UNIVERSIDADE ESTADUAL PAULISTA
"JÚLIO DE MESQUITA FILHO"
Campus de Ilha Solteira

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

"Desenvolvimento de um Objeto de Aprendizagem Para Análise de Sistemas de Energia Elétrica"

MARCIA BEATRIZ CARVALHO PEREIRA

Orientador: Prof. Dr. Laurence Duarte Colvara

Dissertação apresentada à Faculdade de Engenharia - UNESP – Campus de Ilha Solteira, para obtenção do título de Mestre em Engenharia Elétrica.
Área de Conhecimento: Automação

Ilha Solteira – SP
Fevereiro/2008

FICHA CATALOGRÁFICA

Elaborada pela Seção Técnica de Aquisição e Tratamento da Informação
Serviço Técnico de Biblioteca e Documentação da UNESP - Ilha Solteira.

Pereira, Marcia Beatriz Carvalho
P436d Desenvolvimento de um objeto de aprendizagem para análise de sistemas de energia elétrica / Marcia Beatriz Carvalho Pereira. -- Ilha Solteira : [s.n.], 2008
102 p. : il.

Dissertação (mestrado) - Universidade Estadual Paulista. Faculdade de Engenharia de Ilha Solteira. Área de conhecimento: Automação, 2008

Orientador: Laurence Duarte Colvara
Bibliografia: p. 87-89

1. Estabilidade transitória. 2. Programação orientada a objetos (Computação). 3. Objeto de aprendizagem.

unesp **UNIVERSIDADE ESTADUAL PAULISTA**
CAMPUS DE ILHA SOLTEIRA
FACULDADE DE ENGENHARIA DE ILHA SOLTEIRA

CERTIFICADO DE APROVAÇÃO

TÍTULO: DESENVOLVIMENTO DE UM OBJETO DE APRENDIZAGEM PARA ANÁLISE DE SISTEMAS DE ENERGIA ELÉTRICA

AUTORA: MARCIA BEATRIZ CARVALHO PEREIRA
ORIENTADOR: Prof. Dr. LAURENCE DUARTE COLVARA

Aprovada como parte das exigências para obtenção do Título de MESTRE em ENGENHARIA ELÉTRICA pela Comissão Examinadora:

Prof. Dr. LAURENCE DUARTE COLVARA
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira

Prof. Dr. SERGIO AZEVEDO DE OLIVEIRA
Departamento de Engenharia Elétrica / Faculdade de Engenharia de Ilha Solteira

Profa. Dra. SILVIA GALVÃO DE SOUZA CERVANTES
Departamento de Engenharia Elétrica / Universidade Estadual de Londrina

Data da realização: 22 de fevereiro de 2008.

Presidente da Comissão Examinadora
Prof. Dr. LAURENCE DUARTE COLVARA

DEDICATÓRIA

Dedico este trabalho a meus pais, Davi Dutra Pereira e Márcia Carvalho Pereira, aos meus irmãos, Fábio e Naíra e a meu marido Sérgio, pelo amor, carinho, apoio e confiança.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que nos momentos difíceis me deu forças para superar os obstáculos e continuar a caminhar.

Aos meus pais que tanto amo, que muito se empenharam em me proporcionar mais essa conquista profissional.

Ao meu marido, que soube compreender meus momentos de ausência e as madrugadas no computador empenhada nesse trabalho.

Aos meus amigos de sempre Alexsandro dos Santos, Anderson Deizepe, Guilherme Judice, William Yonenaga, pela ajuda pronta e incondicional, incentivo e torcida de sempre.

Ao meu orientador Laurence Duarte Colvara que soube com paciência e sabedoria me conduzir durante esse trabalho. Agradeço de coração tudo o que me ensinou e a atenção que teve comigo.

Ao professor Messias Meneguette Júnior pelo apoio e incentivo de sempre.

Ao professor Sérgio Azevedo de Oliveira pelas observações e contribuições dadas para esse trabalho.

Aos professores Percival Bueno de Araújo e Marcelo Carvalho Minhoto Teixeira pelo apoio e incentivo durante o curso das disciplinas.

Aos colegas do DEE, Adriano, André, Eduardo, Jaine, Jair, Hélio, Márcio, Marcelo e Regiane, pela disposição em me ajudar quando precisei.

Agradeço a todas as pessoas que direta ou indiretamente contribuíram de alguma forma para a conclusão deste trabalho.

“Vem, vamos embora
Que esperar não é saber
Quem sabe faz a hora
Não espera acontecer”

Geraldo Vandré

RESUMO

Este trabalho tem por objetivo contribuir para o desenvolvimento de técnicas de Educação em Engenharia apresentando uma ferramenta de apoio ao Ensino/Aprendizagem de Sistemas Elétricos de Potência na forma de um software de interface amigável e baixo custo. O software desenvolvido em linguagem C++ segue os padrões de programação orientada a objetos e oferece ao usuário a resolução de fluxos de potência e simulações de casos de contingências transitórias com possibilidade de interatividade, apresentando resultados em ambiente gráfico.

Desenvolvido para aplicação na área de Dinâmica e Estabilidade de Sistemas de Potência, suporta sistemas-exemplos clássicos do IEEE e também disponibiliza ao usuário ferramentas para personalização de arquivos de dados de entrada, tornando-se flexível e de fácil acesso aos casos já inseridos no sistema. O usuário também pode criar novos sistemas de acordo com suas necessidades. Os resultados são exibidos em forma numérica e também através de gráficos tridimensionais e animação, possibilitando fácil visualização de resultados, entendimento de fenômenos e análise de influências sobre o desempenho.

Palavras Chave: Sistemas de Potência, Estabilidade Transitória, fluxo de potência, Programação orientada a objetos, Objeto de Aprendizagem.

ABSTRACT

This work presents a contribution to the development of Engineering Education techniques by means of a Teaching/Learning support tool for Electrical Power System studies resulting in a friendly low cost interface software. It was developed in C++ language and follows the pattern of object oriented programming, offering to the user the resolution of load flow and digital simulations of transient stability with the possibility of interactivity, showing the results in a graphic environment.

It was developed for use in applications of Static and Transient Power Systems Stability and it has enclosed classic sample-systems of IEEE and others and also allows the user to customize input data files. Then the tool is flexible and enables easy access to cases formerly inserted in the system. The user also can create new systems according to his needs. The machines rotors movement can be seen in a graphic animation, allowing easy visualization of events, phenomena comprehension and analysis of influences about the performance.

Key words: Power Systems, Transient Stability, Load flow, Object-oriented programming, Educational Object.

