segundo contendo os equipamentos e dispositivos internos de cada subestação (um terceiro nível poderia ainda ser proposto, contendo a rede elétrica dos alimentadores de distribuição presentes na subestação – cargas no contexto do sistema de transmissão –, entretanto esta abordagem não será tratada neste trabalho). As empresas de energia elétrica figuram nesta hierarquia com abrangência irrestrita sobre todos os níveis de profundidade representados, uma vez que possuem desde áreas inteiras (com todas as subestações e LTs desta área) até alguns equipamentos isolados em subestações que pertencem a outras empresas.

---

O modelo orientado a objetos adotado prioriza fundamentalmente a concordância com a estrutura do SEE real. Neste sentido, todas as classes implementadas bem como seus atributos e funcionalidades estão baseadas em conceitos e objetos reais de um SEE. O diagrama de classes adotado para representar o primeiro nível de profundidade do SEE é mostrado na Figura 13.

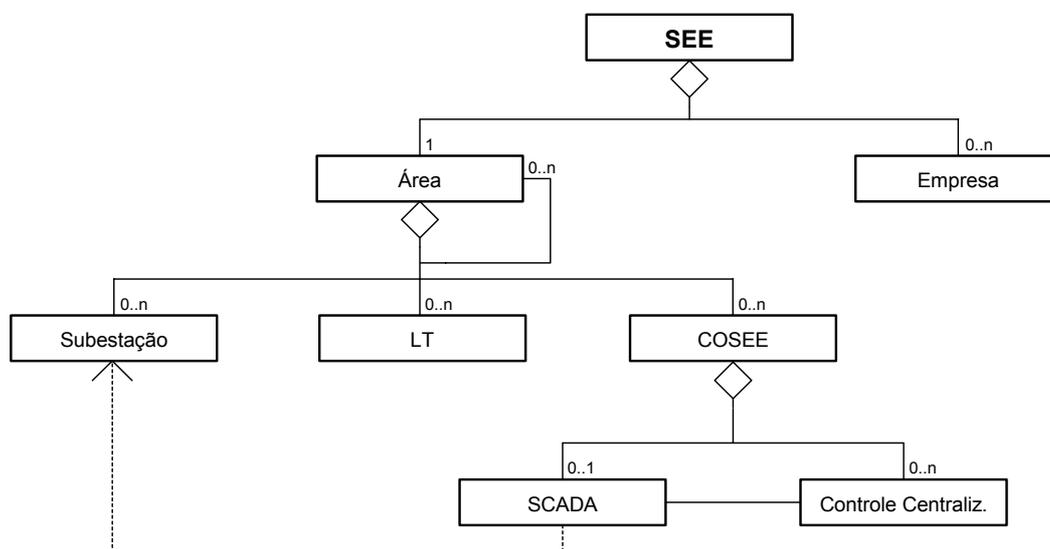


Figura 13 – Diagrama de Classes do SEE

A hierarquia de classes da Figura 13 utiliza uma classe global denominada *SEE* (*Sistema de Energia Elétrica*) para armazenamento e controle de todas as estruturas computacionais que representam o sistema. A classe *SEE* é composta de uma área base que pode conter subestações, linhas de transmissão e um eventual centro de operação do sistema (*COSEE*). A classe *Área* pode ainda abrigar em seu interior outras classes do mesmo tipo (sub-áreas), permitindo assim a hierarquização de diversos níveis de áreas e sub-áreas. O centro de operação do sistema é composto por um sistema de supervisão (*SCADA*) e esquemas de controle centralizado (como o CAG, por exemplo).

O segundo nível de profundidade é representado pela classe *Subestação* (representada genericamente na Figura 13), sendo o diagrama de classes adotado para representar sua estrutura interna mostrado na Figura 14.

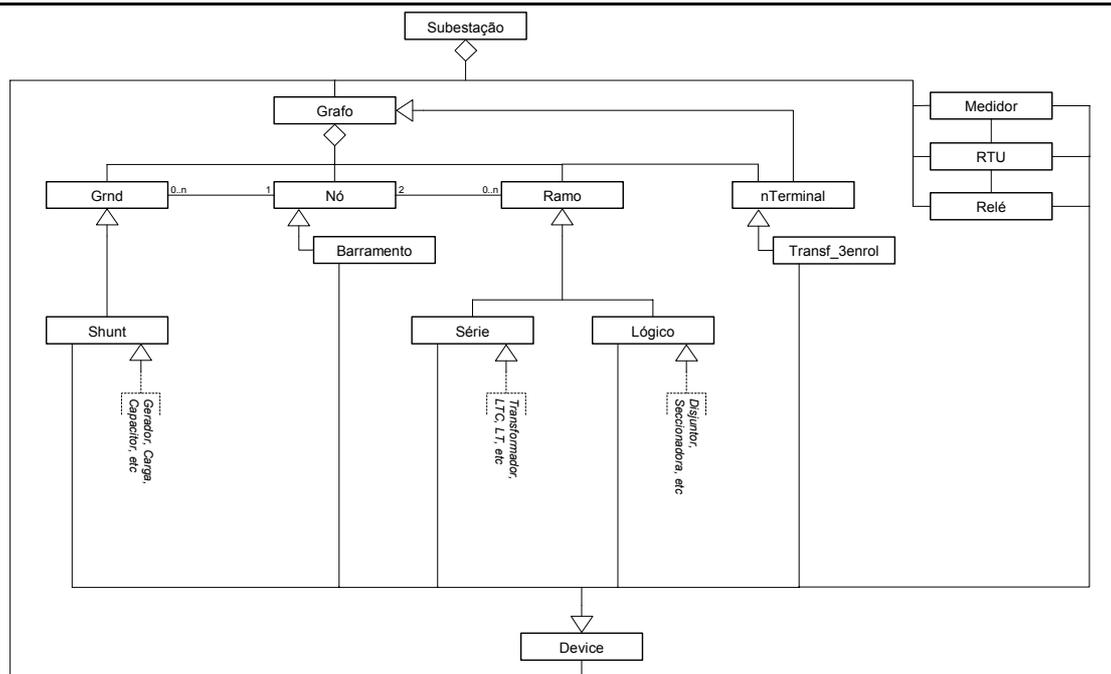


Figura 14 – Diagrama de Classes da Subestação

De forma análoga a uma subestação física real, a classe *Subestação* encerra em seu interior o conjunto de objetos que representam os equipamentos e dispositivos internos de uma subestação. Neste contexto, a classe *Subestação* possui duas funcionalidades principais: armazenar os objetos que representam seus equipamentos e dispositivos internos, e gerenciar a conectividade destes elementos.

A conectividade dos equipamentos de uma subestação (visualizada através do Diagrama Unifilar mostrado no detalhe da Figura 12) é implementada através de um conjunto de classes elementares (*Grnd*, *Ramo*, *nTerminal* e *Nó*) reunidas sob o domínio de uma classe denominada *Grafo* (ver Figura 14). Desta forma, as classes elementares delimitam um conjunto particular de funcionalidades genéricas que implementam e gerenciam os mecanismos responsáveis pela conectividade dos elementos.

A escolha das classes elementares resulta da classificação dos equipamentos de acordo com o seu número de terminais. Assim, abstraindo-se o tipo específico de dispositivo, os equipamentos podem ser classificados em elementos com um terminal (*Grnd*), elementos com dois terminais (*Ramo*) e elementos com três ou mais terminais (*nTerminal*). Cada classe elementar conecta-se a pontos genéricos de conexão (seus nós terminais) que são representados através de uma classe denominada *Nó*. Assim, cada elemento da subestação (*Grnd* ou *Ramo*) está necessariamente conectado a um

---

ou dois nós de conexão (*obs.:* a classe *nTerminal*, e seus descendentes, contêm internamente o conjunto dos seus nós terminais, uma vez que descendem da própria classe *Grafo*). A conectividade entre os elementos e os nós terminais é efetuada através de uma associação bidirecional entre as classes *Grnd-Nó* e *Ramo-Nó* (a Figura 14 mostra estas associações como linhas laterais entre as classes). Esta associação bidirecional facilita a navegabilidade sobre a estrutura topológica dos equipamentos da subestação, definindo um modelo computacional bastante próximo do modelo físico real.

Uma classe denominada *Barramento*, descendente da classe *Nó*, é utilizada para representar os barramentos físicos das subestações. Para isso possui atributos adicionais em relação aos nós, tais como: nome de identificação, nível de tensão, tipo (operação ou transferência), cor, bitola, etc. O conceito de nós e barramentos pode ser melhor entendido através do diagrama unifilar de uma subestação elementar tomada como exemplo e mostrada na Figura 15. Neste exemplo os nós são representados como pontos cheios “•” e os barramentos como barras cheias “|”.

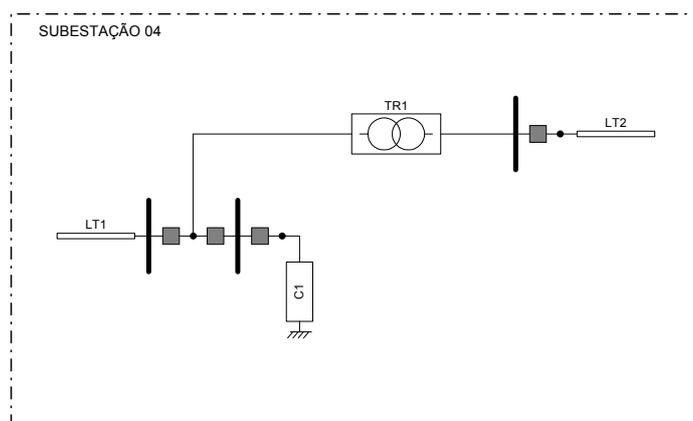


Figura 15 – Diagrama Unifilar de uma Subestação Hipotética

O MOO adotado neste trabalho implementa ainda uma classe especial denominada *LT* para representar as linhas de transmissão do SEE. Sendo um elemento de dois terminais a classe *LT* é descendente natural da classe *Ramo*. No entanto, devido a suas características particulares de conectividade, cada nó terminal pertence a uma subestação distinta, a classe *LT* não pertence ao grafo da subestação mas a área que a contém (ver Figura 13).

Os equipamentos contidos nas subestações possuem características gerais comuns, tais como um identificador do tipo específico de equipamento, seu nome e

---

número de identificação, o módulo a que pertence na subestação, sua empresa proprietária, etc. Estas características são agrupadas em uma classe genérica denominada *Device*, conforme pode ser visualizado na Figura 14. A superposição das características da classe *Device* com as características das classes elementares de conectividade do grafo, através de um mecanismo denominado herança múltipla [2], define um conjunto de classes que constituem a base para a implementação de qualquer equipamento específico do SEE (ver Figura 13 e Figura 14). As classes construtivas base são:

- **Shunt:** classe base para a definição dos equipamentos conectados a apenas um nó terminal (Geradores, Cargas, Capacitores, etc);
- **Série:** classe base para a definição dos equipamentos conectados a dois nós terminais que possuem impedância entre os terminais (Transformadores, LTCs, LTs, TCSCs, etc);
- **Lógico:** classe base para a definição dos equipamentos de manobra (Disjuntores, Seccionadoras, etc). Equipamentos de manobra são conectados a dois nós terminais porém com impedância nula entre os terminais;
- **Controle Centralizado:** classe base para a definição de esquemas de controle centralizado. Estes dispositivos não estão conectados a nós terminais mas associados às demais classes do SEE para a execução de alguma ação de controle de caráter centralizado (um esquema de CAG, por exemplo, importa informações de frequência e fluxo em LTs para atuar nos controles dos geradores).

Os sistemas de medição, proteção e supervisão de uma subestação são implementados através de um conjunto adicional de classes base especialmente projetadas para esta funcionalidade e denominadas *Medidor*, *Relé* e *RTU*, respectivamente. Assim, a classe *Medidor* implementa funções de monitoração de grandezas elétricas e lógicas de um ponto ou equipamento da subestação. A classe *Relé* implementa funções de monitoração e de atuação (sempre através de um dispositivo lógico), permitindo que um determinado equipamento seja desligado/ligado se alguma grandeza monitorada exceder o nível de ajuste do relé (aciona o disjuntor terminal de uma linha de transmissão se a corrente ultrapassar o valor de ajuste do relé, por exemplo). A classe *RTU* recebe/envia informações dos medidores e equipamentos locais para o sistema de supervisão (classe *SCADA*) e/ou diretamente para *RTUs* de outras subestações.

---

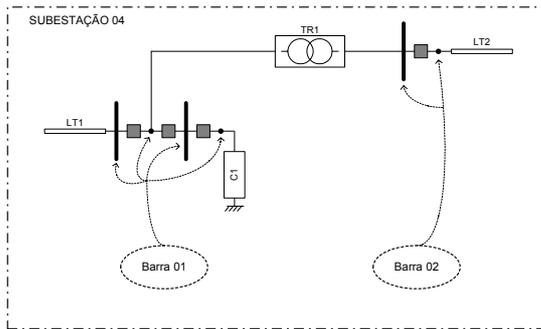
Dispositivos com três ou mais terminais, devido as suas características particulares, são tratados como casos especiais na estrutura proposta, e por esta razão não possuem uma classe base geral para sua implementação. Estes equipamentos são implementados adicionando características particulares do equipamento as características herdadas da classe *nTerminal* (conectividade) e *Device* (características gerais).

### 3.3.2 Descrição Lógica

A descrição física de um SEE normalmente permanece inalterada ao longo de um estudo, uma vez que esta somente é modificada quando novos dispositivos são adicionados ao sistema (entrada em operação de novos equipamentos, construção de subestações e/ou LTs, etc). Por outro lado, um mesmo SEE pode assumir inúmeras configurações de operação distintas, uma vez que manobras nos dispositivos lógicos do sistema (disjuntores, seccionadoras, etc) conectam e/ou desconectam equipamentos, alterando a estrutura da rede elétrica efetivamente ativa deste SEE. Considere o diagrama unifilar da subestação mostrada no exemplo da Figura 15 e reproduzida na Figura 16a. O estado dos disjuntores desta subestação (ABERTO/FECHADO) determinam a configuração operativa da mesma, ou seja, quais elementos estão efetivamente conectados a rede e em que configuração.

Um elemento lógico fechado ou um conjunto de elementos lógicos fechados e conexos determinam um ponto elétrico comum no SEE, composto pelos nós terminais dos dispositivos lógicos. Este ponto comum é aqui denominado *Barra Elétrica*, ou simplesmente *Barra*, e pode ser definido como a agregação de todos os nós conectados eletricamente através de um ou mais dispositivos lógicos fechados (ver *Barra 01* na Figura 16a). Um dispositivo não-lógico (Gerador, Carga, LT, etc) conectado a qualquer nó do ponto elétrico comum está também conectado a barra que define este ponto comum, formando assim uma nova rede elétrica onde os dispositivos lógicos podem ser suprimidos (ver Figura 16b). Esta rede constitui a rede elétrica efetivamente ativa do SEE, sendo denominada neste trabalho de *Descrição Lógica*. O conceito de Descrição Lógica pode ser melhor entendido através da Figura 16a e Figura 16b, onde é mostrado, respectivamente, as descrições física e lógica da subestação exemplo, bem como as *Barras Elétricas* formadas assumindo-se todos os disjuntores fechados.

a) Descrição Física



b) Descrição Lógica

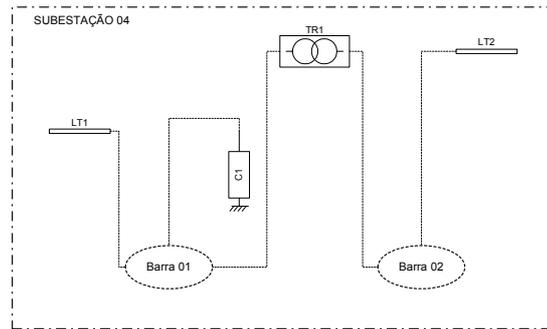


Figura 16 – Processo de Configuração da Subestação

O diagrama de classes adotado para representar a descrição lógica de um SEE é mostrado na Figura 17.

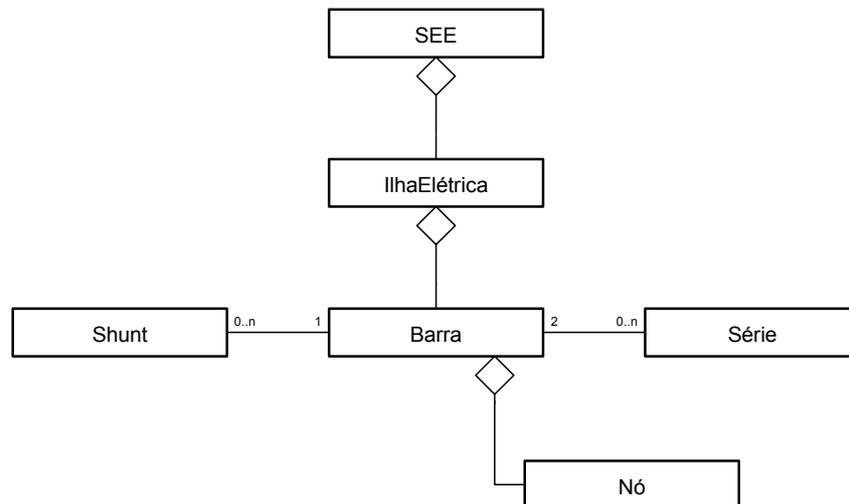


Figura 17 – Diagrama de Classes das Ilhas Elétricas

O diagrama de classes da Figura 17 mostra a introdução de uma classe especial denominada *Barra* para representar as *Barras Elétricas* do SEE, sendo esta representada por uma relação do tipo agregação de um ou mais nós. Para os propósitos da descrição lógica é necessário que apenas os dispositivos descendentes das classes base *Shunt* e *Série* possuam conexão as barras, uma vez que os dispositivos lógicos são suprimidos desta descrição (a classe *nTerminal*, e seus descendentes, contém internamente o seu conjunto de barras).

Um conjunto de barras conexas (através de elementos série) delimitam uma *Ilha Elétrica* no SEE, ou seja uma rede elétrica conexa e seu conjunto de elementos ativos. Este arranjo pode ser visualizado no diagrama de classes da Figura 17 através de

---

uma relação do tipo agregação entre as classes *Ilha Elétrica* e *Barra*. Um sistema normalmente apresenta apenas uma Ilha Elétrica, entretanto em situações de contingências ou durante manobras podem ocorrer desconexões de partes do sistema resultando em “ilhamentos” no SEE, que é traduzido no modelo computacional como o surgimento de Ilhas Elétricas adicionais.

Sob a ótica dos aplicativos apenas a Descrição Lógica do SEE é de real interesse, uma vez que a formulação da maioria dos aplicativos está centrada nos elementos da rede elétrica efetivamente ativa do SEE, desconsiderando qualquer dispositivo lógico.

### 3.3.3 Coexistência das Descrições

Entre as características desejáveis do MOO desenvolvido está a unicidade, por esta razão é necessário que ambas descrições do SEE (física e lógica) coexistam de forma harmônica e consistente. Para isso, cada equipamento possui conectividade dupla, ou seja implementa mecanismos que permite conexão aos *Nós* (ponto físico da subestação, herdado das classes *Grnd* e *Ramo*), e mecanismos de conexão as *Barras* (ponto comum da descrição lógica, herdado das classes *Shunt* e *Série*). No entanto, em ambas descrições o dispositivo, ou objeto que o representa, é exatamente o mesmo.

A característica de conectividade dupla define dois caminhos de acesso a um determinado dispositivo do SEE, o primeiro através da sua Descrição Física (*SEE* → *Área* → *Subestação* → *Equipamento*) e o segundo através da sua Descrição Lógica (*SEE* → *IlhaElétrica* → *Barra* → *Equipamento*). A Figura 18 ilustra a coexistência das duas descrições como planos de agrupamentos de elementos (representado computacionalmente através de uma relação de agregação entre as classes). No plano mais baixo figuram os dispositivos que compõem o SEE (LTs, cargas, transformadores, etc) e seus nós de conexão. Cada novo plano agrupa os elementos do plano inferior (indicado pelas linhas tracejadas entre os planos), sendo que os planos superiores podem ser divididos em dois semiplanos correspondentes as duas descrições possíveis. Por fim, no nível superior, um plano que representa o SEE agrupa ambas as descrições. A ilustração da Figura 15 mostra que os dispositivos que compõem o SEE (plano inferior) são compartilhados pelas duas descrições de forma harmônica e inequívoca. O MOO adotado pode então ser entendido como uma descrição computacional genérica com duas descrições topológicas coexistentes.

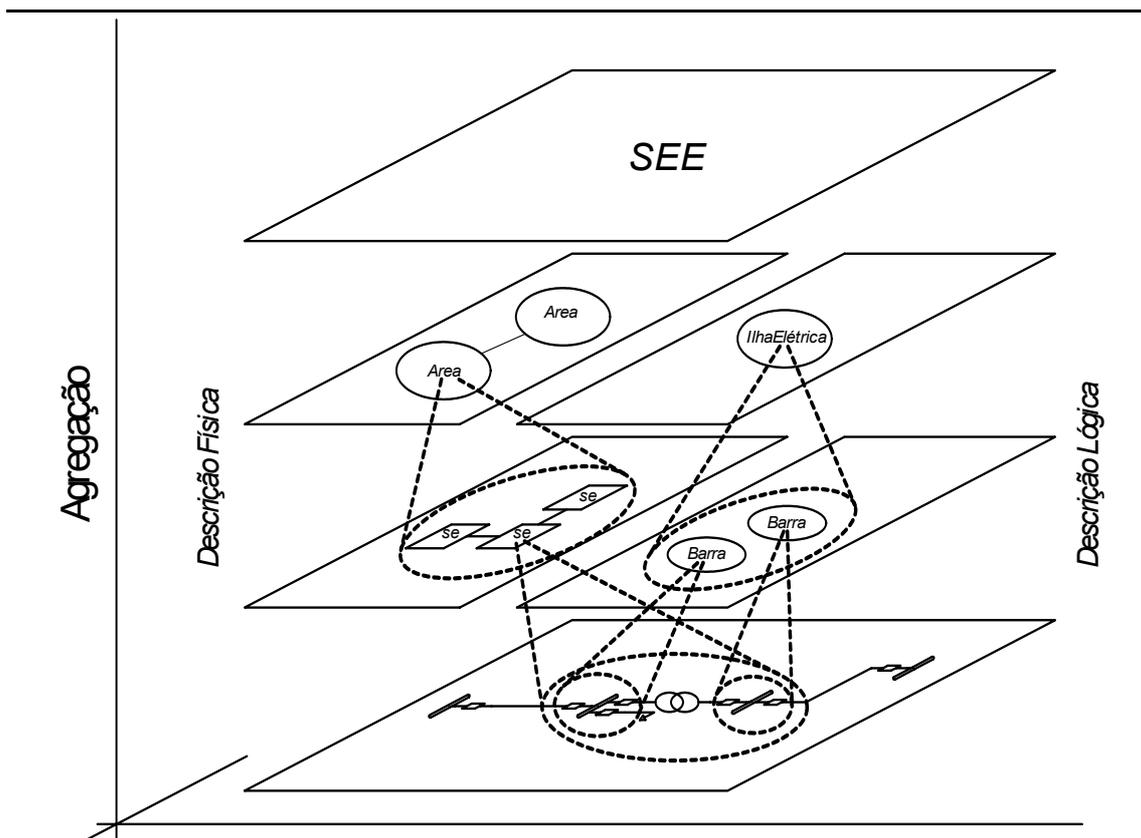


Figura 18 – Estrutura Organizacional do MOO

A utilização das duas descrições topológicas simultâneas adotadas neste trabalho deve-se a diretiva inicial de projetar um MOO para propósitos gerais em SEE, desta forma o MOO torna possível contemplar também aplicações que requeiram a descrição do SEE em nível de diagrama unifilar detalhado das subestações (como por exemplo um aplicativo para simulação do restabelecimento do SEE). Aliado a isso, um MOO projetado sem considerar a representação detalhada dificultaria a sua expansão futura para contemplar esta potencialidade, uma vez que decisões já teriam sido tomadas sobre a estrutura projetada. Por outro lado, a descrição detalhada sempre pode ser simplificada para uma descrição bastante próxima da representação tradicional (*barra-injeção*), simplesmente não representando os dispositivos lógicos nas subestações e conectando os equipamentos diretamente aos barramentos.

A Figura 19 mostra o *Diagrama Geral de Classes* adotado neste trabalho, onde estão reunidos todos os diagramas de classes apresentados anteriormente em um diagrama único.

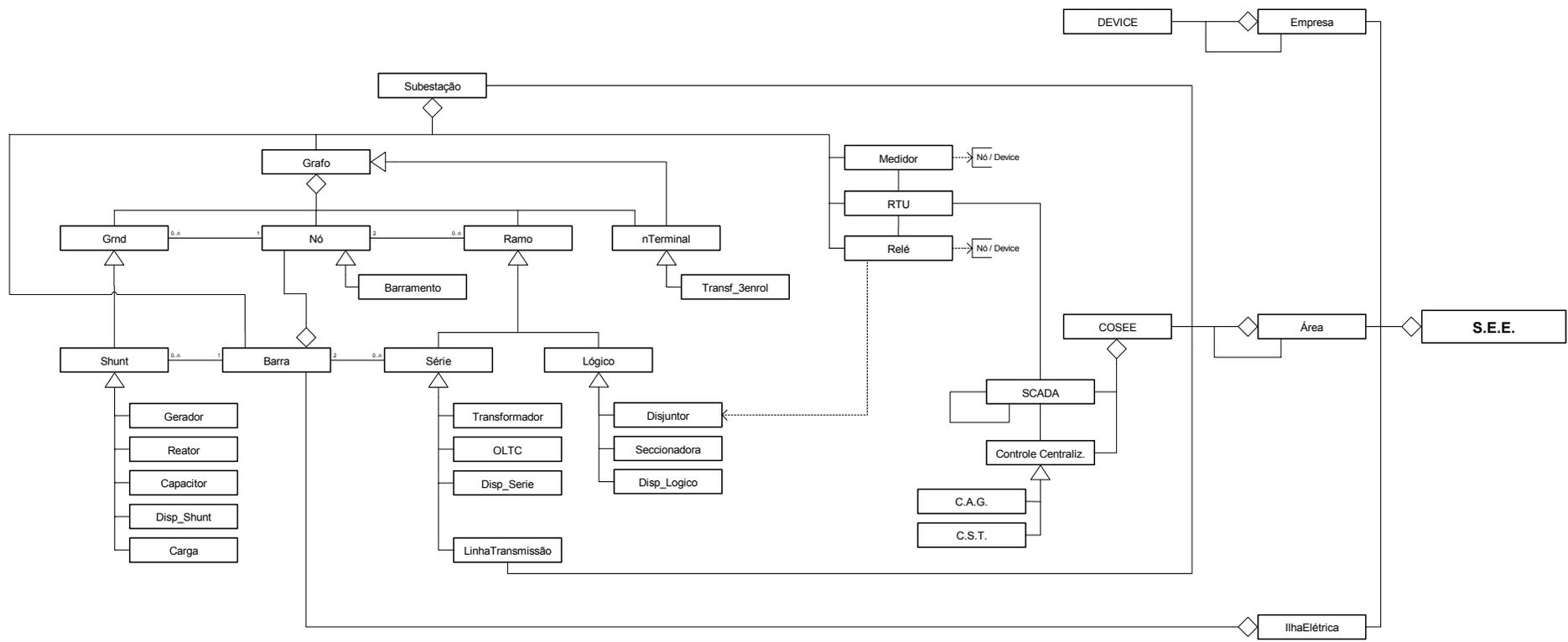


Figura 19 – Diagrama Geral de Classes

---

### 3.3.4 A Configuração da Rede Elétrica

A construção da Descrição Lógica do SEE é efetuada pelo Configurador de Redes [36]. Esta ferramenta trabalha integrada ao modelo computacional que descreve o SEE, atuando como uma funcionalidade de transcrição da Descrição Física do SEE para a sua Descrição Lógica correspondente. Uma vez detectada alguma alteração no estado de um dispositivo lógico do SEE, seja através de alguma ação externa ao MOO (comando explícito de chaveamento pelo usuário) ou pela ação de algum dispositivo de proteção (atuação de relé), o modelo imediatamente aciona a funcionalidade de reconfiguração da rede, mantendo assim sempre atualizada a Descrição Lógica do SEE.

O algoritmo adotado para a função de configuração da rede é uma variação da técnica de *Passeio pelos Arcos do Grafo* do SEE [36], adaptada para explorar eficientemente as potencialidades da POO. A escolha deste algoritmo deve-se principalmente as suas características de varredura única nos arcos do sistema, velocidade de execução e simplicidade de implementação. A configuração da rede elétrica é executada em duas etapas: configuração das subestações e configuração da rede elétrica global. Na primeira etapa, realiza-se a configuração das subestações onde os estados dos dispositivos lógicos são processados gerando Barras Elétricas e Ilhas Elétricas locais a cada subestação. Na etapa seguinte, realiza-se a etapa de configuração da rede elétrica global, onde percorre-se as linhas de transmissão das áreas do SEE reunindo as Ilhas Elétricas locais que estão conexas e gerando uma ou mais Ilhas Elétricas globais. Desta forma, o processo de reconfiguração (atualização da configuração) da rede pode ser otimizado, uma vez que a primeira etapa é realizada apenas nas subestações onde ocorreram chaveamentos nos dispositivos lógicos. Além disto, o mesmo algoritmo de configuração pode ser utilizado sem alterações significativas tanto na etapa de configuração das subestações como na etapa de configuração da rede elétrica global.

Características próprias da Programação Orientada a Objetos e da linguagem C++ introduzem algumas particularidades ao algoritmo de configuração de redes que o diferem da metodologia tradicional. São elas:

- O uso de listas encadeadas alocadas dinamicamente e de ponteiros evitam varreduras constantes nos vetores de dados para renumeração das seções de barras configuradas, otimizando a performance computacional desta metodologia;

- 
- A configuração é realizada em duas etapas somente: configuração das subestações e da rede elétrica global. A etapa de construção das tabelas equipamento-barras da metodologia tradicional (mapeam os equipamentos conectados as seções de barramento nas barras configuradas) pode ser eliminada do processo [36]. O mapeamento é definido automaticamente através da conectividade existente entre as barras e os objetos que representam os dispositivos (objetos contém internamente todos os dados e funcionalidades dos elementos que descrevem).
  - O configurador de redes não é um aplicativo pertencente à cadeia de funções tradicionais da operação em tempo real mas uma funcionalidade intrínseca do MOO que descreve o SEE, uma vez que é necessária e indispensável para manter as duas descrições coerentes.

### 3.4 Funcionalidades Específicas

Diferentemente da descrição topológica, completamente independente dos aplicativos, as funcionalidades dos dispositivos são altamente dependentes do aplicativo em uso. De fato, o comportamento e os dados de um dispositivo podem mudar completamente de um aplicativo para outro. Um gerador, por exemplo, possui dados e comportamento para o Fluxo de Potência completamente diferente dos dados e comportamento que possui para a Análise da Estabilidade Transitória ou Cálculo de Curto-Circuito, no entanto conceitualmente o dispositivo *Gerador* ainda é o mesmo para todas as aplicações.

O MOO adotado implementa mecanismos que permitem que dados e funcionalidades específicas sejam adicionadas aos dispositivos conforme a aplicação e removidos quando não mais necessários. A estratégia adotada para alcançar este objetivo define cada dispositivo (derivado da classe *Device*) como sendo uma composição de duas estruturas especializadas: um *estado*, que determina sua condição de operação (normalmente uma injeção na rede elétrica) e está rigidamente acoplado ao dispositivo, e um *modelo*, que atualiza o *estado* deste dispositivo e pode ser alterado conforme as necessidades do aplicativo em uso. A classe genérica *Device* contém então apenas funções de caráter geral, todos os dados e funcionalidades específicas são deslocados para o *modelo*. De fato, deve-se evitar que características específicas de uma ou outra aplicação sejam adicionadas a esta classe. A Figura 20a

e Figura 20b mostram, respectivamente, a estrutura geral de um dispositivo genérico e o diagrama de classes que o representa.

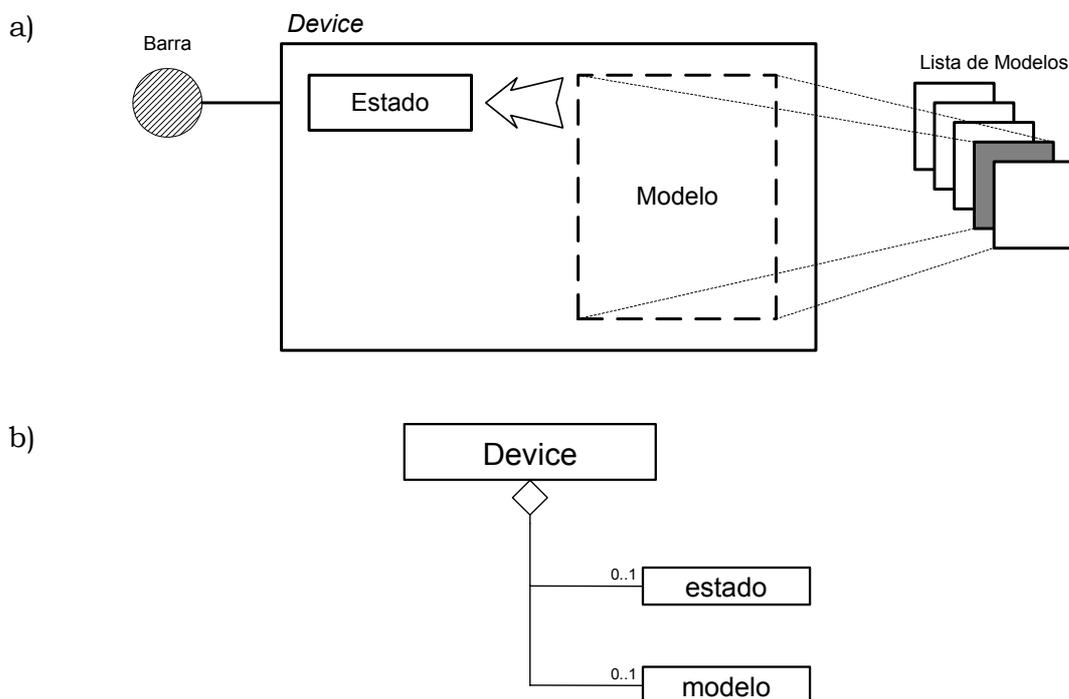


Figura 20 – Estrutura Geral de um Dispositivo Genérico

A ilustração da Figura 20a mostra que o *modelo* de um dispositivo pode ser livremente substituído alterando assim o comportamento deste dispositivo de acordo com as necessidades dos aplicativos (dados e funcionalidades específicas). Entretanto, o dispositivo agregador (descendente da classe *Device*) permanece fixo entre as aplicações, concordando com a premissa de que o elemento em si não muda entre aplicativos, mas apenas o seu comportamento (ou seu *modelo*). A classe *Device* deve então gerenciar adequadamente a substituição do seu *modelo*, quando necessário, e prover mecanismos para que as aplicações tenham livre acesso ao *estado* e ao *modelo*.

A classe base *Device* implementa ainda mecanismos que permitem aos seus descendentes selecionar qual característica estará habilitada para utilização (*estado e/ou modelo*), adicionando flexibilidade ao dispositivo genérico. Assim, equipamentos do tipo Proteção & Medição (classes *Medidor* e *Relé*) e o sistema de Supervisão (classes SCADA e RTU) possuem funcionalidades fixas para qualquer tipo de aplicativo, e por esta razão não precisam implementar mecanismos para troca de funcionalidades

---

nem armazenar estados, ou seja o estado e o modelo estão inibidos neste conjunto de classes. As classes descendentes de *Controle Centralizado* possuem o modelo habilitado e o estado inibido, uma vez que não necessitam armazenar estados. Os demais grupos descendentes da classe genérica *Device* (*Shunt*, *Série*, *Lógico*) possuem estado e modelo habilitados.

A decomposição de cada dispositivo em *estado* e *modelo*, evita que sejam necessárias alterações nos dispositivos para cada nova aplicação implementada, simplesmente é adicionado um novo modelo ao dispositivo com os dados e funcionalidades específicas para o aplicativo. De maneira geral, é possível dizer que, com a abordagem proposta, apenas os dados e as funcionalidades dedicadas do dispositivo são alterados conforme muda a aplicação, o elemento conceitual (classe agregadora) continua o mesmo. Além disso, em uma representação integrada (dispositivos representados através de uma única classe com dados e funcionalidades) os dispositivos tenderiam a tornarem-se classes com um amontoado de métodos e atributos, resultantes da implementação de vários aplicativos sobre a mesma estrutura.

A seguir serão apresentados detalhes dos modelos computacionais adotados para o estado e modelo dos dispositivos.

### **3.4.1 Estado de um Dispositivo**

A classe *estado* desempenha a função de reter no dispositivo o seu estado operativo mais atual (para um gerador, por exemplo, o estado é representado pela injeção de potência ou corrente na rede elétrica). Esta característica é particularmente necessária quando há troca de modelos ou aplicativos em tempo de execução. Assim, um novo modelo adicionado ao dispositivo pode atualizar suas variáveis internas de forma a concordar com o último estado deste dispositivo, e, de maneira semelhante, a solução de um aplicativo pode ser utilizada como ponto de partida para um outro aplicativo qualquer. Por estas razões, a classe *estado* não pode ser substituída em tempo de execução, permanecendo rigidamente acoplada ao dispositivo durante o tempo de vida deste.

Cada grupo de dispositivos possui um conjunto de variáveis que definem o seu estado particular, com exceção dos *Controles Centralizados* que não possuem *estado*. Para os dispositivos *Shunt* e *Série* o estado é composto pela injeção complexa nas

barras terminais do dispositivo ( $S$  ou  $I$ , de acordo com o modelo) e uma parcela dependente da tensão, representada por admitâncias em derivação e/ou em série entre as barras, conforme mostra a Figura 21. Uma estrutura mais simples é definida para os dispositivos *Lógico*, contendo apenas a sua situação operativa (ABERTO-FECHADO). A classe *Barra* não pode ser considerada um dispositivo do SEE, entretanto esta define o estado da rede elétrica, representado pelas tensões complexas e frequência ao longo de toda a rede do SEE, por este motivo possui um estado especial composto apenas pela sua tensão complexa nodal.

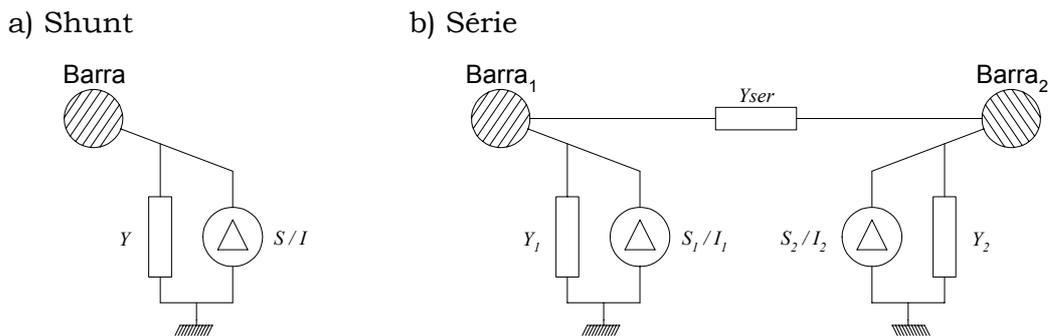


Figura 21 – Estado dos Dispositivos

Cada termo componente do estado de um dispositivo pode ser livremente acessado pelo modelo deste dispositivo ou pelo aplicativo (operação de leitura), entretanto apenas o modelo tem permissão para modificar o estado dos dispositivos (operação de escrita). Assim a ação do modelo pode ser entendida como uma operação particular do dispositivo que é exteriorizada para o sistema pelo estado deste dispositivo. De forma semelhante, o estado das barras pode ser acessado pelos dispositivos e aplicativos, porém apenas os aplicativos tem permissão de escrita, uma vez que as ações dos aplicativos tem caráter sistêmico e não localizado como no caso dos dispositivos. Um modelo pode utilizar ou não todos os termos que compõem o estado, assim uma Linha de Transmissão, por exemplo, representada pelo seu modelo  $\pi$  atualizará os termos  $Y_{ser}$ ,  $Y_1$  e  $Y_2$  do estado, mantendo as injeções  $S_1$ - $S_2$  ou  $I_1$ - $I_2$  nulas (ver Figura 21b).

### 3.4.2 Modelo de um Dispositivo

A classe *modelo* contém as características específicas de um determinado dispositivo, sempre voltadas para um aplicativo ou conjunto de aplicativos. O modelo encerra então os dados, equações e ações individuais dos dispositivos para um

---

determinado aplicativo. No entanto, nas aplicações mais usuais de SEE, é possível identificar um conjunto de funcionalidades para os dispositivos (ou para os modelos) que são comuns a uma grande diversidade de aplicativos. As funcionalidades são:

- **Armazenamento de Dados e Equações:** os dados, parâmetros, variáveis e equações que definem um modelo devem ser armazenados e gerenciados no interior da classe *modelo*. É conveniente salientar que os dados e equações de um dispositivo podem variar de acordo com o aplicativo;
- **Determinação das Condições Iniciais:** valendo-se do estado do dispositivo e do conjunto de equações que definem o modelo, este deve ser capaz de inicializar suas variáveis e parâmetros internos de forma a reproduzir o mesmo estado anterior, porém agora para outros dados e com outras funcionalidades;
- **Solução e Derivação das Equações:** esta talvez seja a funcionalidade mais importante de um modelo, uma vez que é responsável por ações como resolver o conjunto de equações do modelo, atualizar o estado do dispositivo e calcular derivadas parciais das equações que definem o modelo em relação ao conjunto de variáveis de estado;
- **Definição de Funcionalidades Específicas:** aplicativos que requerem ações particulares de um determinado dispositivo podem ainda introduzir tais ações em classes derivadas da classe modelo (através do mecanismo da herança).

A identificação do conjunto de funcionalidades citadas acima, permite definir uma interface de utilização comum para estas funcionalidades, de tal forma que todos os aplicativos reconheçam e saibam como tratar estas funções, independente do modelo específico que está associado ao dispositivo. Isto permite que um dispositivo conserve o mesmo modelo para vários aplicativos, desde que o modelo seja adequado ao aplicativo (o modelo  $\pi$  para uma linha de transmissão, por exemplo, pode manter-se o mesmo para uma grande quantidade de aplicativos). Além disso, a padronização da interface de acesso às funcionalidades comuns generaliza os aplicativos que utilizam esta interface, uma vez que tal aplicativo sabe como tratar o modelo sem conhecer necessariamente sua estrutura interna. Por esta razão, os aplicativos passam a tratar novos modelos automaticamente, sem a necessidade de alterações no código do aplicativo. Assim, um programa de fluxo de potência e um programa de simulação dinâmica completa, por exemplo, incorporam automaticamente qualquer

---

novo modelo adicionado ao SEE sem qualquer alteração no código do programa (um novo equipamento FATCS ou um novo modelo de regulador de tensão).

Para alcançar o grau de generalidade descrito acima, o modelo deve armazenar e gerenciar eficientemente o conjunto de parâmetros, variáveis e equações que definem o comportamento do dispositivo. Uma estrutura computacional especialmente projetada para este fim foi implementada e será descrita a seguir.

### 3.4.2.1 Estrutura Computacional do Modelo de um Dispositivo

Uma equação matemática qualquer pode ser representada através de um conjunto de blocos elementares organizados na forma de um *Diagrama de Blocos*, onde cada bloco elementar representa uma operação matemática singular. A Figura 22 mostra uma equação exemplo e sua correspondente representação em diagrama de blocos.

$$f(x,y) = 2.x + \text{sen} ( 3.y + x ) - y$$

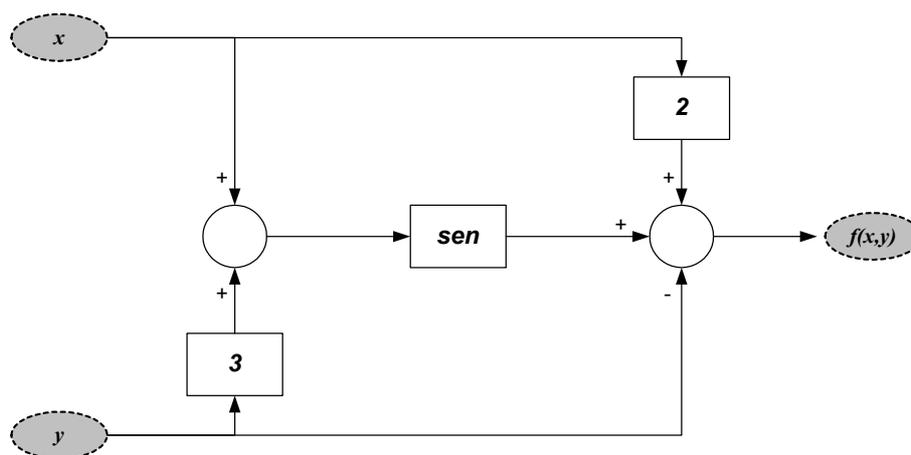


Figura 22 – Diagrama de Blocos Representativo de uma Equação

A decomposição de uma equação em seu correspondente diagrama de blocos permite definir uma estrutura orientada a objetos especial onde blocos construtivos elementares (objetos no contexto da modelagem orientada a objetos) são interligados para formar o conjunto de equações que definem o modelo de um dispositivo. Esta forma de representar as equações do modelo adiciona flexibilidade à estrutura computacional, uma vez que permite ao modelo conhecer e manipular a estrutura da equação armazenada. Além disso, os parâmetros e equações do modelo podem ser construídos em tempo de execução do programa, permitindo aos usuários definirem

---

seus próprios dispositivos (geradores, linhas de transmissão, CAG, etc.) em um conceito denominado **Dispositivo Definido pelo Usuário**.

O elemento central da estrutura orientada a objetos implementada para descrever equações é um bloco genérico (classe *Blc*) sem operação matemática específica associada, e que será utilizado como base para a construção de todos os demais blocos elementares (através do mecanismo da herança). Abstraindo-se o tipo específico de operação matemática associada ao bloco, pode-se definir um conjunto de funcionalidades que serão comuns a todos os blocos elementares que definem um modelo, destas a mais fundamental trata da conectividade entre os blocos, uma vez que esta funcionalidade dita a estrutura da equação armazenada. Para este fim, o bloco base gerencia um conjunto de classes especiais auxiliares (denominadas variáveis) responsáveis pela interconectividade dos blocos. Cada bloco pode possuir “*n*” variáveis de entrada, denominadas *VarInp* ( $u_1 \dots u_n$ ), e “*m*” variáveis de saída, denominadas *VarOut* ( $y_1 \dots y_m$ ), sendo o número de entradas e saídas de cada bloco dependente da operação matemática que este executa (1:1 para um bloco ganho, n:1 para um bloco somador, etc). A variável de saída (*VarOut*) armazena o valor numérico resultante da operação matemática elementar, podendo conectar-se a uma ou mais variáveis de entrada de outros blocos. As variáveis de entrada (*VarInp*), por sua vez, podem conectar-se a somente uma variável de saída de outro bloco, evitando assim ambigüidades (1 entrada associada a diversas saídas). A Figura 23 mostra a estrutura geral do bloco base e suas variáveis de entrada/saída, bem como as conexões possíveis entre as variáveis.

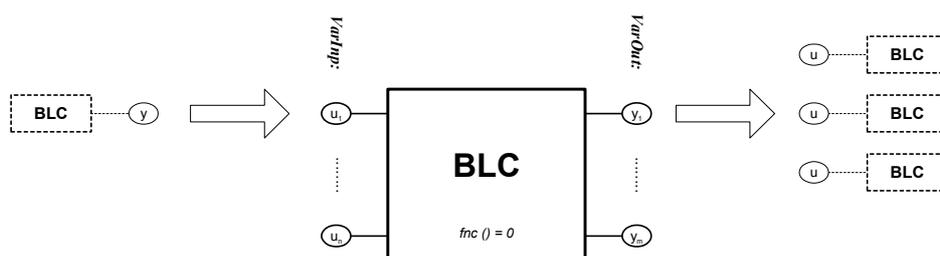


Figura 23 – Estrutura de Conexão de um Bloco Construtivo Elementar

Cada bloco derivado do bloco base implementa uma operação matemática específica (operação virtual no bloco base) que relaciona uma saída a uma ou mais entradas do bloco. Por exemplo, um bloco ganho (1 entrada “*u*” e 1 saída “*y*”) executa uma função do tipo  $y=K*u$ , onde *K* define o ganho do bloco.

O modelo de um determinado dispositivo é geralmente composto de diversos blocos e parâmetros que definem seu conjunto de equações, além disso o modelo deve permitir ainda a inclusão de sub-modelos em sua estrutura. O diagrama de classes que descreve a estrutura computacional do modelo é mostrado na Figura 24.

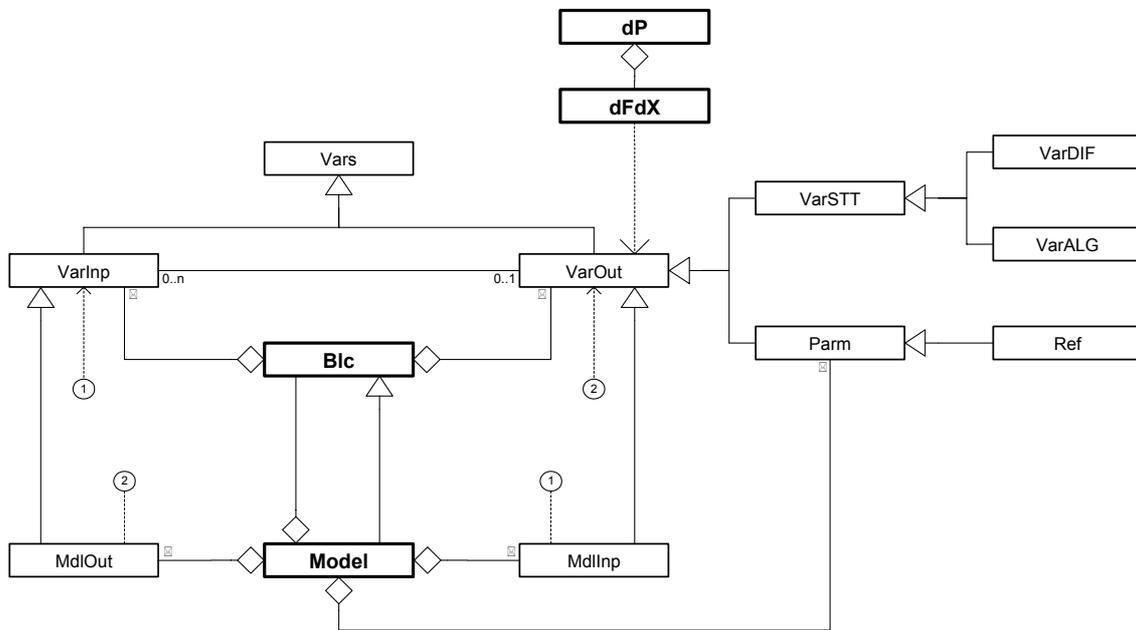


Figura 24 – Diagrama de Classes Geral para o Modelo de um Dispositivo

O diagrama de classes da Figura 24 mostra o relacionamento do bloco base (classe *Blc*) com o seu conjunto de variáveis de entrada/saída (classes *VarInp* e *VarOut*, respectivamente) através de uma relação de agregação, e a conectividade entre as variáveis de entrada/saída através de uma relação de associação. Dois grupos de classes derivadas de *VarOut* constituem o conjunto de parâmetros (classes *Parm* e *Ref*) e variáveis de estado do modelo (classes *VarSTT*, *VarDIF* e *VarALG*). Parâmetros e referências são um tipo especial de variáveis de saída sem bloco associado, ou seja, um parâmetro pode ser entendido como uma variável de saída com valor numérico fixo e que pode ser conectado às entradas dos demais blocos do modelo. Variáveis de estado, por sua vez, representam um conjunto especial de variáveis de saída de determinados blocos (integradores, por exemplo) e definem os estados internos de um modelo. Assim, qualquer bloco de natureza dinâmica (integrador, washout, etc) necessariamente possuirá variáveis de saída do tipo diferencial (classe *VarDIF*). De forma semelhante, um bloco algébrico (somador,

ganho, seno, etc) pode ter sua saída promovida ao “*status*” de variável algébrica (classe *VarALG*). Esta promoção deve ser especificada na construção do modelo.

O modelo de um dispositivo (classe *Model*) é constituído de um conjunto de blocos matemáticos elementares e de um conjunto de parâmetros, em um arranjo estrutural denominado *Composite*<sup>1</sup> [4] que lhe permite hierarquizar a estrutura final em vários níveis e sub-níveis (o modelo de um gerador, por exemplo, pode ser constituído de vários sub-modelos representando a máquina síncrona, seu conjunto de controladores, a turbina, etc). A classe *Model* implementa ainda variáveis de entrada/saída adicionais *MdlOut* e *MdlInp*, derivadas de *VarInp* e *VarOut*, que são responsáveis pelo “*by-pass*” das conexões do modelo para os seus blocos internos.

A Figura 25 mostra um *modelo* construído para representar uma unidade de geração, onde três sub-modelos representam os componentes internos do gerador (Máq. Síncrona, Reg. de Tensão e Reg. de Velocidade). Para simplificar o desenho apenas o sub-modelo do Regulador de Tensão é mostrado em detalhes.

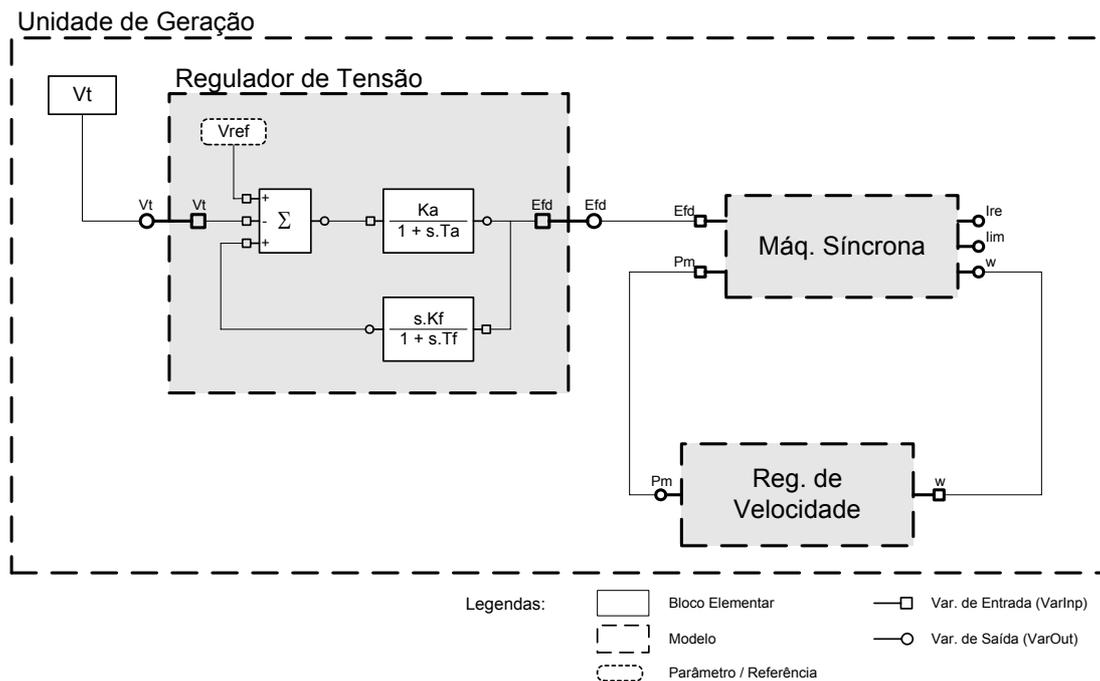


Figura 25 – Modelo de uma Unidade de Geração

<sup>1</sup> *Composite* é um Padrão Estrutural de Projeto que permite representar objetos em estrutura de árvore para representar hierarquias todo-parte. A relação de herança e agregação entre as classes *Blc* e *Model* permite tratar de maneira uniforme objetos individuais e composição de objetos [4].

---

A construção de modelos, usando a estrutura computacional proposta, não precisa necessariamente obedecer ao arranjo estrutural clássico do equipamento (para um gerador: Máq. Síncrona + Reg. de Tensão + Reg. de Velocidade). Assim qualquer arranjo estrutural ou novo equipamento (existente ou que venha a existir) pode ser introduzido no modelo sem nenhum esforço de implementação computacional.

### **3.4.2.2 Mecanismo de Solução e Derivação das Equações**

A estrutura de armazenamento das equações nos modelos permite definir um mecanismo automático de solução e cálculo das derivadas parciais das equações, conferindo aos aplicativos um alto grau de generalização, uma vez que não necessitam conhecer previamente o tipo de dispositivo conectado as barras do sistema para executar tais ações.

Para o cálculo das derivadas fez-se necessário implementar um conjunto de classes específicas (classes  $dP$  e  $dFdX$ ) para armazenar e gerenciar as derivadas parciais das equações do modelo em relação ao seu conjunto de variáveis de estado (classes  $VarDIF$  e  $VarALG$ ). O diagrama de classes da Figura 24 mostra o relacionamento entre as classes  $dP$  e  $dFdX$ , e com as demais classes que definem o modelo. Nesta estrutura, a classe  $dFdX$  armazena a derivada da equação em relação a uma única variável de estado, sendo constituída de um coeficiente numérico e uma associação a variável de saída correspondente ao estado ( $VarOut$ ). A classe  $dP$ , por sua vez, representa o conjunto de derivadas parciais de uma determinada equação em relação a todas as variáveis de estado que a equação depende, sendo implementada como um conjunto de classes  $dFdX$  (relação de agregação). Utilizando esta estrutura é possível padronizar o tratamento das derivadas de tal forma que os aplicativos não precisam conhecer o tipo específico de dispositivo que gerou aquele conjunto de derivadas parciais. Ou seja, os aplicativos dispõem de uma estrutura genérica de tratamento de derivadas parciais que é independente do dispositivo. A seguir será apresentado detalhes do algoritmo de cálculo das derivadas parciais [60][61] através de um exemplo para facilitar o entendimento do mecanismo.

Tomando como exemplo a equação mostrada na Figura 22, aqui reapresentada na Figura 26, e atribuindo as variáveis  $x$  e  $y$  os valores 0.1 e 0.3, respectivamente, a equação exemplo e suas derivadas parciais em relação as variáveis de interesse assumem os valores apresentados abaixo:

$$f(x,y) = 2.x + \text{sen}(3.y + x) - y$$

$$f(x,y) = 0.74$$

$$\bullet \partial f(x,y) / \partial x = 2 + \cos(x + 3.y)$$

 $\Rightarrow$ 

$$\bullet \partial f(x,y) / \partial x = 2.54$$

$$\bullet \partial f(x,y) / \partial y = 3. \cos(x + 3.y) - 1$$

 $(x = 0.1; y = 0.3)$ 

$$\bullet \partial f(x,y) / \partial y = 0.62$$

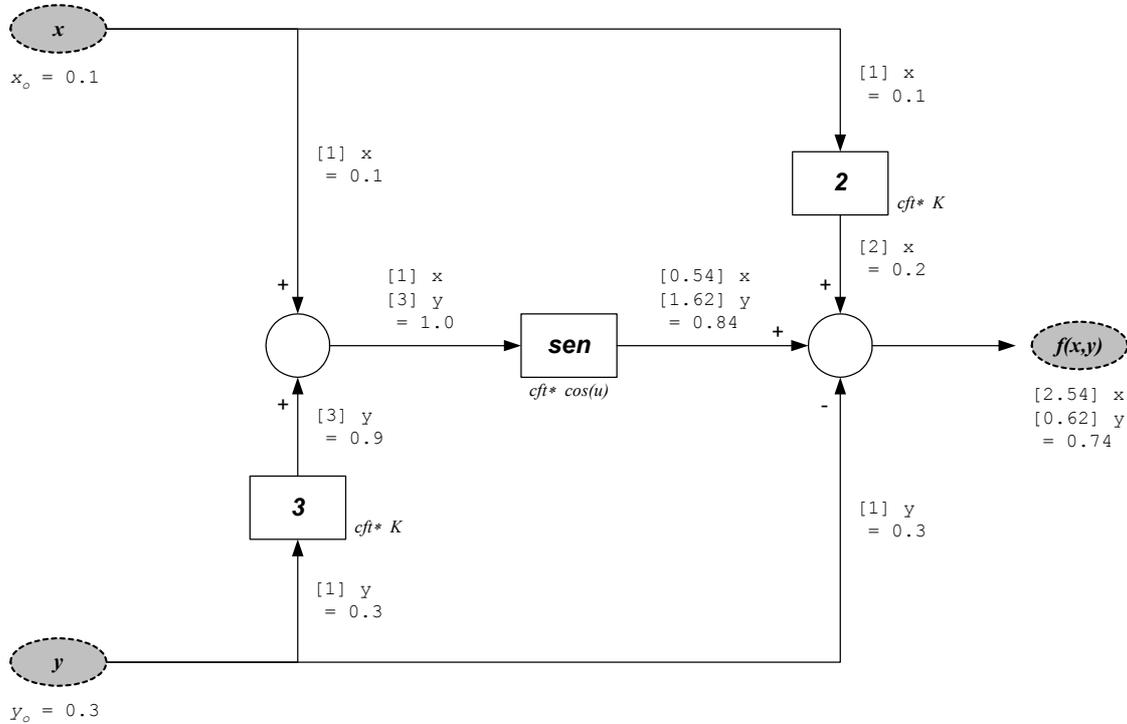


Figura 26 – Mecanismo de Solução e Derivação do Modelo

O mecanismo de solução e derivação das equações parte da variável de saída que define a equação que se deseja obter a solução e/ou o conjunto de derivadas parciais (ponto  $f(x,y)$  na Figura 26). A partir deste ponto percorre-se o caminho seguindo a orientação inversa dos blocos (saída  $\rightarrow$  entrada) até que uma variável de estado seja encontrada ( $VarDIF$  ou  $VarALG$ ), no exemplo da Figura 26 todos os caminhos levam as variáveis  $x$  e  $y$ . A escolha do caminho inverso é adequada devido a existência de apenas uma opção possível para cada variável de entrada dos blocos, uma vez que cada entrada é associada a apenas uma saída de outro bloco. Aliado a isto, é possível implementar um algoritmo de busca recursiva que marca naturalmente o caminho percorrido. As variáveis de estado encontradas nesta busca determinam o conjunto de termos das derivadas parciais que serão calculados para a equação.

Após a identificação do conjunto de variáveis de estado que a equação é dependente percorre-se o caminho direto, a partir das variáveis de estado (o caminho

direto já foi devidamente marcado pela busca recursiva, de tal forma que não a necessidade de buscá-lo), atualizando, a cada passagem por um bloco matemático, a estrutura especial projetada para armazenamento das derivadas parciais (estrutura composta por instâncias da classe  $dFdX$ ). A Figura 26 mostra os valores armazenados na estrutura auxiliar para todos os pontos do diagrama de blocos no caminho direto percorrido, sendo:

$$\begin{array}{l}
 [coeficiente] \text{ variável de estado} \rightarrow dFdX \\
 [coeficiente] \text{ variável de estado} \rightarrow dFdX \\
 = \text{valor parcial da equação}
 \end{array}
 \left. \vphantom{\begin{array}{l} \\ \\ \\ \end{array}} \right\} dP$$

Cada vez que a classe  $dP$  passa por um bloco do diagrama uma operação particular é executada sobre esta estrutura. A operação depende do tipo de operação matemática executada no bloco, porém uma lei de formação geral pode ser formulada. A lei de formação é aplicada a todos os termos  $dFdX$  de  $dP$  e obedece a seguinte regra:

$$[coeficiente]_{antigo} * [d f_{blc} / du]_{u = u_0} \rightarrow \text{novo coeficiente associado a variável de estado}$$

onde o novo coeficiente de  $dFdX$  é obtido multiplicando-se o antigo coeficiente pela derivada da operação matemática implementada no bloco em relação a entrada. Tomando com o exemplo o bloco **sen** do diagrama de blocos da Figura 26 a lei de formação assume a seguinte forma:

$$[coeficiente]_{novo} = [coeficiente]_{antigo} * [cos(u)]_{u = u_0}$$

ou seja, para a variável de estado  $y$  teremos:

$$\begin{array}{ccc}
 [3] \ y & \boxed{\text{sen}} & [ [3] * cos(1.0) = 1.62 ] \ y \\
 = 1.0 & & = sen(1.0) = 0.84
 \end{array}$$

O conjunto de coeficientes (termos  $dFdX$ ) obtidos no ponto que define a equação são as derivadas parciais em relação ao conjunto de variáveis de estado identificadas pelo algoritmo, conforme mostram os resultados da Figura 26 concordando com os obtidos de forma analítica.

A detecção de um parâmetro, ou uma referência, durante o percurso inverso para identificação do conjunto de variáveis de estado também interrompe o processo de busca. Nesta situação é adotado o mesmo procedimento das variáveis de estado, porém a classe *dFdX* assume um coeficiente nulo que determina a eliminação desta derivada (isto concorda com o fato da derivada de uma função em relação a uma constante ser nula).

### 3.4.2.3 Relação dos Blocos Elementares para a Construção dos Modelos

A Figura 27 mostra um diagrama de classes com todos os blocos construtivos elementares que são derivados do bloco base. Estes blocos constituem a base para a construção de qualquer modelo de dispositivo no MOO que representa o SEE.

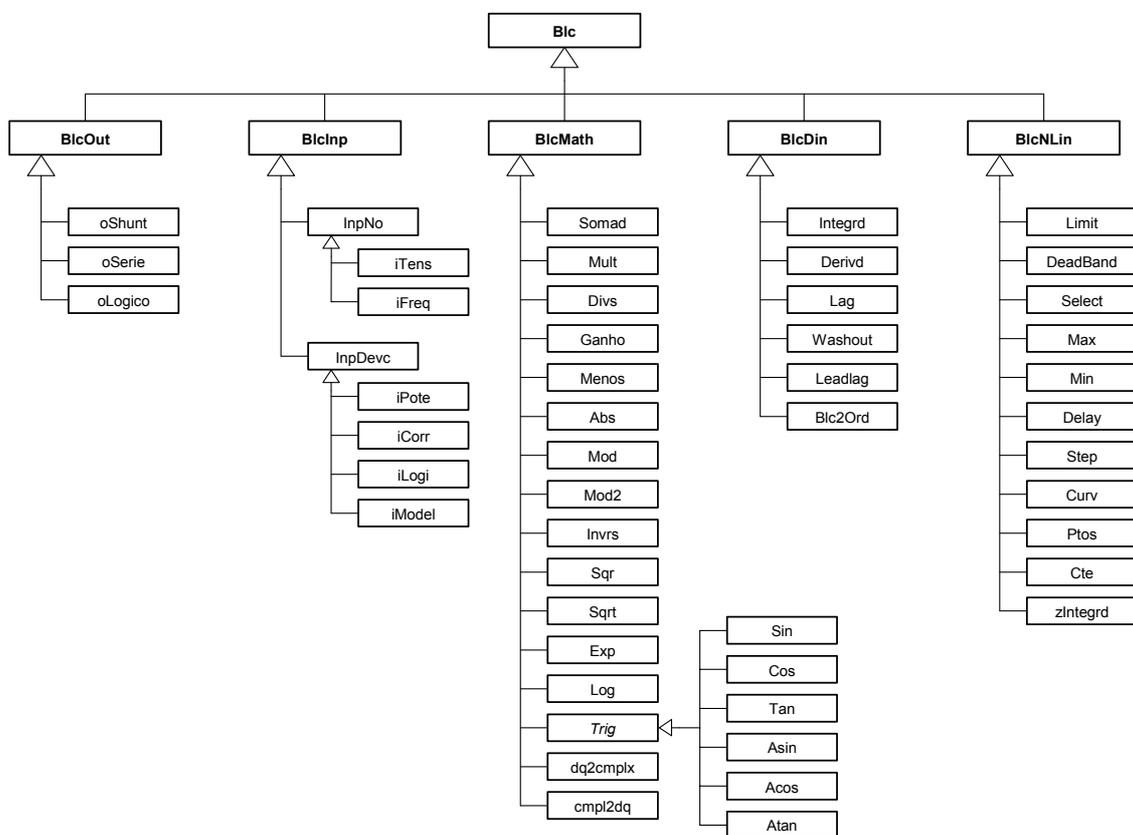


Figura 27 – Estrutura Geral dos Blocos Componentes do Modelo

Os blocos elementares estão organizados em grupos conforme características comuns. Os grupos são:

- 
- **Blocos de Saída (classe *BlcOut*):** compreendem os blocos que exteriorizam a ação do modelo para o SEE, através do estado do dispositivo. Existem três blocos básicos de saída, um para cada grupo de dispositivos que possui estado. Os blocos de saída são responsáveis pela alteração do estado dos dispositivos e pelo cálculo das derivadas parciais da injeção do dispositivo na rede elétrica (potência ou corrente de acordo com a solicitação do aplicativo). Cada modelo de dispositivo pode conter apenas um único bloco de saída, e este deve concordar com o tipo de dispositivo (por exemplo, modelos de dispositivos derivados da classe *Shunt* podem conter somente blocos do tipo *oShunt*).
  - **Blocos de Entrada (classe *BlcInp*):** blocos de entrada fazem a aquisição de variáveis do SEE, ou de outros modelos, para o interior de um determinado modelo. Qualquer tensão nodal ou frequência do SEE, injeção dos dispositivos na rede elétrica (potência e/ou corrente), estado operativo dos dispositivos lógicos, bem como variáveis de estado de outros dispositivos podem ser adquiridos pelos blocos de entrada e utilizados no modelo;
  - **Blocos Matemáticos (classe *BlcMath*):** blocos matemáticos realizam operações algébricas elementares, tais como soma, multiplicação, ganho, seno, cosseno, etc (ver Figura 27). Os blocos matemáticos, derivados da classe base *BlcMath*, não produzem variáveis de estado no modelo, porém, se desejado, podem ter suas saídas promovidas a variáveis de estado algébricas (*VarALG*), e desta forma passam a definir derivadas parciais (correntes de eixo direto e quadratura de geradores síncronos –  $I_d$  e  $I_q$  – são um bom exemplo do uso de variáveis de estado algébricas);
  - **Blocos Dinâmicos (classe *BlcDin*):** são o conjunto de blocos que possuem uma equação diferencial associada, produzindo obrigatoriamente uma variável de estado no modelo. Os blocos dinâmicos sempre impõem suas saídas como variáveis de estado do modelo;
  - **Blocos Não-Linearizáveis (classe *BlcNLin*):** representam um conjunto de blocos que não tem derivada definida, tais como limitadores, funções de máximo e mínimo, banda morta, etc. No entanto, estes blocos ainda tem o seu efeito considerado no processo de cálculo das derivadas parciais das equações do modelo.

---

### 3.4.2.4 Funções Especiais das Classes *BlcOut* e *dP/dFdX*

Os blocos de saída dos modelos (derivados de *BlcOut*) possuem características especiais que os diferem dos demais blocos. Tais blocos são os responsáveis pela conectividade dos modelos à rede elétrica do SEE, uma vez que atualizam o estado dos dispositivos (são os únicos que possuem permissão de escrita no estado dos dispositivos *Shunt*, *Série* e *Lógico*). Por esta razão, cada bloco de saída deve ser coerente com o estado do dispositivo associado, ou seja deve possuir tantas variáveis de entrada quanto forem os termos componentes do estado, permitindo assim que o modelo possa atualizar cada termo que compõe o estado individualmente. Tomando como exemplo os dispositivos *Shunts*, que possuem como estado uma injeção complexa e uma admitância em derivação (ver Figura 21a), o bloco de saída correspondente (*oShunt*) deve possuir 4 variáveis de entrada, uma para cada termo componente do estado. A Figura 28 mostra a estrutura de um bloco do tipo *oShunt*.

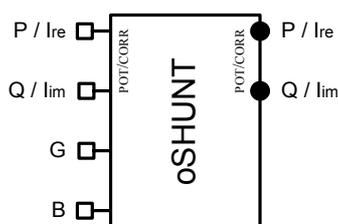


Figura 28 – Estrutura Geral de um Bloco de Saída do Tipo *oShunt*

A definição do tipo de injeção de entrada no bloco (POT:P/Q ou CORR: $I_{re}/I_{im}$ ) deve ser definida durante a construção do modelo, sendo coerente com as equações de injeção definidas no modelo. A injeção total do dispositivo na rede elétrica é então composta pela injeção de entrada no bloco mais uma parcela dependente da tensão (entradas G e B). De forma semelhante a injeção de entrada, também o tipo de injeção na rede elétrica pode ser definido em tempo de execução (POT:P/Q ou CORR: $I_{re}/I_{im}$ ), cabendo ao bloco *oShunt* a tarefa de conversão ( $S_{modelo} \rightarrow I_{rede}$  ou  $I_{modelo} \rightarrow S_{rede}$ ) se houver necessidade. Assim, um modelo pode ser formulado como injeção de corrente e escrever no estado do dispositivo associado tanto corrente como potência, ficando por conta dos blocos *BlcOut* as conversões que se fizerem necessárias. Esta característica permite que um mesmo modelo possa ser usado tanto para aplicativos que trabalham na formulação de balanço de potência como para aqueles que trabalham na formulação de balanço de corrente (em termos práticos, o mesmo modelo pode ser utilizado em um Fluxo de Potência – tradicionalmente formulado

---

como injeção de potência – e em um programa de Análise Modal – tradicionalmente formulado como injeção de corrente – sem que seja necessário acrescentar qualquer nova funcionalidade ao modelo). O mecanismo de conversão POT ↔ CORR também é válido para o cálculo das derivadas parciais, ou seja é possível construir um modelo que escreve corrente nas variáveis de entrada dos blocos *BlcOut* e solicitar derivadas parciais da injeção de potência deste dispositivo na rede elétrica.

Da mesma forma que um aplicativo pode ser formulado como injeção de potência ou corrente para os dispositivos, este também pode ser formulado com tensões em coordenadas polares ou retangulares. As classes de armazenamento das derivadas parciais  $dP$  e  $dFdX$  foram dotadas de funções que encarregam-se de fazer as devidas conversões entre coordenadas polares e retangulares para as derivadas das tensões. O tipo de coordenada desejado é um parâmetro de definição da classe  $dP$  e esta armazena as derivadas das tensões nas coordenadas desejadas. Assim, se definido coordenadas retangulares, por exemplo, e um bloco de entrada fornece o módulo de uma tensão qualquer, a classe  $dP$  automaticamente encarrega-se de efetuar a conversão das derivadas parciais para o tipo de coordenada desejado. Através dos mecanismos de conversão descritos qualquer modelo de dispositivo implementado pode fornecer derivadas da sua injeção de corrente ou potência na rede elétrica, e com coordenadas polares ou retangulares para as tensões.

Como já foi dito anteriormente, a utilização de uma estrutura geral para solução das equações e cálculo das derivadas parciais torna possível a generalização dos aplicativos quanto ao tipo específico de dispositivos instalados no SEE. Somando-se a isso, as novas funcionalidades de conversão dos blocos de saída *BlcOut* e das classes  $dP/dFdX$ , é possível a implementação de uma família de aplicativos que são generalizados quanto a sua formulação (Potência/Corrente – Polar/Retangular), e quanto ao tipo e modelo de dispositivos instalados no SEE. Algoritmos generalizados permitem ao usuário definir, em tempo de execução, que tipo de injeção na rede elétrica é mais adequada e/ou qual o tipo de coordenadas para as tensões deve ser utilizado, além, é claro, de incorporar automaticamente qualquer novo modelo adicionado ao SEE. Assim, por exemplo, se um aplicativo de Fluxo de Potência apresenta problemas de convergência com a formulação tradicional (injeção de potência / coordenadas polares), o usuário pode alterar a formulação do aplicativo para injeção de corrente e coordenadas retangulares, sem que um novo aplicativo tenha que ser implementado para isso.

---

- **Um Modelo de Carga Exemplo**

A seguir a estrutura computacional proposta neste trabalho será utilizada para a construção de um modelo de Carga com dependência da tensão terminal (modelo ZIP). A Figura 29 mostra o diagrama de blocos do modelo implementado, onde são mostrados os blocos componentes do modelo, as conexões entre as variáveis de entrada/saída dos blocos e os parâmetros. Um bloco de entrada do tipo *iTENS* faz a aquisição do módulo da tensão terminal do dispositivo, e um bloco de saída do tipo *oSHUNT* capta o valor das potências ativa e reativa calculadas no modelo para fins de atualização do estado do dispositivo na rede elétrica.

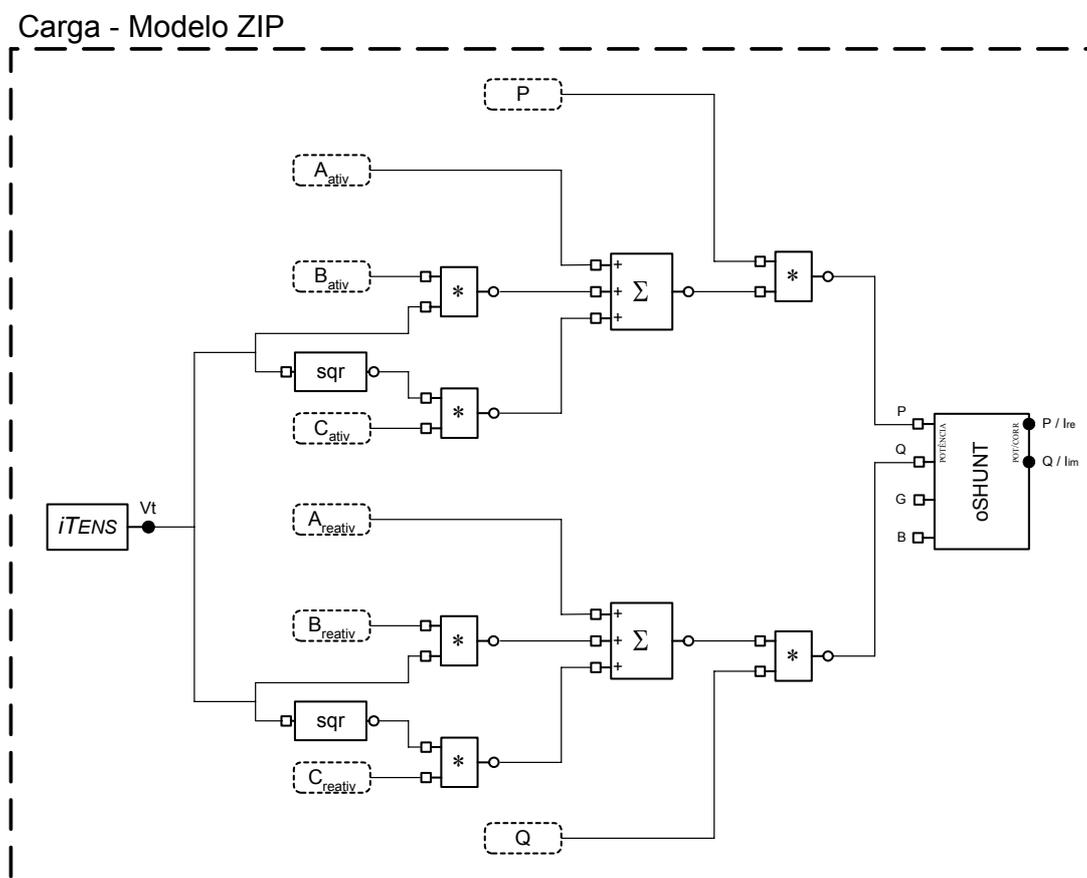


Figura 29 – Modelo de Carga ZIP

A estrutura computacional proposta permite que todos os modelos sejam definidos em tempo de execução, para isto foi elaborado uma linguagem própria para a construção dos modelos. Não é propósito deste trabalho apresentar detalhes da linguagem elaborada, porém um exemplo da sintaxe definida é apresentada a seguir para o modelo de carga tomado como exemplo .

---

<< Linguagem Utilizada para Definição do Modelo Carga ZIP >>

```
MODL:  CARGA#Mdl:ZIP      .
#
#      [--ID.--] REF VISIBL [--Valor--] T P
#  === Parametros ===
#  PARM:      P REF          $
#  PARM:      Q REF          $
#  PARM:      Aa             1.000 %
#  PARM:      Ba             0.000 %
#  PARM:      Ca             0.000 %
#  PARM:      Ar             1.000 %
#  PARM:      Br             0.000 %
#  PARM:      Cr             0.000 %
#
#  === INPUT ===
#  [--ID.--] [Vre/Vmd] [Vim/Van] [TIP OUT]
#  TENS:  INP001          Vt          POLAR
#
#  [--ID.--] [--INP--] [--OUT--] STT
#  === Blocos ===
#  SQR :  BLC001          Vt          V2
#
#  --- ( P ) ---
#  MULT:  BLC002          Ba          Ia
#  ....          V
#  DFIM
#  MULT:  BLC003          Ca          Za
#  ....          V2
#  DFIM
#  SOMD:  BLC004 +          Aa          %P
#  ....          +          Ia
#  ....          +          Za
#  DFIM
#  MULT:  BLC005          P          Po
#  ....          %P
#  DFIM
#
#  --- ( Q ) ---
#  MULT:  BLC006          Br          Ir
#  ....          V
#  DFIM
#  MULT:  BLC007          Cr          Zr
#  ....          V2
#  DFIM
#  SOMD:  BLC008 +          Ar          %Q
#  ....          +          Ir
#  ....          +          Zr
#  DFIM
#  MULT:  BLC009          Q          Qo
#  ....          %Q
#  DFIM
#
#  === OUTPUT ===
#  OSHT:  OUT001          POTENCIA POTENCIA
#  OUT :          Po          Qo
#  DFIM
#  DFIM
```

<< Final do Fragmento >>

Os mecanismos de conversão descritos no item 3.4.2.4 serão mostrados através de um exemplo numérico. O modelo de carga exemplo foi adicionado a uma carga conectada em um sistema hipotético qualquer. A tensão terminal da barra é de

1.0126 pu com ângulo de  $-3.6870^\circ$ , e o valor total da carga é 80 MW e 30 MVAR, sendo as parcelas do modelo ZIP definidas da seguinte forma: *parte ativa*  $\rightarrow$  40%  $P_{cte}$ , 0%  $I_{cte}$ , 60%  $Z_{cte}$ ; e *parte reativa*  $\rightarrow$  20%  $P_{cte}$ , 0%  $I_{cte}$ , 80%  $Z_{cte}$ . A Tabela 1 mostra os coeficientes das derivadas parciais calculadas para a carga do exemplo em todas as formulações possíveis (POTÊNCIA/CORRENTE – POLAR/RETANGULAR), onde cada retângulo do interior da tabela representa o coeficiente associado a uma variável de estado (colunas da tabela) para uma determinada equação de injeção na rede elétrica (linhas da tabela).

Tabela 1 - Derivadas do Modelo ZIP de Carga

		dP V (POLAR)		dP Vre (RETANG)	
		dP $\theta$ (POLAR)		dP Vim (RETANG)	
model->tipo (POTÊNCIA)	P	$\partial P / \partial V =$ 1.0777	$\partial P / \partial \theta =$ 0.0000	$\partial P / \partial V_{re} =$ 1.0754	$\partial P / \partial V_{im} =$ -0.0693
	Q	$\partial Q / \partial V =$ 0.4766	$\partial Q / \partial \theta =$ 0.0000	$\partial Q / \partial V_{re} =$ 0.4756	$\partial Q / \partial V_{im} =$ -0.0306
model->tipo (CORRENTE)	$I_{re}$	$\partial I_{re} / \partial V =$ 0.1743	$\partial I_{re} / \partial \theta =$ 0.3530	$\partial I_{re} / \partial V_{re} =$ 0.1964	$\partial I_{re} / \partial V_{im} =$ 0.3366
	$I_{im}$	$\partial I_{im} / \partial V =$ -0.1895	$\partial I_{im} / \partial \theta =$ 0.8682	$\partial I_{im} / \partial V_{re} =$ -0.1340	$\partial I_{im} / \partial V_{im} =$ 0.8678

### 3.5 Ferramentas Matemáticas

Uma base computacional geral para SEE deve contar ainda com um conjunto de ferramentas que disponibilizem o suporte matemático necessário aos aplicativos. Estas ferramentas devem ser responsáveis pelo armazenamento, gerenciamento e solução de sistemas de equações lineares esparsos e de grande porte.

Embora sejam imprescindíveis para a construção e solução da maioria dos aplicativos em SEE, conceitualmente, estas ferramentas não pertencem ao domínio do SEE. Por esta razão, as ferramentas matemáticas devem constituir uma estrutura completamente independente da estrutura utilizada para representar o SEE, constituindo-se assim em um *toolkit* autocontido que poderá ser utilizado em qualquer outra aplicação, não necessariamente na área de sistemas elétricos.

A estrutura computacional implementada é derivada do modelo proposto em [23][24] para armazenamento e fatoração matricial, porém expandida e generalizada

---

para sistemas lineares. Esta ferramenta geral foi denominada CAL<sup>++</sup> (*Classes para Álgebra Linear*) e seu diagrama de classes é apresentado na Figura 30.