LISTA DE FIGURAS

SUMÁRIO

CAPITULO 1

INTRODUÇÃO

O Ministério da Educação do Brasil através do programa Rede Internacional Virtual de Educação RIVED (2007) tem incentivado a produção de conteúdos pedagógicos digitais, na forma de objetos de aprendizagem. Além de promover a produção e publicar na *web* os conteúdos digitais para acesso gratuito, o RIVED realiza capacitações sobre a metodologia para produzir e utilizar os objetos de aprendizagem nas instituições de ensino superior e na rede pública de ensino.

Os objetos de aprendizagem (OA) podem ser compreendidos como “qualquer recurso digital que possa ser utilizado para o suporte ao ensino” (WILEY, 2000). Os estudos sobre OA são recentes, de forma que não há um consenso universal sobre seu conceito e sua definição. Os OA podem ser criados em qualquer mídia ou formato, podendo ser simples como uma animação ou uma apresentação em slides ou complexos como uma simulação em linguagem orientada a objetos. Não há limite de tamanho para um Objeto de Aprendizagem, porém existe o consenso de que ele deve ter um propósito educacional definido, um elemento que estimule a reflexão do estudante e que sua aplicação não se restrinja a um único contexto.

A flexibilidade é uma das principais características dos Objetos de Aprendizagem que são construídos de forma a ser reutilizados, a pouco custo de manutenção. Também são facilmente atualizados e customizados para adequação das exigências do curso pretendido e da instituição estudada.

De acordo com Sá Filho e Machado (2004) a informática pode ser um recurso auxiliar para a melhoria do processo de ensino e aprendizagem, no qual o foco da educação passa a ser o aluno, construtor de novos conhecimentos, em um ambiente Construcionista, Contextualizado e Significativo definido por Valente (2002) como um ambiente favorável que desperta interesse do aluno e o motive a explorar, a pesquisar, a descrever, a refletir e a depurar suas idéias.

Uma das formas de resolver os problemas que nascem na sala de aula e abrangem o cotidiano de diversas áreas comerciais e científicas é o uso de

ferramentas computacionais que possibilitam cálculos, visualização, modelagem e geração de simulações. Assim os objetos de aprendizagem se encaixam perfeitamente na necessidade do docente passar essas experiências aos seus alunos (SCHLÜNZEN, 2000).

Inspirados no crescente interesse da comunidade científica no desenvolvimento de ferramentas para proporcionar ao aluno uma melhor aprendizagem e interação com o conteúdo pedagógico, o presente trabalho tem como objetivo contribuir para o processo de ensinar e aprender no cotidiano da graduação e pós-graduação em engenharia elétrica, apresentando uma ferramenta computacional que favorece a interação entre os alunos e o professor com vistas à aprendizagem nas disciplinas de análise de Sistemas de Energia Elétrica (SEE).

No presente trabalho, foi desenvolvido um software seguindo os paradigmas de programação orientada a objetos, no qual foram incluídos os principais elementos de sistemas de potência, objetivando-se a resolução de fluxo de potência e a análise de estabilidade estática e transitória de sistemas multimáquinas. O programa traz incorporados os dados correspondentes a diversos sistemas-exemplo da literatura especializada, e permite que o usuário modifique parâmetros e até mesmo acrescente dados de outros sistemas de uma maneira bastante amigável. Com isto, as possibilidades de criação e análise de casos ficam praticamente ilimitadas, restritas apenas pelos modelos de representação dos sistemas implementados, possibilitando estudos do desempenho de Sistemas de Energia Elétrica sob condições estáticas e transitórias.

1.1 Exemplos de Softwares para Educação em Engenharia

No campo das relações ensino/aprendizagem de Sistemas de Energia Elétrica, um Objeto de Aprendizagem voltado ao estudo de Estabilidade de Sistemas de Energia Elétrica foi desenvolvido por Vieira e Colvara (2007a) e Vieira e Colvara (2007b) suportando análises de casos de estabilidade em sistemas de uma e de três máquinas. Este trabalho resultou no desenvolvimento de um software educacional intitulado Estabilidade Visual, no qual foi utilizada programação orientada a objeto para verificação da estabilidade transitória de sistemas de energia elétrica considerando-se o modelo clássico para análise de primeira oscilação. O software oferece ao usuário a possibilidade de alterar os sistemas disponíveis, exibindo além

de respostas numéricas, opções de visualização gráfica com controle tridimensional e animação virtual representando o comportamento dinâmico das máquinas síncronas que compõem o sistema.

O software “Estabilidade Visual” possui limitações como tamanho dos sistemas lidos, alteração dos parâmetros de simulação, cálculo dos fluxos de potência pré e pós-falta, que foram superadas no presente trabalho. Na Figura 1 é mostrada a tela inicial do *software* “Estabilidade Visual”.

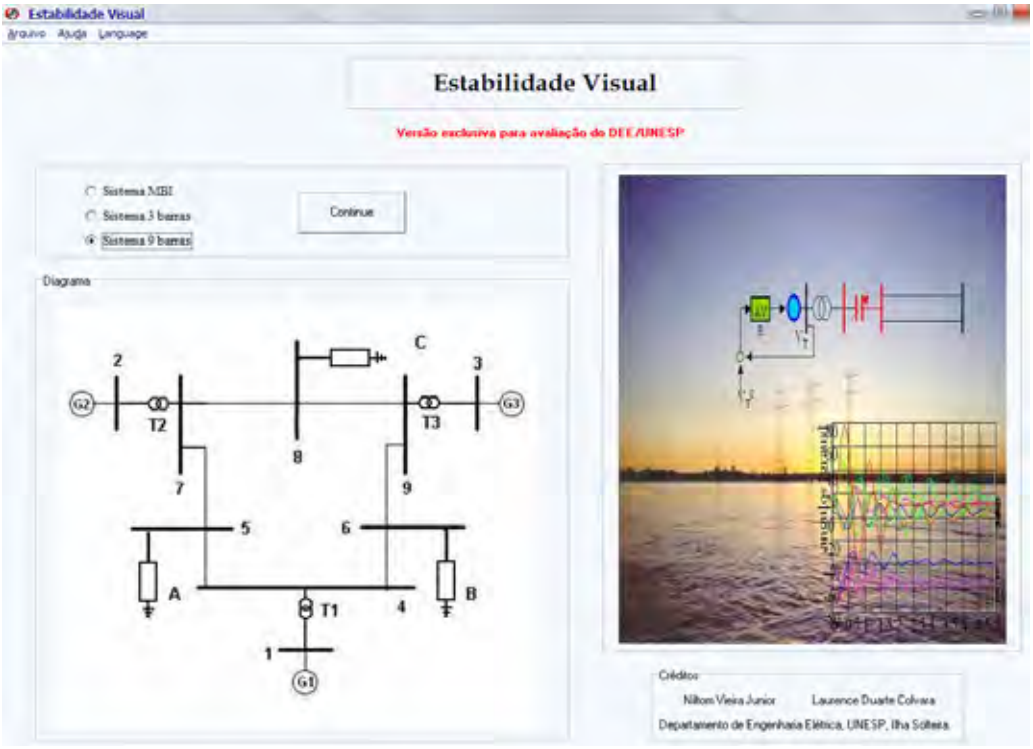


Figura 1: Tela Inicial do Software Estabilidade Visual (VIEIRA JUNIOR e COLVARA, 2007a).

No trabalho de Agostini, Decker e Silva (2002) foi desenvolvido uma base computacional orientada a objetos para aplicações de sistemas de energia elétrica, onde foi descrito a aplicação de técnicas de Modelagem Orientada a Objetos (MOO) na modelagem de SEE. Pretendeu-se desenvolver uma base computacional orientada a objetos que represente de forma eficiente os componentes físicos dos SEE e suas metodologias de análise e síntese. O Software foi desenvolvido utilizando Framework, que consiste num ambiente de desenvolvimento com suporte a MOO (Modelagem Orientada a Objetos).

Estruturar em classes os elementos de um sistema de energia elétrica permite certo grau de generalização tal que possa suportar especializações futuras no sentido de se implementar, de forma integrada, as mais diversas metodologias de análise e

síntese na área. As motivações para o estudo foram as atuais limitações impostas pelos métodos e linguagens tradicionalmente utilizados no desenvolvimento dos softwares na área de SEE, tais como baixo nível de integração entre módulos e a necessidade de elevados esforços para manutenções e atualizações. A metodologia de trabalho consiste em se modelar os SEE e suas aplicações, tendo por base métodos de projeto orientados a objetos, com especial atenção ao método *Object Modeling Tecnique* (OMT). No trabalho de Agostini, Decker e Silva (2002) foram implementados dois módulos representando metodologias de análise, visando a validação das estruturas propostas: um Fluxo de Potência Linearizado e uma Simulação da Dinâmica de SEE com Modelagem Detalhada, como está representado na Figura 2. A documentação do software pode ser encontrada em (http://www.labplan.ufsc.br/newplan/).

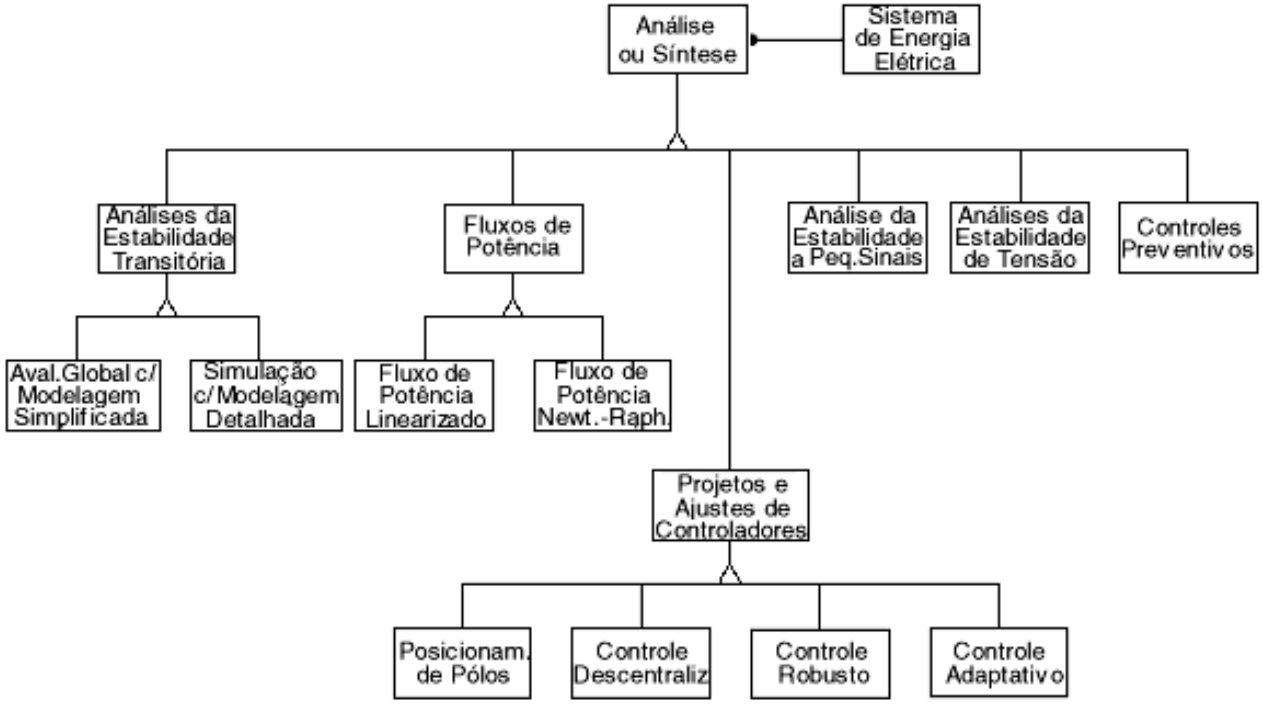


Figura 2: Estruturas de Classes Newplan (AGOSTINI E DECKER, 2002).

No âmbito internacional há vários trabalhos ligados ao desenvolvimento de softwares para educação em engenharia como é o caso do PSAT encontrado em Milano, Vanfretti e Morataya (2007) que consiste numa *toolbox* para Matlab, seguindo os padrões da GNU/Octave. A GNU/Octave é uma alternativa de uso de programas com plataforma Matlab sem uso de licença.

Apesar de haver uma versão do Matalb para plataformas Unix/Linux, a mesma não é gratuita. O Matlab é um software proprietário cuja licença tem um valor

alto para o usuário comum. O Octave provê uma interface em linha de comando para solução de problemas lineares e não-lineares. Ele também funciona como uma linguagem de programação em *batch*. Possui uma extensa biblioteca para resolução dos problemas mais comuns em álgebra linear. É facilmente extensível e customizável usando funções definidas pelo usuário criadas em linguagem própria do Octave ou carregando módulos escritos em C/C++, Fortran e outras linguagens. Assim o PSAT faz parte dos softwares *open source*, com distribuição livre e gratuita. Essa *toolbox* oferece ao usuário recursos como fluxo de potência, fluxo de potência continuado, análises no domínio do tempo, estudos de modelos dinâmicos para máquinas síncronas e assíncronas e também estudos de reguladores e FACTS (*Flexible AC Transmission Systems*).