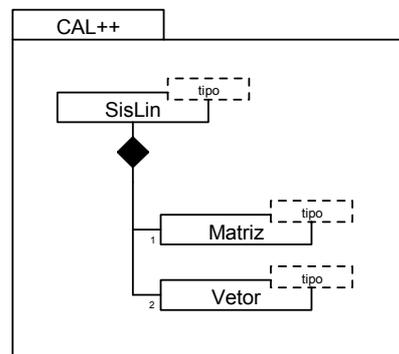


Figura 30 – Diagrama de Classes das Ferramentas Matemáticas

O diagrama da Figura 30 mostra três classes básicas abrigadas sob um pacote, que lhe confere a característica de independência do MOO utilizado para representar o SEE. Algumas características das classes componentes são apresentadas a seguir:

- **Matriz:** estrutura computacional utilizada para armazenamento e gerenciamento de matrizes esparsas de grande porte, simétricas ou não-simétricas. A classe *matriz* utiliza uma estrutura de armazenamento baseado em um conjunto de listas encadeadas alocadas dinamicamente, o que permite que seja alocado em memória apenas o espaço necessário ao seu armazenamento. Conforme metodologia proposta em [23], a fatoração LU elimina quase completamente buscas sobre as listas encadeadas durante o processo de fatoração (principal deficiência do armazenamento matricial através de listas encadeadas), otimizando assim ao máximo a performance computacional. A classe *matriz* conta ainda, até o presente momento, com operações elementares entre matrizes e entre matriz e vetor (\*, + e -), cálculo do determinante, inversa parcial, autovalores e autovetores, e decomposição em valores singulares.
- **Vetor:** a classe *vetor* é utilizada para o armazenamento e gerenciamento de vetores esparsos ou cheios. De forma análoga à classe *matriz*, a classe *vetor* utilizam uma estrutura de armazenamento baseada em listas encadeadas alocadas dinamicamente.
- **Sistema Linear:** a classe *SistLin* é constituída por uma instância da classe *matriz* (*A*) e duas instâncias da classe *vetor* (*x* e *b*), constituindo internamente a

---

expressão matricial  $A \cdot x = b$ . Uma vez atribuídos valores aos termos da matriz  $A$  e do vetor  $b$ , a classe *SistLin* é capaz de determinar a solução deste sistema de equações, disponibilizando-a no vetor  $x$ . O processo de solução é realizado através da fatoração LU da matriz  $A$  e posterior substituição direta-inversa entre os fatores e o vetor  $b$ . De forma análoga à fatoração LU, a metodologia utilizada para substituição direta-inversa elimina quase completamente buscas sobre as listas encadeadas, otimizando ao máximo a performance do processo de solução do sistema de equações lineares.

Convém salientar que sendo a matriz  $A$  da classe *SistLin* uma instância da classe *matriz*, todas as funcionalidades disponíveis para a classe *matriz*, ou que venham a ser implementadas no futuro, estarão disponíveis para a matriz de coeficientes do sistema linear, e desta forma disponíveis também para todos os aplicativos que utilizam o pacote CAL<sup>++</sup>. Isto permite que, por exemplo, uma vez implementado um aplicativo para Análise Modal de sistemas elétricos, o aplicativo de Fluxo de Potência pelo método de Newton pode avaliar os autovalores e/ou autovetores da sua matriz Jacobiana, ou o número de condicionamento desta matriz, sem nenhum esforço adicional de programação.

Todas as classes (*matriz*, *vetor* e *SistLin*) foram definidas como sendo do tipo *template*, onde o tipo básico dos elementos armazenados é passado como parâmetro para as classes. Isto permite que a classe *SistLin* possa implementar sistemas de equações lineares com coeficientes reais, complexos, sub-matrizes (2x2, 3x3, ...), bem como qualquer objeto válido que possua pelo menos as operações elementares definidas (\*, /, + e -).

### 3.6 Aplicativos

Em uma base computacional para propósitos gerais em SEE, tão importante quanto a entidade global que descreve o SEE (classe *SEE*) são os aplicativos que serão construídos sobre esta estrutura computacional, uma vez que são estes elementos os que efetivamente resolvem um problema específico de engenharia (Fluxo de Potência, Análise de Contingências, Cálculo de Sensibilidades, etc). No MOO adotado, os aplicativos encontram-se no mesmo nível hierárquico da classe *SEE*, sendo definida uma classe base para a construção destes aplicativos. Assim, todos os aplicativos construídos sobre o MOO adotado devem descender da classe

---

base, denominada *Aplicativos*. O uso de uma classe base impõe uma hierarquia obrigatória para os aplicativos e permite que alguns aspectos comuns possam ser definidos no nível mais básico, tais como ordenação ótima das barras, geração da matriz admitância de barras, etc. Estas características serão herdadas por todos os aplicativos descendentes da classe base.

A classe base *Aplicativos* possui uma associação lógica com a classe *SEE* e uma associação conceitual com o pacote  $CAL^{++}$  (no sentido de que não há mecanismos físicos conectando ambas classes). A Figura 31 mostra o diagrama de classes desta estrutura, onde observa-se a associação do pacote  $CAL^{++}$  apenas com a classe *Aplicativos*, concordando com a premissa de que as ferramentas matemáticas não pertencem ao contexto do SEE, porém pertencem ao contexto dos aplicativos..

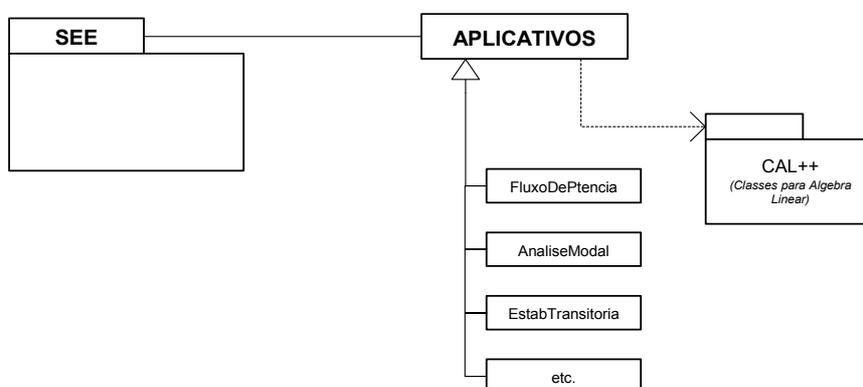


Figura 31 – Diagrama de Classes do SEE x APLICATIVOS

A estrutura geral proposta disponibiliza todos os mecanismos necessários a construção de um aplicativo, conforme mostra a Figura 32. Estes mecanismos são:

- **Descrição Topológica:** a classe *SEE* fornece todo o suporte referente à conectividade dos elementos do SEE bem como seu arranjo estrutural, permitindo a representação e o gerenciamento do SEE nos dois tipos de descrição topológica possíveis de forma integrada. Isto permite implementar tanto aplicativos que utilizam a descrição lógica do SEE (Fluxo de Potência, Análise Linear, etc) como àqueles que utilizam a descrição física (Restabelecimento de SEE, Coordenação da Proteção).
- **Dados e Funcionalidades Específicas:** suporte fornecido diretamente pelos modelos de cada dispositivo do SEE. Este mecanismo fornece os dados, as

funcionalidades comuns (inicialização, solução e derivação das equações do modelo) e funcionalidades específicas dos dispositivos que forem necessárias para a construção do aplicativo, sendo estas características acessadas através da classe *SEE*.

- **Ferramentas Matemáticas:** suporte fornecido pelo pacote *CAL++*. Todo o suporte matemático para o aplicativo é fornecido por este conjunto de classes, assim funções como armazenamento de matrizes e vetores esparsos e de grande porte, solução de sistemas lineares, cálculo de autovalores e autovetores, etc estão prontamente disponíveis no *toolkit* *CAL++* para utilização pelos aplicativos.

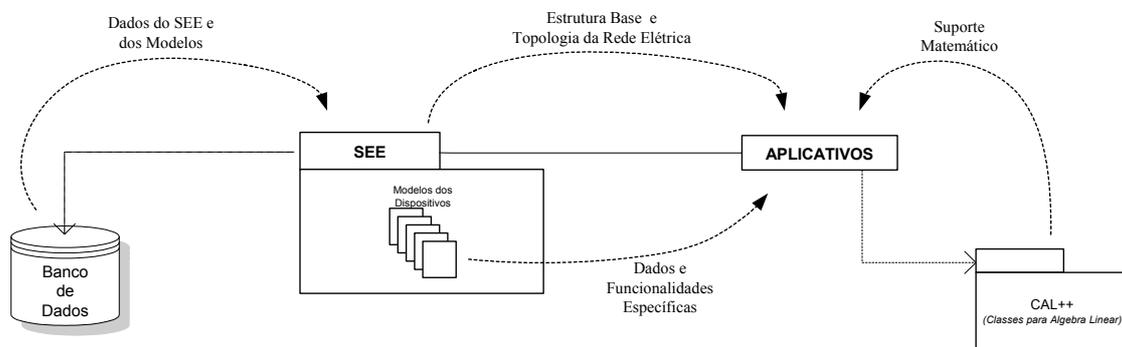


Figura 32 – Estrutura Geral do MOO

Cada nova aplicação utiliza estes mecanismos para construir o seu problema de engenharia específico. Neste sentido, o aplicativo propriamente dito, sob o enfoque do problema de engenharia, fica desacoplado dos problemas de natureza computacional, tais como conectividade dos elementos do SEE, armazenamento e solução de sistemas lineares esparsos, gerenciamento de dados e informações dos dispositivos do SEE, etc. Isto permite ao usuário, ou ao engenheiro, uma maior dedicação ao desenvolvimento de aspectos práticos e teóricos do problema de engenharia a ser resolvido, uma vez que a estrutura computacional base já está disponível e pronta para uso.

Um aplicativo executável pode ser entendido como a união de várias estruturas descritivas e funcionais dispostas em círculos concêntricos compondo níveis de especialização até o aplicativo final, conforme mostra a Figura 33. No núcleo do modelo está a representação da estrutura topológica geral do SEE, nenhuma característica ou informação específica é representada neste nível; no segundo nível, abrigado sobre a estrutura topológica, figuram os modelos dos dispositivos do SEE,

onde são adicionados (e removidos quando necessário) dados e características específicas dos dispositivos; por fim, utilizando as informações e funcionalidades de todos os níveis inferiores estão os aplicativos. Uma última estrutura, exterior ao modelo base, fornece o suporte matemático aos aplicativos.

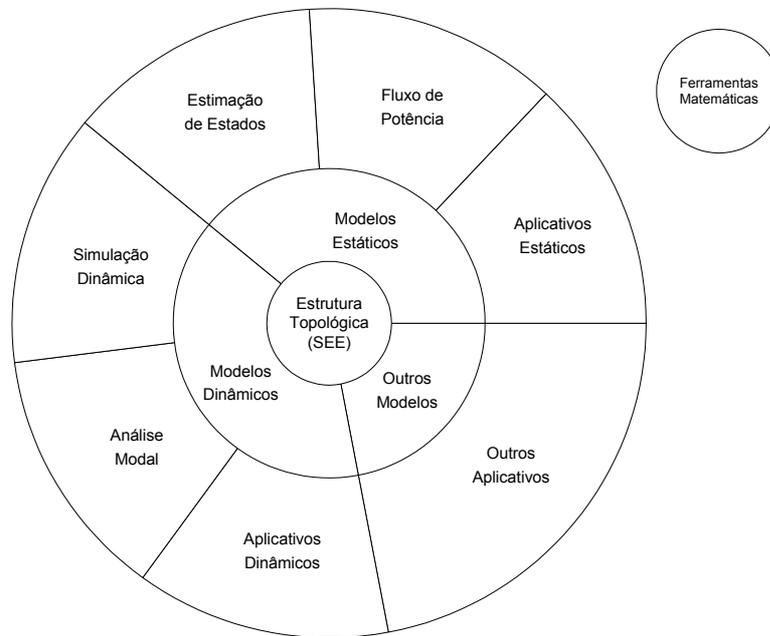


Figura 33 – Estrutura de Utilização

Embora os modelos delimitem regiões de abrangência para os aplicativos (modelos estáticos somente são utilizados por aplicativos estáticos, o mesmo vale para os modelos dinâmicos), o MOO implementado permite certas invasões destas regiões. Assim, um programa de Fluxo de Potência poderá, por exemplo, considerar na sua formulação um modelo detalhado de gerador (máquina síncrona e controladores), ou um programa de Análise Modal poderá considerar na sua formulação uma barra de tensão constante do Fluxo de Potência (barra PV), bastando para isso simplesmente adicionar o modelo desejado no dispositivo e executar o aplicativo. Isto somente é possível para aplicativos que utilizam exclusivamente as funcionalidades padronizadas dos modelos (*inicialização, solução e derivação*).

### 3.6.1 Um Aplicativo Exemplo

Com o objetivo de ilustrar a utilização do MOO e exemplificar a utilização do mecanismo para construção de aplicativos será mostrado a seguir um aplicativo geral, escrito em  $C^{++}$ , para cálculo das tensões complexas de uma rede elétrica qualquer

---

usando a equação matricial:  $[V] = [Y]^{-1} \cdot [I]$

```
00:
01: SEE *ps = load("file.dat");
02:
03: SistLin<complex> SL(ps->maxbarra());
04:
05: //=====
06: //=== Monta Matriz Ybarra ===
07: //=====
08: for (word i=1; i<=ps->maxbarra(); i++)
09: {
10:     word    lin = ps->barra(i)->num();
11:     complex Yii = admt(0,0);
12:
13:     //=== Elemento Diagonal ===
14:     for (word j=1; j<=ps->barra(i)->maxshunt(); j++)
15:     {
16:         ps->barra(i)->shunt(j)->model()->solve();
17:         Yii += ps->barra(i)->shunt(j)->state()->Yi();
18:     }
19:     for (word j=1; j<=ps->barra(i)->maxserie(); j++)
20:     {
21:         ps->barra(i)->serie(j)->model()->solve();
22:         Yii += ps->barra(i)->serie(j)->state()->Yser();
23:         Yii += ps->barra(i)->serie(j)->state()->Yi();
24:     }
25:     SL.A(lin,lin) = Yii;
26:
27:     //=== Elemento Off-Diagonal ===
28:     for (word j=1; j<=ps->barra(i)->maxserie(); j++)
29:     {
30:         word col = ps->barra(i)->barra_ser(j)->num();
31:         SL.A(lin,col) -= ps->barra(i)->serie(j)->state()->Yser();
32:     }
33: }
34: //=====
35: //=== Calcula INJECAO de Corrente ===
36: //=====
37: for (word i=1; i<=ps->maxbarra(); i++)
38: {
39:     word    lin = ps->barra(i)->num();
40:     complex Ii = corr(0,0);
41:
42:     for (word j=1; j<=ps->barra(i)->maxshunt(); j++)
43:     {
44:         Ii -= ps->barra(i)->shunt(j)->state()->Ii();
45:     }
46:     for (word j=1; j<=ps->barra(i)->maxserie(); j++)
47:     {
48:         Ii -= ps->barra(i)->serie(j)->state()->Ii();
49:     }
50:     SL.b(lin) = Ii;
51: }
52: //=====
53: //=== Resolve o SISTEMA LINEAR ===
54: //=====
55: SL.lu();
56: SL.solve();
57: //=====
58: //=== Atualiza as Tensoes ===
59: //=====
60: for (word i=1; i<=ps->maxbarra(); i++)
61: {
62:     ps->barra(i)->V( SL.x(i) );
63: }
```

Para facilitar o entendimento do código escrito acima a Figura 34 mostra o diagrama de classes de alguns dos principais elementos do MOO (*Shunt*, *Série* e *Barra*) que são utilizados no aplicativo exemplo. O diagrama de classes é simplificado e mostra apenas os métodos públicos de cada classe (interface de utilização).

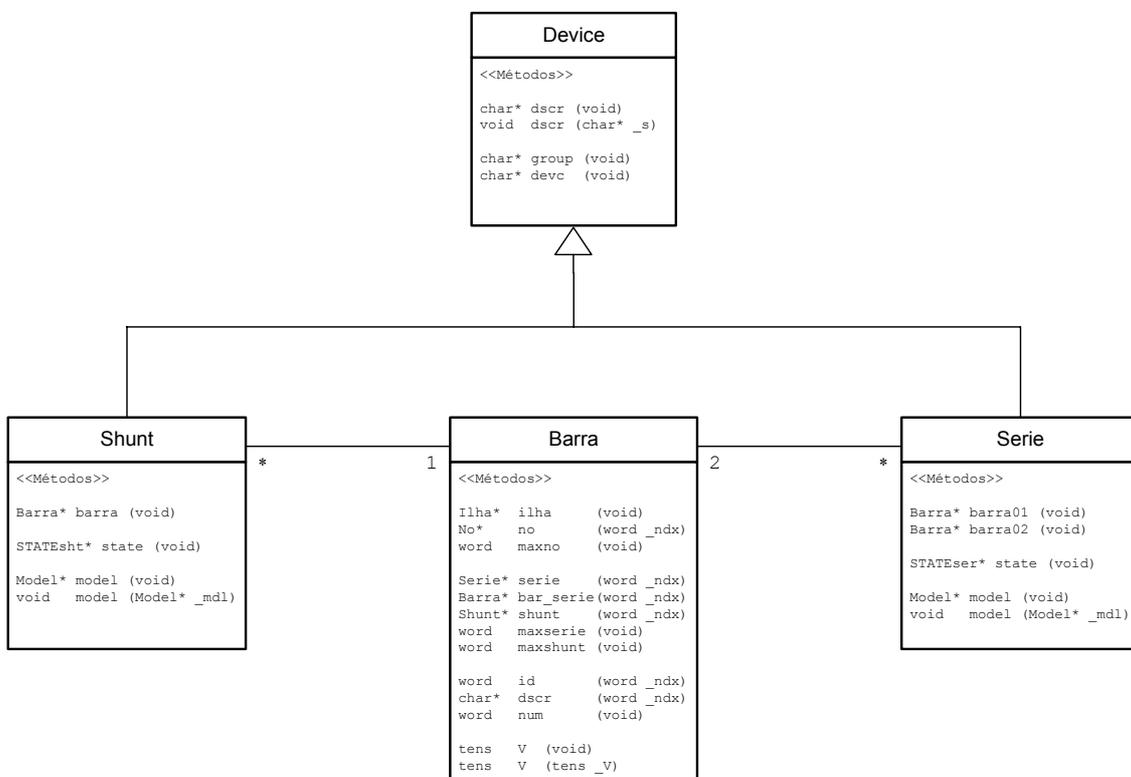


Figura 34 – Diagrama das Classes Utilizadas no Aplicativo Exemplo

Em L:01 do código fonte exemplo são carregados e armazenados, no objeto *ps* (instância da classe *SEE*), todos os dados de um sistema qualquer. Estes dados ficam disponíveis para utilização, em ambas estruturas descritivas (física e lógica), de forma organizada e coerente.

O objeto *ps* possui funções de acesso a ambas estruturas descritivas do SEE, que permitem a livre navegação sobre os dados armazenados. No entanto, para o aplicativo exemplo somente à descrição lógica do SEE é necessária, ou seja apenas as funções de acesso às barras elétricas do SEE (*barra(index)* e *maxbarra()*), e aos dispositivos conectados as barras (*serie(index)*, *maxserie()*, *shunt(index)* e *maxshunt()*) são necessários, demonstrando assim o uso da descrição topológica do SEE para a construção de um aplicativo.

---

Em L:17, L:22, L:23, L:31, L:44 e L:48 o código exemplo mostra acesso aos termos individuais dos estados dos dispositivos (*Shunt* e *Serie*), sendo estes termos utilizados para a construção da matriz admitância de barras e do vetor de injeções de corrente. Em L:16 e L:21 é solicitada a solução do conjunto de equações do modelo (função padronizada do modelo – `solve()`) com a finalidade de atualizar o estado do dispositivo, e suas variáveis de estado internas, se estas existirem. É importante destacar que nenhuma referência é feita ao tipo específico de dispositivo (gerador, linha de transmissão, carga, etc), sendo dado um tratamento generalizado aos dispositivos através das classes base *Shunt* e *Serie* e das funcionalidades padronizadas dos modelos.

O *toolkit* de ferramentas matemáticas CAL<sup>++</sup> é utilizado para construir o sistema linear que executará a solução da rede elétrica, sendo declarado e dimensionado o objeto *SL*, em L:03, para este fim. A utilização do *toolkit* é bastante intuitiva e amigável, ficando o gerenciamento da estrutura esparsa das matrizes e vetores totalmente oculto dos usuários do *toolkit*. A definição dos coeficientes da matriz *A*, em L:25 e L:31, e dos coeficientes do vetor *b*, em L:50, mostram a facilidade com que estes dados são introduzidos na estrutura. Em L:55 e L:56 são executados a fatoração LU da matriz de coeficientes e a solução direta-inversa do sistema linear, respectivamente. Em uma última etapa (L:60 até L:63), a solução do sistema linear é utilizada para atualizar as novas tensões calculadas.

### 3.7 Considerações Finais

A proposição de um *framework* para sistemas de energia elétrica capaz de acomodar todo e qualquer aplicativo em sua estrutura é um objetivo extremamente difícil de ser alcançado, senão impossível. Embora a premissa básica seja representar o sistema elétrico tal qual sua estrutura física real, supondo com isso que o modelo será capaz de suportar uma infinidade de aplicativos, é muito difícil afirmar que o modelo computacional será adequado para todos os aplicativos possíveis na área de sistemas de energia elétrica. É conveniente salientar que um modelo computacional é projetado através da visão do problema dos desenvolvedores, e estes possuem conhecimento limitado a tão somente algumas sub-áreas do domínio geral do problema. Assim, problemas como análise harmônica, planejamento da expansão, simulação de fenômenos eletromagnéticos, e outros não foram sequer considerados

---

na etapa de projeto do modelo orientado a objetos, muito embora isto não signifique que o modelo não será capaz de acomodar estes ou outros aplicativos. A inclusão de aplicativos que os desenvolvedores não consideraram na etapa de projeto pode forçar a reformulação de alguns aspectos do modelo. Cabe ainda lembrar, que um bom projeto orientado a objetos deve levar sempre em conta que o sistema pode necessitar de mudanças ao longo de sua vida. Até o presente momento o MOO proposto foi capaz de suportar satisfatoriamente aplicativos para cálculo de Fluxo de Potência (formulação generalizada e clássica), Análise Modal, Análise de Sensibilidades, Simulação Rápida da Dinâmica e Simulação da Dinâmica Completa pelos métodos Alternado e Simultâneo.

---

## Capítulo 4

# Metodologia de Simulação Rápida no Tempo

### 4.1 Considerações Gerais

Problemas relacionados ao controle e estabilidade da tensão tem sido relatados nos sistemas elétricos de vários países do mundo há alguns anos [37]. No setor elétrico brasileiro, devido ao crescimento das interligações e a falta de investimentos no setor, também já foram relatados alguns incidentes desta natureza. Atualmente, o novo cenário institucional do setor elétrico brasileiro, baseado em um ambiente desregulamentado e competitivo, onde o sistema passa a operar com fraco suporte de reativos e em condições de reduzidas margens de segurança, resulta em um cenário propício para o aparecimento de problemas de instabilidade de tensão, sobretudo na ocorrência de contingências. A estabilidade de tensão é atualmente o assunto de maior interesse envolvendo estudos de estabilidade de sistemas elétricos.

Os estudos de estabilidade de tensão no sistema brasileiro vêm sendo realizados utilizando-se metodologias baseadas no fluxo de potência convencional e na simulação dinâmica completa no tempo. Limitações inerentes a estas metodologias requerem que novas técnicas sejam investigadas para os estudos de instabilidade de tensão. Recentemente, foi proposto um método de simulação que tem se mostrado bastante promissor nestes estudos [38][42][43]. Trata-se do simulador rápido de médio e longo prazo, aqui denominado apenas Simulador Rápido, e que constitui uma área de pesquisa na atualidade. Este método apresenta as mesmas vantagens de desempenho computacional do fluxo de potência enquanto mantém representadas as dinâmicas relevantes de média e longa duração.

---

Neste capítulo a metodologia de simulação rápida de médio e longo prazo é apresentada em sua formulação original e em uma versão modificada, que se mostrou capaz de estudar uma faixa mais ampla de fenômenos que a formulação original.

## 4.2 Estabilidade de Tensão

A estabilidade de tensão (também denominada estabilidade da carga) está associada à incapacidade do sistema em manter um razoável perfil de tensões quando sujeito a um distúrbio. Um sistema entra em instabilidade de tensão quando a ocorrência de um distúrbio, como o aumento da carga ou uma alteração das condições de operação do sistema, causam um progressivo e descontrolado decaimento nas tensões do sistema. Um critério para a detecção da instabilidade é a sensibilidade V-Q para cada uma das barras do sistema, ou seja se a magnitude da tensão de cada barra do sistema aumentar para um correspondente aumento da potência reativa injetada o sistema está estável quanto às tensões (sensibilidade positiva), no entanto se em pelo menos uma barra do sistema a magnitude da tensão diminuir para um correspondente aumento na potência reativa injetada diz-se que o sistema está em processo de instabilidade de tensão (sensibilidade negativa). A instabilidade de tensão esta normalmente associada a sistemas com carregamento elevado ou com fraco suporte de reativos, em que os principais fatores que contribuem para um eventual colapso de tensão são: atuação dos limitadores de potência reativa dos geradores, características da carga, características dos dispositivos de compensação reativa, e a ação de dispositivos de controle de tensão como os LTCs.

### 4.2.1 Definições em Estabilidade de Tensão

Como resultado da intensificação dos estudos na área de estabilidade de tensão uma série de conceitos e definições foram propostos para descrever determinados comportamentos envolvendo o fenômeno. A seguir são apresentados alguns conceitos associados ao estudo de estabilidade de tensão e sua relação com os conceitos tradicionalmente utilizados nos estudos de estabilidade de sistemas elétricos [37][41].

- **Estabilidade de Sistemas Elétricos de Potência:** propriedade que permite a um sistema de energia elétrica permanecer em equilíbrio quando submetido a condições normais de operação ou de alcançar um estado de equilíbrio aceitável após ter sido submetido a um distúrbio. Tradicionalmente, o problema da

---

estabilidade de sistemas elétricos têm sido associado à estabilidade angular do rotor, no entanto instabilidades em sistemas elétricos podem se manifestar de diferentes formas dependendo da configuração do sistema, da sua condição de operação e do tipo de distúrbio.

- **Estabilidade Angular do Rotor:** habilidade das máquinas síncronas de um sistema elétrico de permanecerem em sincronismo sob condições normais de operação e de retornarem a um ponto de equilíbrio aceitável após terem sido submetidas a um distúrbio. A análise da estabilidade angular do rotor envolve o estudo das oscilações eletromecânicas das máquinas síncronas, onde a questão fundamental é a maneira pela qual as máquinas mantêm o balanço de potência quando acontecem variações nos torques e aparecem oscilações nos rotores. Conforme a natureza do fenômeno, a estabilidade angular pode ser dividida em duas categorias: estabilidade de pequenos-sinais (ou pequenas perturbações), relacionada a pequenos distúrbios como por exemplo um aumento gradual da carga; e estabilidade transitória, relacionada a grandes distúrbios como por exemplo um curto-circuito.
- **Estabilidade de Tensão:** habilidade de um sistema elétrico de manter as tensões de operação de todas as barras do sistema em níveis aceitáveis quando submetido a condições normais de operação e de retornar a um ponto de equilíbrio aceitável após ter sido submetido a um distúrbio. Um sistema entra em estado de instabilidade de tensão quando um distúrbio ou mudanças nas condições de operação do sistema causam uma progressiva e descontrolada queda de tensão. A principal causa associada a esta condição de operação é a incapacidade do sistema em manter um balanço de potência reativa adequado para a condição de operação a que está submetido.
- **Estabilidade de Tensão a Grandes Perturbações:** é a habilidade do sistema em manter tensões de operação aceitáveis após a ocorrência de um grande distúrbio, tal como um curto circuito ou abertura de uma linha de transmissão. A ocorrência deste fenômeno deve-se às características da carga e a interação entre os dispositivos de controle e proteção do sistema. A resposta do sistema irá envolver uma grande excursão das tensões. A estabilidade de tensão a grandes perturbações requer o exame do comportamento dinâmico não-linear do sistema num período de tempo suficiente para capturar a interação dos dispositivos que exercem influência sobre este tipo de fenômeno.

- 
- **Estabilidade de Tensão a Pequenas Perturbações:** É a habilidade do sistema em controlar a tensão seguindo um pequeno distúrbio, tal como um aumento gradual de carga, de tal forma que as tensões retornem ao ponto de equilíbrio inicial ou próximo do ponto de operação pré-distúrbio. A ocorrência deste fenômeno deve-se às características dinâmicas do sistema, tal como o comportamento e restauração da carga, e ações de controle discretas e contínuas quando submetidos a pequenos distúrbios em um dado instante de tempo. As perturbações são suficientemente pequenas para permitir a análise pela linearização das equações dinâmicas em torno de um ponto de operação.
  - **Colapso de Tensão:** É o processo que se segue a uma instabilidade de tensão, sendo que um sistema de potência sofrerá colapso se a tensão de equilíbrio pós-distúrbio próxima às cargas estiver abaixo de um valor aceitável de operação. Geralmente este processo é composto de uma série de eventos. O colapso de tensão pode atingir todo o sistema (blecaute) ou somente uma área.
  - **Segurança de Tensão:** É a habilidade do sistema, não somente de operar de forma estável, mas também de permanecer nesta condição após a ocorrência de uma contingência de grande porte ou após mudanças adversas no sistema.

#### 4.2.2 Escalas de Tempo

Embora do ponto de vista da estabilidade de um sistema de potência todos os efeitos possuam uma interligação entre si, é necessário e desejável tratar os problemas divididos em categorias. Neste sentido, a divisão da estabilidade deve levar em conta fatores que contribuem com maior ou menor intensidade em cada categoria. Dois fatores principais estão envolvidos nesta divisão: a intensidade do distúrbio considerado, e as diferentes escalas de tempo de atuação dos equipamentos e seus dispositivos de controle e proteção.

- **Instabilidade de Tensão de Curta Duração (0-10 segs):** instabilidade que ocorre no período de tempo imediatamente após uma grande adversidade no sistema. Este é o tempo clássico dos estudos de estabilidade transitória. Nesta faixa de tempo não existe ainda uma clara distinção entre instabilidade de tensão e instabilidade angular, sendo que na maioria das ocorrências envolvendo problemas associados à instabilidade de tensão houve o envolvimento em alguma proporção de ambos fenômenos. Componentes típicos como HVDC e motores de

---

indução apresentam efeitos severos no processo de instabilidade de tensão nesta faixa de tempo (também denominada Instabilidade de Tensão Transitória). Devido às grandes excursões de tensão e freqüência no sistema, onde as não-linearidades tem efeito destacado, utilizam-se ferramentas de simulação dinâmica completa no tempo para o estudo dos fenômenos nesta escala de tempo. Variáveis associadas à dinâmica lenta do sistema ainda não respondem e podem ser consideradas constantes.

- **Instabilidade de Tensão de Média Duração (alguns minutos):** se o sistema encontra um ponto de equilíbrio factível após o período transitório, manifesta-se então a dinâmica lenta do sistema. Nesta faixa de tempo assume-se que as oscilações de potência sincronizante intermáquinas foram totalmente amortecidas, resultando em uma freqüência uniforme para o sistema. A dinâmica do sistema é conduzida preferencialmente pelos equipamentos automáticos de ação mais lenta como os LTCs, OELs, relés de sobrecorrente temporizados, etc. Nesta faixa de tempo ainda não é possível a intervenção do operador.
- **Instabilidade de Tensão de Longa Duração (várias horas):** a instabilidade de tensão manifesta-se em uma faixa de tempo maior ainda. Aqui a atuação do operador para a inserção de equipamentos de potência reativa ou corte de carga pode ser determinante para evitar a instabilidade. Fatores tais como as características da demanda de carga (curva de carga), ação de caldeiras em unidades térmicas, perda da diversidade da carga pelas baixas tensões, etc podem ser importantes no estudo do fenômeno.

#### **4.2.3 Métodos de Avaliação da Estabilidade de Tensão**

A classificação da estabilidade de tensão em escalas de tempo permite diferenciar os fenômenos que podem ser examinados utilizando uma análise estática, daqueles que devem ser examinados utilizando uma análise dinâmica não-linear. A análise por técnicas estáticas permite o exame de uma ampla faixa de condições de operação do sistema, e quando usadas apropriadamente, dizem muito sobre a natureza do problema e os fatores contribuintes principais. Já a análise dinâmica é útil para o estudo detalhado de situações específicas de colapso de tensão, coordenação da proteção, e da investigação de como será atingido um ponto de equilíbrio de regime permanente. Uma discussão mais detalhada sobre ambas abordagens é apresentada a seguir.

---

#### 4.2.3.1 *Análise Estática*

Embora o problema de estabilidade de tensão possua características dinâmicas, existem várias abordagens que tratam o problema como estático, fazendo uso da suposição de que o mecanismo de instabilidade de tensão acontece lenta e progressivamente (instabilidade de média e longa duração). Com isso, os fenômenos dinâmicos rápidos do sistema já se manifestaram e o acompanhamento do problema pode ser obtido através da resolução do conjunto de equações do fluxo de potência, sendo portanto computacionalmente mais eficiente que a análise dinâmica.

A forma usualmente empregada para conduzir os estudos de estabilidade de tensão nesta abordagem consiste em submeter o sistema a um aumento gradual de carga, respeitando um determinado padrão de incremento, até um ponto de operação que pode submeter o sistema ao colapso de tensão. Vários métodos para detectar o ponto de colapso são propostos na literatura [45][46], a maioria deles está associado à determinação do ponto de singularidade da matriz Jacobiana do Fluxo de Potência ou de matrizes derivadas desta, relacionando a não convergência do Fluxo de Potência com o ponto crítico da ocorrência do colapso de tensão. Dentre as metodologias para detecção da proximidade do ponto de singularidade da matriz Jacobiana e avaliação da estabilidade de tensão destacam-se: o número de condicionamento, decomposição em valores singulares, análise modal (autovalores/autovetores), determinante reduzido, método do vetor tangente, método da continuação, índices baseado em sensibilidade e índices baseado em otimização.

No entanto, a utilização exclusiva de ferramentas de análise estática para estudos de estabilidade de tensão não é recomendável, uma vez que essas podem conduzir a resultados imprecisos na determinação das margens de estabilidade. Estudos [44] mostram significantes discrepâncias entre os resultados encontrados no estudo da estabilidade de tensão quando foram comparados modelos estáticos e simulação completa no tempo. É conhecido que modelos estáticos podem determinar o chamado ponto de bifurcação *Sela-Nó*, que está associado ao limite máximo de transmissão e é identificado pelo popular ponto crítico da curva PV (*nariz da curva*) [38][46]. Este tipo de bifurcação está associado ao decaimento monotônico das tensões de uma barra ou um grupo destas. Entretanto, a análise considerando a dinâmica completa do sistema revela que outros tipos de bifurcações podem ocorrer, como por exemplo, as bifurcações de *Hopf*, de *Singularidade Induzida*, *Induzida por Limites*, etc [45][53][52].

---

#### 4.2.3.2 *Análise Dinâmica*

A abordagem dinâmica da estabilidade de tensão considera a descrição do sistema elétrico como conjunto de equações algébrico-diferenciais não-lineares, dado pela equação:

$$\begin{aligned} \bullet \\ x &= f(x, y) \\ 0 &= g(x, y) \end{aligned} \tag{1}$$

onde  $x$  é o vetor das variáveis de estado,  $y$  é o vetor das tensões complexas nodais e  $f$  e  $g$  são vetores de funções não-lineares que descrevem, respectivamente, as equações diferenciais dos elementos dinâmicos do sistema (geradores e seus controladores, FACTS, motores de indução, etc) e as equações algébricas da rede elétrica.

Em sistemas de energia elétrica é usual a classificação dos problemas de estabilidade de tensão em diferentes escalas de tempo. Esta classificação merece destaque dentro da análise dinâmica, uma vez que os resultados obtidos são diretamente influenciados pelos dispositivos que tiveram seu comportamento dinâmico explicitamente representado na análise do problema. Ou seja, em estudos de estabilidade de tensão de curta duração a análise deve considerar as equações diferenciais dos dispositivos rápidos do sistema, tais como os geradores e seu conjunto de controladores, HVDC, motores de indução, etc, e pode desconsiderar o comportamento dinâmico dos dispositivos de ação mais lenta, que não possuem influência significativa nesta faixa de tempo. Com a adoção de uma escala de tempo mais longa, como nos estudos de estabilidade de médio e longo prazo, os tempos envolvidos são da ordem de minutos ou horas e as dinâmicas dos elementos rápidos podem ou não ser desprezadas, entretanto devem ser incorporados na análise os equipamentos dinâmicos de ação mais lenta, tais como caldeiras para as unidades térmicas, limitadores de sobre-excitação, e esquemas de controle centralizado (CAG-CST), além da ação de dispositivos discretos como LTCs, chaveamento de reatores ou capacitores e a própria evolução da carga ao longo do tempo (curva de carga). O conjunto completo das equações do sistema assume então a seguinte forma:

$$\begin{aligned} \bullet \\ x &= f(x, y, z(k)) \\ 0 &= g(x, y, z(k)) \\ z_{(k+1)} &= h(x, y, z(k)) \end{aligned} \tag{2}$$

---

onde  $z$  é um vetor de variáveis de ação discreta e  $h$  é um vetor de funções de controle.

As metodologias baseadas na abordagem dinâmica são então capazes de identificar um conjunto maior de fenômenos ocorridos no sistema, podendo identificar, por exemplo, fenômenos como o colapso de tensão ou a instabilidade oscilatória. As metodologias de análise da estabilidade dinâmica de tensão são: simulação dinâmica completa no domínio do tempo, simulação rápida (quase-estática), análise modal, teoria de bifurcações e metodologias baseadas em função energia.

Dentre os métodos de análise da estabilidade de tensão a simulação dinâmica completa é a que fornece a resposta mais exata para o comportamento dinâmico do sistema. Por este motivo tem sido utilizada quando há a necessidade de um estudo detalhado dos fenômenos dinâmicos envolvidos ou como "*benckmark*" para a verificação dos resultados obtidos com outras metodologias de análise. A simulação dinâmica completa consiste em resolver o conjunto de equações algébrico-diferenciais não-lineares que descrevem o comportamento dinâmico do sistema, onde o conjunto de equações diferenciais é algebrizado através de algum método de integração numérica (normalmente a regra trapezoidal implícita) e então resolvido passo-a-passo ao longo do tempo, juntamente com o conjunto de equações algébricas.

Se por um lado à simulação dinâmica completa fornece precisamente o comportamento dinâmico do sistema, por outro demanda um elevado esforço computacional, além de não fornecer com facilidade informações a respeito de sensibilidades e do grau de instabilidade do sistema. A determinação do local e causa da provável instabilidade envolve normalmente a análise de um grande número de curvas e um elevado número de simulações. Recentemente, foi proposto um método de simulação que representa as dinâmicas relevantes de médio e longo prazo enquanto mantém as mesmas vantagens de desempenho computacional do fluxo de potência. Este método denomina-se simulação rápida e tem se mostrado bastante promissor em estudos de estabilidade de tensão de médio e longo prazo.

---

### 4.3 Simulação Rápida

Os fenômenos envolvidos no estudo da estabilidade de tensão de médio e longo prazo são de natureza lenta, normalmente na faixa de minutos ou horas. Com isso, os fenômenos dinâmicos rápidos já se manifestaram e a dinâmica do sistema é conduzida preferencialmente pelas atuações dos elementos discretos e pelas variações na carga. Assim, a dinâmica transitória pode ser considerada estável e instantânea e substituída pela sua correspondente equação de equilíbrio:

$$\dot{x} = 0 = f(x, y, z(k)) \quad (3)$$

Esta constatação deu origem a um método simplificado de simulação [42][43], denominado Simulação Rápida, que consiste em aproximar a evolução das variáveis no tempo através do cálculo de uma sucessão de *Pontos de Equilíbrio*. Assim, o conjunto de equações:

$$\begin{aligned} 0 &= f(x, y, z(k)) \\ 0 &= g(x, y, z(k)) \\ z(k+1) &= h(x, y, z(k)) \end{aligned} \quad (4)$$

pode ser resolvido pelo método de Newton, fornecendo um novo ponto de equilíbrio para o sistema, assumida uma determinada trajetória de carga e situação das variáveis discretas de controle. O novo ponto de equilíbrio é então utilizado para verificar se alguma variável discreta de controle necessita ser modificada. Isto ocorre, por exemplo, quando o erro de tensão de alguma barra controlada por um LTC viola os limites permitidos, fazendo com que o tap deste LTC seja deslocado em uma posição para tentar corrigir o desvio de tensão.

O fato da dinâmica transitória ser desconsiderada (conforme Equação 3) torna desnecessária a integração numérica das equações diferenciais. Isto, aliado a uma substancial simplificação que pode ser obtida nas equações diferenciais dos elementos dinâmicos (um gerador e seu conjunto completo de controladores pode ser descrito por apenas três variáveis de estado, como será mostrado a seguir) torna possível obter um método de simulação muito eficiente computacionalmente.

---

### 4.3.1 Modelos Matemáticos

A seguir serão deduzidas as equações de equilíbrio para os principais componentes do sistema nos estudos de estabilidade de médio e longo prazo. O procedimento geral adotado despreza as equações diferenciais dos elementos com dinâmica rápida ( $dx/dt=0$ ), fazendo as variações das variáveis de estado no tempo serem consideradas como instantâneas.

#### 4.3.1.1 Geradores

Uma unidade de geração, em sistemas de energia elétrica, é basicamente constituída pela máquina síncrona, sistemas de controle da excitação, sistemas de controle de velocidade e pelo sistema de geração de vapor quando a unidade for térmica. No entanto, a modelagem deduzida neste trabalho não considera a dinâmica do sistema de geração de vapor das unidades térmicas, supondo estoque infinito de vapor para estas unidades.

##### **Máquina Síncrona:**

Os modelos matemáticos normalmente utilizados para representar as máquinas síncronas em estudos de estabilidade transitória encontram-se muito bem estabelecidos [37][38]. Estes modelos são derivados do modelo geral de Park para máquinas síncronas, onde assume-se um sistema de coordenadas que gira na mesma velocidade do rotor da máquina, definindo um eixo em fase com os pólos do rotor (eixo direto -  $d$ ) e outro 90° elétricos atrasado em relação ao primeiro (eixo quadratura -  $q$ ), conforme ilustra a Figura 35a.