O PSAT teve como referências outros trabalhos que também tiveram o objetivo de construir softwares educacionais para engenharia como: *Power System Toolbox* (PST), MatPower, Toolbox (VST), MatEMTP, SimPowerSystems (SPS), *Power Analysis Toolbox* (PAT), and the *Educational Simulation Tool* (EST) (MILANO, VANFRETTI e MORATAYA, 2007), mais detalhes deste último software podem ser encontrados em (http://www.power.uwaterloo.ca/~fmilano/).

Apesar de já existirem *softwares* com propósitos semelhantes ao do “Objeto de aprendizagem para sistemas de energia elétrica”, ainda tinha-se a necessidade de desenvolver um que pudesse ser usado pedagogicamente no ensino da engenharia, sem restrições de custo ou de licenças e possibilidade de reunir numa só ferramenta os principais conceitos abordados na disciplina de análise de estabilidade de sistemas de energia elétrica.

CAPITULO 2

CONCEITOS DE OBJETOS DE APRENDIZAGEM

O uso de computador como ferramenta de aprendizagem tem sido sustentada desde as pesquisas de 1960 nos Estados Unidos onde eles foram usados para realização de tarefas de cálculo e o auxílio das atividades de ensino. Desde então a comunidade cientifica desenvolve diversos softwares tutoriais educacionais que ganharam força após os investimentos de grandes empresas na área de informática como a IBM, RCA, Digital.

A partir de 1980, com a popularização do uso do computador, surgiram novas linguagens de programação que deram suporte ao desenvolvimento de softwares mais eficazes não só em realização de cálculos, mas na execução de tarefas diárias e entretenimento. Usar o computador apenas para digitar textos e realizar cálculos não era mais suficiente, os usuários buscavam uma interação maior homem-máquina, onde pudessem usar o computador de forma mais fácil e eficaz. Assim nos anos 90 surgiram os primeiros computadores com processadores MMX que permitiram adicionar ao sistema operacional recursos áudios-visuais. Essa mudança também foi um facilitador para a popularização da internet, que deixou de ser apenas um ambiente de comunicação e pesquisa para ser fonte de lazer e entretenimento e aprendizagem.

A aplicação do agente computacional como mediador no processo de ensino-aprendizagem oferece uma abordagem alternativa ao modelo de ensino tradicional. Se antes o aprendiz atuava como agente receptor de informações emitidas pelo professor, a presença do computador cria um ambiente de aprendizagem. Este ambiente atua como um tutor, apresentando os principais tópicos da matéria e testando o desempenho do aluno. Sua vantagem reside no fato de poder dispor de sons e animações para catalisar o aprendizado (VALENTE, 2002).

Numa próxima etapa, este ambiente de aprendizagem pode fornecer a opção de simular determinado fenômeno a ser estudado. Neste caso, há uma comunicação bidirecional entre o aprendiz e o sistema. Na medida em que o aluno altera o valor

das variáveis, o simulador retorna as saídas. De acordo com Valente (1993), a simulação oferece a possibilidade de o aluno desenvolver hipóteses, testá-las, analisar resultados e refinar os conceitos. Ou seja, a geração de diferentes cenários simulados promove o aprendizado do aluno.

As novas tecnologias de comunicação conseguem aumentar a produtividade e a qualidade na transmissão de conhecimentos. Diferentes recursos de mídia podem ser utilizados visando a maior clareza e precisão na transmissão da informação e ainda permitem o efetivo envolvimento com o aluno. Com a utilização de imagens, sons e experiências de simulação e experimentação, a atividade multimídia desenvolve no aprendiz um nível de absorção de conhecimento que o ensino tradicional dificilmente alcançaria.

A utilização de múltiplos formatos de informação desempenha um papel importante na aquisição do conhecimento, pois os programas de multimídia têm a vantagem de envolver múltiplos sentidos simultaneamente, acomodando uma grande variedade de estilos de aprendizagem.

Uma das estruturas utilizadas para passar conhecimento para os aprendizes são objetos de aprendizagem. Tais objetos são baseados na teoria de aprendizagem construtivista. Esta estrutura consiste em uma entidade (digital ou não digital) que pode ser utilizada para aprendizagem, educação ou treinamento. São, na verdade, as menores unidades de aprendizagem.

A idéia destes objetos é fragmentar o conteúdo educacional em pequenas partes que podem ser reutilizadas em diferentes ambientes de aprendizagem. Ou seja, um mesmo objeto “encapsula” determinados conceitos que podem ser aplicados em áreas do conhecimento distintas. Além disso, cada objeto pode ser utilizado de modo independente ou em conjunto com outros objetos, dependendo do objetivo do professor.

Objeto de aprendizagem é uma unidade de instrução/ensino que é reutilizável. Segundo o grupo *Learning Objects Metadata Workgroup* do Institute of Electrical and Electonics Engineers (IEEE, 2007), Objetos de Aprendizagem (*Learning Objects*) podem ser definidos por "qualquer entidade, digital ou não digital, que possa ser utilizada, reutilizada ou referenciada durante o aprendizado suportado por tecnologias". Um objeto de aprendizagem pode ser usado em diferentes contextos e em diferentes ambientes virtuais de aprendizagem. Para atender a esta característica, cada objeto tem sua parte visual, que interage com o aprendiz

separada dos dados sobre o conteúdo e os dados instrucionais do mesmo. A principal característica dos objetos de aprendizagem é sua reusabilidade, que é posta em prática através de repositórios, que armazenam os objetos logicamente, permitindo serem localizados a partir da busca por temas, por nível de dificuldade, por autor ou por relação com outros objetos. Nos países de língua inglesa existe um vasto número de repositórios disponíveis, utilizados e reutilizados em contextos diversos.

O conceito de objetos de aprendizagem (*Learning Objects* - LO) é muito amplo e surgiu com um objetivo: localizar conteúdos educacionais na Web, para serem reutilizados em diferentes cursos e plataformas e, assim, possibilitar a redução do custo de produção dos materiais desses cursos.

Os objetos de aprendizagem podem ser empregados pelo professor para:

- Contextualização do tema curricular por meio de uma situação-problema.

Para os alunos, os objetos permitem:

- Observação do fenômeno;
- Interação com a situação-problema;
- Interferência nos resultados observados.

Várias organizações empreenderam esforços para desenvolver padrões de descrição dos *Learning Objects* a fim de atender sua característica fundamental: a reutilização. A redução de custos está vinculada, porém, ao desenvolvimento dos *Learning Objects*, pois sua construção com qualidade tem um custo alto, em conseqüência das etapas de design iniciais, que são demoradas, e também da distribuição dos mesmos.