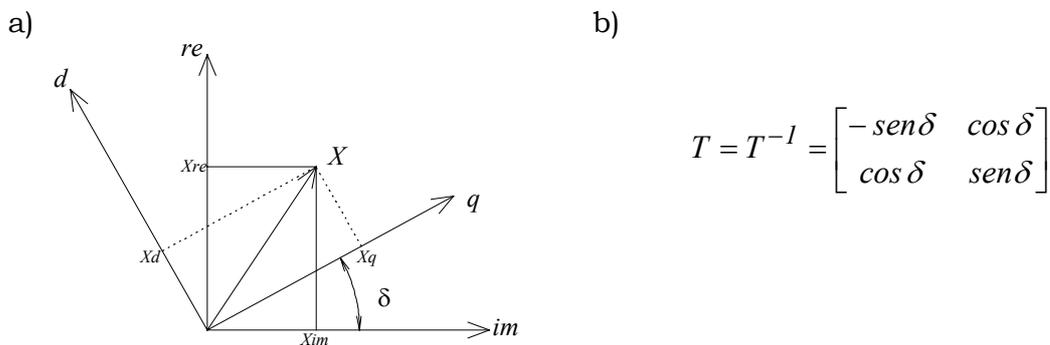


Figura 35 – Sistema de Referência e Matriz de Transformação

Assumindo-se algumas hipóteses simplificadoras no modelo geral [37][38] é possível definir um conjunto de equações diferenciais ordinárias para descrever o comportamento dinâmico dos rotores das máquinas síncronas, constituído de um subconjunto de equações elétricas e um subconjunto de equações eletromecânicas. Para uma máquina de pólos salientes e sendo desprezados os efeitos subtransitórios, as equações (5)(6) e (7) descrevem o comportamento dinâmico do rotor.

$$\dot{E}'_q = \frac{1}{T'_{do}} \cdot [E_{fd} + (x_d - x'_d) \cdot I_d - E'_q] \quad (5)$$

$$\dot{w} = \frac{1}{2 \cdot H} \cdot (P_m - P_e) \quad (6)$$

$$\dot{\delta} = w_r \cdot (w - 1) \quad (7)$$

Os enrolamentos estatóricos das máquinas síncronas podem ter suas constantes de tempo desprezadas, sendo então representados por um conjunto de equações algébricas. Considerando a resistência da armadura nula, para simplificar as manipulações algébricas, as equações do estator assumem a seguinte forma:

$$\begin{aligned} -V_d &= r \cdot I_d + x_q \cdot I_q \\ E'_q - V_q &= r \cdot I_q - x'_d \cdot I_d \end{aligned} \Rightarrow \begin{aligned} I_q &= -\frac{V_d}{x_q} \\ I_d &= -\frac{E'_q - V_q}{x'_d} \end{aligned} \quad (8)$$

onde,

$$\begin{aligned} V_d &= -V_{re} \cdot \text{sen} \delta + V_{im} \cdot \text{cos} \delta \\ V_q &= V_{re} \cdot \text{cos} \delta + V_{im} \cdot \text{sen} \delta \end{aligned} \quad (9)$$

A potência elétrica da máquina síncrona e a corrente injetada na rede elétrica são dadas pelas equações (10) e (11), respectivamente.

$$P_e = V_d \cdot I_d + V_q \cdot I_q \quad (10)$$

$$\begin{aligned} I_{re} &= -I_d \cdot \text{sen} \delta + I_q \cdot \text{cos} \delta \\ I_{im} &= I_d \cdot \text{cos} \delta + I_q \cdot \text{sen} \delta \end{aligned} \quad (11)$$

O modelo matemático descrito acima pode ser utilizado em estudos de estabilidade transitória para representar máquinas síncrona de pólos salientes onde são considerados os efeitos transitórios do campo e desconsiderados os efeitos

---

subtransitórios devido aos enrolamentos amortecedores. Este modelo é o mais simples que considera os efeitos transitórios para o campo.

**Sistema de Controle da Excitação (Regulador de Tensão):**

A modelagem matemática utilizada para reguladores de tensão é bastante vasta e permite representar a maioria dos reguladores de tensão atualmente em uso. Um modelo simplificado de 1ª ordem é descrito pela equação (12) e pode ser utilizado para representar reguladores que possuem somente elementos estáticos. Este modelo, apesar de simplificado, mantém as características dinâmicas do regulador de tensão.

$$\begin{aligned} \dot{E}_{fd} &= \frac{K_a}{T_a} \cdot (V_{ref} - V_t) - \frac{1}{T_a} \cdot E_{fd} \\ V_t &= \sqrt{V_{re}^2 + V_{im}^2} \end{aligned} \quad (12)$$

**Sistema de Controle de Velocidade (Turbina + Regulador de Velocidade):**

O sistema de controle de velocidade dos geradores possuem efeito mais destacado em estudos de média e longa duração, e, nestes casos, devem ser devidamente representados. A equação (13) descreve um sistema de controle de velocidade simplificado que ainda mantém suas características dinâmicas.

$$\dot{P}_m = \frac{1}{T_c} \cdot \left( P_{ref} - \frac{1}{R} \cdot (w - 1) \right) - \frac{1}{T_c} \cdot P_m \quad (13)$$

**Simplificações Assumidas no Modelo de Gerador**

Os modelos matemáticos apresentados podem ser utilizados tanto para estudos de longa duração quanto para estudos de estabilidade transitória, quando não é necessário um modelo detalhado das máquinas síncronas e sistemas de controle. No entanto, as hipóteses assumidas para o simulador rápido desconsideram as dinâmicas transitórias, fazendo as variações das variáveis de estado no tempo instantâneas ( $dx/dt = 0$ ). Assim, as seguintes simplificações são assumidas no modelo:

Fazendo  $dE'_q/dt = 0$ ,  $dw/dt = 0$  e  $d\delta/dt = 0$ , respectivamente nas equações (5),(6) e (7) da máquina síncrona, e eliminando-se as constantes de tempo  $T'_{do}$  e  $H$  obtém-se as equações (14) e (15).

$$E'_q = E_{fd} + (x_d - x'_d) \cdot I_d \quad (14)$$

---


$$P_m = P_e \quad (15)$$

Substituindo a equação  $E'_q = E_q + (x_d - x'_d) \cdot I_d$  em (14) e (8), para eliminar a tensão transitória de eixo quadratura  $E'_q$ , obtém-se o conjunto de equações de regime permanente para a máquina síncrona:

$$E_q = E_{fd} \quad (16)$$

$$P_m = V_d \cdot I_d + V_q \cdot I_q \quad (17)$$

$$I_q = -\frac{V_d}{x_q} \quad (18)$$

$$I_d = -\frac{E_q - V_q}{x_d}$$

No sistema de controle da excitação, fazendo  $dE_{fd}/dt = 0$  na equação (12) e eliminando a constante de tempo  $T_a$  obtém-se a seguinte equação de regime permanente:

$$E_{fd} = K_a \cdot \left( V_{ref} - \sqrt{V_{re}^2 + V_{im}^2} \right) \quad (19)$$

Procedimento semelhante pode ser feito para o sistema de controle de velocidade, onde  $dP_m/dt = 0$  em (13) conduz a equação de regime permanente:

$$P_m = P_{ref} - l/R \cdot (\omega - 1) \quad (20)$$

### **Conjunto de Equações do Modelo do Gerador**

Substituindo (20) em (17) e eliminando as variáveis algébricas auxiliares  $I_d$  e  $I_q$ , obtém-se um modelo de regime permanente para a máquina síncrona com três equações algébricas ( $f_1, f_2$  e  $f_3$ ), além das duas equações de injeção na rede elétrica:

$$f_1 \Rightarrow 0 = E_{fd} - E_q \quad (21)$$

$$f_2 \Rightarrow 0 = P_{ref} - l/R \cdot (\omega - 1) + \left( \frac{V_d \cdot (E_q - V_q)}{x_d} + \frac{V_q \cdot V_d}{x_q} \right) \quad (22)$$

$$f_3 \Rightarrow 0 = K_a \cdot \left( V_{ref} - \sqrt{V_{re}^2 + V_{im}^2} \right) - E_{fd} \quad (23)$$

$$\begin{aligned}
g_1 \Rightarrow I_{re} &= \left[ \frac{E_q - V_q}{x_d} \right] \cdot \text{sen} \delta + \left[ -\frac{V_d}{x_q} \right] \cdot \text{cos} \delta \\
g_2 \Rightarrow I_{im} &= \left[ -\frac{E_q - V_q}{x_d} \right] \cdot \text{cos} \delta + \left[ -\frac{V_d}{x_q} \right] \cdot \text{sen} \delta
\end{aligned} \tag{24}$$

sendo

$$\begin{aligned}
V_d &= -V_{re} \cdot \text{sen} \delta + V_{im} \cdot \text{cos} \delta \\
V_q &= V_{re} \cdot \text{cos} \delta + V_{im} \cdot \text{sen} \delta
\end{aligned} \tag{25}$$

A variável  $w$  representa a velocidade das máquinas síncronas e, uma vez que as oscilações de potência não existem neste modelo, é igual para todas as máquinas do sistema. Assim, assume-se apenas uma variável associada à velocidade das máquinas síncronas, sendo esta substituída pela frequência do sistema se for conveniente.

#### 4.3.1.2 Rede Elétrica

Os elementos que formam a rede elétrica são representados através de modelos estáticos (modelo  $\pi$  para a grande maioria dos dispositivos) e, sendo assim, podem ser descritos pela equação complexa que representa o balanço de corrente em todas as barras do sistema.

$$\bar{I}_{(x,y,z_k)} = \bar{Y} \cdot \bar{V} \tag{26}$$

Decompondo a equação (26) em suas componentes real e imaginária obtém-se:

$$\begin{aligned}
g_1 \Rightarrow \Re[\bar{I}_{i(x,y,z_k)}] &= \sum_{j \in \Omega_i} (G_{ij} \cdot V_{re_j} - B_{ij} \cdot V_{im_j}) \\
g_2 \Rightarrow \Im[\bar{I}_{i(x,y,z_k)}] &= \sum_{j \in \Omega_i} (G_{ij} \cdot V_{im_j} + B_{ij} \cdot V_{re_j})
\end{aligned} \tag{27}$$

onde  $\Re[\bar{I}_{i(x,y,z_k)}]$  e  $\Im[\bar{I}_{i(x,y,z_k)}]$  são as componentes real e imaginária da injeção de corrente dos dispositivos *shunt* (geradores, cargas, etc.) na barra  $i$ .

#### 4.3.1.3 Cargas Estáticas

As cargas estáticas podem ser representadas através de uma injeção constante de potência na rede elétrica (equação (28) mantendo-se  $P$  e  $Q$  constantes), ou através de um modelo que assumem a variação da carga devido a alterações na tensão da

---

barra terminal através de uma função polinomial ou exponencial (equação (28) fazendo-se  $P_{(v)}$  e  $Q_{(v)}$ ).

$$\bar{I}_{(y, z_k)} = -\frac{P - j \cdot Q}{\bar{V}^*} \quad (28)$$

Em estudos de longa duração deve-se considerar ainda a variação da potência nominal da carga no intervalo de tempo simulado, devido a alterações na demanda da carga que está sendo representada. Assim, o modelo comumente utilizado para representar este efeito descreve o comportamento da carga segundo uma curva discretizada ao longo do intervalo de simulação.

## 4.4 Simulação Rápida Modificada

Na simulação rápida original são assumidas hipóteses que não somente simplificam as equações dinâmicas dos modelos mas também eliminam variáveis de estado que não influenciam o cálculo dos pontos de equilíbrio do sistema (as variáveis de estado associadas aos PSSs, por exemplo, podem ser completamente removidas do modelo de regime permanente). Estas simplificações visam obter o modelo de regime permanente dos dispositivos através da eliminação das dinâmicas rápidas, e por consequência disso, tornam o simulador rápido original incapaz de detectar fenômenos dinâmicos de natureza oscilatória (como as que ocorrem na presença de bifurcações de Hopf, por exemplo). Por esta razão, a seguir é proposta uma modificação no método de simulação rápida que permite a detecção destes e outros fenômenos dinâmicos que podem ocorrer no sistema.

A modificação proposta consiste em manter as equações diferenciais dos modelos em seu formato original, sem nenhuma simplificação ou eliminação de constantes de tempo. Isto faz com que o conjunto de equações que descreve o sistema contenha a representação da dinâmica completa dos dispositivos, como será demonstrado a seguir.

### 4.4.1 Modelos Matemáticos

A modificação proposta para o simulador rápido tem efeito apenas para os elementos dinâmicos que tiveram suas equações diferenciais simplificadas. Assim, os modelos matemáticos apresentados a seguir se limitarão aos dispositivos dinâmicos

(máquinas síncronas e conjunto de controladores), uma vez que qualquer dispositivo estático do sistema, tais como a rede elétrica ou as cargas estáticas, mantém a formulação original do método de simulação rápida (ver itens 4.3.1.2 e 4.3.1.3).

#### 4.4.1.1 Geradores

O modelo de gerador apresentado em 4.3.1.1 representa uma máquina síncrona de pólos salientes com efeitos transitórios do campo considerados, e dois modelos simplificados para o sistema de excitação e controle de velocidade que ainda mantém suas características dinâmicas. As equações diferenciais destes dispositivos são rerepresentadas a seguir.

##### **Máquina Síncrona:**

$$\dot{E}'_q = \frac{1}{T'_{do}} \cdot [E_{fd} + (x_d - x'_d) \cdot I_d - E'_q] \quad (29)$$

$$\dot{w} = \frac{1}{2 \cdot H} \cdot (P_m - P_e) \quad (30)$$

$$\dot{\delta} = w_r \cdot (w - I) \quad (31)$$

##### **Sistema de Controle da Excitação (Regulador de Tensão):**

$$\dot{E}_{fd} = \frac{K_a}{T_a} \cdot (V_{ref} - V_t) - \frac{1}{T_a} \cdot E_{fd} \quad (32)$$

$$V_t = \sqrt{V_{re}^2 + V_{im}^2}$$

##### **Sistema de Controle de Velocidade (Turbina + Regulador de Velocidade):**

$$\dot{P}_m = \frac{1}{T_c} \cdot \left( P_{ref} - \frac{1}{R} \cdot (w - I) \right) - \frac{1}{T_c} \cdot P_m \quad (33)$$

##### **Conjunto de Equações do Gerador no Simulador Rápido Modificado**

A hipótese básica assumida para o simulador rápido desconsidera as dinâmicas transitórias, fazendo as variações das variáveis de estado no tempo instantâneas ( $dx/dt = 0$ ). Para o conjunto de equações da máquina síncrona e seus controladores faz-se simplesmente:

$$\dot{E}'_q = 0; \quad \dot{w} = 0; \quad \dot{\delta} = 0; \quad \dot{E}_{fd} = 0; \quad \dot{P}_m = 0;$$

mantendo assim as equações em seu formato original. Isto conduz a um modelo de regime permanente para a máquina síncrona utilizando as equações tradicionalmente usadas nos estudos de estabilidade transitória e/ou análise modal. O modelo contém cinco equações algébricas ( $f_1, f_2, f_3, f_4$  e  $f_5$ ), além das duas equações de injeção na rede elétrica:

$$f_1 \Rightarrow 0 = \frac{1}{T'_{do}} \cdot [E_{fd} + (x_d - x'_d) \cdot I_d - E'_q] \quad (34)$$

$$f_2 \Rightarrow 0 = \frac{1}{2 \cdot H} \cdot \left( P_m - \left( \frac{V_d \cdot (E'_q - V_q)}{x'_d} + \frac{V_q \cdot V_d}{x_q} \right) \right) \quad (35)$$

$$f_3 \Rightarrow 0 = w_r \cdot (w - 1) \quad (36)$$

$$f_4 \Rightarrow 0 = \frac{K_a}{T_a} \cdot (V_{ref} - \sqrt{V_{re}^2 + V_{im}^2}) - \frac{1}{T_a} \cdot E_{fd} \quad (37)$$

$$f_5 \Rightarrow 0 = \frac{1}{T_c} \cdot \left( P_{ref} - \frac{1}{R} \cdot (w - 1) \right) - \frac{1}{T_c} \cdot P_m \quad (38)$$

$$g_1 \Rightarrow I_{re} = \left[ \frac{E'_q - V_q}{x'_d} \right] \cdot \text{sen} \delta + \left[ -\frac{V_d}{x_q} \right] \cdot \text{cos} \delta \quad (39)$$

$$g_2 \Rightarrow I_{im} = \left[ -\frac{E'_q - V_q}{x'_d} \right] \cdot \text{cos} \delta + \left[ -\frac{V_d}{x_q} \right] \cdot \text{sen} \delta$$

sendo

$$\begin{aligned} V_d &= -V_{re} \cdot \text{sen} \delta + V_{im} \cdot \text{cos} \delta \\ V_q &= V_{re} \cdot \text{cos} \delta + V_{im} \cdot \text{sen} \delta \end{aligned} \quad (40)$$

#### 4.4.2 Considerações sobre os Modelos Matemáticos e suas Implicações

Observa-se, nas equações do simulador rápido original, que as constantes de tempo ( $T'_{do}$ ,  $H$ ,  $T_a$  e  $T_c$ ) são eliminadas, uma vez que estas não têm qualquer influência para o cálculo do ponto de equilíbrio. No entanto, para o simulador rápido modificado todas as constantes de tempo são mantidas no conjunto de equações resultante ( $f_1, f_2, f_4$  e  $f_5$ ). Desta forma, os coeficientes obtidos no processo de linearização das equações para solução pelo método de Newton ( $\partial f / \partial x$  e  $\partial f / \partial y$  na obtenção da matriz Jacobiana) contém os efeitos da dinâmica completa do sistema.

---

A linearização do conjunto de equações do sistema, considerando as variáveis discretas constantes durante a solução pelo método de Newton, resulta em:

$$\begin{bmatrix} \dot{\Delta x} = 0 \\ 0 \end{bmatrix} = \underbrace{\begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \end{bmatrix}}_{J_{xy}} \cdot \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \quad (41)$$

Devido a retenção da dinâmica completa do sistema na formulação base do simulador rápido modificado, a matriz Jacobiana é idêntica àquela utilizada para a análise modal [45][53]. Ou seja, os autovalores da *Matriz de Estados*, obtida através da equação (42), representam a dinâmica completa do sistema para um determinado ponto de operação, encerrando tanto as dinâmicas transitórias quanto as dinâmicas de natureza mais lenta.

$$A = \begin{bmatrix} \frac{\partial f}{\partial x} \end{bmatrix} - \begin{bmatrix} \frac{\partial f}{\partial y} \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial g}{\partial y} \end{bmatrix}^{-1} \cdot \begin{bmatrix} \frac{\partial g}{\partial x} \end{bmatrix} \quad (42)$$

Desta forma, a análise dos autovalores da *Matriz de Estados*, obtida ao longo da trajetória dos pontos de equilíbrio, permite que bifurcações locais possam ser detectadas e analisadas. Além disto, outras valiosas informações podem ainda ser obtidas das matrizes  $A$  e  $J_{xy}$ , tais como sensibilidades para o projeto ou sintonia de controladores (através dos autovetores e/ou fatores de participação associados a um determinado autovalor) e a indicação de direções para uma eventual ação corretiva visando afastar o sistema de regiões susceptíveis a problemas de instabilidade de tensão, tanto de natureza oscilatória como monotônica.

Para os propósitos de cálculo do ponto de equilíbrio ambas as representações conduzem rigorosamente a mesma solução, porém, como já foi dito, as equações do simulador rápido modificado encerram a dinâmica completa do sistema (sem reduções ou simplificações) contendo assim uma riqueza maior de informações.

Outra vantagem do método modificado é que os modelos matemáticos utilizados para representar os dispositivos dinâmicos do sistema são os mesmos utilizados nos estudos de estabilidade transitória e/ou análise modal, o que evita a necessidade de substituição de modelos e dados para a execução de um ou outro programa. Além disto, os modelos apresentados neste trabalho para as máquinas síncronas e controladores podem ser considerados modestos para os estudos de estabilidade transitória e/ou análise modal, porém a metodologia apresentada suporta

---

modelos tão complexos e sofisticados quanto àqueles utilizados nas simulações de estabilidade transitória.

Cabe ainda comentar que a dimensão da matriz  $J_{xy}$  do simulador rápido modificado pode ser significativamente maior que a matriz  $J_{xy}$  do simulador rápido original (sobretudo se modelos detalhados para os geradores forem utilizados). Entretanto, este aumento na dimensão não acarreta grande esforço computacional adicional, uma vez que o principal esforço computacional para a solução do sistema de equações (41) reside na fatoração da submatriz de  $J_{xy}$  referente às equações algébricas da rede elétrica ( $\partial g/\partial y$ ), que são idênticas para ambas abordagens.

## 4.5 Comparação entre as Metodologias

Alguns aspectos das metodologias de simulação rápida original e modificada serão mostrados através de resultados obtidos para um sistema reduzido tomado como exemplo. O sistema exemplo possui 9 barras e três geradores, sendo todos os geradores representados pelos modelos dinâmicos apresentados neste capítulo. O diagrama unifilar do sistema exemplo é mostrado na Figura 36.

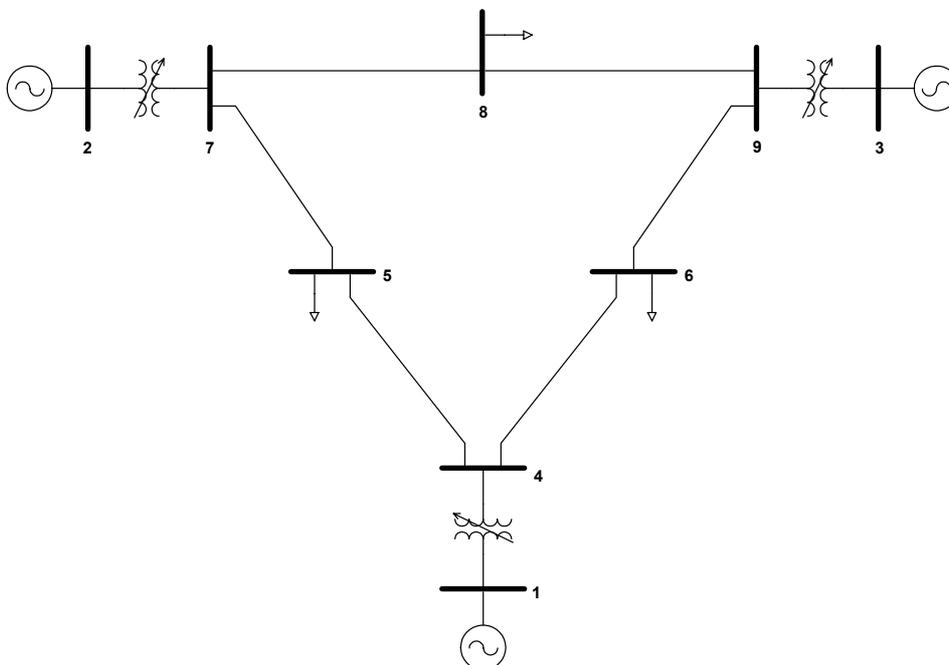


Figura 36 – Sistema Exemplo de 9 Barras

---

A característica principal do simulador rápido é a simulação da trajetória lenta do sistema no tempo, através do cálculo de uma sucessão de *Pontos de Equilíbrio*, evitando a integração das equações diferenciais. Assim, para os propósitos de cálculo do ponto de equilíbrio, ambas metodologias devem conduzir rigorosamente a mesma solução. As figuras a seguir mostram grandezas do sistema exemplo quando este foi submetido a um acréscimo de 50% na carga distribuído ao longo de 60 segs. As curvas foram obtidas com um simulador rápido nas versões original e modificada, e mostram que ambas metodologias obtêm rigorosamente a mesma solução.

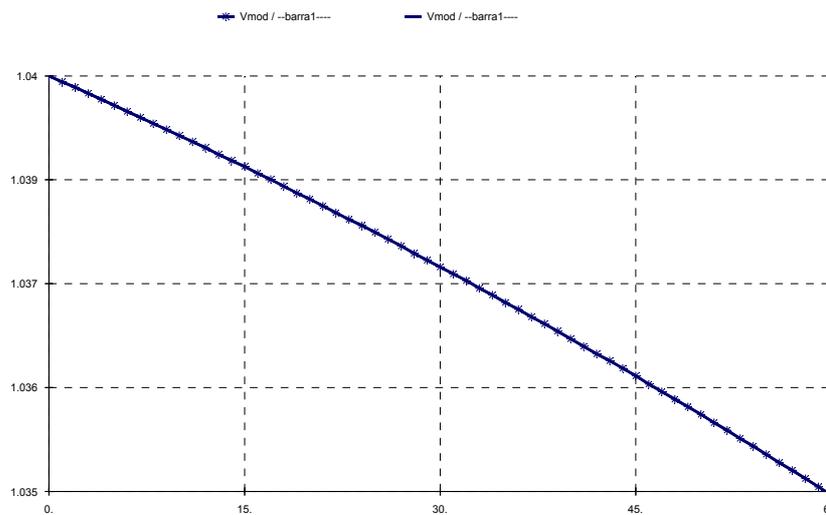


Figura 37 – Tensão Terminal do Gerador 1 em ambas Metodologias

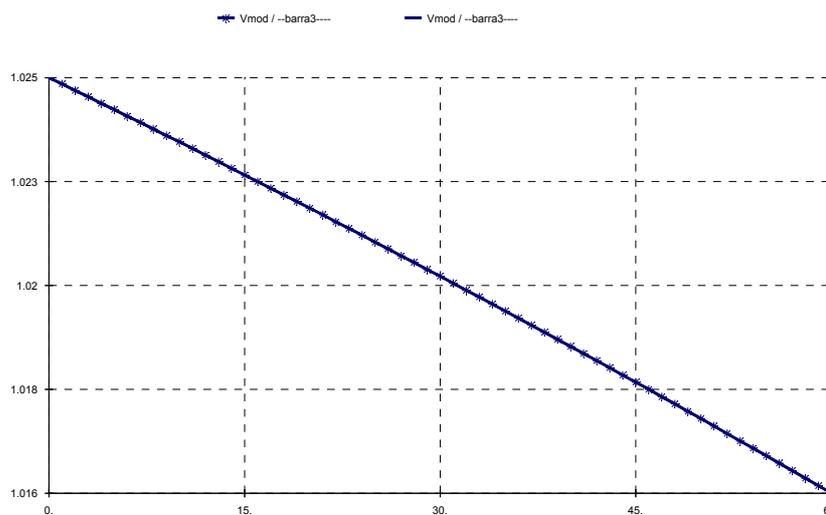


Figura 38 – Tensão Terminal do Gerador 3 em ambas Metodologias

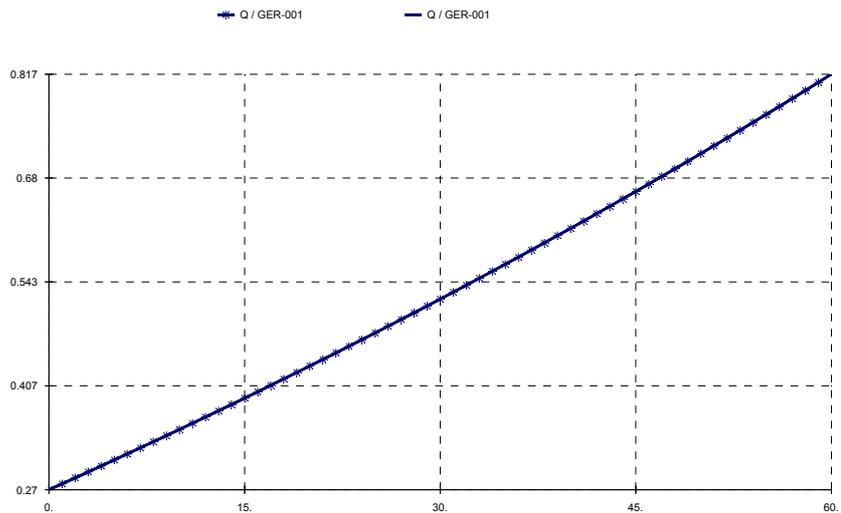


Figura 39 – Potência Reativa do Gerador 1 em ambas Metodologias

A retenção da dinâmica do sistema sem simplificações na formulação base do simulador rápido modificado permite a análise da estabilidade do sistema ao longo da trajetória, obtida através do cálculo dos autovalores da *Matriz de Estados*. Esta característica é exclusiva do simulador rápido modificado e permite detectar e analisar bifurcações locais. A Figura 40 mostra a trajetória dos autovalores do sistema exemplo para o acréscimo de carga do caso anterior. Os resultados mostram a ocorrência de uma bifurcação de *Hopf* no sistema (causada pelo cruzamento de um par de autovalores complexos conjugados para o semiplano direito do plano complexo), bem como os fatores de participação principais para alguns autovalores selecionados.

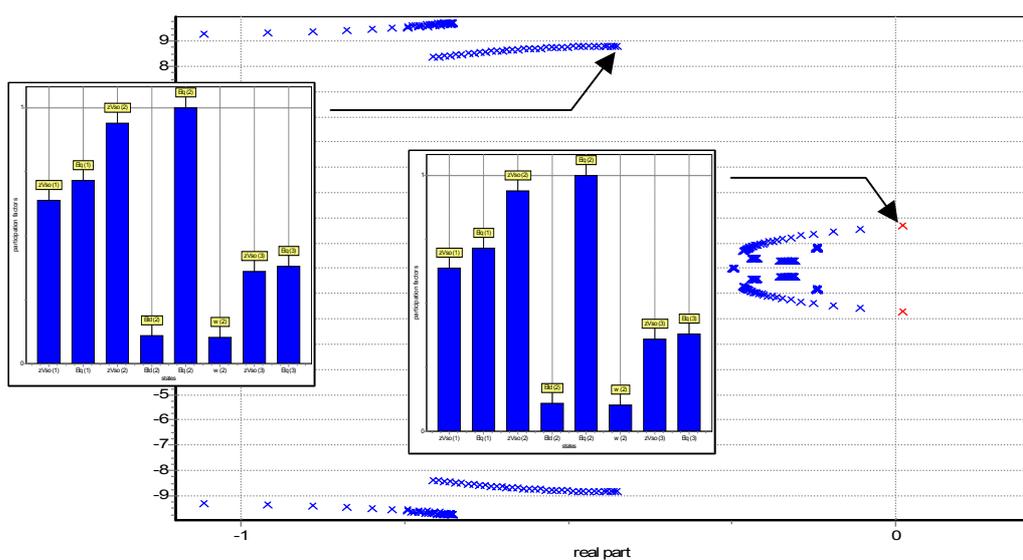


Figura 40 – Análise da Estabilidade ao Longo da Trajetória

Em estudos onde é necessário representar um grande impacto no sistema, como a abertura de linhas de transmissão ou o desligamento de grandes blocos de carga, a simulação rápida pode conduzir a resultados incorretos. Considere o sistema exemplo da Figura 36 em que foi aplicado um degrau de carga de 30% no sistema no instante 10 segs. A Figura 41 mostra todas tensões do sistema para a simulação deste distúrbio com o simulador rápido modificado e a análise dos autovalores do sistema para as situações antes e depois do distúrbio. Os resultados indicam que o sistema suporta satisfatoriamente o distúrbio, mantendo o perfil de tensões em patamares adequados e não excedendo nenhum limite dos reguladores de tensão. Além disso, os dois pontos de equilíbrio obtidos são estáveis, uma vez que todos os autovalores localizam-se no semiplano esquerdo do plano complexo.

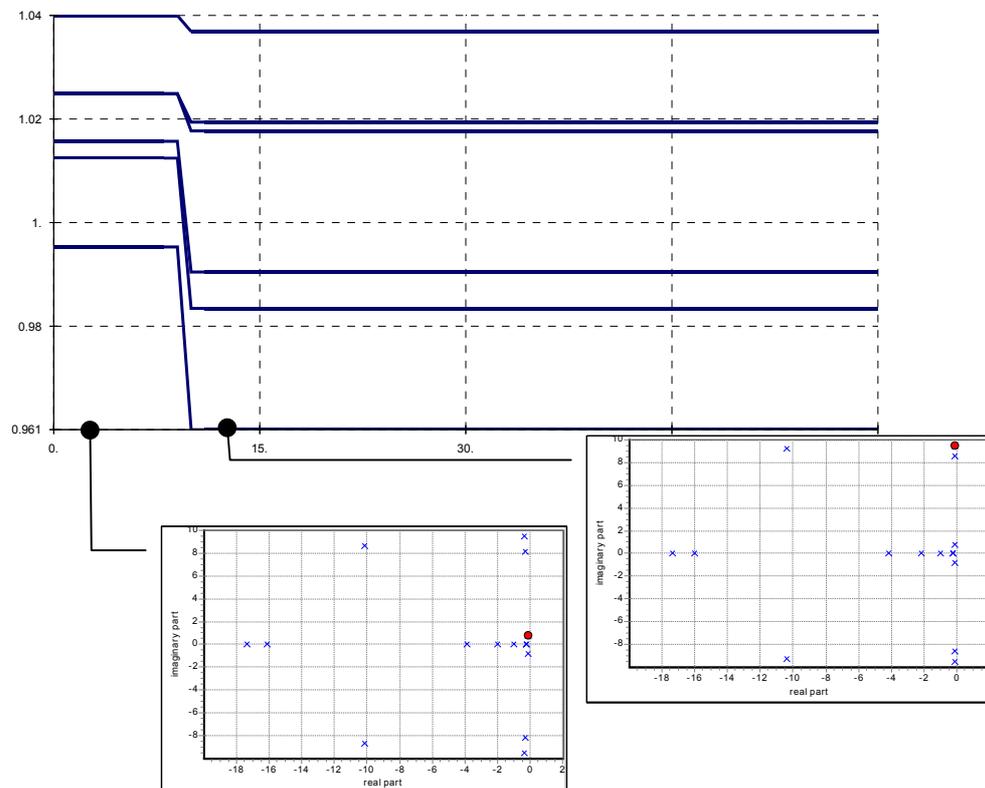


Figura 41 – Tensões e Análise Modal para um Degrau de Carga de 30%

Este mesmo distúrbio, no entanto, se simulado com uma ferramenta de simulação dinâmica completa produz as curvas de tensão e potência reativa injetada pelos geradores apresentadas na Figura 42 e Figura 43, respectivamente.

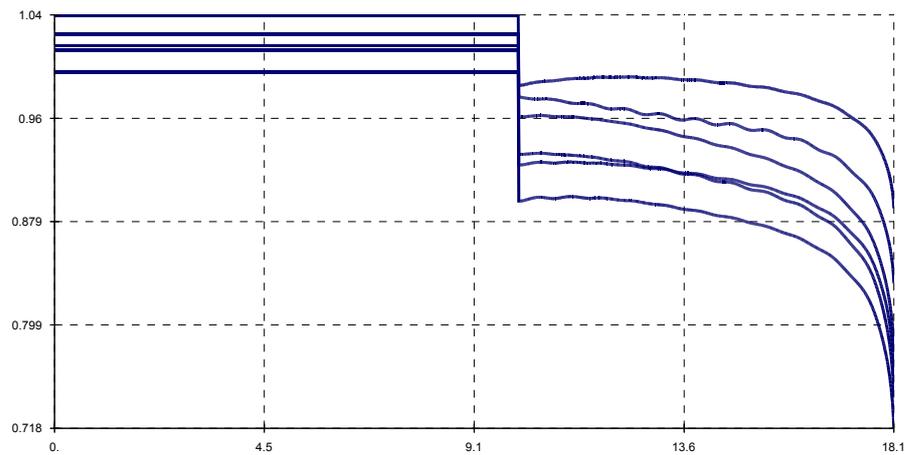


Figura 42 – Tensões nas Barras para um Degrau de Carga de 30%  
(Simulação Dinâmica Completa)

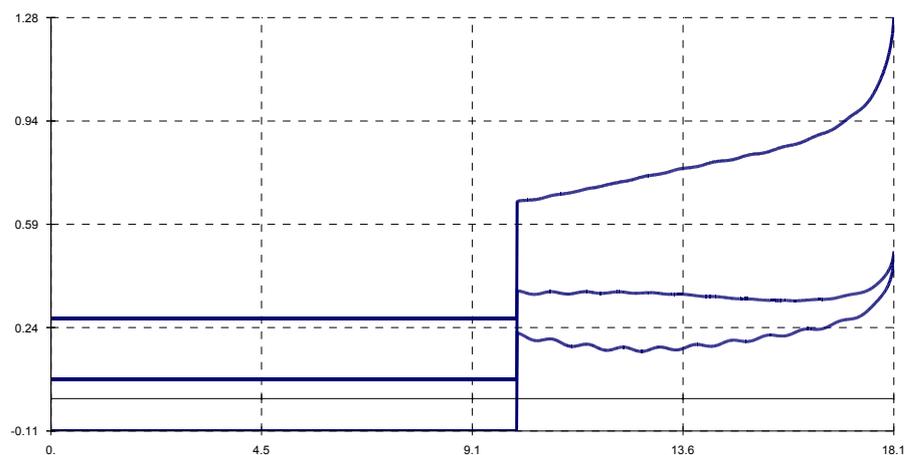


Figura 43 – Potências Reativas Geradas para um Degrau de Carga de 30%  
(Simulação Dinâmica Completa)

Os resultados mostram que, embora os dois pontos de equilíbrio sejam estáveis, a trajetória dinâmica do sistema não consegue alcançar o ponto de equilíbrio pós-distúrbio, caracterizando uma situação de instabilidade transitória.

---

## 4.6 Simulação Rápida e Completa Combinadas

Para estudos que necessitam a simulação de distúrbios de grande impacto no sistema, a simulação rápida somente poderá ser utilizada quando há a certeza de que a resposta transitória do sistema é estável, ou seja, o ponto de equilíbrio pós-distúrbio pode ser alcançado. Em situações onde não há esta certeza, é necessário analisar a aplicação do distúrbio com uma ferramenta de simulação dinâmica completa em um intervalo de tempo suficiente para a identificação da estabilidade do sistema.

Uma alternativa é combinar a metodologia de simulação rápida modificada com a de simulação dinâmica completa, de tal forma que seja possível chavear facilmente de uma para outra metodologia durante a simulação. Assim, é possível conduzir a simulação com a metodologia rápida durante o intervalo de tempo em que as dinâmicas lentas ditam a trajetória do sistema, chaveando para a simulação completa sempre que for detectado um distúrbio de grande impacto no sistema durante um intervalo de tempo suficiente para a detecção da estabilidade da trajetória (tipicamente 5 segs), quando então a metodologia pode novamente ser chaveada para a simulação rápida para o próximo intervalo de simulação.

A Figura 44 e Figura 45 mostram os resultados de uma simulação combinada para o sistema exemplo onde foi aplicado um degrau de 10% na carga da barra 5. A simulação dinâmica completa foi executada durante 5 segs a partir da aplicação do distúrbio quando então novamente foi chaveada para a simulação rápida.

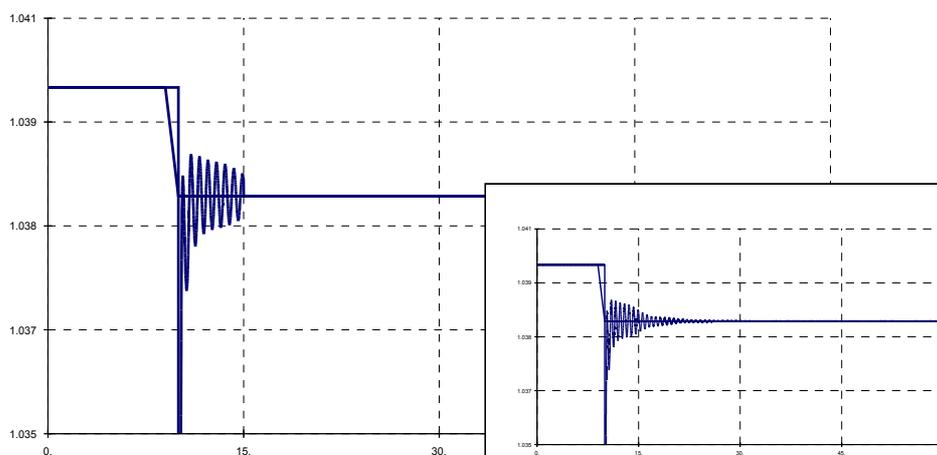


Figura 44 – Tensão Terminal do Gerador 1 para um Degrau de Carga

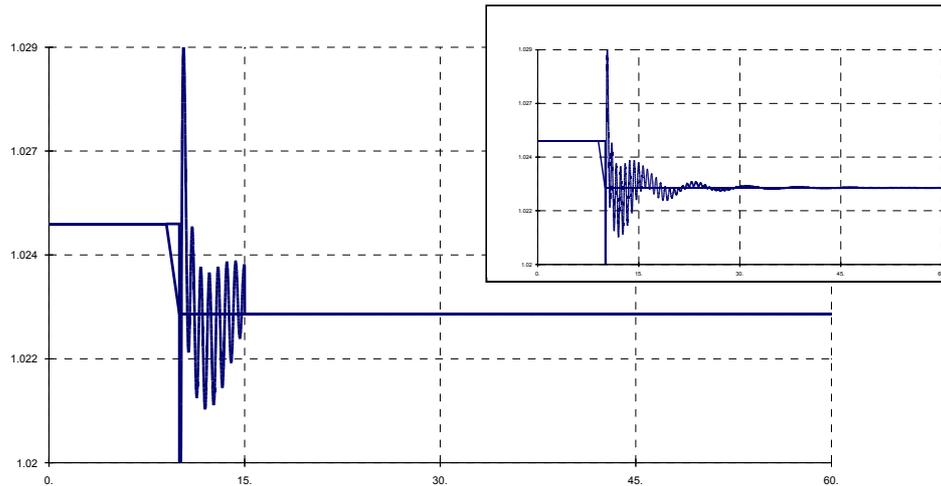


Figura 45 – Tensão Terminal do Gerador 3 para um Degrau de Carga

#### 4.6.1 Aspectos Computacionais

A representação completa das equações dos elementos dinâmicos na simulação rápida modificada torna a transição de metodologias algo simples e natural, uma vez que não há substituição de modelos mas apenas da metodologia de simulação. Isto permite que o chaveamento entre as duas metodologias possa ser executada a qualquer instante da simulação e quantas vezes for necessário.

O chaveamento entre as metodologias de simulação pode ser realizado de forma simples devido a duas características principais: a retenção da dinâmica dos elementos sem simplificações na formulação modificada, permitindo a utilização dos mesmos modelos tanto para a simulação da dinâmica completa quanto para a simulação rápida modificada; e a representação das equações dos modelos dos dispositivos como blocos construtivos elementares – ver item 3.4.2.1. Esta última característica faz com que o comportamento dos blocos dinâmicos do sistema (*Integrador*, *Derivador*, *Lag*, *Washout*, *LeadLag* e *Blc2ord*), durante o processo de solução e derivação das equações, determine qual metodologia de simulação está sendo executada. O exemplo a seguir ilustra melhor o mecanismo de chaveamento de metodologias utilizando como exemplo um o bloco dinâmico geral, mostrado na Figura 46.