Se for desenvolvida uma animação ou simulação para um único curso, o custo torna-se alto, mas quando desenvolvida a utilização em muitos outros cursos, esse custo cai. Assim, aqueles objetos que foram planejados e são utilizados dentro de uma instituição ou rede têm seus valores amortizados à medida que são reutilizados. Quanto aos *Learning Objects* que se tornam públicos, estes podem ser utilizados com ou sem custo.

Os metadados dos objetos de aprendizagem (*Learning Objects Metadatas* - LOM), do IEEE (Institute of Electrical and Electronics Engineers), têm categorias onde são descritos os direitos e o custo do objeto para o caso da reutilização.

Para permitir o armazenamento e o desenvolvimento dos objetos de aprendizagem, além do relacionamento existente entre eles, não se pode esquecer dos chamados repositórios de objetos de aprendizagem. Tem-se como exemplo o CLOE (*Cooperative Learning Object Exchange*), desenvolvido na Universidade de Waterloo, no Canadá (CLOE, 2007).

CAPITULO 3

REPRESENTAÇÃO DO SISTEMA ELÉTRICO DE POTÊNCIA

Rede Elétrica

Considerando a rede elétrica em regime permanente, desconsiderando as dinâmicas eletromagnéticas, a sua representação genérica é:

$$I = YE \tag{3.1}$$

sendo:

I = vetor injeções de correntes de barra;
E = vetor de tensões na barra;
Y = matriz admitância de barra;

assim,

$$Y = G + jB \tag{3.2}$$

para

G = matriz de condutância de barra;
B = matriz susceptância de barra.

Máquina Síncrona

Para a máquina síncrona, considerando modelo de dois eixos (δ, ω, E'_d e E'_q), como no diagrama fasorial mostrado na Figura 3, as equações de estado podem ser escritas como:

$$\dot{\delta} = \omega \tag{3.3}$$

$$\dot{\omega} = \frac{1}{M}(-D\omega + P_m - P_e) \tag{3.4}$$

$$\tau'_{do}\, E'_q = -E'_q - (x_d - x'_d)i_d + E_{fd} \tag{3.5}$$

$$\tau'_{qo}\, E'_d = -E'_d - (x_q - x'_q)i_q \tag{3.6}$$

(3.7)

$$E_{fd} = E'_q - (X_d - X'_d) i_d$$

sendo:

δ = posição angular medida em relação a um eixo que gira à velocidade síncrona;

ω = desvio de velocidade angular da máquina síncrona com relação à velocidade síncrona;

D = constante de amortecimento [s];

E'_d = tensão proporcional ao enlace de fluxo de eixo direto;

E'_q = tensão proporcional ao enlace de fluxo de eixo quadratura;

E_{fd} = tensão de excitação;

i_d = corrente de eixo direto nos terminais da máquina;

$M = \frac{2H}{2\pi f_o}$ = constante de inércia;

P_e = potência elétrica entregue pela máquina síncrona;

P_m = potência mecânica de entrada (fornecida a máquina síncrona);

τ'_{do} = constante de tempo de circuito aberto de eixo direto;

τ'_{qo} = constante de tempo de circuito aberto de eixo em quadratura;

x'_d = reatância transitória de eixo direto;

x_d = reatância de eixo direto;

x'_q = reatância transitória de eixo em quadratura; e

x_q = reatância de eixo em quadratura.

A potência elétrica (P_e) fornecida pela máquina é dada por:

$$P_e = v_d i_d + v_q i_q$$

em que:

$$v_d = E'_d + x'_q i_d$$

$$v_q = E'_q - x'_d i_d$$

sendo:

v_d = tensão de eixo direto no terminal da máquina e

v_q = tensão de eixo em quadratura no terminal da máquina.

portanto:

$$P_e = E'_d\, i_d + E'_q\, i_q + (x'_q - x'_d) i_d i_q \tag{3.11}$$

para as máquinas de rotor liso:

$$x'_q \cong x'_d \tag{3.12}$$

então:

$$P_e = E'_d\, i_d + E'_q\, i_q \tag{3.13}$$

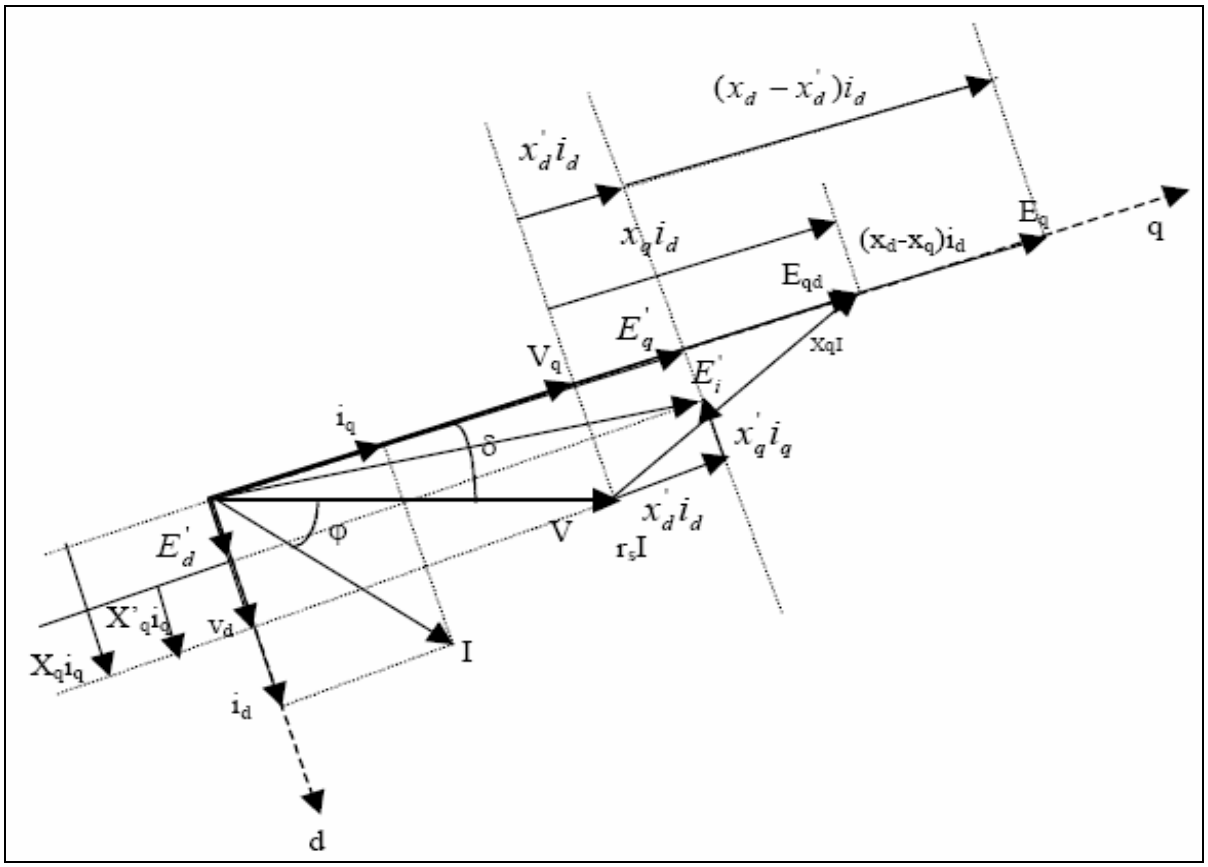


Figura 3: Diagrama Fasorial de Máquina Síncrona para modelo de dois eixos.

CAPITULO 4

FLUXO DE POTÊNCIA

O objetivo dos estudos de fluxo de potência é determinar as tensões nas barras do sistema em suas magnitudes e ângulos, bem como as potências ativa e reativa que “fluem” nas linhas de transmissão de um dado Sistema de Energia Elétrica em regime permanente. Para isto, já foram desenvolvidas diversas técnicas de cálculo (referência), e neste trabalho se consideram os métodos de Newton-Raphson e Newton Desacoplado para a resolução dos sistemas de equações algébricas não-lineares envolvidas (MONTICELLI e GARCIA, 2000). Os resultados de cada execução do fluxo de potência representam um retrato do sistema de potência para as condições do regime permanente considerado. Cálculos de fluxo de potência são realizados para o projeto, planejamento da expansão, planejamento da operação e para supervisão e controle de sistemas elétricos de potência.

A resolução de problemas de fluxo de potência requer que numerosas equações algébricas sejam resolvidas. A formulação do problema é trivial, porém, a solução de um grande número de equações que consideram as especificidades de cada tipo de barra tornam a solução inviável sem o auxílio de computadores.

4.1 Método de Newton-Raphson

Para a solução de sistemas de equações algébricas não-lineares o método de Newton-Raphson baseia-se em linearizações sucessivas das funções a partir de uma condição inicial arbitrária. As linearizações são obtidas através da expansão em Série de Taylor.

Uma grande vantagem do uso deste método é o fato de que a convergência é quadrática, tornando o processo mais veloz que a maioria dos demais algoritmos. Além disso, este método é menos sensível aos fatores que poderiam perturbar a convergência (como a escolha da barra de referência, por exemplo). Pode-se considerar como a maior desvantagem do método a necessidade de construir e inverter a matriz Jacobiano.

Conforme Stevenson (1986), supondo que o conjunto de equações não-lineares a resolver tenha a forma:

$$\begin{aligned} f_1(x_1, x_2, ..., x_n) &= K_1 \\ f_2(x_1, x_2, ..., x_n) &= K_2 \\ &... \\ f_i(x_1, x_2, ..., x_n) &= K_i \end{aligned} \tag{4.1}$$

em que:

f_i - i-ésima função a resolver;

$x_1, x_{2,...,} x_n$ - variáveis do problema;

K_i - i-ésima constante do problema;

pode-se escrever que as raízes (soluções) destas equações, para a estimativa inicial, sejam representados por $x_1^0, x_2^0, ..., x_n^0$. Os erros entre as estimativas iniciais e o valor correto das soluções podem ser expressos por:

$$\begin{aligned} \Delta x_1^{(0)} &= x_1 - x_1^{(0)} \\ \Delta x_2^{(0)} &= x_2 - x_2^{(0)} \\ &... \\ \Delta x_n^{(0)} &= x_n - x_n^{(0)} \end{aligned} \tag{4.2}$$

em que:

$x_n^{(0)}$ - estimativa inicial de x_n;

x_n - valor correto de x_n para a solução;

$\Delta x_n^{(0)}$ - erro entre a estimativa inicial e o valor correto.

Substituindo o conjunto de equações (4.2) na equação (4.1):

$$\begin{aligned} f_1\left(x_1^{(0)} + \Delta x_1^{(0)}, x_2^{(0)} + \Delta x_2^{(0)}, ..., x_n^{(0)} + \Delta x_n^{(0)}\right) &= K_1 \\ f_2\left(x_1^{(0)} + \Delta x_1^{(0)}, x_2^{(0)} + \Delta x_2^{(0)}, ..., x_n^{(0)} + \Delta x_n^{(0)}\right) &= K_2 \end{aligned}$$

...

$$f_i\left(x_1^{(0)} + \Delta x_1^{(0)}, x_2^{(0)} + \Delta x_2^{(0)}, ..., x_n^{(0)} + \Delta x_n^{(0)}\right) = K_i$$

Para resolver as equações acima deve-se recorrer à expansão em Série de Taylor. Reescrevendo as equações considerando a expansão, tem-se:

$$f_1\left(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right) + \Delta x_1^{(0)} \cdot \left.\frac{\partial f_1}{\partial x_1}\right|_{(0)} + \Delta x_2^{(0)} \cdot \left.\frac{\partial f_1}{\partial x_2}\right|_{(0)} + ... + \Delta x_n^{(0)} \cdot \left.\frac{\partial f_1}{\partial x_n}\right|_{(0)} + ... = K_1$$

$$f_2\left(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right) + \Delta x_1^{(0)} \cdot \left.\frac{\partial f_2}{\partial x_1}\right|_{(0)} + \Delta x_2^{(0)} \cdot \left.\frac{\partial f_2}{\partial x_2}\right|_{(0)} + ... + \Delta x_n^{(0)} \cdot \left.\frac{\partial f_2}{\partial x_n}\right|_{(0)} + ... = K_2$$

$$f_i\left(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right) + \Delta x_1^{(0)} \cdot \left.\frac{\partial fi}{\partial x_1}\right|_{(0)} + \Delta x_2^{(0)} \cdot \left.\frac{\partial f_i}{\partial x_2}\right|_{(0)} + ... + \Delta x_n^{(0)} \cdot \left.\frac{\partial f_i}{\partial x_n}\right|_{(0)} + ... = K_i$$

onde:

$\left.\frac{\partial f_i}{\partial x_n}\right|_{(0)}$ - Derivada parcial da i-ésima função em relação a n-ésima variável no instante inicial.