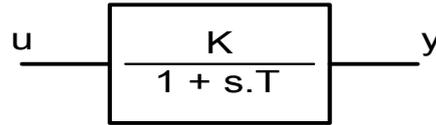


Figura 46 – Bloco Dinâmico do Modelo Orientado a Objetos (*Atraso*)

A equação diferencial que descreve o comportamento do bloco é:

$$\dot{y} = \frac{1}{T} \cdot (K \cdot u - y) \quad (43)$$

A hipótese básica assumida na metodologia da simulação rápida modificada desconsidera as dinâmicas rápidas sem realizar qualquer simplificação nas equações, fazendo com que as variações das variáveis de estado no tempo sejam instantâneas e retendo a dinâmica completa dos modelos. Para a equação acima faz-se simplesmente  $dy/dt = 0$ , e resolve-se a equação juntamente com as demais equações do sistema pelo método de *Newton*. Os coeficientes das derivadas parciais calculados durante o processo de linearização da equação 43 (coeficientes também da matriz Jacobiana do sistema) resultam em:

$$\dot{y} = 0 = \frac{K}{T} \cdot u - \frac{1}{T} \cdot y \Rightarrow f_1 \quad \begin{cases} \frac{\partial f_1}{\partial y} = -\frac{1}{T} \\ \frac{\partial f_1}{\partial u} = \frac{K}{T} \end{cases} \quad (44)$$

Na simulação da dinâmica completa a hipótese anterior não pode mais ser mantida e a equação 43 deve ser discretizada por algum método de integração numérica e resolvida juntamente com as demais equações do sistema. Aplicando o método trapezoidal implícito a equação 43 obtém-se a seguinte expressão:

$$y_{(t+\Delta t)} - y_{(t)} = \frac{\Delta t}{2} \cdot \left[ \frac{K}{T} \cdot u_{(t+\Delta t)} - \frac{1}{T} \cdot y_{(t+\Delta t)} + \frac{K}{T} \cdot u_{(t)} - \frac{1}{T} \cdot y_{(t)} \right] \quad (45)$$

Agrupando os termos semelhantes através de manipulações algébricas chega-se a expressão:

$$y_{(t+\Delta t)} = \frac{K \cdot \Delta t}{2 \cdot T + \Delta t} \cdot u_{(t+\Delta t)} + F \quad (46)$$

onde  $F$  agrupa os termos que dependem unicamente do passo anterior, sendo, portanto, constante durante o processo de cálculo do passo atual. O valor da

---

constante  $F$  é dado pela seguinte equação:

$$F = \frac{K \cdot \Delta t}{2 \cdot T + \Delta t} \cdot u_{(t)} + \frac{2 \cdot T - \Delta t}{2 \cdot T + \Delta t} \cdot y_{(t)} \quad (47)$$

A equação 46, agora discretizada pelo método trapezoidal, deve então ser resolvida juntamente com as demais equações do sistema também pelo método de *Newton* (para o esquema simultâneo implícito). Os coeficientes das derivadas parciais calculados durante o processo de linearização da equação 46 resultam em:

$$0 = \frac{\Delta t \cdot K}{2 \cdot T + \Delta t} \cdot u_{(t+\Delta t)} - y_{(t+\Delta t)} + F \Rightarrow f_2 \quad \begin{cases} \frac{\partial f_2}{\partial y} = -1 \\ \frac{\partial f_2}{\partial u} = \frac{K \cdot \Delta t}{2 \cdot T + \Delta t} \end{cases} \quad (48)$$

Tanto para o método de simulação rápida modificada (equação 44) como para o método da simulação da dinâmica completa (equação 48), a solução da equação que descreve o comportamento do bloco dinâmico exemplo resultam em dois coeficientes ( $\partial f_{1,2}/\partial y$  e  $\partial f_{1,2}/\partial u$ ), ambos advindos do processo de linearização da equação para a montagem da matriz Jacobiana do sistema. O mecanismo de derivação automática dos blocos dinâmicos deve selecionar os coeficientes adequados à metodologia de simulação utilizada. Para o chaveamento de metodologias basta então que cada bloco dinâmico dos modelos troque os coeficientes durante o processo de cálculo automático das derivadas do modelo, assim a matriz Jacobiana montada será coerente com a metodologia de simulação requerida.

## 4.7 Considerações Finais

A metodologia de simulação rápida modificada mostrou-se adequada para os estudos de estabilidade de tensão de médio e longo prazo. A representação sem simplificações para os modelos dinâmicos permite que uma análise de bifurcações locais possa ser conduzida durante a simulação, através do cálculo dos autovalores do sistema para cada ponto de equilíbrio obtido. Este processo pode deteriorar o desempenho do método para sistemas de médio e grande porte, porém este problema pode ser contornado através do cálculo dos autovalores a intervalos maiores que o passo de simulação e a utilização de técnicas de cálculo parcial de autovalores.

---

A simulação rápida modificada e a simulação combinada podem ser utilizadas para expandir a faixa de fenômenos que a simulação rápida pode investigar. A última permite que distúrbios de grande impacto no sistema sejam simulados sem o risco dos resultados conduzirem a conclusões incorretas sobre o comportamento do sistema. Entretanto alguns aspectos da simulação combinada ainda precisam ser investigados, sobretudo os que levam a determinação de: “Quando um distúrbio é grande o suficiente para exigir o chaveamento para a simulação completa ?”, ou, “Como detectar de forma automática se a trajetória da simulação completa é estável, possibilitando o retorno para a simulação rápida, ou instável ?”. Estas e outras interrogações mostram a necessidade de incorporar na simulação combinada uma metodologia de análise automática da estabilidade transitória [54][55][56][57], determinando assim se a trajetória é estável ou instável. Uma alternativa é a utilização de métodos baseados nos conceitos de Energia Potencial Generalizada [58][59]. No entanto, este assunto não será abordado neste trabalho.

É conveniente salientar que embora a simulação rápida modificada e a simulação combinada possam ser utilizadas para estudos de uma ampla faixa de fenômenos em estabilidade de tensão, a simulação dinâmica completa ainda é ferramenta indispensável nestes estudos, sendo as metodologias complementares. De fato, quando há a necessidade de um estudo detalhado do comportamento do sistema, em situações específicas de colapso de tensão, ou para a investigação de como será atingido um ponto de equilíbrio de regime permanente a simulação dinâmica completa é a mais indicada.

---

## Capítulo 5

### Resultados

#### 5.1 Considerações Gerais

A estrutura computacional orientada a objetos proposta no Capítulo 3 foi utilizada para a implementação de um *Framework* para Análise de Sistemas de Energia Elétrica, denominado *FASEE*. Este *framework* foi desenvolvido em C++ e constitui a estrutura computacional base sobre a qual foram implementados vários aplicativos para sistemas de energia elétrica. Todos os aplicativos implementados estão integrados em um mesmo ambiente computacional, e atuam sobre uma base de dados única.

Neste capítulo são apresentados alguns resultados obtidos com os aplicativos do ambiente computacional implementado. Estes resultados são utilizados para a validação e avaliação do desempenho da estrutura orientada a objetos proposta, e procuram ilustrar a capacidade e as potencialidades do ambiente de desenvolvimento implementado.

O capítulo está dividido em duas partes principais, a primeira relativa a validação dos resultados dos aplicativos implementados com programas comerciais da área. São também apresentados resultados obtidos com a metodologia de simulação rápida modificada, descrita no Capítulo 4, para as simulações de longo prazo. Por fim, são apresentados alguns exemplos que demonstram a capacidade da estrutura proposta quanto a flexibilidade de formulação dos aplicativos e quanto a facilidade de inclusão de novos modelos e/ou dispositivos no ambiente.

---

Na segunda parte deste capítulo é realizada, sob diversas formas, uma avaliação do desempenho computacional dos aplicativos implementados. São apresentados estudos de desempenho computacional para os programas de fluxo de potência e simulação da dinâmica completa para vários sistemas de diferentes tamanhos, sendo os resultados comparados a programas escritos em linguagens tradicionais. Nesta parte do capítulo é também apresentado uma simulação de longo prazo para o sistema interligado Brasileiro utilizando os modelos reais deste sistema.

## 5.2 Base de Aplicativos Implementada

O conjunto de aplicativos atualmente implementado está integrado em um ambiente computacional de simulação e análise de sistemas elétricos. Este ambiente é suportado por uma base de dados única que alimenta a estrutura computacional com os dados e modelos conforme a necessidade do aplicativo. Desta forma, uma única base de dados deve ser gerenciada, cabendo ao aplicativo específico buscar na base de dados as informações e modelos mais adequados a sua execução. O ambiente descrito conta com os seguintes aplicativos:

- **Fluxo de Potência – FLXPOT++:** o ambiente de estudo conta com aplicativos para solução das equações do fluxo de potência pelos métodos de Newton-Raphson (coordenadas polares e retangulares) e Newton-Raphson Desacoplado Rápido, em suas formulações clássicas [62]. Uma versão generalizada do algoritmo de Newton, que utiliza a metodologia de derivação automática descrita no item 3.4.2.2 - Mecanismo de Solução e Derivação das Equações, resolve todas as equações do fluxo de potência simultaneamente, incluindo as equações dos controles. Esta versão permite representar qualquer modelo de dispositivo definido pelo usuário, usando suas formulações de injeção de potência ou corrente, e coordenadas polares ou retangulares para as tensões;
- **Análise de Estabilidade de Pequenos-Sinais – AEPS++:** um programa de análise de estabilidade de pequenos-sinais pode também ser executado no ambiente de análise. Este programa calcula autovalores e autovetores para o sistema (cálculo completo apenas), fatores de participação e *mode-shape* para cada um dos autovalores do sistema dinâmico, apresentando estes resultados em uma interface gráfica amigável;

- 
- **Simulação da Dinâmica Completa – SIMSEE++:** um programa de simulação da dinâmica completa do sistema no tempo está também disponível no ambiente de simulação implementado. São possíveis simulações pelos métodos simultâneo implícito (*SIMSEEs++*) e alternado implícito (*SIMSEEs++*)[47], sendo que ambos programas utilizam o método trapezoidal implícito para algebrização das equações diferenciais do sistema algébrico-diferencial. Os resultados das simulações frente a vários tipos de eventos e distúrbios podem ser visualizados graficamente na tela na forma de curvas plotadas para as grandezas monitoradas;
  - **Simulação Rápida – FASTSIM++:** o ambiente conta ainda com um programa de simulação rápida modificada, conforme descrito no Capítulo 4. Este programa pode ser utilizado nas simulações em que a dinâmica rápida do sistema pode ser desprezada sendo a sua trajetória conduzida preferencialmente por fenômenos de natureza lenta e de longa duração. Uma versão modificada deste programa que combina a simulação rápida e a simulação da dinâmica completa também esta disponível no ambiente de simulação implementado, conforme metodologia proposta no item 4.6 - Simulação Rápida e Completa Combinadas.

Todos os programas descritos acima utilizam a metodologia de derivação automática descrita no item 3.4.2.2 - Mecanismo de Solução e Derivação das Equações, com exceção dos aplicativos de fluxo de potência pelas formulações clássicas, que implementam o cálculo das derivadas parciais em código, e o aplicativo de simulação da dinâmica completa pelo método alternado implícito, que não necessita de derivadas parciais.

### 5.3 Simulações e Avaliação da Flexibilidade do M.O.O.

Para ilustração da capacidade e potencial do ambiente implementado na realização de simulações e análises típicas para sistemas de energia elétrica serão apresentados alguns resultados de estudos realizados com os programas disponíveis no ambiente implementado. Comparações com os resultados de programas comerciais da área são apresentados para validação dos aplicativos implementados. Todos os eventos e situações foram definidos sobre um sistema de pequeno porte para melhor visualização dos efeitos representados e interpretação dos resultados.

### 5.3.1 Sistema Exemplo

O sistema tomado como exemplo para validação do modelo orientado a objetos proposto é um equivalente do sistema elétrico interligado da região Sul do Brasil, onde foram representadas algumas áreas por equivalentes de rede. Este sistema é composto por 45 barras, 56 linhas de transmissão, 17 transformadores e 10 geradores, conforme mostra a Figura 47. O sistema possui ainda dois compensadores síncronos, mas estes elementos não são representados nas simulações realizadas.

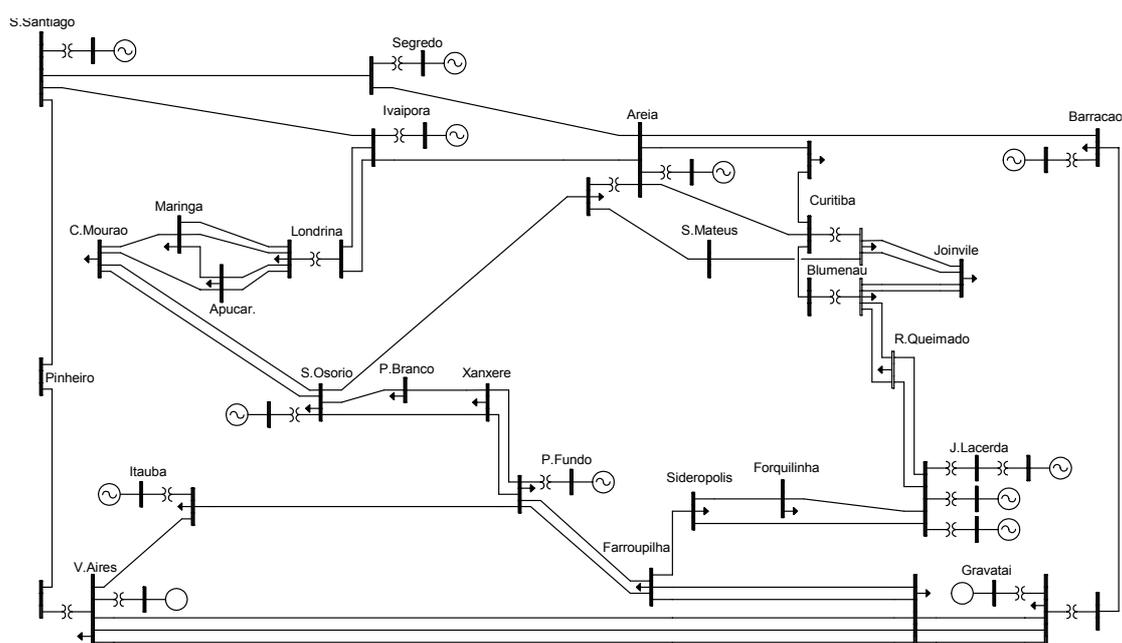


Figura 47 – Sistema Exemplo de 45 barras

A carga base total do sistema é de 6473.1 MW / 942.1 MVAR. Ainda que este sistema seja considerado de pequeno porte, serve perfeitamente aos propósitos de demonstrar as potencialidades do modelo orientado a objetos proposto.

### 5.3.2 Fluxo de Potência

Os resultados de fluxo de potência que seguem foram obtidos com o aplicativo de fluxo de potência generalizado – *FLXPOT++*. Este aplicativo monta e resolve automaticamente as equações do fluxo de potência considerando dispositivos definidos pelo usuário. Além disso, permite selecionar a formulação adotada quanto ao tipo de injeção dos dispositivos na rede elétrica (injeção de potência ou injeção de corrente) e quanto ao tipo de coordenadas para as tensões (coordenadas polares ou

coordenadas retangulares), sendo capaz de mudar a formulação do problema ainda em tempo de execução. Os resultados de tensões para todas as barras do sistema e de potência ativa e reativa para as barras de geração são mostrados, respectivamente, na Figura 48 e Figura 49. Estes resultados foram obtidos utilizando-se a formulação de injeção de potência e tensões em coordenadas polares. Todas as cargas foram representadas pelo modelo de potência constante.

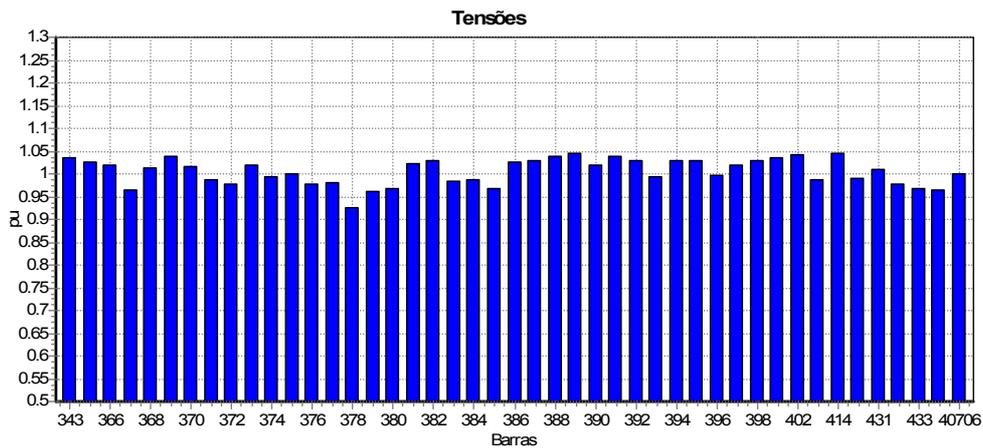


Figura 48 – Tensões das Barras (*FLXPOT++*)

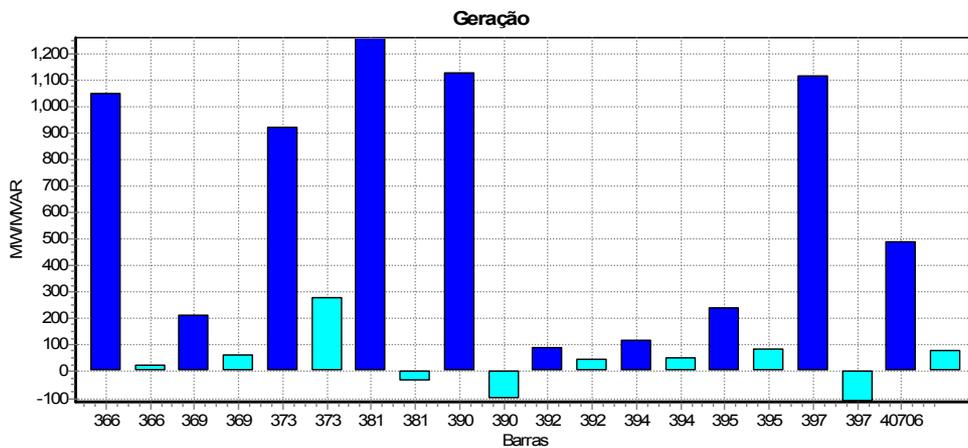


Figura 49 – Potência Ativa e Reativa dos Geradores (*FLXPOT++*)

Os resultados mostrados na Figura 48 e na Figura 49 são relativos ao caso base de fluxo de potência utilizado para todas as demais simulações apresentadas a seguir.

### 5.3.2.1 Comparação dos Resultados com o Programa ANAREDE

Com a finalidade de validar o programa de fluxo de potência os resultados foram comparados com os obtidos através do Programa de Análise de Redes do CEPEL – ANAREDE. Os resultados mostrados na Figura 50 e na Figura 51 correspondem, respectivamente, a diferenças encontradas nas tensões calculadas por ambos programas e nas potências ativa e reativa dos geradores, onde observa-se uma diferença máxima de  $6.0E-4$  nas tensões e de  $9.0E-3$  nas potências geradas.

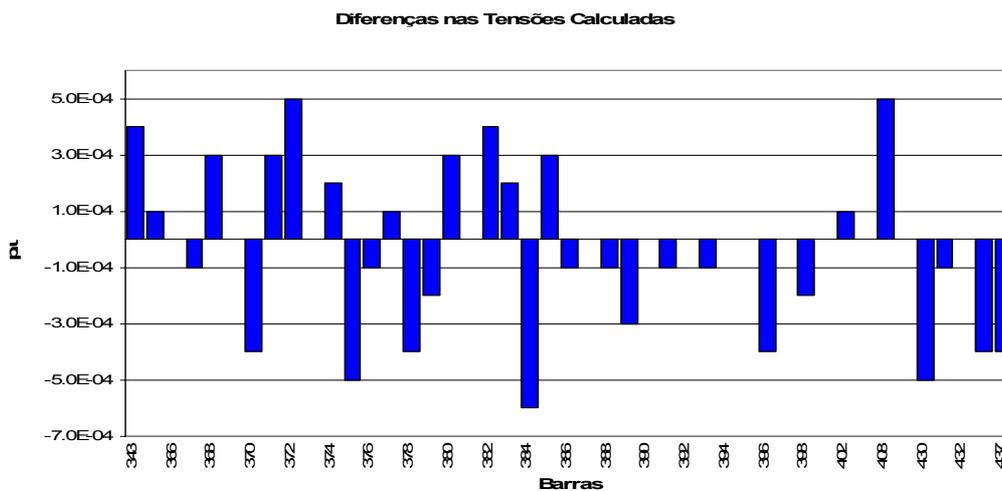


Figura 50 – Diferença nas Tensões entre os Programas *FLXPOT++* e ANAREDE

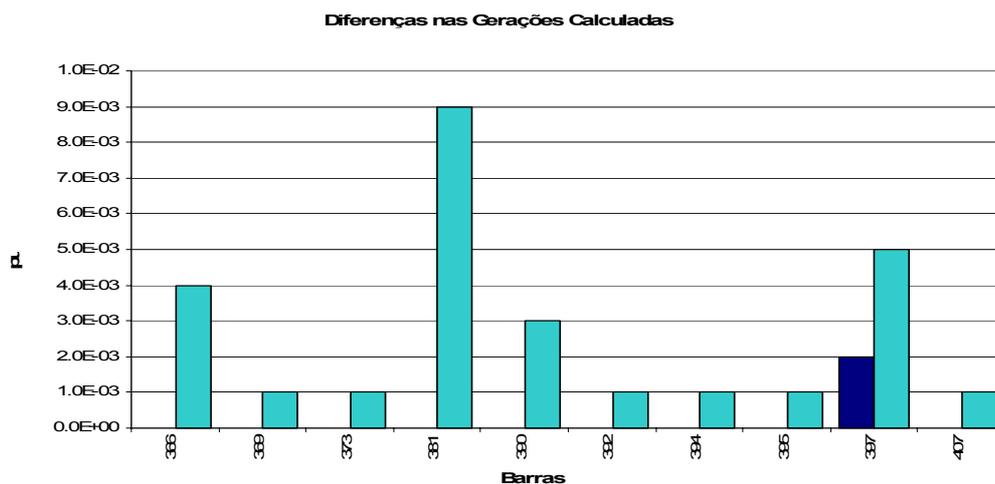


Figura 51 – Diferença nas Gerações entre os Programas *FLXPOT++* e ANAREDE

### 5.3.3 Análise de Estabilidade de Pequenos-Sinais

O aplicativo de análise de estabilidade de pequenos-sinais fornece a matriz de estados do sistema, os autovalores e autovetores desta matriz, bem como fatores de participação e *mode-shape* para qualquer autovalor selecionado. Este aplicativo conta com as mesmas características do programa de fluxo de potência generalizado quanto a sua capacidade de mudança na formulação e de inclusão automática de novos modelos ou dispositivos.

Na Figura 52 são mostrados os autovalores do sistema exemplo plotados no plano complexo (apenas os autovalores mais próximos ao eixo imaginário do plano complexo foram plotados). A figura mostra também os fatores de participação principais de um autovalor selecionado (visualizados através de um histograma de participação) e os correspondentes *mode-shape* de velocidade associados a este autovalor.

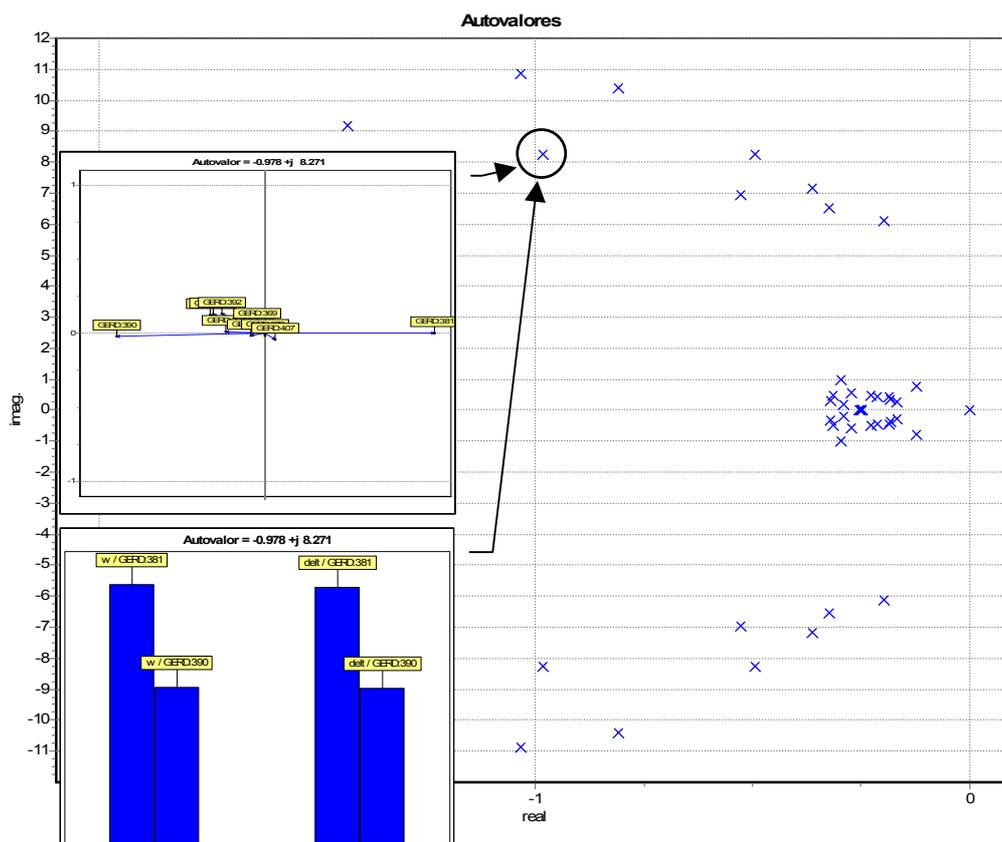


Figura 52 – Autovalores e Fatores de Participação / Mode Shape para um Autovalor Selecionado (AEPS++)

O modelo dinâmico utilizado representa as máquinas síncronas através de um modelo de 5<sup>ª</sup> ordem com efeitos transitórios e subtransitórios representados. Os reguladores de tensão e de velocidade foram representados por modelos simplificados de 2<sup>ª</sup> ordem, e as cargas através de seus modelos de impedância constante.

### 5.3.3.1 Comparação dos Resultados com o Programa PACDYN

Com a finalidade de validar o programa de análise de estabilidade de pequenos-sinais os resultados foram comparados com os obtidos através do Programa de Análise de Estabilidade de Pequenos-Sinais do CEPEL – PACDYN, sendo os autovalores calculados por ambos programas apresentados a seguir:

No.	PACDYN		AEPS++	
	Real	Imaginário	Real	Imaginário
1	-354.4320		-354.432	
2	-354.4190		-354.418	
3	-354.4088		-354.408	
4	-354.4047		-354.404	
5	-354.4001		-354.400	
6	-354.3961		-354.396	
7	-354.3931		-354.393	
8	-354.3906		-354.390	
9	-354.3886		-354.388	
10	-354.3880		-354.387	
11	-120.4879		-120.500	
12	-100.0054		-100.008	
13	-100.0036		-100.007	
14	-100.0022		-100.006	
15	-100.0012		-100.006	
16	-100.0010		-100.006	
17	-100.0006		-100.006	
18	-100.0004		-100.006	
19	-100.0004		-100.006	
20	-100.0003		-100.005	
21	-100.0003		-100.001	
22	-99.01880		-99.0589	
23	-58.41376		-58.4278	
24	-47.75062		-47.7510	
25	-39.18521		-39.1944	
26	-38.64028		-38.6410	
27	-36.51148		-36.5119	
28	-33.31290		-33.3137	
29	-29.79459		-29.7950	
30	-29.08141		-29.0843	
31	-25.38615		-25.3902	
32	-20.18562		-20.1954	
33	-18.45029		-18.4539	
34	-17.95156		-17.9572	
35	-16.14729		-16.1525	
36	-14.96566		-14.9622	
37	-13.57413		-13.5758	
38	-12.51192		-12.5203	
39	-1.037704	±j 10.88265	-1.03689	±j 10.88920
41	-.8106690	±j 10.45940	-0.80765	±j 10.47802
43	-10.36106		-10.3616	
44	-1.434989	±j 9.168389	-1.42869	±j 9.202729
46	-9.056631		-9.05734	
47	-.9812848	±j 8.233150	-0.97771	±j 8.271292
49	-.4932066	±j 8.236013	-0.49188	±j 8.260576
51	-.3602380	±j 7.133622	-0.35783	±j 7.167945
53	-.5279705	±j 6.927608	-0.52541	±j 6.971739
55	-.3288478	±j 6.494136	-0.32640	±j 6.532770

---

57	-.1960263	±j	6.062183	-0.19458	±j	6.109363
59	-.2840350	±j	0.9911228	-0.28323	±j	0.990794
61	-.2684090	±j	0.5552425	-0.26978	±j	0.554517
63	-.3139665	±j	0.4823697	-0.31426	±j	0.482092
65	-.2243066	±j	0.5047911	-0.22408	±j	0.504004
67	-.1788216	±j	0.4559459	-0.17916	±j	0.455484
69	-.2085646	±j	0.4370201	-0.20867	±j	0.436562
71	-.3223018	±j	0.3025792	-0.32278	±j	0.301490
73	-.1808788	±j	0.3436238	-0.18100	±j	0.343282
75	-.2930363	±j	0.2062649	-0.29300	±j	0.203130
77	-.1660255	±j	0.2915048	-0.16608	±j	0.291227
79	-.1239749	±j	0.2381705	-0.12187	±j	0.786790
81	-.2516305			-0.25318		
82	-.2498809			-0.24953		
83	-.2498630			-0.24818		
84	-.2498208			-0.24800		
85	-.2497282			-0.24785		
86	-.2496798			-0.24701		
87	-.2494717			-0.24675		
88	-.2492826			-0.24626		
89	-.2483445			-0.24619		
90	-.0003917			-0.00005		

Pequenas diferenças podem ser observadas nos autovalores calculados pelos programas, entretanto estas discrepâncias podem ser atribuídas a precisão do ponto de operação fornecido pelo fluxo de potência (observar a diferença encontrada no autovalor “nulo” do sistema – linha sombreada).

### 5.3.4 Simulação da Dinâmica Completa

O ambiente computacional orientado a objetos conta com dois algoritmos de simulação no tempo da dinâmica completa do sistema: **algoritmo simultâneo implícito** e **algoritmo alternado implícito** [47], sendo possível o chaveamento de metodologias a qualquer instante do processo de simulação.

A seguir serão mostrados resultados referentes a aplicação de um curto-circuito trifásico na barra de *Itaúba* do sistema exemplo em 1.0 seg., e sua posterior remoção 100 msecs após a aplicação. O modelo dinâmico utilizado representa as máquinas síncronas através de um modelo de 5<sup>o</sup> ordem com efeitos transitórios e subtransitórios. Os reguladores de tensão e de velocidade foram representados por modelos simplificados de 2<sup>o</sup> ordem, e as cargas através do seu modelo de impedância constante. Os itens 5.3.4.1 e 5.3.4.2 mostram, respectivamente, os resultados das simulações obtidas com os métodos simultâneo implícito (programa *SIMSEEs++*) e alternado implícito (programa *SIMSEEA++*).

### 5.3.4.1 Método Simultâneo Implícito – SIMSEEs++

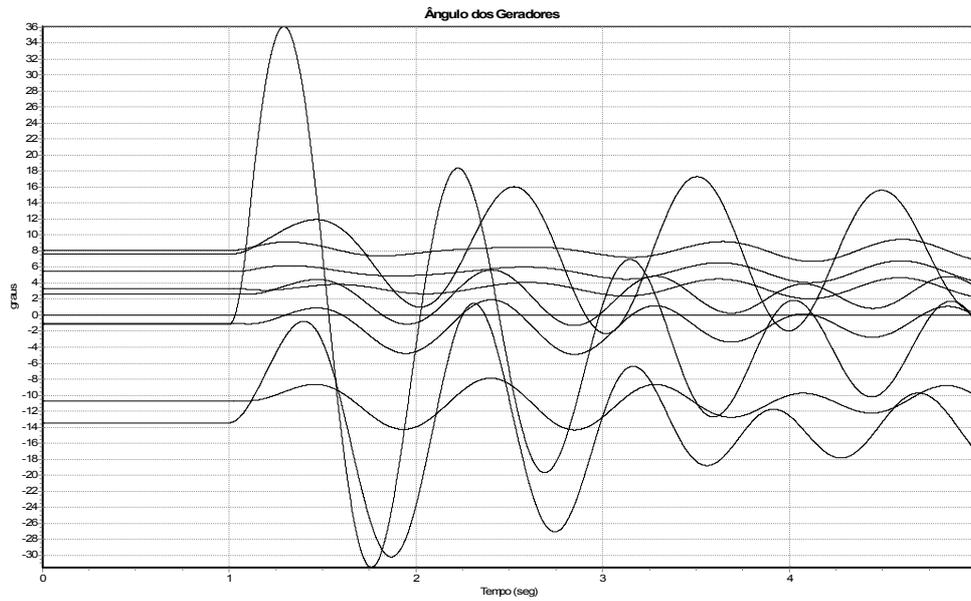


Figura 53 – Ângulo dos Geradores (SIMSEEs++)

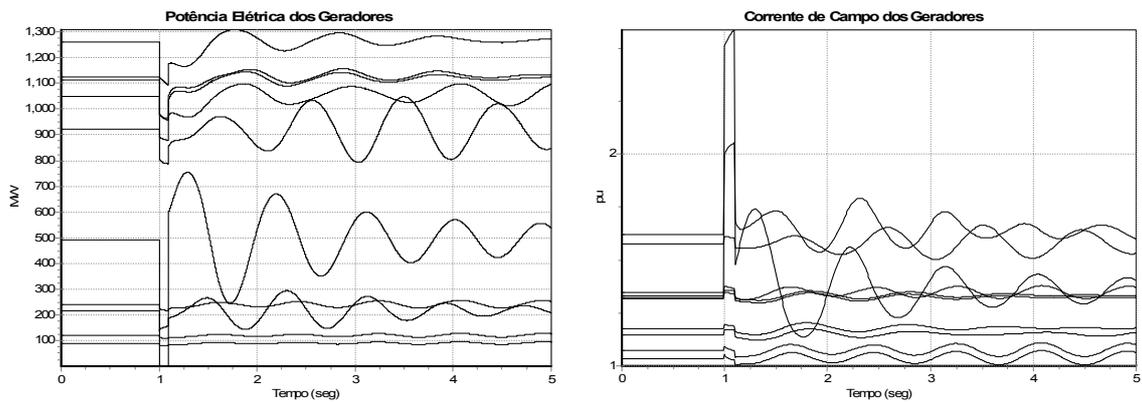


Figura 54 – Potência Elétrica e Corrente de Campo dos Geradores (SIMSEEs++)

### 5.3.4.2 Método Alternado Implícito – SIMSEEa++

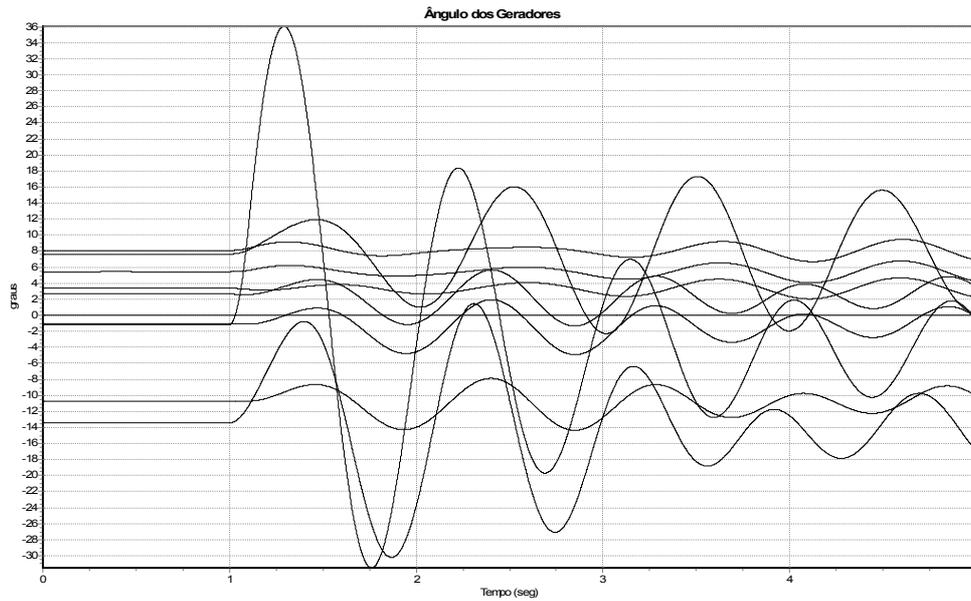


Figura 55 – Ângulo dos Geradores (SIMSEEa++)

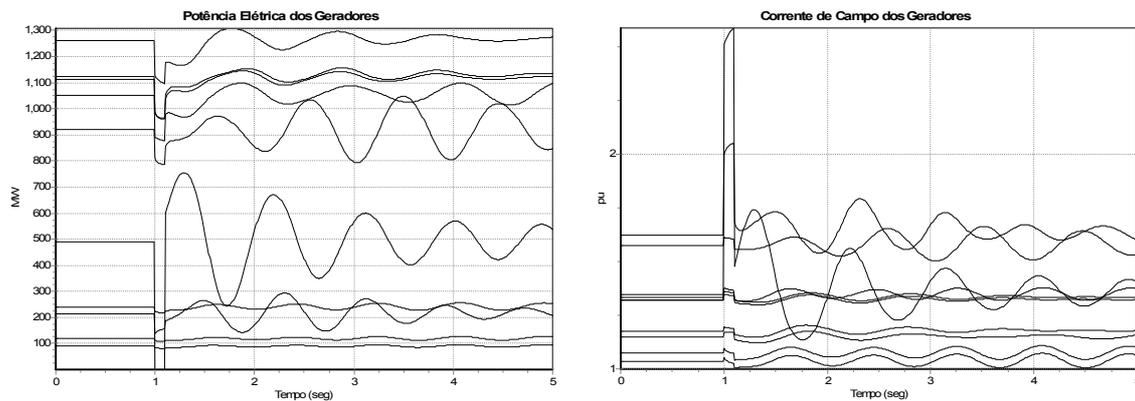


Figura 56 – Potência Elétrica e Corrente de Campo dos Geradores (SIMSEEa++)

### 5.3.4.3 Comparação dos Resultados com o Programa ANATEM

Com a finalidade de validar os programas de análise de estabilidade transitória os resultados foram comparados com os obtidos através do Programa de Análise de Transitórios Eletromecânicos do CEPEL – ANATEM. Por questões de espaço apenas serão mostradas resultados de um dos geradores do sistema, escolhido aleatoriamente.

---

A Figura 57 e Figura 58 mostram, respectivamente, o ângulo e a tensão de campo do gerador da usina de *Itaúba* obtidos com os três programas (*SIMSEEs++*, *SIMSEEs++* e *ANATEM*) e plotados em um único gráfico, onde observa-se que os resultados encontrados são rigorosamente idênticos.

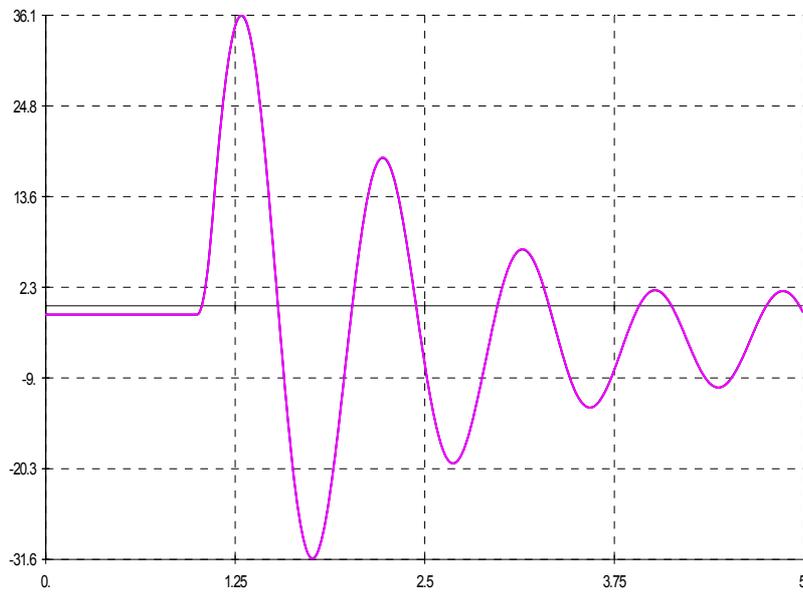


Figura 57 – Ângulo do Gerador da Usina de Itaúba  
(*SIMSEEs++*/*SIMSEEs++*/*ANATEM*)

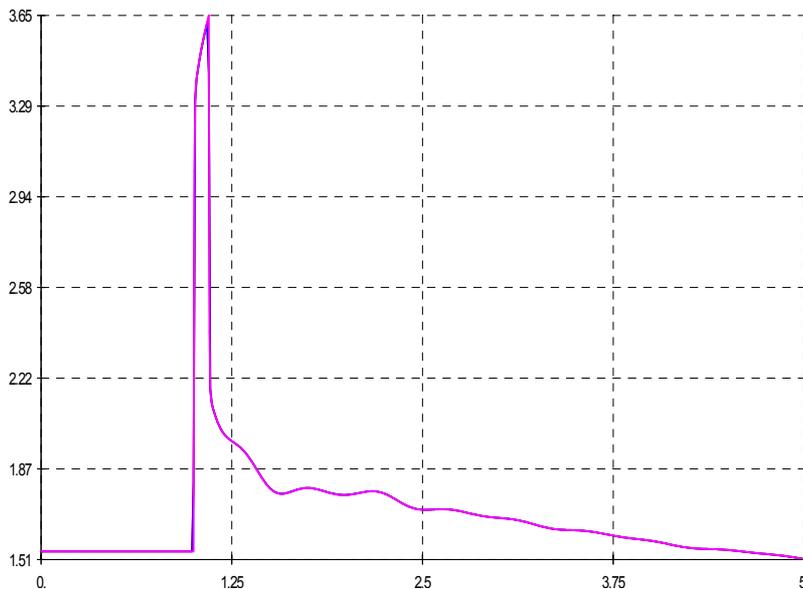


Figura 58 – Tensão de Campo do Gerador da Usina de Itaúba  
(*SIMSEEs++*/*SIMSEEs++*/*ANATEM*)

---

### 5.3.5 Simulação Rápida

Quando as dinâmicas rápidas não tem efeito significativo nos fenômenos em estudo o método de simulação rápida torna-se atrativo em função da sua precisão e bom desempenho computacional. O ambiente computacional implementado conta com um programa de simulação rápida modificada (*FASTSIM++*), conforme metodologia descrita no Capítulo 4, cujos resultados serão mostrados a seguir.