No método de Newton-Raphson as derivadas parciais de ordem superior a um são desconsideradas e pode-se escrever as anteriores na forma matricial:

$$\begin{bmatrix} K_1 - f_1\left(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right) \\ K_2 - f_2\left(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right) \\ ... \\ K_i - f_n\left(x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)}\right) \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & ... & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & ... & \frac{\partial f_2}{\partial x_n} \\ ... & ... & ... & ... \\ \frac{\partial f_i}{\partial x_1} & \frac{\partial f_i}{\partial x_2} & & \frac{\partial f_i}{\partial x_n} \end{bmatrix} \begin{bmatrix} \Delta x_1^{(0)} \\ \Delta x_2^{(0)} \\ ... \\ \Delta x_n^{(0)} \end{bmatrix} \quad (4.3)$$

A matriz de derivadas parciais é conhecida como matriz Jacobiano. Considerando o vetor ΔK_i como o vetor dos erros entre os valores das estimativas iniciais e os valores corretos, pode-se reescrever a matriz acima, simplificando-a:

$$\begin{bmatrix} \Delta K_1^{(0)} \\ \Delta K_2^{(0)} \\ ... \\ \Delta K_i^{(0)} \end{bmatrix} = J^{(0)} \begin{bmatrix} \Delta x_1^{(0)} \\ \Delta x_2^{(0)} \\ ... \\ \Delta x_n^{(0)} \end{bmatrix}$$

Com a equação matricial acima pode-se facilmente obter os valores das variações nas variáveis de interesse. Como truncou-se a série de Taylor (derivadas

parciais de ordem superior a um), os valores obtidos para $\Delta x_1^{(0)}$ não são os valores corretos. Desta forma, deve-se repetir todo o processo novamente considerando novas estimativas para as variáveis:

$$x_1^{(1)} = x_1^{(0)} + \Delta x_1^{(0)}$$

$$x_2^{(1)} = x_2^{(0)} + \Delta x_2^{(0)}$$

$$\ldots$$

$$x_n^{(1)} = x_n^{(0)} + \Delta x_n^{(0)}$$

O processo deve ser interrompido quando os erros forem toleráveis (escolha da precisão para a convergência).

Para utilizar todo o equacionamento acima para as equações do fluxo de potência é conveniente utilizar as equações que regem os desbalanços de potência. As equações que representam tais desbalanços variam conforme o tipo de barra:

a) Barras do tipo PQ (barras de carga):

$$\Delta P_i = P_i^{esp} - P_i = P_i^{esp} - V_i \cdot \sum \left(G_{ik} \cos \delta_{ik} + B_{ik} sen \, \delta_{ik} \right) V_k = 0$$

$$\Delta Q_i = Q_i^{esp} - Q_i = Q_i^{esp} - V_i \cdot \sum \left(G_{ik} sen \, \delta_{ik} - B_{ik} \cos \delta_{ik} \right) V_k = 0$$

b) Barras do tipo PV (barras de tensão controlada):

$$\Delta P_i = P_i^{esp} - P_i = P_i^{esp} - V_i \cdot \sum \left(G_{ik} \cos \delta_{ik} + B_{ik} sen \, \delta_{ik} \right) V_k = 0$$

em que:

P_i^{esp} - potência ativa especificada para a barra i;

P_i - potência ativa calculada para a barra i;

Q_i^{esp} - potência reativa especificada para a barra i;

Q_i - potência reativa calculada para a barra i;

G_{ik} - condutância entre a barra i e a barra k;

B_{ik} - susceptância entre a barra i e a barra k;

V_k - tensão na barra k;

δ_{ik} - ângulo entre as tensões na barra i e k.

Considerando as equações para cada tipo de barra, pode-se reescrever a equação (4.3) de forma a torná-la específica ao estudo de fluxo de potência:

$$\begin{bmatrix} \Delta P_2 \\ \dots \\ \Delta P_i \\ \Delta Q_2 \\ \dots \\ \Delta Q_i \end{bmatrix} = \begin{bmatrix} \frac{\partial P_2}{\partial \delta_2} & \dots & \frac{\partial P_2}{\partial \delta_i} & \frac{\partial P_2}{\partial |V_2|} & \dots & \frac{\partial P_2}{\partial |V_i|} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{\partial P_i}{\partial \delta_2} & \dots & \frac{\partial P_i}{\partial \delta_i} & \frac{\partial P_i}{\partial |V_2|} & \dots & \frac{\partial P_i}{\partial |V_i|} \\ \frac{\partial Q_2}{\partial \delta_2} & \dots & \frac{\partial Q_2}{\partial \delta_i} & \frac{\partial Q_2}{\partial |V_2|} & \dots & \frac{\partial Q_2}{\partial |V_i|} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \frac{\partial Q_i}{\partial \delta_2} & \dots & \frac{\partial Q_i}{\partial \delta_i} & \frac{\partial Q_i}{\partial |V_2|} & \dots & \frac{\partial Q_i}{\partial |V_i|} \end{bmatrix} \cdot \begin{bmatrix} \Delta \delta_2 \\ \dots \\ \Delta \delta_i \\ \Delta V_2 \\ \dots \\ \Delta V_i \end{bmatrix} \tag{4.4}$$

Cabe ressaltar que, na equação (4.4) o índice inicial utilizado foi 2 ou invés de 1. Essa adoção foi proposital já que a barra de referência normalmente é numerada com o índice 1 e este tipo de barra não necessita de representação na matriz Jacobiano.

Para facilitar a confecção da matriz Jacobiano, utiliza-se uma divisão em quatro partes iguais, conforme o diagrama seguinte:

$$\begin{bmatrix} \Delta P_2 \\ \dots \\ \Delta P_i \\ \Delta Q_2 \\ \dots \\ \Delta Q_i \end{bmatrix} = \begin{bmatrix} \frac{\partial P_2}{\partial \delta_2} & \dots & \frac{\partial P_2}{\partial \delta_i} & \frac{\partial P_2}{\partial |V_2|} & \dots & \frac{\partial P_2}{\partial |V_i|} \\ \dots & \mathbf{H} & \dots & \dots & \mathbf{N} & \dots \\ \frac{\partial P_i}{\partial \delta_2} & \dots & \frac{\partial P_i}{\partial \delta_i} & \frac{\partial P_i}{\partial |V_2|} & \dots & \frac{\partial P_i}{\partial |V_i|} \\ \frac{\partial Q_2}{\partial \delta_2} & \dots & \frac{\partial Q_2}{\partial \delta_i} & \frac{\partial Q_2}{\partial |V_2|} & \dots & \frac{\partial Q_2}{\partial |V_i|} \\ \dots & \mathbf{M} & \dots & \dots & \mathbf{L} & \dots \\ \frac{\partial Q_i}{\partial \delta_2} & \dots & \frac{\partial Q_i}{\partial \delta_i} & \frac{\partial Q_i}{\partial |V_2|} & \dots & \frac{\partial Q_i}{\partial |V_i|} \end{bmatrix} \cdot \begin{bmatrix} \Delta \delta_2 \\ \dots \\ \Delta \delta_i \\ \Delta V_2 \\ \dots \\ \Delta V_i \end{bmatrix} \tag{4.5}$$