A Figura 59, Figura 60 e Figura 61 mostram os resultados de uma simulação rápida conduzida para o sistema exemplo, onde um acréscimo de 20% de carga em rampa (ativa e reativa) foi aplicado ao sistema durante 60 segs. As máquinas síncronas foram representadas através de um modelo de 5<sup>o</sup> ordem com efeitos transitórios e subtransitórios representados. Os reguladores de tensão e de velocidade foram representados por modelos simplificados de 2<sup>o</sup> ordem, e as cargas através do seus modelos de potência constante. Os limitadores dos reguladores de tensão e velocidade dos geradores não foram considerados nesta simulação (limites abertos), no entanto o programa é capaz de considerar o efeito de tais limitadores.

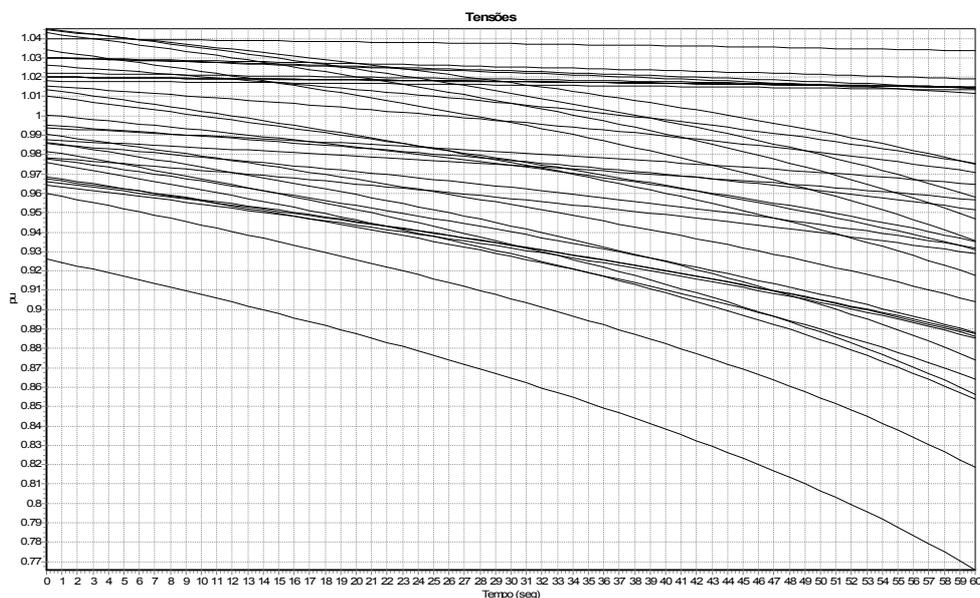


Figura 59 – Tensões das Barras (*FASTSIM++*)

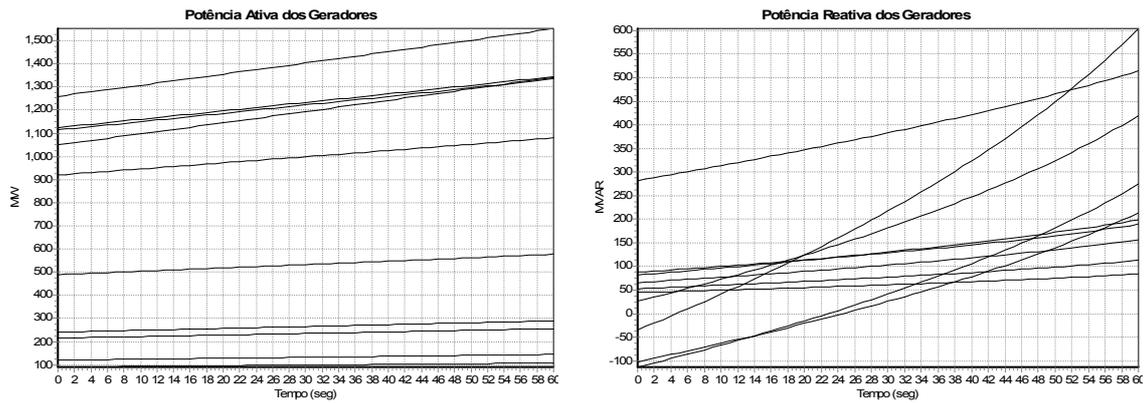


Figura 60 – Potência Ativa e Reativa dos Geradores (*FASTSIM++*)

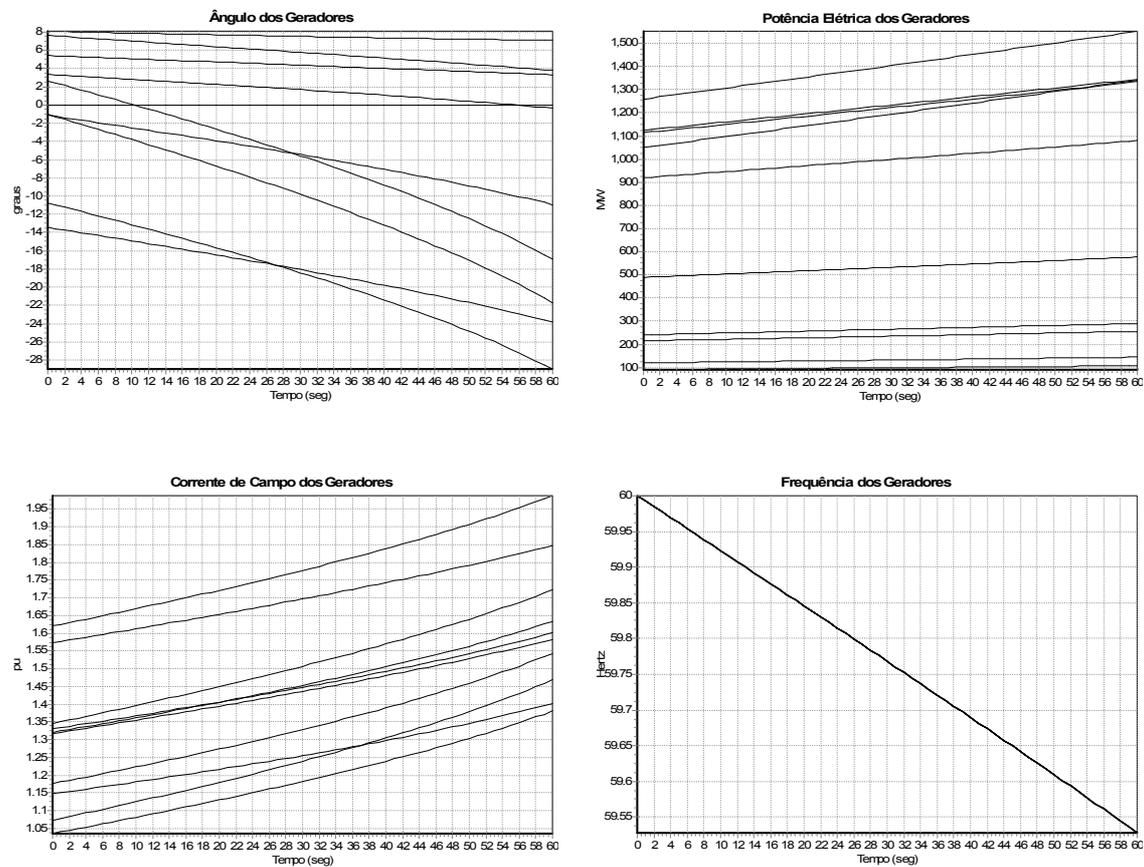


Figura 61 – Outras Grandezas Elétricas dos Geradores (*FASTSIM++*)

A retenção da dinâmica do sistema na metodologia de simulação rápida modificada (ver item 4.4 - Simulação Rápida Modificada) permite o cálculo dos autovalores do sistema ao longo da simulação. Isto permite a detecção e o estudo de uma faixa mais ampla de fenômenos que podem ocorrer no sistema, como, por

exemplo, o estudo de bifurcações locais que podem sinalizar instabilidades no sistema. A Figura 62 mostra a trajetória dos autovalores do sistema para todos os instantes de tempo da rampa de carga aplicada.

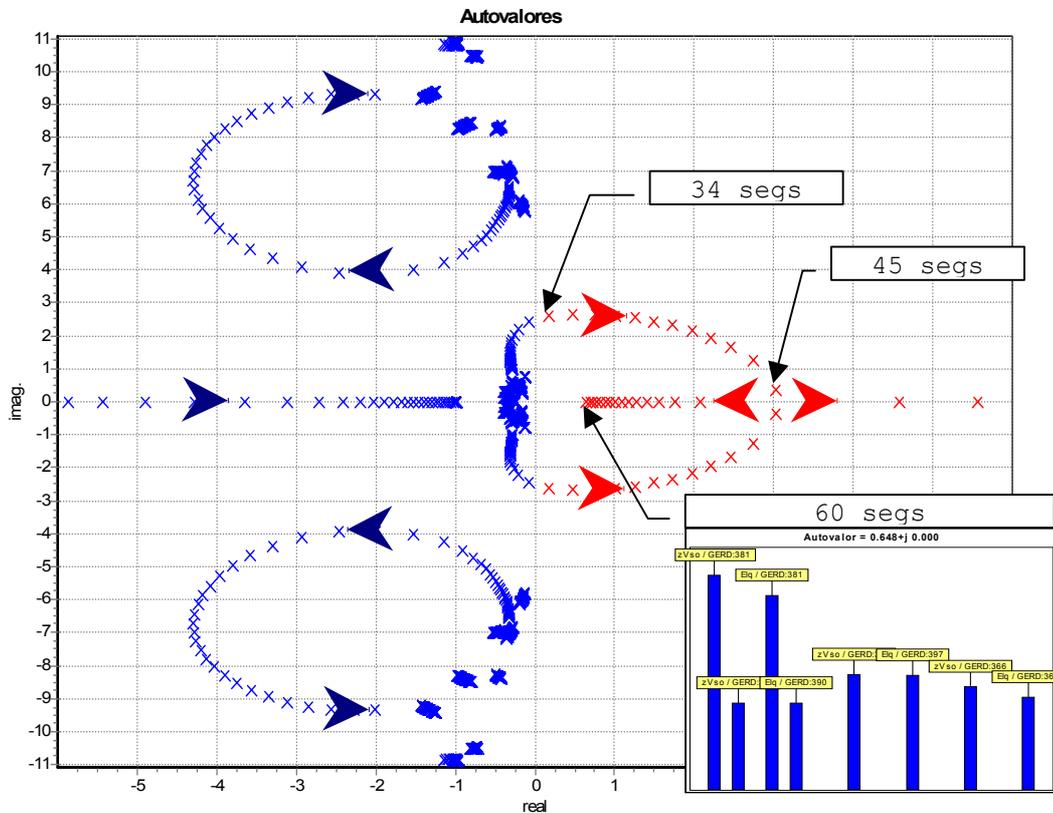


Figura 62 – Trajetória dos Autovalores para uma Rampa de Carga (FASTSIM++)

A trajetória dos autovalores da Figura 62 mostra a ocorrência de uma bifurcação de *Hopf* no sistema em 34 segs (causada pelo cruzamento de um par de autovalores complexos conjugados para o semiplano direito do plano complexo), o que torna o sistema instável a partir deste instante. Este tipo de instabilidade não pode ser detectado pelo método tradicional de simulação rápida [42][43][38].

Os resultados mostram também que, mantendo-se a atual taxa de crescimento da carga um dos autovalores do sistema alcançará a origem, caracterizando uma bifurcação do tipo *Sela-Nó* e conseqüentemente uma singularidade na Matriz de Estados do sistema. Os fatores de participação calculados para este autovalor e mostrados no detalhe da Figura 62 indicam claramente que esta instabilidade é de natureza de tensão (a figura mostra ainda as variáveis de estado correspondentes aos fatores de participação plotados).

---

### 5.3.6 Simulações de Longo Prazo

A metodologia de simulação rápida é particularmente adequada em estudos de longo prazo, nestes estudos os fenômenos de interesse ocorrem após vários minutos de simulação ou mesmo após várias horas, como, por exemplo, estudos de ajuste de CAG (Controle Automático da Geração). Nestes estudos, normalmente os fenômenos rápidos não são o principal interesse, o que torna a metodologia de simulação rápida bastante atrativa. Para demonstrar as potencialidades da ferramenta de simulação rápida em estudos de longo prazo será aplicada uma curva de carga diária típica ao sistema exemplo adotado e realizada uma simulação de 24 horas. A curva diária de carga simulada, com a metodologia da simulação rápida, será comparada com a obtida através da simulação da dinâmica completa do sistema para fins de validação dos resultados. Em ambos casos as máquinas síncronas serão representadas através de um modelo de 5<sup>o</sup> ordem com efeitos transitórios e subtransitórios representados. Os reguladores de tensão e de velocidade representados por modelos simplificados de 2<sup>o</sup> ordem, e as cargas através do seus modelo de impedância constante.

#### 5.3.6.1 Simulação Rápida

Os resultados obtidos com a metodologia da simulação rápida são apresentados na Figura 63, Figura 64 e Figura 65.

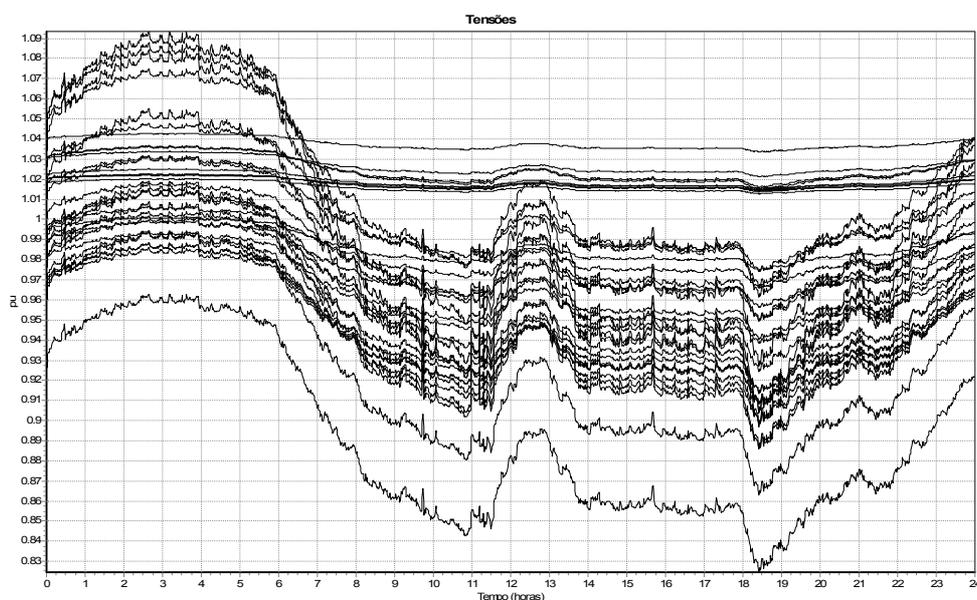


Figura 63 – Tensões das Barras (*FASTSIM++*)

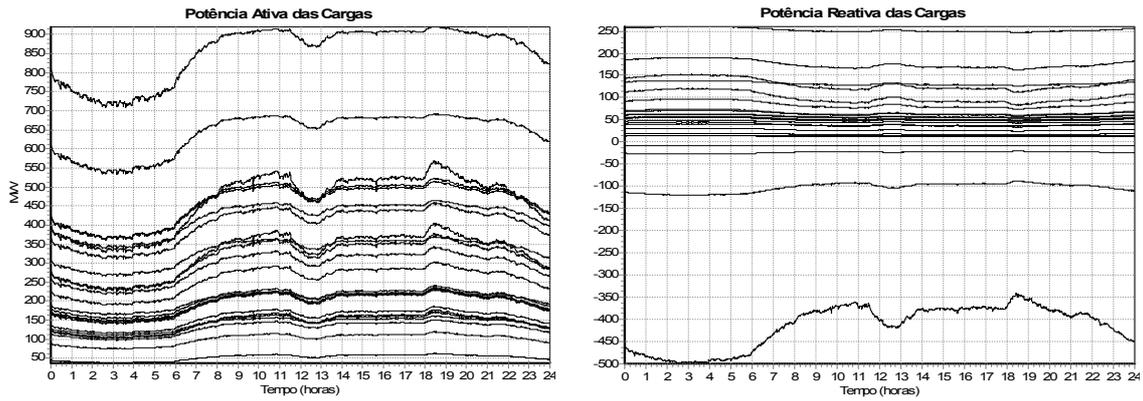


Figura 64 – Potência Ativa e Reativa das Cargas (*FASTSIM++*)

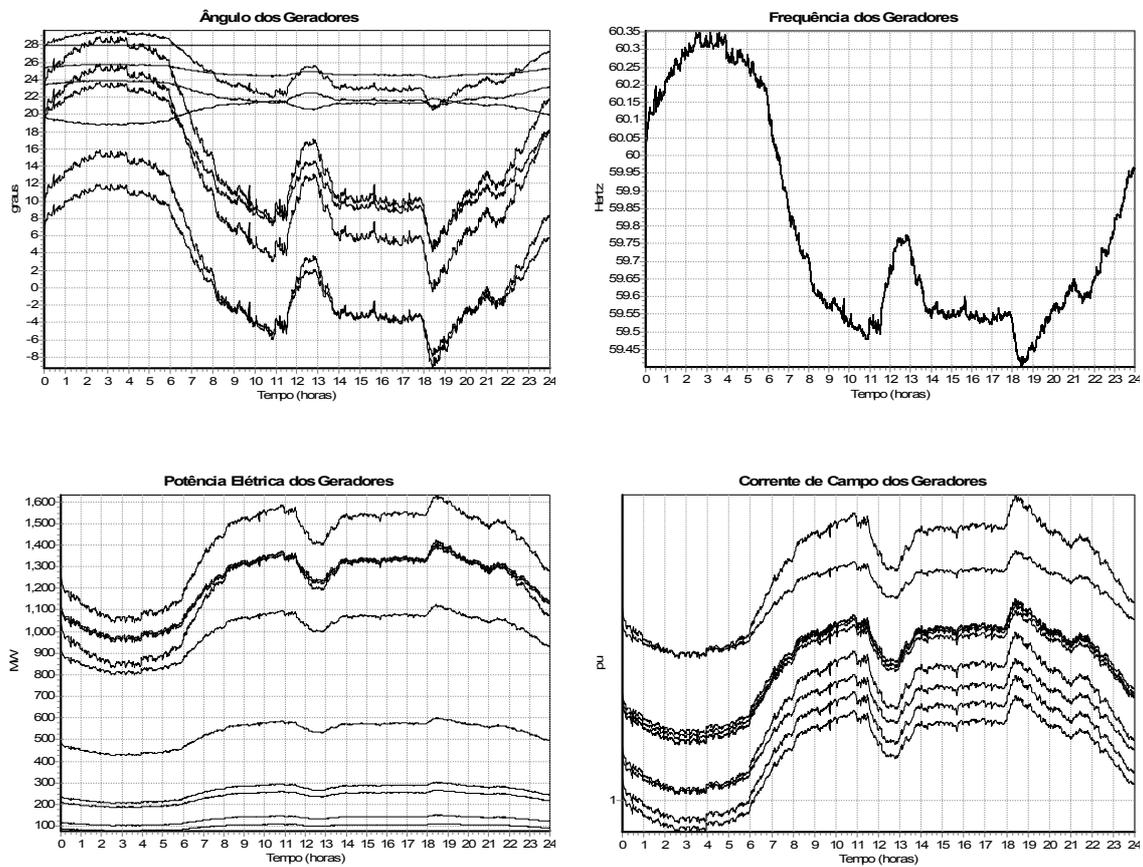


Figura 65 – Grandezas Elétricas dos Geradores (*FASTSIM++*)

### 5.3.6.2 Simulação da Dinâmica Completa

Os resultados obtidos com a simulação da dinâmica completa são apresentados na Figura 66, Figura 67 e Figura 68. Os resultados mostrados a seguir foram obtidos através do método simultâneo implícito (*SIMSEEs++*), porém simulações foram realizadas com o método alternado implícito (*SIMSEEs++*) e os resultados mostraram-se idênticos.

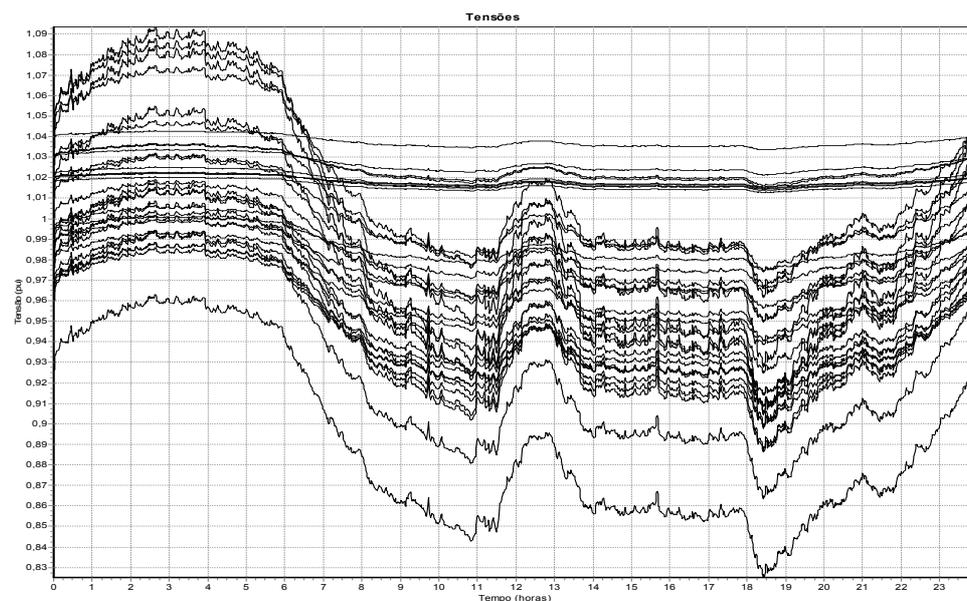


Figura 66 – Tensões das Barras (*SIMSEEs++*)

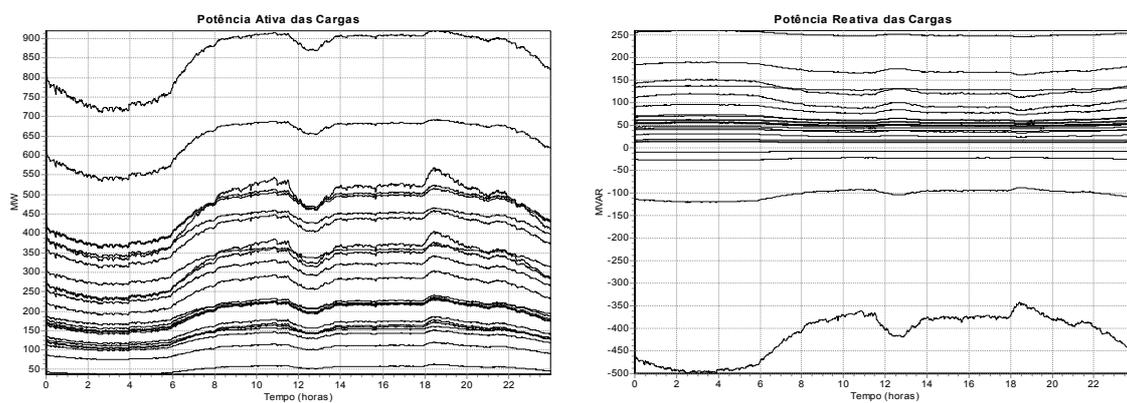


Figura 67 – Potência Ativa e Reativa das Cargas (*SIMSEEs++*)

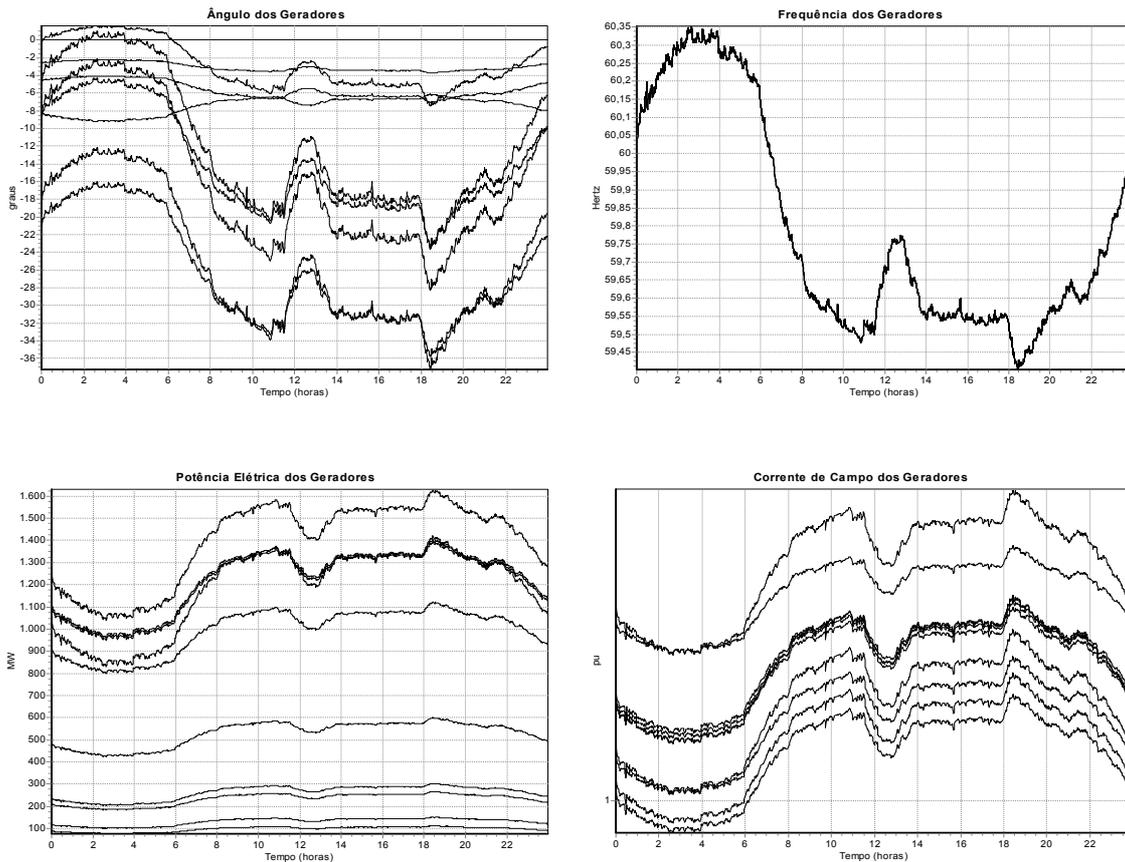


Figura 68 –Grandezas Elétricas dos Geradores (*SIMSEEs++*)

Os resultados obtidos com a simulação rápida (*FASTSIM++*) mostraram-se bastante satisfatórios, quanto a precisão das curvas obtidas, quando comparados com os obtidos com a simulação da dinâmica completa, uma vez que para o horizonte de simulação desejado as curvas podem ser consideradas idênticas. Quanto ao desempenho computacional, o método de simulação rápida mostrou-se bastante superior aos métodos de simulação da dinâmica completa. A Tabela 2 mostra os resultados de desempenho computacional dos programas utilizados para as simulações de longo prazo.

Tabela 2 - Desempenho Computacional dos Programas para as Simulações de Longo Prazo

<b>FASTSIM++</b>		<b>SIMSEEA++</b>			<b>SIMSEEs++</b>		
$\Delta t$	Tempo (secs)	$\Delta t$	Tempo (secs)	Tempo (horas)	$\Delta t$	Tempo (secs)	Tempo (horas)
60	9.06	0.005	25736	7.1	0.010	70833	19.7

---

Os resultados mostraram que o tempo de simulação do método rápido é cerca de 2800 vezes mais rápido que o método alternado implícito com passo fixo de 0.005 segs (passos maiores não foram possíveis com este método), e de 7800 vezes mais rápido que o método simultâneo implícito com passo fixo de 0.010 segs. No entanto, uma análise do desempenho computacional mais elaborada é apresentada na segunda parte deste capítulo.

### **5.3.6.3 Simulação Rápida e Completa Combinadas**

A metodologia de simulação rápida mostrou-se adequada para simulações de longa duração onde os fenômenos rápidos podem ser desprezados, no entanto quando distúrbios severos e de grande impacto são aplicados no sistema, esta hipótese não pode mais ser mantida, e a simulação da dinâmica completa é a metodologia mais adequada. No item 4.4 – Simulação Rápida Modificada – foi proposta uma metodologia de simulação que combina a simulação rápida modificada com a simulação da dinâmica completa, sendo ambas metodologias chaveadas conforme a necessidade. Para mostrar as potencialidades da metodologia de simulação rápida e completa combinadas foi simulada a mesma curva de carga diária do caso anterior, porém agora sendo aplicado um curto-circuito trifásico na barra da usina de *Itaúba* no instante 66300 segs (18:25 hs) com duração de 100 msecs. O curto-circuito é removido com a abertura da LT *Itaúba–Passo Fundo*, permanecendo a LT desligada pelo resto do período de simulação. Para esta simulação as máquinas síncronas foram representadas através de um modelo de 5<sup>o</sup> ordem com efeitos transitórios e subtransitórios representados. Os reguladores de tensão e de velocidade foram representados por modelos simplificados de 2<sup>o</sup> ordem, e as cargas através do seu modelo de impedância constante.

Os resultados da Figura 69 e Figura 70 mostram as tensões das barras e ângulos dos geradores para todo o horizonte de simulação, e nos detalhes, um “zoom” com a janela de tempo onde a metodologia de simulação foi chaveada para simulação da dinâmica completa. A janela da simulação dinâmica completa permanece por 5 segs. (ou até que a estabilidade transitória do sistema frente ao distúrbio seja comprovada – esta verificação, no entanto, não foi implementada), voltando após decorrido este tempo para a metodologia rápida e permanecendo assim até o final da simulação.

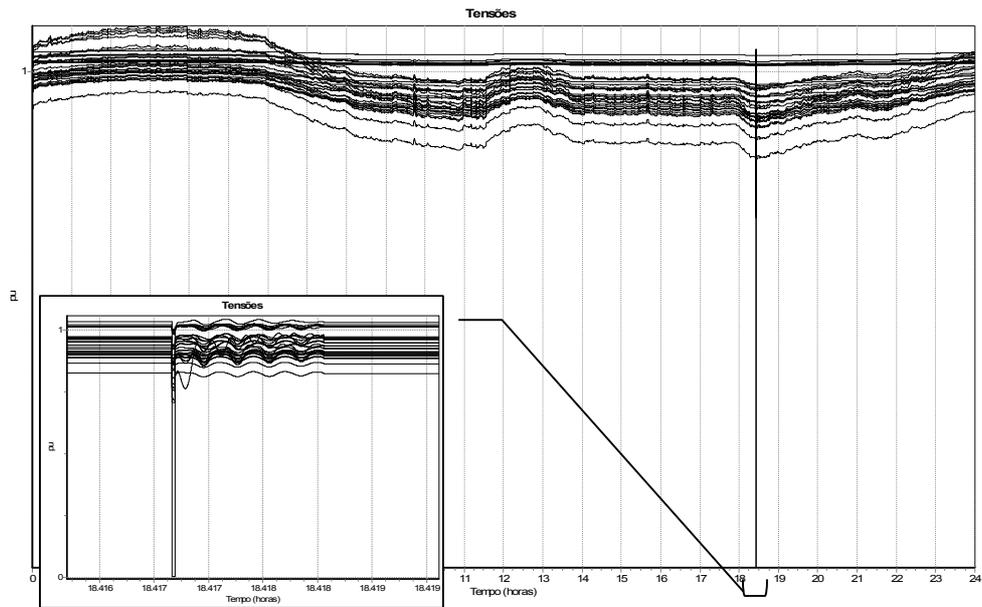


Figura 69 – Tensões das Barras (*FASTSIM++*)

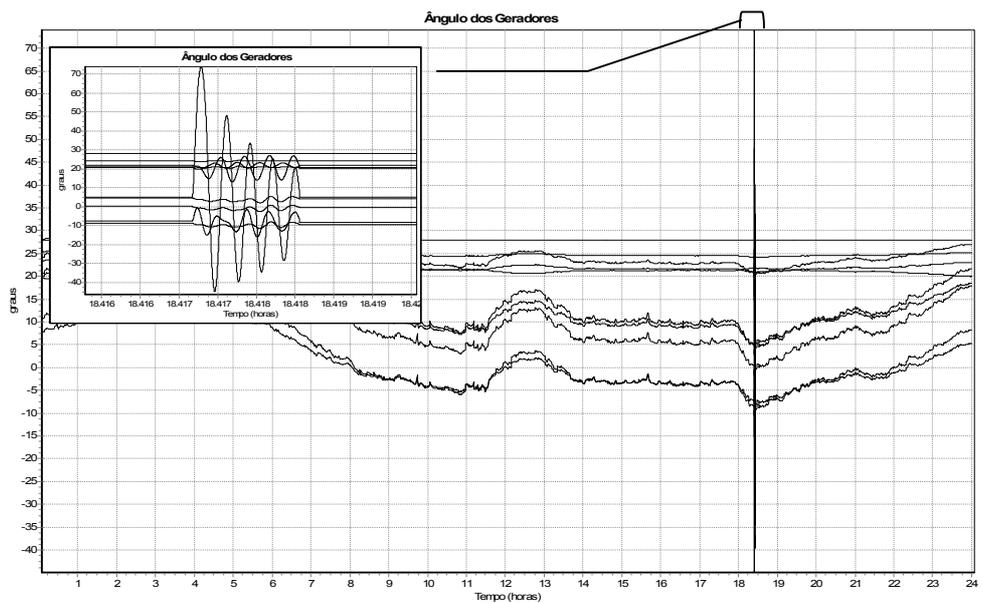


Figura 70 – Ângulos dos Geradores (*FASTSIM++*)

### 5.3.7 Flexibilidade de Formulação

O mecanismo de solução e derivação automática dos modelos da estrutura orientada a objetos utilizada neste trabalho permite que um problema específico possa ser formulado conforme definição do usuário (ver item 3.4.2.2 – Mecanismo de Solução e Derivação das Equações). Assim, o tipo de injeção dos dispositivos na rede

elétrica (injeção de potência ou injeção de corrente) e o tipo de coordenada para as tensões (coordenadas polares ou retangulares) pode ser definido em tempo de execução, permitindo que a formulação do problema mude conforme a necessidade ou definição do usuário. Isto é possível devido a funcionalidades especiais dos blocos de Entrada/Saída dos modelos (ver item 3.4.2.4 – Funções Especiais das Classes BlcOut e dP/dFdX) que executam as conversões necessárias para a formulação desejada.

Para ilustrar esta potencialidade serão apresentados resultados apenas para o programa de fluxo de potência generalizado, entretanto todos os aplicativos que utilizam a metodologia generalizada de solução e/ou derivação automática das equações dos modelos possuem esta capacidade. A Tabela 3 mostra os resultados de convergência do fluxo de potência generalizado para as quatro formulações possíveis.

Tabela 3 – Convergência do Fluxo de Potência

	<b>Tensão Coord. Polares</b>	<b>Tensão Coord. Retangulares</b>
<b><i>Injeção de Potência</i></b>	2 iterações mismatch: 79.93 → 0.543 cond(J) : 7105.79	2 iterações mismatch: 79.93 → 0.997 cond(J) : 7037.82
<b><i>Injeção de Corrente</i></b>	2 iterações mismatch: 66.59 → 0.870 cond(J) : 6843.29	2 iterações mismatch: 66.59 → 0.681 cond(J) : 6771.91

*obs: Todos os cálculos foram executados com partida plana para as tensões.*

### 5.3.8 Flexibilidade de Modelagem

O alto grau de generalização obtido com os mecanismos de solução e derivação automática das equações dos modelos permite que qualquer dispositivo seja adicionado ao sistema e tratado pelos aplicativos do ambiente de forma totalmente automática, ou seja, qualquer novo modelo construído pelo usuário é automaticamente reconhecido e considerado por todos os aplicativos do ambiente em tempo de execução do programa. Para ilustrar esta potencialidade será adicionado um Compensador Estático de Reativos – SVC (sigla do inglês *Static VAR Compensator*) no sistema exemplo, cujo modelo é mostrado na Figura 71.

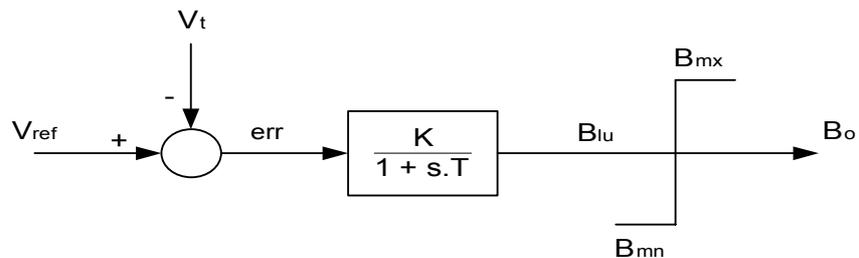


Figura 71 – Modelo do SVC Introduzido na Barra de Joinville

O novo dispositivo é construído e adicionado ao sistema exemplo através de uma linguagem de modelagem específica para o ambiente computacional implementado. Não é propósito deste trabalho apresentar detalhes da linguagem de modelagem, porém a definição do modelo nesta linguagem é apresentada a seguir apenas para propósitos de ilustração.

<< Linguagem Utilizada para Definição do Modelo do SVC >>

```

-----
--- COMPENSADOR ESTATICO DE REATIVOS ---
-----
MODL:SVC#Mdl:01      .
#
# =====
# === PARMS ===
# =====
#      [--ID.--] REF VISIBL [--Valor--]
#      PARM:      K          50.00
#      PARM:      T          0.05
#      PARM:      Bmn       -0.50
#      PARM:      Bmx        2.00
#      PARM:      Vref REF   0.98
#
# === INPUT ===
#      [--ID.--] [Vre/Vmd] [Vim/Van] [TIP OUT]
#      TENS:    INP001      Vt          POLAR
#
# =====
# === EQUACAO DO SVC ===
# =====
#      [--ID.--] [--INP--] [--OUT--] STT [----A----] [----B----] [---C---]
#      SOMD:    BLC001 +    Vref      err
#              .... -    Vt
#      DFIM
#      LAG :    BLC002      err      Blu      K          1.0      T
#      LIMT:    BLC003      Blu      Bo      Bmn      Bmx
#
# === OUTPUT C/A REDE ===
#      OSHT:    OUT001      POTENCIA POTENCIA
#              [-P/Ire-] [-Q/Iim-] [---G---] [---B---]
#      OUT :
#              B0
#      DFIM
#      DFIM

```

<< Final do Fragmento >>

Após a construção do modelo, o mesmo foi adicionado ao sistema exemplo na barra de *Joinville* (barra 378), com a tensão de referência ajustada em 0.98 pu. Todos os aplicativos do ambiente computacional passam agora a considerar o novo modelo

---

nas simulações e análises executadas. A seguir serão mostrados alguns resultados obtidos com os aplicativos do ambiente computacional considerando o modelo de SVC adicionado ao sistema. Não é demais salientar que nenhuma linha de código dos aplicativos foi alterada para a inclusão do SVC.

### 5.3.8.1 Fluxo de Potência

O fluxo de potência generalizado foi executado em sua formulação de injeção de potência e tensões em coordenadas polares, todas as cargas representadas pelo seus modelos de potência constante. Os resultados das tensões das barras para os casos sem SVC e com SVC são apresentados na Figura 72 e Figura 73, respectivamente.

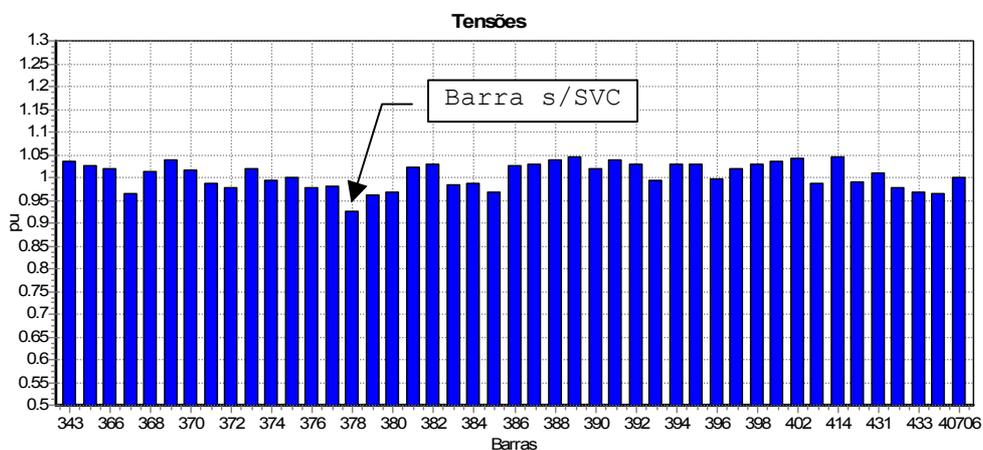


Figura 72 – Tensões das Barras Sem a Inclusão do SVC (*FLXPOT++*)

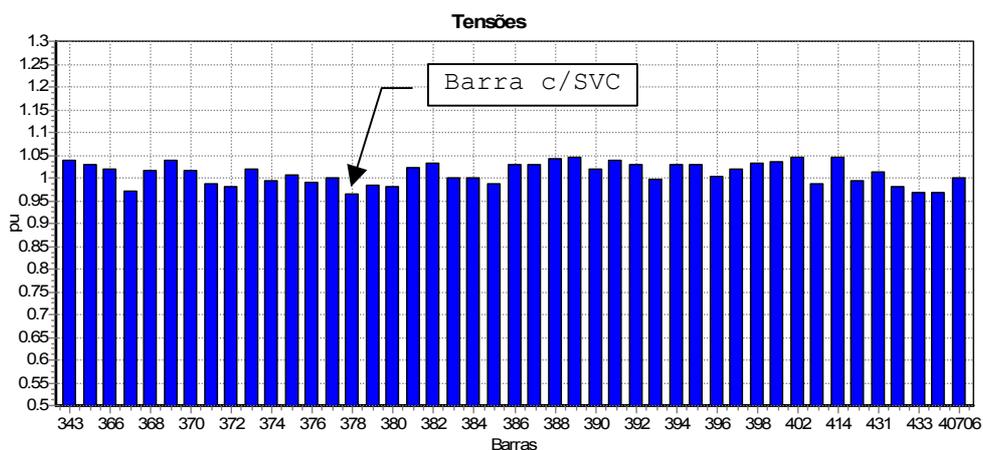


Figura 73 – Tensões das Barras Com a Inclusão do SVC (*FLXPOT++*)

A Figura 74 mostra os resultados calculados para as potências ativas e reativas injetadas no sistema pelos geradores e pelo SVC. O sinal negativo da potência reativa injetada pelo SVC deve-se a convenção adotada para dispositivos definidos pelo usuário (mesma convenção adotada para as cargas do sistema).

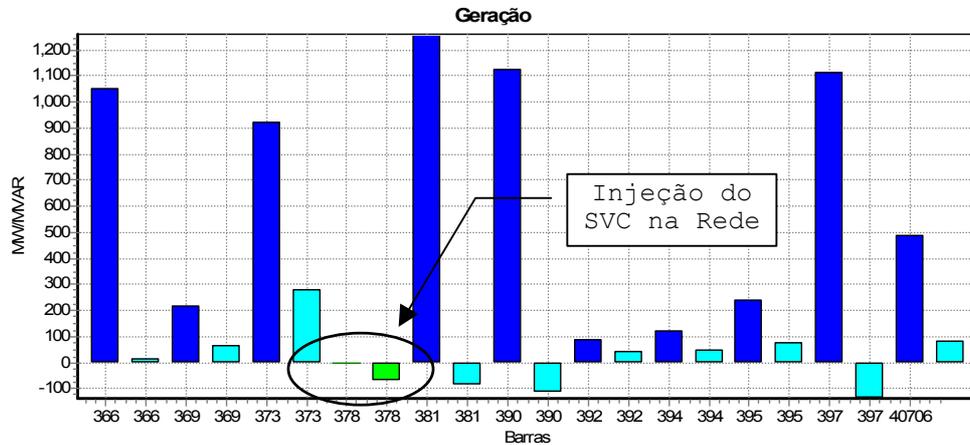


Figura 74 – Potências Geradas pelos Geradores e pelo SVC (*FLXPOT++*)

### 5.3.8.2 Análise de Estabilidade de Pequenos-Sinais

De forma idêntica ao fluxo de potência generalizado, o aplicativo de análise de estabilidade de pequenos-sinais reconhece e passa a considerar automaticamente o novo equipamento na sua formulação. A nova estrutura da matriz Jacobiana, considerando agora o SVC, é apresentada na Figura 75, onde as linhas tracejadas dividem a estrutura da matriz Jacobiana em sub-matrizes relativas aos estados ( $J_1$ ), a rede elétrica ( $J_4$ ), e em duas sub-matrizes de acoplamento ( $J_2$  e  $J_3$ ). A linha e a coluna introduzidas na matriz Jacobiana devido ao estado interno do SVC (denominado  $B_{li}$ ) são mostradas na Figura 75 através de linhas cheias, sendo marcados com setas os elementos que surgem na matriz Jacobiana devido as derivadas parciais das equações do modelo do SVC e das equações de injeção na rede elétrica. A estrutura da matriz mostrada na Figura 75 foi obtida diretamente do programa de análise de estabilidade a pequenos-sinais após a inclusão do SVC.

O modelo dinâmico considerado nesta análise representou as máquinas síncronas através de um modelo de 5<sup>o</sup> ordem com efeitos transitórios e subtransitórios representados. Os reguladores de tensão e de velocidade foram representados por modelos simplificados de 2<sup>o</sup> ordem, e as cargas através do seu modelo de impedância constante.

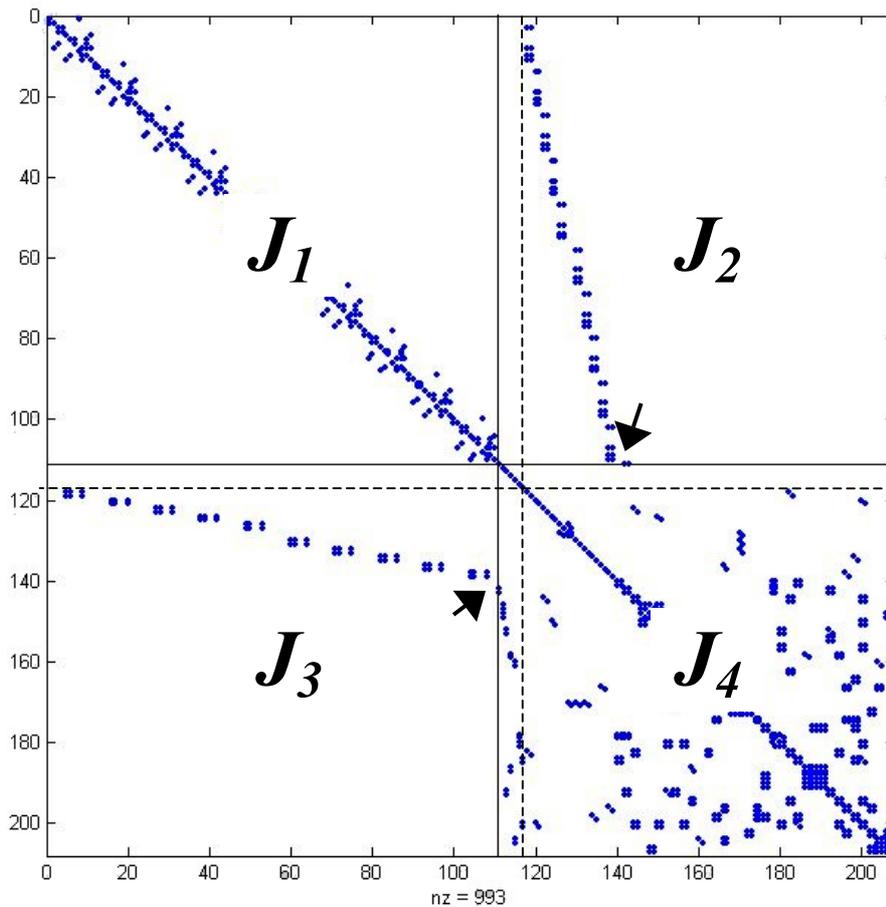


Figura 75 – Estrutura da Matriz Jacobiana Com a Inclusão do SVC (*AEPS++*)

Os autovalores do sistema dinâmico, considerando o SVC, foram calculados e são mostrados na Figura 76 (apenas os autovalores mais próximos ao eixo imaginário do plano complexo foram plotados). Nesta figura foi selecionado um autovalor específico e calculados os seus correspondentes fatores de participação (mostrados no detalhe da Figura 76), onde observa-se uma significativa influência do estado interno do SVC ( $B_{lv}$ ) neste autovalor.

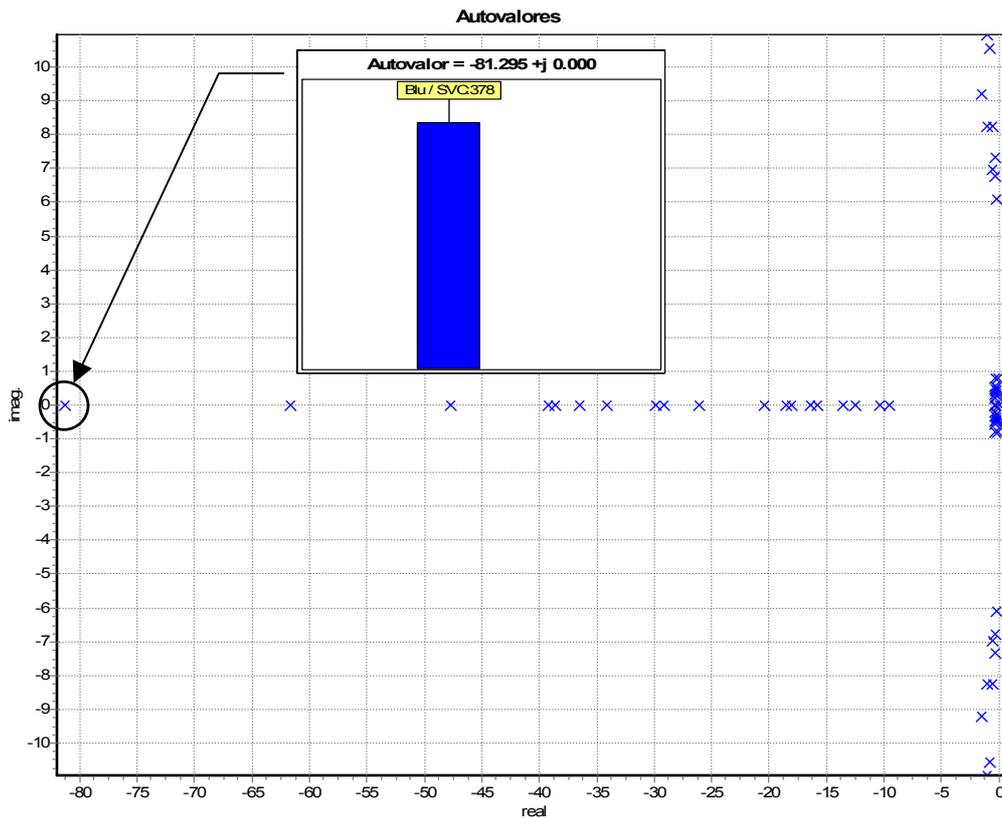


Figura 76 – Autovalores do Sistema Com a Inclusão do SVC (*AEPS++*)

### 5.3.8.3 Simulação da Dinâmica Completa

A simulação da dinâmica completa do sistema foi realizada utilizando o método simultâneo implícito (*SIMSEEs++*). Nesta simulação foi aplicado um curto-circuito trifásico na barra da usina de *Itaúba*, sendo este eliminado 100 msecs após sua aplicação. As máquinas síncronas foram representadas através de um modelo de 5<sup>o</sup> ordem com efeitos transitórios e subtransitórios representados. Os reguladores de tensão e de velocidade foram representados por modelos simplificados de 2<sup>o</sup> ordem, e as cargas através do seu modelo de impedância constante.

A Figura 77 e Figura 78 mostram os resultados obtidos para a simulação da dinâmica completa do sistema, onde são apresentados resultados relativos a variáveis do SVC adicionado ao sistema.

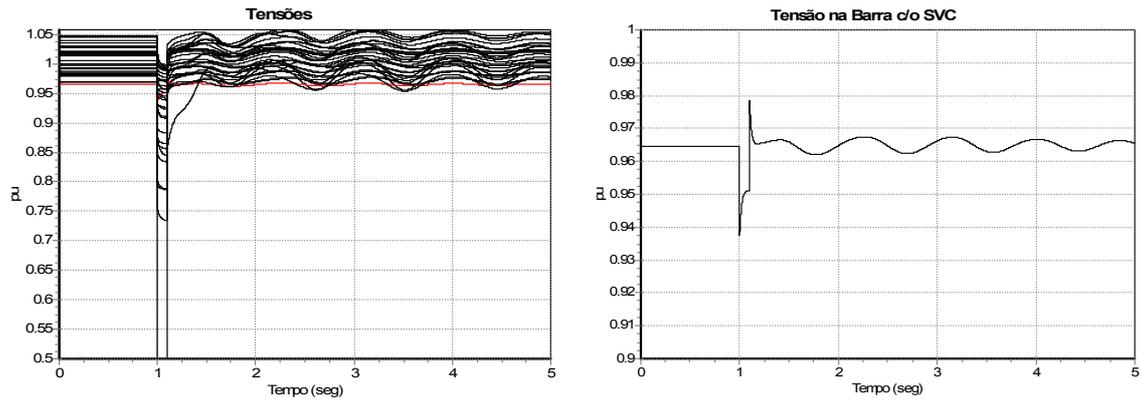


Figura 77 – Tensões das Barras e Tensão na Barra do SVC (*SIMSEEs++*)

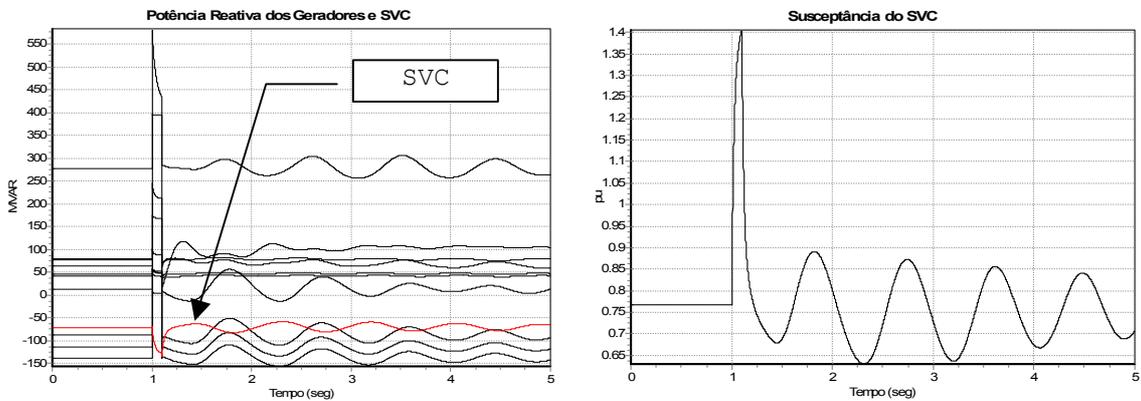


Figura 78 – Potência Reativa e Susceptância do SVC (*SIMSEEs++*)

#### 5.3.8.4 Simulação Rápida

A Figura 79 e Figura 80 mostram os resultados de uma simulação rápida conduzida para o sistema exemplo considerando o SVC, onde um acréscimo de 20% de carga em rampa (ativa e reativa) foi aplicado ao sistema durante 60 segs.

Para esta simulação as máquinas síncronas foram representadas através de um modelo de 5<sup>o</sup> ordem com efeitos transitórios e subtransitórios representados. Os reguladores de tensão e de velocidade foram representados por modelos simplificados de 2<sup>o</sup> ordem, e as cargas através do seus modelos de potência constante.

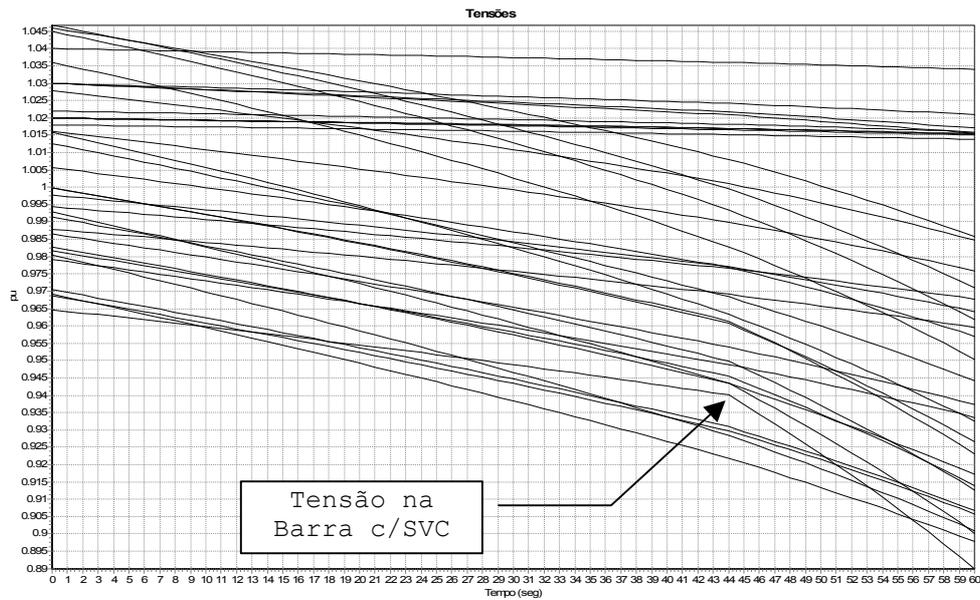


Figura 79 – Tensões das Barras (FASTSIM++)

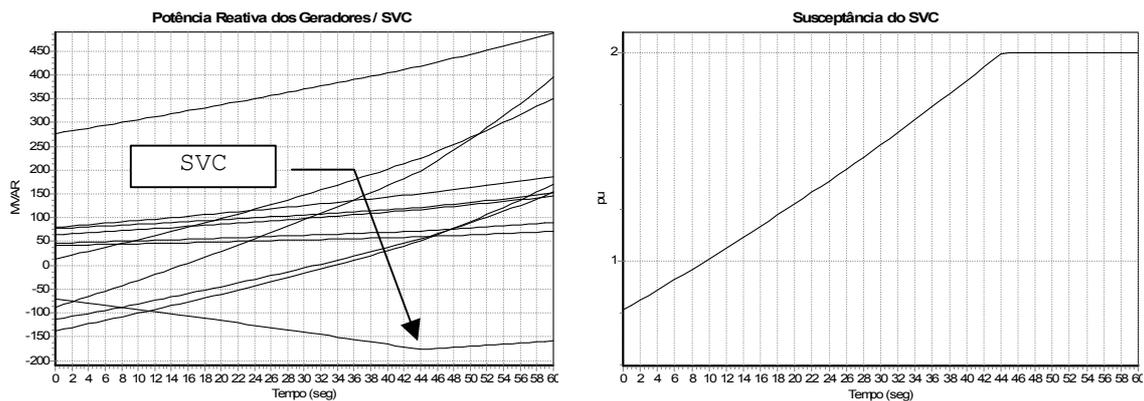


Figura 80 – Potência Reativa dos Geradores/SVC e Susceptância do SVC (FASTSIM++)

Os resultados das simulações acima indicam que o SVC alcança o seu limite de controle da tensão no instante 44 segs (conforme mostra claramente a Figura 80 – Susceptância do SVC), quando então o SVC perde a sua capacidade de controle de tensão e passa a comportar-se como um capacitor. A partir deste instante a tensão terminal do SVC decai de forma mais acentuada (ver Figura 79), e a potência reativa injetada decai proporcionalmente ao quadrado da tensão na barra (ver Figura 80 - Potência Reativa dos Geradores/SVC).

---

### 5.3.9 Comentários Gerais

Os resultados apresentados demonstraram a capacidade e as potencialidades do ambiente orientado a objetos implementado, sendo este capaz de estudar uma grande faixa de fenômenos em sistemas elétricos, tanto de natureza estática como de natureza dinâmica. Os aplicativos são todos integrados em um mesmo ambiente de estudo, onde é possível trocar, a qualquer instante, a ferramenta de análise de forma natural e totalmente transparente para o usuário. Todos os aplicativos implementados mostraram-se bastante robustos para as simulações e análises realizadas não apresentando durante as simulações problemas de convergência ou instabilidade numérica. Aliado a estas características encontra-se ainda a facilidade para manutenção e futuras expansões proporcionada pela modelagem orientada a objetos, e a grande versatilidade para a incorporação de novos modelos e dispositivos ao ambiente orientado a objetos. Neste sentido, a estrutura orientada a objetos proposta mostrou-se bastante flexível quanto a capacidade de incorporar automaticamente, e para todos os aplicativos do ambiente, qualquer novo modelo ou dispositivo definido pelo usuário.

Os programas de fluxo de potência generalizado, análise de estabilidade de pequenos-sinais e simulação da dinâmica completa pelos método alternado e simultâneo foram validados com programas comerciais da área de sistemas de energia elétrica comprovando os resultados apresentados.

O método de simulação rápida modificada proposto mostrou-se capaz de executar simulações de longo prazo com precisão, robustez e eficiência computacional, permitindo ainda o estudo de uma faixa mais ampla de fenômenos que a formulação convencional.

## 5.4 Desempenho Computacional do M.O.O.

Dentre os objetivos deste trabalho destaca-se a análise da viabilidade do emprego da programação orientada a objetos e da linguagem C<sup>++</sup> aos problemas de simulação e análise de sistemas elétricos. Assim, foi realizada uma análise comparativa da eficiência computacional de alguns aplicativos do ambiente orientado a objetos com programas comerciais da área ou, na falta destes, com programas em implementações tradicionais em C<sup>++</sup> ou FORTRAN. Esta análise procura ainda avaliar o

---

desempenho computacional dos programas orientados a objetos para sistemas elétricos de várias dimensões.

A seguir são apresentados os sistemas utilizados para a análise de desempenho computacional realizada.

#### 5.4.1 Descrição dos Sistemas Utilizados

Foram utilizados cinco sistemas teste de pequeno, médio e grande porte. Os parâmetros descritivos da dimensão destes sistemas são apresentados na tabela abaixo.

Tabela 4 - Dimensões dos Sistemas Utilizados

Sistema	Barras	LTs e Transf.	Cargas	Geradores	
				Fluxo de Potência	c/ Modelo Dinâmico
<i>S0045</i>	45	72	24	10	11
<i>S0188</i>	188	264	79	36	25
<i>S0730</i>	730	1146	392	116	82
<i>S2000</i>	1916	2788	1157	198	93
<i>S2800</i>	2806	3931	1452	296	108

O sistema S0045 é um equivalente do sistema elétrico da região Sul do Brasil. Todos os demais sistemas são equivalentes, em diversos níveis de detalhes, do sistema interligado do Brasil, onde foram representadas algumas sub-áreas por equivalentes estáticos de rede. As máquinas síncronas com modelo dinâmico representado são descritas através de um modelo de 5<sup>o</sup> ordem com efeitos transitórios e subtransitórios representados. O reguladores de tensão e de velocidade foram representados por modelos simplificados de 2<sup>o</sup> ordem, e as cargas através do seu modelo de potência ou impedância constante.

Todas as simulações apresentadas para a análise de desempenho computacional foram realizadas em um microcomputador DELL com processador INTEL PENTIUM IV – 2.8 GHz de “clock” e 256Mb de memória RAM. O compilador utilizado sob esta plataforma foi o Borland C++ Builder versão 6.0 com opções de otimização habilitadas.

## 5.4.2 Desempenho Computacional do Fluxo de Potência

A análise de desempenho computacional para o fluxo de potência será realizada comparando os resultados de desempenho da formulação generalizada (*FLXPOT++*) com duas implementações tradicionais escritas em C++, uma em coordenadas polares e outra em coordenadas retangulares [62] (infelizmente os programas comerciais de fluxo de potência não informam o tempo de CPU de execução). Os programas tradicionais escritos em C++ possuem implementação em código (em tempo de compilação) do cálculo das derivadas parciais e montagem da matriz Jacobiana, por esta razão não permitem o reconhecimento automático de novos modelos introduzidos, ficando limitados a um conjunto de dispositivos pré-determinados.

A Tabela 5 mostra os resultados de desempenho computacional obtidos com as três implementações de fluxo de potência avaliadas. Os resultados para o sistema *S0045* não foram apresentados devido a baixa precisão para medir tempos de CPU muito pequenos na plataforma INTEL.

Tabela 5 - Desempenho Computacional dos Programas de Fluxo de Potência

	FLXPOT++		Fluxo de Pot. Clássico (Retang.)			Fluxo de Pot. Clássico (Polares)		
	Tempo (segs)	Iters	Tempo (segs)	Iters	FD	Tempo (segs)	Iters	FD
<i>S0045</i>	-	-	-	-	-	-	-	-
<i>S0188</i>	0.032	2	0.016	2	0.5	0.016	2	0.5
<i>S0730</i>	0.140	2	0.078	3	0.6	0.078	3	0.6
<i>S2000</i>	0.560	3	0.406	6	0.7	0.516	6	0.9
<i>S2800</i>	0.859	3	0.594	5	0.7	0.750	5	0.9

Obs: o programa *FLXPOT++* utilizou formulação com injeção de potência e coordenadas polares para as tensões.

Os resultados apresentados na Tabela 5 mostram o tempo total em segundos para a execução do cálculo do fluxo de potência, o número de iterações e, para os dois programas tradicionais em C++, um índice definido como *Fator de Desempenho* do programa (*FD*). Este fator representa a relação de desempenho entre o caso analisado e um caso tomado como base, conforme define a equação 43. Assim, *Fatores de Desempenho* abaixo de 1.0 indicam que o programa analisado é mais rápido que o caso base, e *Fatores de Desempenho* acima de 1.0 indicam que o programa analisado é mais lento que o caso base. Nos resultados da Tabela 5 o caso base assumido

corresponde ao fluxo de potência generalizado (cujos tempos de referência são destacados na coluna sombreada).

$$FD = \frac{\text{tempo\_caso\_analizado}}{\text{tempo\_caso\_base}} \quad (49)$$

Os *FDs* para todos os sistemas podem ser visualizados graficamente na Figura 81.

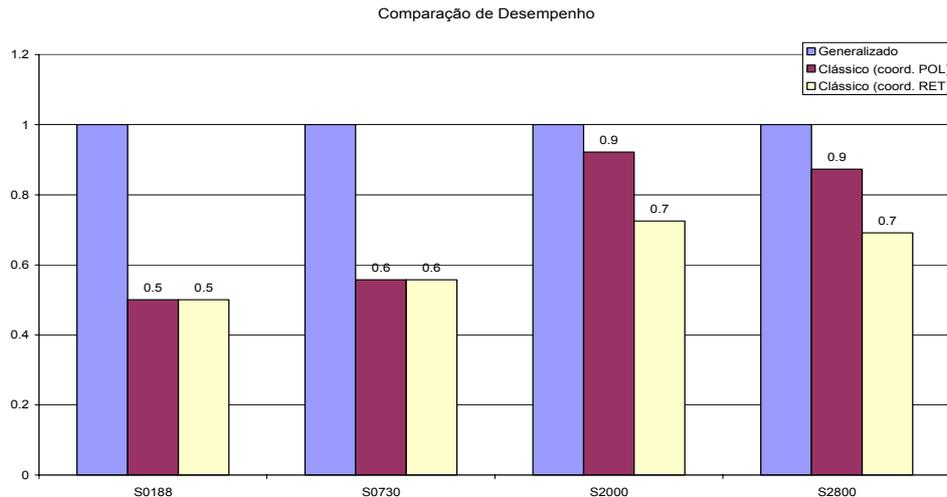


Figura 81 – Comparação de Desempenho do Programa de Fluxo de Potência Generalizado com o Fluxo de Potência Convencional

Dos resultados apresentados observa-se :

- O desempenho do programa de fluxo de potência generalizado mostrou-se inferior às implementações tradicionais para todos os sistemas avaliados. Este resultado, no entanto, já era esperado uma vez que o cálculo das derivadas parciais e montagem da matriz Jacobiana não é implementado em código mas determinado automaticamente pelo programa em tempo de execução (ver item 3.4.2.2 – Mecanismo de Solução e Derivação das Equações);
- Os programas tradicionais são cerca de 2.0 vezes mais rápidos que o generalizado para os sistemas de pequeno e médio porte, porém esta superioridade decai a medida que a dimensão do sistema aumenta. Isto ocorre porque o fluxo de potência generalizado converge em menos iterações que as implementações tradicionais, ficando a diferença de tempo em torno de 10% para o programa em coordenadas polares e 30% para o programa em coordenadas retangulares;

A seguir será realizada uma análise mais detalhada de uma iteração do processo de solução do fluxo de potência para as três implementações avaliadas.

#### 5.4.2.1 Composição de uma Iteração do Processo

A Figura 82 e Figura 83 mostram a composição de uma iteração do processo de solução dos fluxos de potência avaliados, onde os resultados apresentados são percentuais em relação ao tempo total de uma iteração do método generalizado.

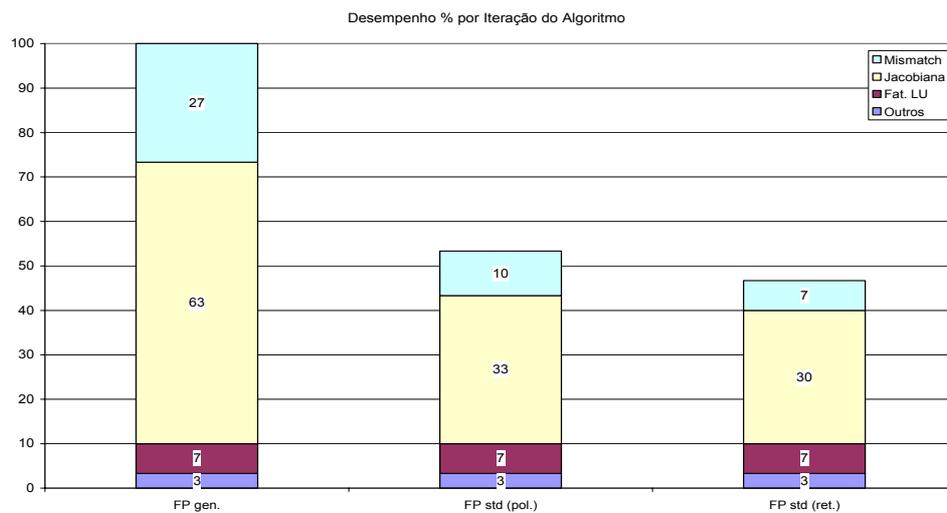


Figura 82 – Comparação de Desempenho de uma Iteração do Processo para o Sistema S2000

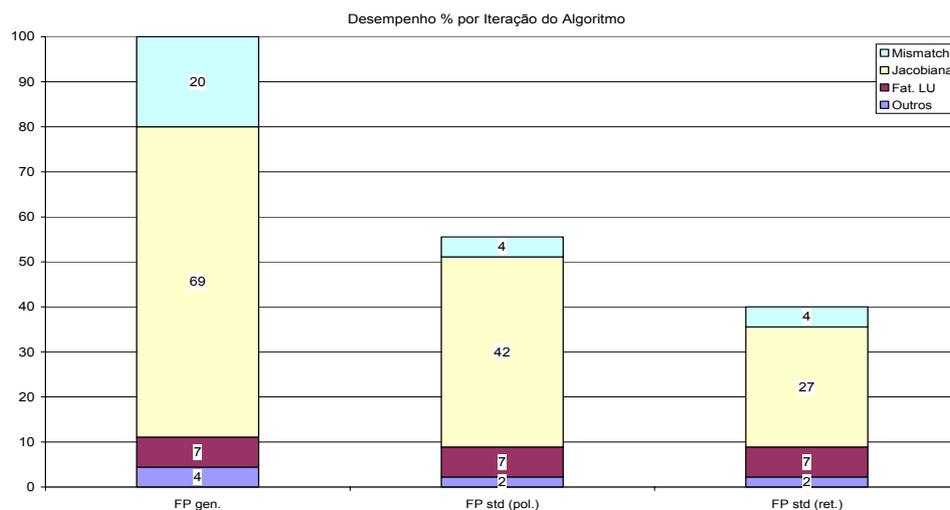


Figura 83 – Comparação de Desempenho de uma Iteração do Processo para o Sistema S2800

---

Os resultados apresentados comprovam que o método generalizado é cerca de duas vezes mais lento que os demais por iteração. Isto ocorre devido, principalmente, ao mecanismo de solução e derivação automática das equações dos modelos, uma vez que a grande diferença nos tempos de execução é justamente onde este mecanismo está mais presente, durante o cálculo das derivadas parciais e montagem da matriz Jacobiana e, em menor proporção, no cálculo do vetor de *mismatches* do sistema linear. Nas duas implementações tradicionais estas tarefas são implementadas em código, o que as torna extremamente eficientes. Assim, é possível afirmar que o mecanismo de solução e derivação automática é a principal vantagem e maior “*gargalo*” da metodologia generalizada.

#### **5.4.3 Desempenho Computacional da Simulação da Dinâmica Completa**

A análise de desempenho computacional para os programas de simulação da dinâmica completa será realizada comparando-se os resultados de desempenho dos programas *SIMSEEs++* (método simultâneo implícito) e *SIMSEEs++* (método alternado implícito) com um programa comercial escrito em FORTRAN. O programa em FORTRAN utilizado foi o *Programa para Análise de Transitórios Eletromecânicos - ANATEM* desenvolvido pelo CEPEL. Este programa utiliza o esquema alternado implícito para a solução do conjunto de equações algébrico-diferenciais do sistema.

A análise comparativa é limitada a estudos típicos de estabilidade transitória, sendo que, para esta análise, todos os programas utilizaram os mesmos modelos para os dispositivos do sistema, a mesma tolerância de convergência do processo de solução, o mesmo passo de integração, e o mesmo critério de convergência. O evento simulado foi a aplicação de um curto-circuito trifásico em uma das barras do sistema com duração de 100 msecs, e o tempo total de simulação foi de 5 segs.

A Tabela 6 mostra os resultados de desempenho computacional obtidos com o programa ANATEM e com os dois programas de simulação da dinâmica completa implementados.

Tabela 6 – Comparação de Desempenho dos Programas de Simulação da Dinâmica Completa

	ANATEM			SIMSEEA++		SIMSEEs++	
	Tempo (secs)		No. ITERS/passos	Tempo (secs)	FD	Tempo (secs)	FD
	C/CDU	S/CDU					
<b>S0045</b>	<b>1.97</b>	1.26	2.93	4.64	<b>2.4</b>	17.54	<b>8.9</b>
<b>S0188</b>	<b>3.01</b>	2.18	2.84	21.29	<b>7.1</b>	69.85	<b>23.2</b>
<b>S0730</b>	<b>8.37</b>	6.75	2.89	97.40	<b>11.6</b>	326.75	<b>39.0</b>
<b>S2000</b>	<b>15.04</b>	13.29	2.84	203.26	<b>13.5</b>	766.98	<b>51.0</b>
<b>S2800</b>	<b>26.08</b>	23.72	3.69	261.46	<b>10.0</b>	984.18	<b>37.7</b>

Obs: todos os resultados são com passo de integração fixo e igual a 0.001 seg.

Os resultados apresentados na Tabela 6 mostram o tempo total em segundos para a execução das simulações, e o *Fator de Desempenho* dos programas SIMSEEA++ e SIMSEEs++, tomando-se como caso base o programa ANATEM (cujo tempo de referência foi destacado na coluna sombreada da tabela). Novamente, *Fatores de Desempenho* abaixo de 1.0 indicam que o programa analisado é mais rápido que o caso base, e *Fatores de Desempenho* acima de 1.0 indicam que o programa analisado é mais lento que o caso base.

Os *FDs* para todos os sistemas podem ser visualizados graficamente na Figura 84.

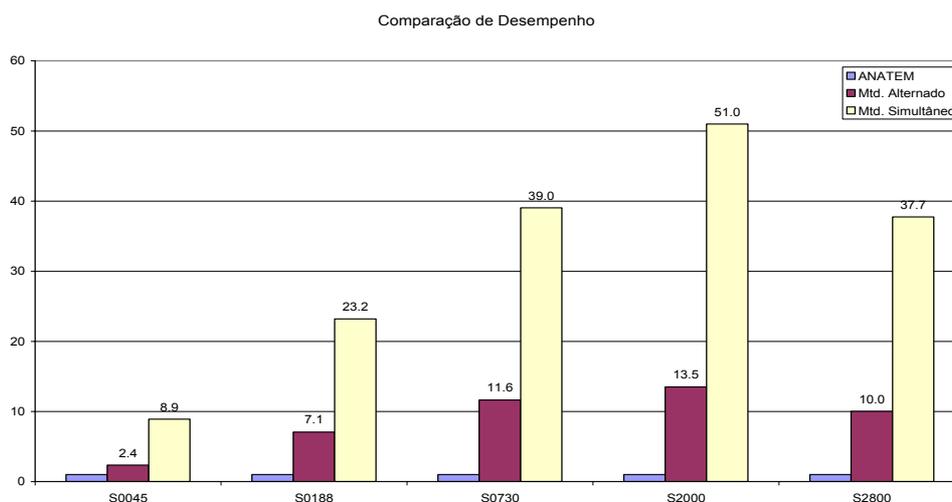


Figura 84 – Comparação de Desempenho dos Programas de Simulação da Dinâmica Completa

---

Dos resultados apresentados observa-se :

- Considerando-se as condições estipuladas para as simulações, o desempenho dos programas *SIMSEEA++* e *SIMSEES++* mostraram-se, para todos os casos, mais lentos que o programa ANATEM;
- O programa *SIMSEEA++* (que utiliza o método alternado implícito, mesmo método utilizado pelo ANATEM) é cerca de 10 vezes mais lento que o programa ANATEM para os sistemas de grande porte. Esta relação diminui um pouco para o sistema S2800 devido ao aumento do número médio de iterações por passo que ocorre no programa ANATEM;
- O programa *SIMSEES++* (que utiliza o método simultâneo implícito com passo fixo) chega a ser 51 vezes mais lento que o programa ANATEM para os sistemas de grande porte. Este resultado, no entanto, já era esperado, uma vez que no método simultâneo implícito a matriz Jacobiana do sistema é calculada e fatorada para cada passo de integração (ou a cada “n” passos, de acordo com a heurística de atualização da matriz Jacobiana adotada). O método alternado implícito, por outro lado, utiliza a matriz  $Y_{barra}$  do sistema em sua formulação (muito menor que a matriz Jacobiana) e mantém esta matriz inalterada por quase toda a simulação, apenas nos instantes onde ocorrem chaveamentos na rede elétrica a matriz é atualizada e refatorada.

#### **5.4.3.1 Análise do Desempenho do Método Alternado Implícito**

O baixo desempenho do método alternado implícito orientado a objetos (*SIMSEEA++*), quando comparado com o programa ANATEM, é devido a generalização obtida com a utilização do modelo orientado a objetos adotado. O programa *SIMSEEA++*, assim como todos os demais programas do ambiente computacional desenvolvido, reconhece e passa a considerar automaticamente qualquer novo modelo ou dispositivo introduzido no sistema, valendo-se, para isto, das rotinas de solução generalizadas das equações dos modelos. Programas comerciais como o ANATEM também possuem a capacidade de incorporar automaticamente novos modelos definidos pelo usuário (através dos *Controladores Definidos pelo Usuário – CDUs*), uma vez que novos controladores estão sempre surgindo no mercado e devem ser rapidamente incorporados aos programas. No entanto, o programa ANATEM permite a definição de apenas alguns tipos de controladores (reguladores de tensão e

velocidade, estabilizadores, etc), dispositivos como LTs, transformadores, cargas, e outros não podem ser representados através de CDUs, ou seja são implementados em código. O programa *SIMSEEA++*, por outro lado, permite que qualquer tipo de dispositivo seja definido pelo usuário. A Tabela 7 mostra como os dispositivos são representados em cada um dos programas.

Tabela 7 - Tipos de Modelos Utilizados

<b>Modelos</b>	<b>SIMSEEA++</b>	<b>ANATEM</b>
<i>Máq. Síncrona</i>	CDU	em código
<i>Regl. de Tensão</i>	CDU	CDU
<i>Regl. de Velocidade</i>	CDU	CDU
<i>Carga</i>	CDU	em código
<i>LT - Transformador</i>	CDU	em código
<i>Reator - Capacitor</i>	CDU	em código

Os CDUs utilizados no programa ANATEM causam um *overhead* no tempo total de execução do programa de menos de 10% para os sistemas de grande porte, conforme pode ser observado nos resultados da Tabela 6 – ANATEM C/CDU e S/CDU. Assim, a maior contribuição no tempo total de simulação é devido a solução do sistema linear  $I=Y.V$ , e devido ao cálculo das injeções de corrente na rede elétrica pelos dispositivos do sistema (máquinas síncronas, cargas, LTs, transformadores, etc). A solução do sistema linear (fatoração LU e solução direta-inversa) é equivalente para ambos programas, logo a diferença no tempo de execução ocorre devido ao cálculo das injeções de corrente nos dispositivos do sistema, justamente onde os dispositivos do ANATEM possuem implementação em código (ver Tabela 7) e os do *SIMSEEA++* utilizam os mecanismos de solução generalizada das equações dos modelos.

Outros aspectos contribuem ainda para a diferença de desempenho encontrada:

- Qualquer chaveamento no sistema elétrico implica em reconfiguração da rede elétrica no ambiente orientado a objetos, isto deve-se a características de representação detalhada dos elementos do sistema (em nível de subestação);
- A implementação em C<sup>++</sup> utiliza listas encadeadas para o armazenamento e gerenciamento de suas estruturas internas. Este tipo de estrutura de dados permite o autodimensionamento do programa para o porte do sistema em estudo, em detrimento de um *overhead* adicional para acesso aos elementos da lista.

### 5.4.3.2 Análise do Desempenho do Método Simultâneo Implícito

O desempenho do método simultâneo implícito mostrou-se até 51 vezes mais lento que o programa ANATEM para sistemas de grande porte. Este resultado ocorre devido a natureza do método simultâneo implícito que necessita calcular e fatorar a matriz Jacobiana do sistema para cada passo de integração. A Figura 85 mostra a composição percentual do tempo de CPU para uma iteração do método simultâneo.

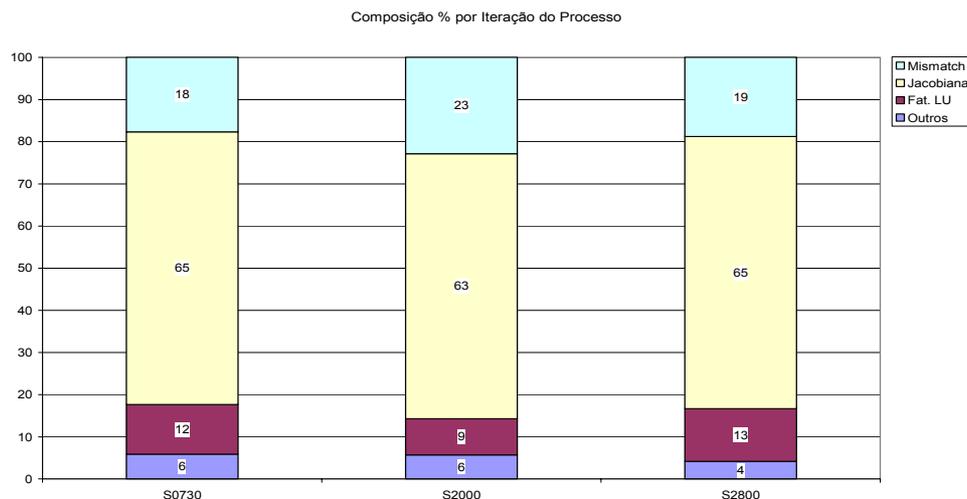


Figura 85 – Composição de uma Iteração do Processo de Cálculo do Método Simultâneo Implícito (*SIMSEEs++*)

Os resultados da Figura 85 mostram que aproximadamente 80% do tempo de cada iteração é utilizado para calcular a matriz Jacobiana ( $\approx 60\%$ ) e os *mismatches* do sistema linear ( $\approx 20\%$ ), comprovando a afirmação do parágrafo anterior. Entretanto, se por um lado o método simultâneo é inerentemente mais lento que o método alternado, por outro a inexistência de erros de interface entre os sistemas algébrico e diferencial permite que passos de integração mais elevados possam ser utilizados (o método alternado não permite passos de integração elevados). Assim, tomando-se o sistema *S0188* como exemplo, alguns testes de desempenho com passos de integração mais elevados foram executados.

Duas simulações adicionais foram executadas para o sistema *S0188* e levantados os correspondentes tempos de execução, uma com passo de integração 10 vezes maior que o utilizado até então, e outra adotando uma estratégia simples de variação automática do passo de integração. A estratégia de variação do passo utilizada é

proposta em [47] e consiste em dobrar o passo de integração atual sempre que o número de iterações do processo de solução for inferior a um valor mínimo especificado, e reduzir a metade sempre que o número de iterações for superior a um valor máximo especificado. Esta estratégia é bastante simples e assume que o número de iterações do processo é uma boa estimativa do erro local da regra trapezoidal, sendo utilizado como referência para o controle do passo de integração. Os resultados obtidos são mostrados na Tabela 8.

Tabela 8 - Desempenho do Sistema S0188 para Variação no Passo de Integração

Tempo de CPU (segs)		
$\Delta t = 0.001$	$\Delta t = 0.010$	$\Delta t = \text{AUTO}$
69.85	8.16	3.89

Os resultados da Tabela 8 podem ser visualizados graficamente na Figura 86.

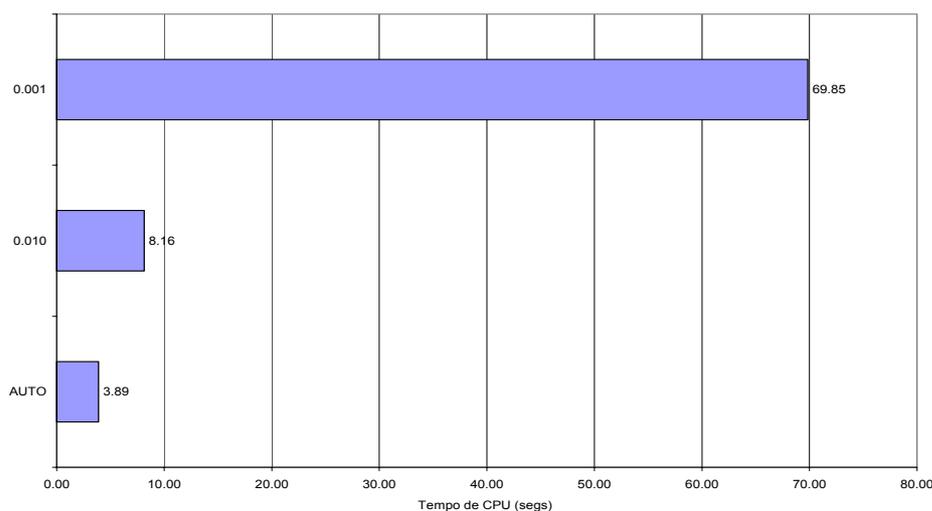


Figura 86 – Desempenho do Sistema S0188 para Variação no Passo de Integração (SIMSEEs++)

A redução no tempo de processamento obtida com o aumento do passo de integração para 0.010 segs chega a 8.5 vezes o tempo base (simulação com passo fixo de 0.001 segs), chegando a 17.9 vezes se o algoritmo de variação automática do passo de integração for utilizado. A redução no tempo de simulação obtida com o algoritmo de variação do passo de integração tornou o tempo de processamento do método simultâneo implícito (3.89 segs) compatível com o obtido com programa ANATEM (3.01 segs), para o sistema S0188. Entretanto maiores ganhos podem ser

---

obtidos com algoritmos mais sofisticados de variação do passo de integração [48][49][50] e/ou para simulações de mais longo prazo, onde passos de integração maiores podem ser utilizados. A Figura 87 mostra que o erro introduzido com o aumento do passo de integração não compromete os resultados das simulações, neste exemplo foram plotados o ângulo de um dos geradores do sistema *S0188* para uma simulação com passo de integração de 0.001 segs e para uma simulação com passo de integração 0.010 segs (10 vezes maior que o primeiro).

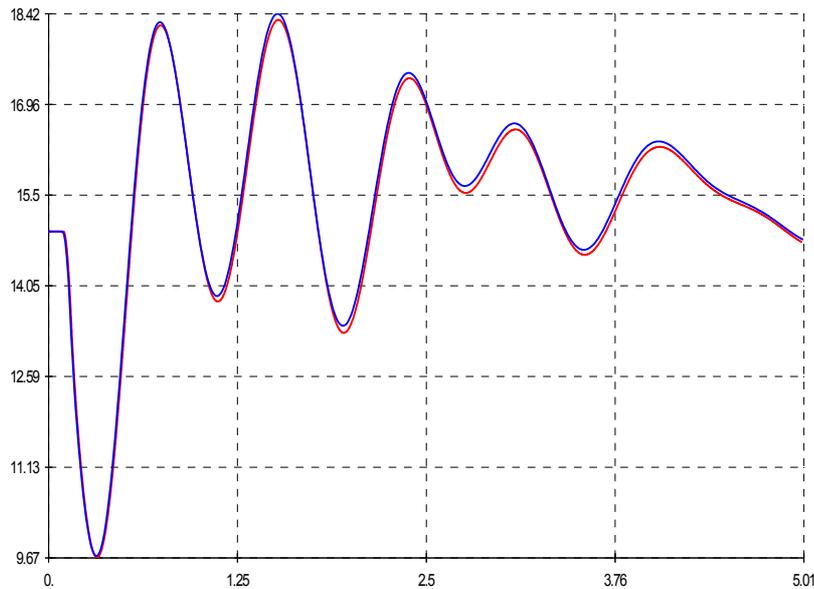


Figura 87 – Ângulos do Gerador para  $\Delta t = 1$  mseg e  $\Delta t = 10$  mseg (*SIMSEEs++*)

#### 5.4.4 Simulação de Longo Prazo para o Sistema Brasileiro

Para demonstrar a robustez e o desempenho computacional da ferramenta de simulação rápida em estudos de longo prazo será aplicada uma curva de carga diária típica a uma região do sistema interligado brasileiro e realizada uma simulação de 24 horas. O sistema simulado é composto de 2806 barras, 3931 circuitos e 296 geradores, sendo 77 destes representados por seus modelos reais de máquinas síncronas e controladores. A região onde a curva de carga será aplicada é a área do Rio de Janeiro (área Light), sendo a curva de carga discretizada a intervalos de 1 minuto (tempo também utilizado como passo de simulação). Todas as cargas foram representadas através do seu modelo de impedância constante.

Os resultados obtidos para a simulação da curva diária de carga no sistema brasileiro são apresentados na Figura 88 e Figura 89.

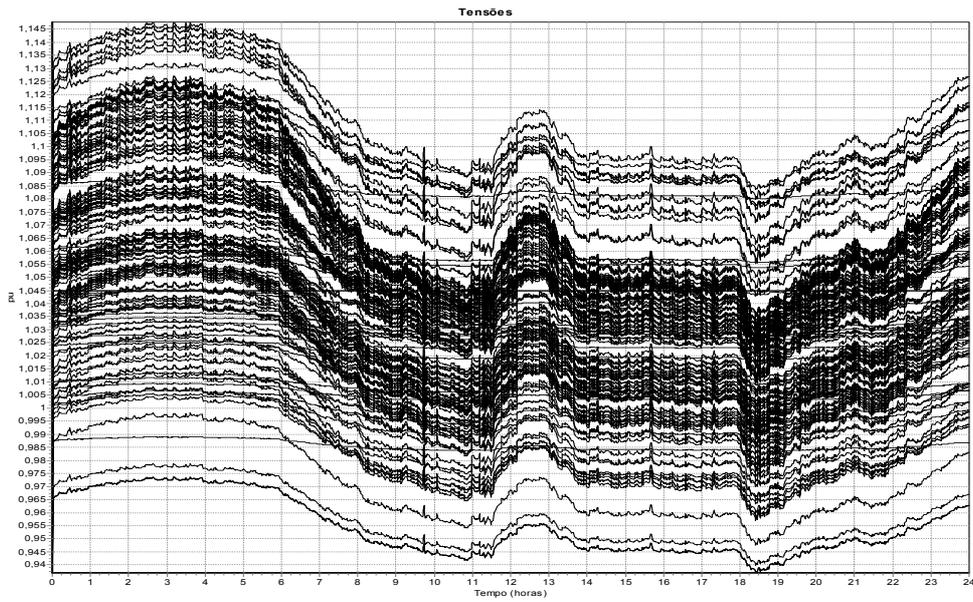


Figura 88 – Tensões das Principais Barras do Sistema (*FASTSIM++*)

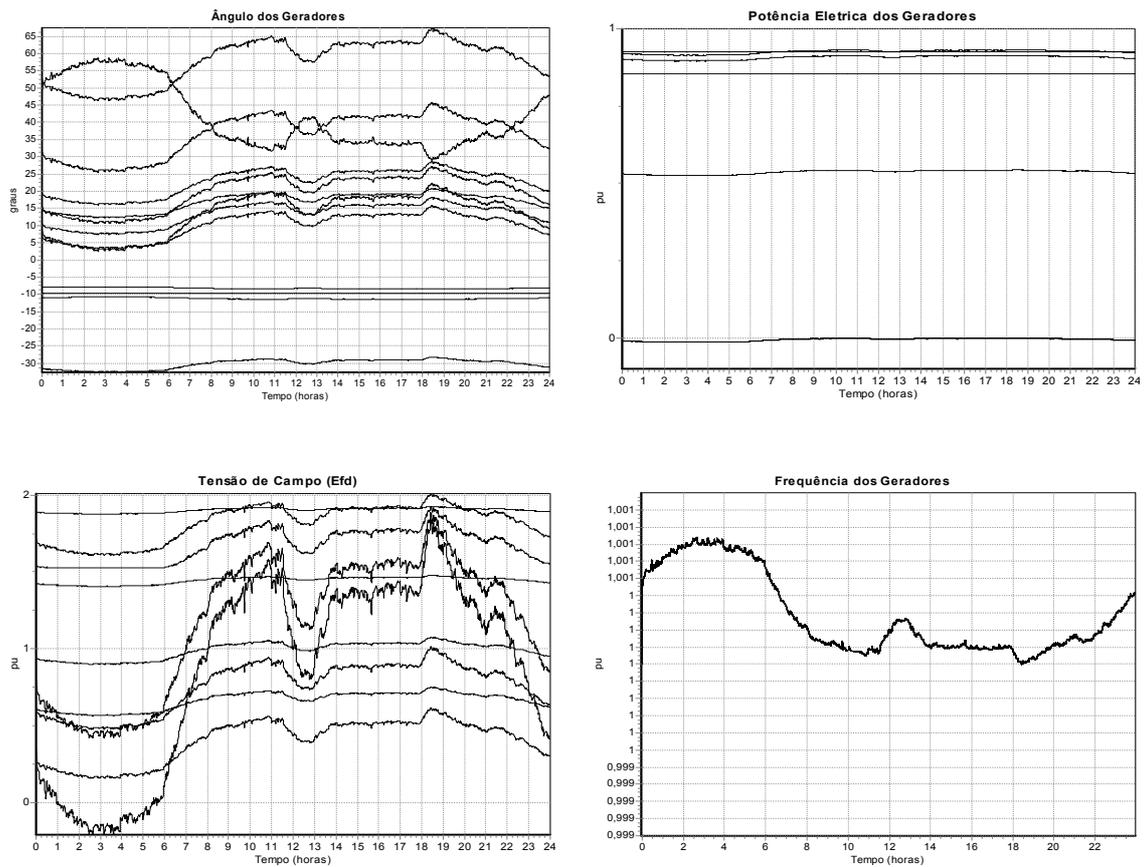


Figura 89 – Grandezas Elétricas de Alguns Geradores do Sistema (*FASTSIM++*)

---

Os resultados obtidos com o *FASTSIM++* para o sistema brasileiro mostraram um desempenho computacional bastante satisfatório da metodologia de simulação rápida, conforme mostra a Tabela 9.

Tabela 9 - Desempenho do Sistema Brasileiro com Modelos Reais

Tempo de CPU	
segs	min.
427	7.1

O resultados de desempenho computacional mostraram que a metodologia de simulação rápida executa uma simulação de 24 horas no sistema interligado brasileiro em apenas 7.1 minutos. Fazendo uma estimativa da mesma simulação para o programa ANATEM, considerando a relação de 5.21 vezes o tempo real para o sistema *S2800* (da Tabela 6), o tempo de CPU seria de aproximadamente 450144 segs para simular 24 horas, ou 5.21 dias de CPU. Nesta simulação o *FASTSIM++* é cerca de 1050 vezes mais rápido que o ANATEM.

#### 5.4.5 Simulação de Médio Prazo para o Sistema Brasileiro

Muito mais freqüentes que as simulação de longo prazo em estudos de sistemas de energia elétrica são as simulações de médio prazo, onde o horizonte de simulação abrange alguns minutos. Sendo assim, a mesma curva de carga do item anterior será aplicada a uma região do sistema interligado brasileiro, porém agora apenas os 30 minutos iniciais serão simulados. Após 10 minutos de simulação será aberta uma LT de baixo impacto no sistema (LT *CAMPOS-VITÓRIA*), permanecendo esta aberta pelo resto da simulação A região onde a curva de carga será aplicada novamente será a área do Rio de Janeiro (área Light), sendo a curva de carga é discretizada a intervalos de 1 minuto (tempo também utilizado como passo de simulação). Todas as cargas são representadas através do seu modelo de impedância constante.

Os resultados obtidos para a simulação de médio prazo no sistema brasileiro são apresentados na Figura 90 e Figura 91.

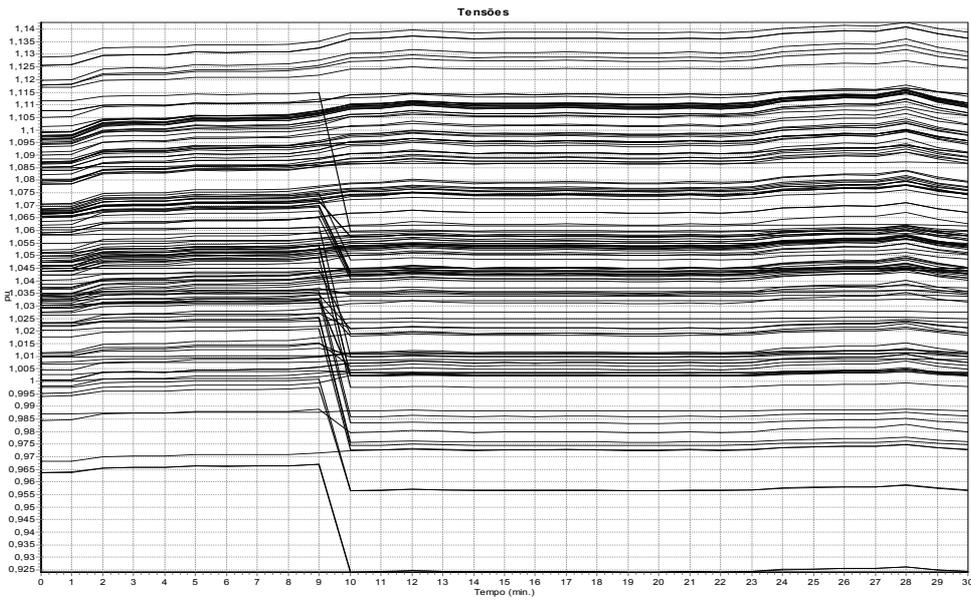


Figura 90 – Tensão das Principais Barras do Sistema (*FASTSIM++*)

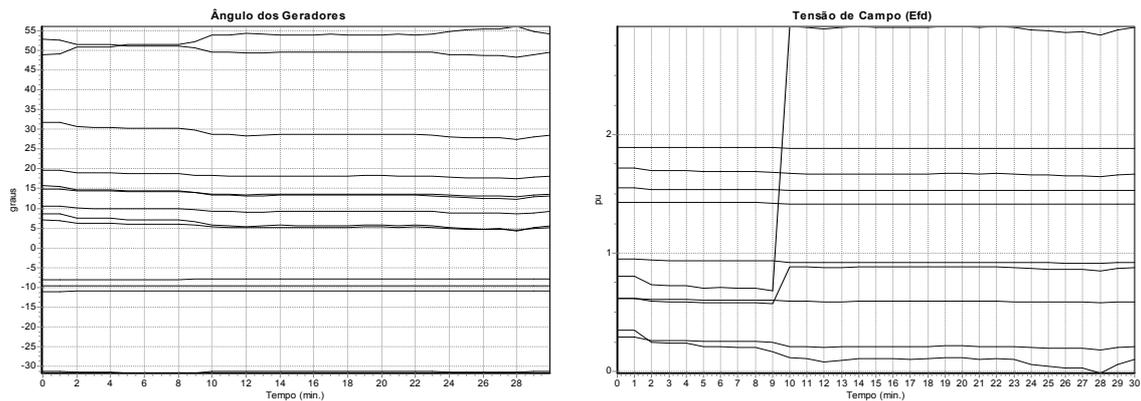


Figura 91 – Ângulo e Tensão de Campo dos Geradores (*FASTSIM++*)

O tempo de CPU para a simulação de 30 minutos no sistema interligado brasileiro é de apenas 12.86 segundos. Fazendo uma estimativa da mesma simulação para o programa ANATEM, e novamente considerando a relação de 5.21 vezes o tempo real para o sistema S2800, o tempo de CPU seria de aproximadamente 9378 segs para simular os 30 minutos, ou 2.6 horas de CPU. Nesta simulação o *FASTSIM++* é cerca de 730 vezes mais rápido que o ANATEM.

---

#### 5.4.6 Comentários Gerais

O programa de fluxo de potência generalizado foi comparado quanto ao seu desempenho computacional com duas implementações tradicionais em C++, ficando o desempenho do programa orientado a objetos cerca de 2 vezes mais lento por iteração que as implementações tradicionais. Este *overhead* é perfeitamente aceitável em virtude do ganho obtido com a generalização da metodologia no reconhecimento automático de qualquer novo modelo definido pelo usuário.

Quanto ao desempenho dos programas de simulação da dinâmica completa, o aplicativo de simulação pelo método alternado implícito apresentou um desempenho computacional aproximadamente 10 vezes mais lento que o programa ANATEM, devido principalmente ao seu caráter geral para o tratamento dos equipamentos do sistema (todos os equipamentos são CDU). O método simultâneo implícito, por sua vez, apresentou um desempenho computacional bastante inferior ao programa ANATEM se o passo de integração for mantido fixo. Entretanto, os resultados obtidos para um caso teste indicam que, se uma metodologia de variação automática do passo for utilizada o método simultâneo implícito pode apresentar resultados de desempenho compatíveis com o programa ANATEM, podendo mesmo ser superior ao ANATEM se uma metodologia sofisticada de variação do passo for adotada [48][49][50].

A metodologia de simulação rápida tornou viável a simulação de uma curva diária de carga para o sistema interligado brasileiro, permitindo simular as 24 horas da curva de carga em apenas 7.1 minutos de CPU. Nesta simulação o programa *FASTSIM++* mostrou-se aproximadamente 1050 vezes mais rápido que o programa ANATEM, no entanto esta relação pode mudar em função dos passos de integração utilizados por ambos programas.

### 5.5 Considerações Finais

Neste capítulo foram apresentados resultados relativos à aplicabilidade da modelagem orientada a objetos a sistemas de energia elétrica. Estes resultados foram viabilizados através da implementação de um ambiente de simulação e análise para sistemas elétricos usando a estrutura computacional proposta neste trabalho.

---

Vários resultados foram apresentados para um sistema exemplo com 45 barras, onde o ambiente implementado mostrou-se adequado para as diversas simulações e análises executadas em uma ampla faixa de eventos e situações usuais de sistemas elétricos, comprovando assim a capacidade e a flexibilidade do ambiente implementado. Com base nos resultados apresentados mostrou-se que a utilização do modelo orientada a objetos proposto neste trabalho é perfeitamente viável para aplicações em sistemas de energia elétrica.

Foram apresentados resultados também de desempenho computacional dos programas orientados a objetos para vários sistemas de diferentes portes. A análise comparativa do desempenho dos programas orientados a objetos com programas em implementações tradicionais mostrou um *overhead* adicional nos aplicativos orientados a objetos, embora este seja perfeitamente aceitável em função da enorme flexibilidade e agilidade obtida com os modelos orientados a objetos para sistemas de energia elétrica.

---

# Capítulo 6

## Conclusões

### 6.1 Considerações Gerais

O trabalho de tese descrito neste documento buscou a modelagem de sistemas de energia elétrica através do paradigma da modelagem orientada a objetos. Através desta nova forma de projetar programas computacionais foi projetado e implementado um *framework* orientado a objetos capaz de servir como base única para a construção e integração de diversos aplicativos na área de sistemas de energia. O *framework* proposto dita a arquitetura dos novos aplicativos, define sua estrutura geral, sua divisão em classes e como estas colaboram, permitindo construir aplicações mais rapidamente e com estrutura similar (mais legíveis e consistentes). O modelo mostrou-se adequado para acomodar aplicativos tais como fluxo de potência, análise modal, simulação dinâmica completa e simulação rápida, bem como compartilhar resultados entre estes aplicativos, possibilitando assim que ferramentas mais complexas como as de avaliação de segurança possam facilmente ser implementadas. A utilização do *framework* para a construção de aplicativos na área de sistemas de energia elétrica permite que o engenheiro possa dar maior atenção ao desenvolvimento de aspectos práticos e teóricos do problema específico de engenharia a ser resolvido, uma vez que a estrutura computacional base já está disponível e pronta para uso.

Adicionalmente, obteve-se um grau de generalização para os aplicativos que permitem que um novo equipamento ou modelo seja adicionado ao sistema e assimilado automaticamente por todo o elenco de aplicativos construído sobre o *framework*. Para isso foi projetada uma interface de utilização dos modelos que padroniza a execução de ações como armazenamento dos parâmetros e equações,

---

cálculo das condições iniciais e solução/derivação das equações do modelo. Assim, os aplicativos sabem como executar estas tarefas sem o conhecimento prévio do tipo de dispositivo que está associado ao modelo. O mecanismo de solução e derivação das equações dos modelos permite ainda que um determinado aplicativo possa ser formulado como injeção de potência ou corrente na rede elétrica, e em coordenadas polares ou retangulares para as tensões, cabendo ao modelo orientado a objetos todas as conversões necessárias para a troca de formulação. Algoritmos generalizados quanto a formulação permitem que o usuário defina qual o tipo de injeção na rede elétrica é mais adequada e/ou qual o tipo de coordenadas para as tensões deve ser utilizado. Assim, por exemplo, se um aplicativo de Fluxo de Potência apresenta problemas de convergência com a formulação tradicional (injeção de potência / coordenadas polares), o usuário pode alterar a formulação do aplicativo para injeção de corrente e coordenadas retangulares, sem que um novo aplicativo tenha que ser implementado para isso.

O *framework* proposto conta ainda com um *toolkit* para suporte matemático dos aplicativos. Este *toolkit* é responsável pelo armazenamento e gerenciamento de matrizes e vetores esparsos e de grande porte, bem como solução de sistemas de equações lineares do mesmo tipo. Um amplo conjunto de ferramentas matemáticas está também disponível no *toolkit*, tais como operações elementares entre matrizes e entre matriz e vetor, cálculo do determinante, inversa parcial, autovalores, autovetores e decomposição em valores singulares para matrizes simétricas e não-simétricas. O *toolkit* pode ainda ser utilizado em qualquer aplicação, não necessariamente na área de sistemas elétricos.

Em uma segunda etapa do trabalho, foi proposta uma modificação na metodologia de simulação rápida das dinâmicas de médio e longo prazo de sistemas elétricos. Esta modificação consiste na retenção da dinâmica dos modelos na formulação do método modificado possibilitando que técnicas de análise modal possam ser utilizadas durante a simulação da trajetória do sistema, detectando assim fenômenos que a metodologia tradicional é incapaz de detectar, tais como a ocorrência de alguns tipos de bifurcações locais que podem sinalizar instabilidades no sistema. A metodologia de simulação rápida modificada permite um estudo mais amplo dos fenômenos que ocorrem em cenários de instabilidade de tensão. Adicionalmente, e em virtude da flexibilidade obtida com a estrutura computacional, foi proposta uma metodologia de simulação que reuniu o simulador rápido modificado e

---

um simulador da dinâmica completa em uma ferramenta única de simulação, denominada simulação combinada. Nesta metodologia é possível conduzir a simulação com a metodologia rápida durante o intervalo de tempo em que as dinâmicas lentas ditam a trajetória do sistema, chaveando para a simulação completa sempre que for detectado um distúrbio de grande impacto no sistema durante um intervalo de tempo suficiente para a detecção da estabilidade da trajetória, quando então a metodologia pode novamente ser chaveada para a simulação rápida para o próximo intervalo de simulação.

Quanto a performance computacional dos aplicativos orientados a objetos implementados, foram apresentados no Capítulo 5 vários resultados do desempenho destes aplicativos comparados com programas em implementações tradicionais para vários sistemas de diferentes tamanhos. Os resultados mostraram que o programa de fluxo de potência generalizado, quando comparado com duas implementações tradicionais em C<sup>++</sup>, é cerca de 2 vezes mais lento, por iteração, que as implementações tradicionais. Para estudos de estabilidade transitória, dois programas de simulação da dinâmica completa foram comparados ao programa ANATEM do CEPEL. O primeiro, utilizando o método alternado implícito, apresentou um desempenho computacional aproximadamente 10 vezes mais lento que o programa ANATEM, devido principalmente ao seu caráter geral para o tratamento dos equipamentos do sistema, e o segundo, utilizando o método simultâneo implícito, apresentou um desempenho computacional que chegou a ser 51 vezes mais lento que o programa ANATEM se o passo de integração for mantido fixo. Entretanto, se uma metodologia de variação automática do passo for utilizada o método simultâneo implícito pode apresentar resultados de desempenho compatíveis com o programa ANATEM, podendo mesmo ser superior se uma metodologia sofisticada de variação do passo for adotada. De maneira geral todos os aplicativos implementados apresentaram um *overhead* adicional em relação aos programas convencionais, no entanto este *overhead* é perfeitamente aceitável em função da enorme flexibilidade e agilidade obtida com os modelos orientados a objetos em sistemas de energia elétrica. Por fim, uma simulação de longo prazo foi executada para o sistema interligado brasileiro, onde o aplicativo de simulação rápida tornou viável a simulação de uma curva diária de carga de 24 horas de duração levando apenas 7.1 minutos de tempo de CPU. Assim, este trabalho mostrou que a utilização do modelo orientado a objetos proposto é perfeitamente viável para aplicações em sistemas de energia elétrica.

---

Até a presente data vários trabalhos de mestrado e doutorado tem utilizado o *framework* orientado a objetos implementado como base de desenvolvimento para aplicativos em sistemas de energia elétrica. Pode-se citar trabalhos nas áreas de:

- Fluxo de potência ótimo, com função objetivo e restrições totalmente definidas pelo usuário;
- Avaliação da máxima transferência de potência entre áreas de um sistema de energia elétrica;
- Distribuição de energia elétrica, com aplicações em: fluxo de potência pelo método de soma de potências, reconfiguração ótima de alimentadores, alocação de bancos de capacitores e reguladores de tensão.

## **6.2 Sugestões para Futuros Trabalhos**

São apresentadas algumas sugestões de possíveis trabalhos futuros, visando a continuidade da pesquisa iniciada neste trabalho de tese:

- Projetar e implementar uma interface gráfica, aderente a estrutura orientada a objetos implementada, de tal forma que todos os níveis de descrição do sistema possam ser visualizados e gerenciados graficamente;
- Incluir o sistema de proteção no modelo orientado a objetos;
- Investigar a viabilidade da utilização de cálculo parcial de autovalores para otimizar a performance do método de simulação rápida modificado com avaliação da trajetória dos autovalores do sistema;
- Implementar um esquema de variação automática do passo de integração para a metodologia de simulação da dinâmica completa pelo método simultâneo implícito e para a metodologia de simulação rápida;
- Investigar a viabilidade da utilização de técnicas de avaliação automática da estabilidade transitória para automatizar alguns aspectos do algoritmo de simulação combinada.

---

## Referências Bibliográficas

- [1] J. RUMBAUGH, M. BLAHA, W. PREMERLANI, F. EDDY, W. LORENSEM, *Modelagem e Projetos Baseados em Objetos*, Ed. Campus, 1994.f
- [2] J. D. FURLAN, *Modelagem de Objetos através da UML – Análise e Desenho Orientados a Objeto*, Makron Books, 1998.
- [3] G. BOOCH, J. RUMBAUGH, IVAR JACOBSON, *The Unified Modeling Language – User Guide*, Addison-Wesley, 1999.
- [4] E. GAMMA, R. HELM, R. JOHNSON, J. VLISSIDES, *Padrões de Projeto – Soluções Reutilizáveis de Software Orientado a Objetos*, Ed. Bookman, 2000.
- [5] A. F. NEYER, F. F. WU, K. IMHOF, “Object-Oriented Programming for Flexible Software : Example of a Load Flow”, *IEEE Transactions on Power Systems*, Vol. 5, No. 3, pg. 689-696, August 1990.
- [6] D. RUMPEL, S.S. VENKATA, K. PANDJI, C.C. LIU, N.M. NAGDY, “Real-Time Database for Power Systems using Language Oriented Data Structure”, *IEEE Transactions on Power Systems*, Vol. 5, No. 3, pp. 993-1000, August 1990.
- [7] D.G. FLINN, R.C. DUGAN, “A Database for Diverse Power System Simulation Applications”, *IEEE Transactions on Power Systems*, Vol. 7, No. 2, pp. 784-790, May 1992.
- [8] S. N. IROMONGER, M.J. BUSHNELL, R. PATEL, M.E. BRADLEY AND B.W. VAUGHAN, “An Object-Oriented Power System Model and Graphical Information Display System for Control Engineers”, *Power System Control and Management*, Conference Publication No. 421, IEE, pp-120-124, April 1996.
- [9] M. FOLEY, A. BOSE, W. MITCHELL, A. FAUSTINI, “An Object Based Graphical User Interface for Power Systems”, *IEEE Transactions on Power Systems*, Vol. 8, No. 1, pp. 97-104, November 1993.

- 
- [10] M. FOLEY, A. BOSE, "Object-Oriented on-line Network Analysis", *IEEE Transactions on Power Systems*, Vol. 10, No. 1, February 1995.
- [11] S. LI AND S. M. SHAHIDEPOUR, "An Object Oriented Power System Graphics Package for Personal Computer Enviroment", *IEEE Transactions on Power Systems*, Vol. 8, No. 3, pp. 1054-1060, August 1993.
- [12] B. HAKAVIK, A. T. HOLEN, "Power System Modelling and Sparse Matrix Operations Using Object-Oriented Programming", *IEEE Transactions on Power Systems*, Vol. 9, No. 2, pg. 1045-1051, May 1994.
- [13] M. R. V. VANTI, *Implementação Orientada para Objeto de um Simulador para Dinâmica Lenta de um Sistema de Energia Elétrica*, Dissertação de Mestrado, UFSC, Novembro de 1994.
- [14] E. Z. ZHOU, "Object-Oriented Programming, C++ and Power System Simulation", *IEEE/PES Winter Meeting*, paper 95 SW 219-5 PWRS, January/February 1995.
- [15] Z.L. GAING, C.N. LU, B.S. CHANG, C.L. CHENG, "An Object-Oriented Approach for Implementing Power System Restoration Package", *IEEE Transactions on Power Systems*, Vol. 11, No. 1, pp. 483-489, February 1996.
- [16] N.B.P PHILLIPS, J.O. GANN AND M.R. IRVING, "The Simian Architecture - An Object Orientated Framework for Integrated Power System Modelling Analysis and Control", *Power System Control and Management*, Conference Publication No. 421, IEE, pp-148-153, April 1996.
- [17] J. ZHU, D. LUBKEMAN, "Object-Oriented Development of Software Systems for Power System Simulations", *IEEE Transactions on Power Systems*, Vol. 12, No. 2, pp. 1002-1007, May 1997.
- [18] E. HANDSCHIN, M. HEINE, D. KÖNIG, T. NIKODEM, T. SEIBT, R. PALMA, "Object-Oriented Software Engineering for Transmission Planning in Open Access Schemes", *IEEE Transactions on Power Systems*, Vol. 13, No. 1, pp. 94-100, February 1998.
- [19] C.R. FUERTE-ESQUIVEL, E. ACHA, S.G. TAN AND J.J. RICO, "Efficient Object Oriented Power Systems Software for the Analysis of Large-Scale Networks

---

Containing FACTS-Controlled Branches”, *IEEE Transactions on Power Systems*, Vol. 13, No. 2, pp. 464-472, 1998.

- [20] S.J. LEE, S.I. LIM, B.S. AHN, “Service Restoration of Primary Distribution Systems Based on Fuzzy Evaluation of Multi-Criteria”, *IEEE Transactions on Power Systems*, Vol. 13, No. 3, pp. 1156-1163, August 1998.
- [21] J. ZHU, P. JOSSMAN, “Application of Design Patterns for Object-Oriented Modeling of Power Systems”, *IEEE Transactions on Power Systems*, Vol. 14, No. 2, pp. 532-537, 1999.
- [22] M.E. BRADLEY, M.J. BUSHNELL, S.I. MACLEAN, “Object-Oriented Creation of Fault Sequences for On-Line Transient Stability Analysis”, *13<sup>th</sup> PSCC – Power Systems Computation Conference*, Trondheim, pp. 654-660, June/July 1999.
- [23] A. MANZONI, *Desenvolvimento de um Módulo Dinâmico para Simuladores de Ensino e Treinamento em Sistemas de Energia Elétrica Usando Programação Orientada a Objetos*, Dissertação de Mestrado, UFSC, Junho de 1996.
- [24] A. MANZONI, A. S. E SILVA, I. C. DECKER, “Power Systems Dynamics Simulation Using Object-Oriented Programming”, *IEEE Transactions on Power Systems*, Vol. 14, No. 1, pp. 249-255, February 1999.
- [25] L.R. ARAUJO, J.L.R. PEREIRA, “Solução de Redes Elétricas de Grande Porte, usando Programação Orientada a Objetos”, *XIII Congresso Brasileiro de Automática – CBA 2000*, Florianópolis – SC, Brasil. pp. 604-609, Setembro de 2000.
- [26] M.N. AGOSTINI, I.C. DECKER, A.S. E SILVA, “Desenvolvimento e Implementação de uma Base Computacional Orientada a Objetos para Aplicações em Sistemas de Energia Elétrica”, *XIII Congresso Brasileiro de Automática – CBA 2000*, Florianópolis – SC, Brasil. pp. 1850-1856, Setembro de 2000.
- [27] S. PANDIT, S.A. SOMAN, S.A. KHAPARDE, “Object-Oriented Design for Power System Applications”, *IEEE Computer Applications in Power*, pp. 43-47, October 2000.
- [28] S. PANDIT, S.A. SOMAN, S.A. KHAPARDE, “Object-Oriented Network Topology Processor”, *IEEE Computer Applications in Power*, pp. 42-46, April 2001.

- 
- [29] D. BECKER, H. FALK, J. GILLERMAN, S. MAUSER, R. PODMORE, L. SCHNEBERGER, "Standards-Based Approach Integrates Utility Applications", *IEEE Computer Applications in Power*, pp. 13-20, October 2000.
- [30] S. PANDIT, S.A. SOMAN, S.A. KHAPARDE, "Design of Generic Direct Sparse Linear System Solver in C++ for Power System Analysis", *IEEE Transactions on Power Systems*, Vol. 16, No. 4, pp. 647-652, November 2001.
- [31] M.N. AGOSTINI, J.M.F. FERREIRA, I.C. DECKER, A.S. E SILVA, "Object Oriented Matrix Structure for the Development of Computing Tools in Electric Power Systems", *VIII Simpósio de Especialistas em Planejamento da Operação e Expansão Elétrica – SEPOPE*, IP-108, Brasília – Brasil, Maio de 2002.
- [32] L.R. ARAUJO, P.A.N. GARCIA, J.L.R. PEREIRA, "Modelagem Orientada a Objetos Aplicada na Solução de Programas de Distribuição", *XIV Congresso Brasileiro de Automática*, Natal – Brasil, pp. 844-848, Setembro de 2002.
- [33] M.N. AGOSTINI, I.C. DECKER, A.S. E SILVA, "Nova Filosofia para o Projeto de Software na Área de Sistemas de Energia Elétrica utilizando Modelagem Orientada a Objetos", *XIV Congresso Brasileiro de Automática*, Natal – Brasil, pp. 1555-1561, Setembro de 2002.
- [34] M.N. AGOSTINI, *Nova Filosofia para o Projeto de Software para Sistemas de Energia Elétrica usando Modelagem Orientada a Objetos*, Tese de Doutorado, UFSC, 2002.
- [35] D. DOTTA, *Modelagem e Implementação de Aplicações Usando uma Base Computacional Orientada a Objetos para Sistemas de Energia Elétrica*, Dissertação de Mestrado, UFSC, 2003.
- [36] A.I. MURCIA CABRA, *Estudos sobre a Configuração das Redes Elétricas: Avaliação de Diversos Algoritmos e Novas Porposições para Implementação em Tempo Real*, Dissertação de Mestrado, UFSC, 1988.
- [37] P. KUNDUR, *Power System Stability and Control*. 1 ed. New York, McGraw-Hill Inc., 1994.
- [38] T. VAN CUTSEM, C. VOURNAS, *Voltage Stability of Electrical Power Systems*. 1 ed. Kluwer Academic Publishers, 1998.
-

- 
- [39] W.J. CAUSARANO, *Método de Simulação Rápida no Tempo para Avaliação da Estabilidade de Tensão*, Dissertação de Mestrado, COPPE/UFRJ, 1997.
- [40] C.B. BORGES, *Implementação de Funções Utilizadas no Controle Coordenado de Tensão num Simulador Rápido*, Dissertação de Mestrado, COPPE/UFRJ, 2001.
- [41] F.A.B. LEMOS, *Determinação da Instabilidade de Tensão Associada à Bifurcação Sela-Nó*, Tese de Doutorado, UFSC, 2000.
- [42] T. VAN CUTSEM, Y. JACQUEMART, J.N. MARQUET, P. PRUVOT, "A Comprehensive Analysis of Mid-Term Voltage Stability", *IEEE Transactions on Power Systems*, Vol. 10, pp. 1173-1182, 1995.
- [43] T. VAN CUTSEM, C.D. VOURNAS, "Voltage Stability Analysis in Transient and Mid-Term Time Scales", *IEEE Transactions on Power Systems*, Vol. 11, pp. 1173-1182, 1996.
- [44] B.H. CHOWDHURY, C.W. TAYLOR, "Voltage Stability Analysis: V-Q Power Flow Simulation Versus Dynamic Simulation", *IEEE Transactions on Power Systems*, Vol. 15, pp. 1354-1359, 2000.
- [45] C. CAÑIZARES, ed., *Voltage Stability Assessment, Procedures and Guides*, IEEE/PES Power System Stability Subcommittee Special Publication, Final Draft, January 2001, disponível em [www.power.uwaterloo.ca](http://www.power.uwaterloo.ca).
- [46] C. W. TAYLOR, *Power System Voltage Stability*, New York, McGraw-Hill, 1994.
- [47] B. STOTT, "Power System Dynamic Response Calculations", *Proceedings of the IEEE*, Vol. 67, No. 2, February 1979, pg. 219-241.
- [48] J.Y. ASTIC, A. BIHAIN, M. JEROSOLIMSKI, "The Mixed Adams-BDF Variable Step Size Algorithm to Simulate Transient and Long Term Phenomena in Power Systems", *IEEE Transactions on Power Systems*, Vol. 9, No. 2, pp. 929-935, May 1994.
- [49] J.L. JARDIM, X.V. FILHO, "Long Term Dynamics: Its Perspective in Brazil", *V Simpósio de Especialistas em Planejamento da Operação e Expansão Elétrica – SEPOPE*, Recife – Brasil, Maio de 1996.

- 
- [50] J.L. JARDIM, C.A. NETO, M.G DOS SANTOS, P. GOMES, "Design Features of a Dynamic Security Assessment System", *IX Simpósio de Especialistas em Planejamento da Operação e Expansão Elétrica – SEPOPE*, SP-038, Rio de Janeiro – Brasil, Maio de 2004.
- [51] A. A. P. LERM, C. A. CAÑIZARES, F. A. B. LEMOS, AND A. S. E SILVA, "Multi-parameter Bifurcation Analysis of Power Systems," *Proceedings of the North American Power Symposium*, Cleveland, Ohio, October 1998.
- [52] V. VENKATASUBRAMANIAN, H. SCHÄTTLER AND J. ZABORSZKY, "Local Bifurcations and Feasibility Regions in Differential-Algebraic Systems, *IEEE Transactions on Automatic Control*, Vol. 40, No. 12, pp. 1992-2013, December 1995.
- [53] C. RAJAGOPALAN, B. LESIEUTRE, P.W. SAUER, M.A. PAI, "Dynamic Aspects of Voltage/Power Characteristics", *IEEE Transactions on Power Systems*, Vol. 7, pp. 990-1000, 1992.
- [54] T. ATHAY, R. PODMORE, S. VIRMANI, "A Practical Method for Direct Analysis of Transient Stability" *IEEE Transactions on Power Apparatus & Systems*, Vol. PAS-98, 1979."
- [55] Y. XUE, T. VAN CUTSEM, M. RIBBENS-PAVELLA, "A Simple Direct Method for Fast Transient Stability Assessment of Large Power Systems", *IEEE Transactions on Power Systems*, Vol. 3, No. 2, pp. 400-412, May 1988.
- [56] I.C. DECKER, L.G.S. FONSECA, "Iterative Algorithm for Critical Energy Determination in Transient Stability of Power Systems", *IFAC - Symposium of Planning and Operation of Electrical Energy Systems*, Rio de Janeiro, Brasil, pp. 483-489, 1985.
- [57] J.L. JARDIM, B. CORY, N. MARTINS, "Efficient Transient Stability Assessment Using Transient Energy Function", *13<sup>th</sup> PSCC – Power Systems Computation Conference*, Trondheim, pp. 661-668, June/July 1999.
- [58] M. LA SCALA, G. LORUSSO, R. SBRIZZAI, M. TROVATO, "A Qualitative Approach to the Transient Stability Analysis", *IEEE Transactions on Power Systems*, Vol. 11, No. 4, pp. 1996-2002, November 1996.

- 
- [59] M. LA SCALA, R. SBRIZZAI, F. TORELLI, P. SCARPELLINI, "A Tracking Time Domain Simulator for Real-Time Transient Stability Analysis", *IEEE Transactions on Power Systems*, Vol. 13, No. 3, pp. 992-998, August 1998.
- [60] M. JEROSOLIMSKI, L. LECACHER, "A New Method For Fast Calculation of Jacobian Matrices: Automatic Differentiation for Power System Simulation", *IEEE Transactions on Power Systems*, Vol. 9 , No. 2, pp. 700-706, May 1994.
- [61] A. GRIEWANK, *Evaluating Derivatives – Principles and Techniques of Algorithmic Differentiation*, Society for Industrial and Applied Mathematics, 2000.
- [62] A. J. MONTICELLI, *Fluxo de Carga em Redes de Energia Elétrica*, Edgar Blücher LTDA, 1983.