Para o cálculo de cada conjunto de derivadas parciais, serão necessárias as seguintes equações:

a) Quadrante **H**:

$$H_{ii} = -Q_i - v_i^{\,2} B_{ii}$$

e

$$H_{ik} = V_i V_k \left(G_{ik} sen\ \delta_{ik} - B_{ik} \cos\ \delta_{ik} \right)$$

b) Quadrante **N**:

$$N_{ii} = \frac{\left(P_i + V_i^2 G_{ii}\right)}{V_i}$$

e

$$N_{ik} = V_i\left(G_{ik}\cos\ \delta_{ik} + B_{ik} sen\ \delta_{ik}\right)$$

c) Quadrante **M**:

$$M_{ii} = P_i + V_i^2 G_{ii}$$

e

$$M_{ik} = -V_i V_k\left(G_{ik}\cos\ \delta_{ik} + B_{ik} sen\ \delta_{ik}\right)$$

d) Quadrante **L**:

$$L_{ii} = \frac{\left(Q_i + V_i^2 B_{ii}\right)}{v_V}$$

e

$$L_{ik} = V_i\left(G_{ik}\sin\ \delta_{ik} + B_{ik}\cos\ \delta_{ik}\right)$$

Como mencionado anteriormente, a matriz Jacobiano deverá ser atualizada em cada iteração, o esquema para solução do método de Newton-Raphson, encontrado em Stagg e El-Abiad (1968).

4.2 Método de Newton-Desacoplado

Segundo Monticelli (1983), os métodos desacoplado, como o próprio nome sugere, baseiam-se no desacoplamento $P\theta - QV$, ou seja, são obtidos considerando-se o fato de que as sensibilidades $\partial P / \partial \theta$ e $\partial Q / \partial V$ serem mais intensas que as sensibilidades $\partial P / \partial V$ e $\partial Q / \partial \theta$.

O desacoplamento possibilita a adoção de um esquema de resolução segundo o qual os subproblemas $P\theta$ e QV são resolvidos alternadamente: na resolução do subproblema $P\theta$ são utilizados os valores atualizados de V; na resolução do subproblema QV são utilizados os valores atualizados de θ.

Em Monticelli (1983), o algoritmo básico do método de Newton-Raphson é colocado na forma:

$$\begin{aligned}
&\Delta P(V^{\nu},\theta^{\nu}) = H(V^{\nu},\theta^{\nu})\Delta\theta^{\nu} + N(V^{\nu},\theta^{\nu})\Delta\theta^{\nu} \\
&\Delta Q(V^{\nu},\theta^{\nu}) = M(V^{\nu},\theta^{\nu})\Delta\theta^{\nu} + L(V^{\nu},\theta^{\nu})\Delta\theta^{\nu} \\
&\theta^{\nu+1} = \theta^{\nu} + \Delta\theta^{\nu} \\
&V^{\nu+1} = V^{\nu} + \Delta V^{\nu}
\end{aligned} \tag{4.6}$$

A dedução do método de Newton desacoplado é feita em duas etapas: desacoplamento e aplicação do esquema alternado de resolução. Pelo desacoplamento $P\theta - QV$ os termos $N\Delta V$ e $M\Delta\theta$ são ignorados, o que torna possível colocar o algoritmo de Newton na seguinte forma:

$$\begin{aligned}
&\Delta P(V^{\nu},\theta^{\nu}) = H(V^{\nu},\theta^{\nu})\Delta\theta^{\nu} \\
&\Delta Q(V^{\nu},\theta^{\nu}) = L(V^{\nu},\theta^{\nu})\Delta\theta^{\nu} \\
&\theta^{\nu+1} = \theta^{\nu} + \Delta\theta^{\nu} \\
&V^{\nu+1} = V^{\nu} + \Delta V^{\nu}
\end{aligned} \tag{4.7}$$

A recorrência dada pelas equações (4.7) ainda está colocada na forma simultânea, isto é, θ e V são atualizados ao mesmo tempo. A segunda etapa da obtenção do método desacoplado consiste em se aplicar o esquema de resolução alternado, resultando:

$$\begin{aligned}
&\Delta P(V^{\nu},\theta^{\nu}) = H(V^{\nu},\theta^{\nu})\Delta\theta^{\nu} \\
&\theta^{\nu+1} = \theta^{\nu} + \Delta\theta^{\nu} \\
&\Delta Q(V^{\nu},\theta^{\nu+1}) = L(V^{\nu},\theta^{\nu})\Delta\theta^{\nu} \\
&V^{\nu+1} = V^{\nu} + \Delta V^{\nu}
\end{aligned} \tag{4.8}$$

É possível notar que colocando-se o algoritmo na forma alternada como na equação (4.8), as aproximações introduzidas na matriz jacobiana com passagem da equação (4.6) para equação (4.7) são parcialmente compensadas pelo fato de as variáveis θ e V serem atualizadas a cada meia-iteração; na equação (4.8) imediatamente após a obtenção de uma nova estimativa de θ e V, esses valores já são utilizados no cálculo subseqüente de ΔP e ΔQ.

Segundo Monticelli (1983), existem situações nas quais os subproblemas $P\theta$ e QV têm velocidades de convergência distintas: o subproblema $P\theta$ pode convergir antes do subproblema QV. Nesses casos, podem-se obter algumas vantagens computacionais interando-se apenas com um subproblema ainda não resolvido. Em Monticelli (1983) é mostrado o esquema para solução do método de Newton desacoplado.

CAPITULO 5

ESTABILIDADE DE SISTEMAS DE POTÊNCIA

Um dos aspectos mais importantes no estudo de sistemas elétricos de potência consiste na caracterização da estabilidade das máquinas síncronas que pertencem a este sistema (STEVENSON, 1986). As máquinas síncronas, como o nome sugere, são aquelas que se mantêm em sincronismo em condições normais de operação. Se uma máquina síncrona tende a se afastar ligeiramente da velocidade de sincronismo, as forças de sincronismo fazem com que ela tenda novamente a funcionar à velocidade síncrona. Contudo, existem algumas condições de funcionamento em que as forças de sincronismo intrínsecas ao sistema não são suficientes para manter o sincronismo de uma ou mais máquinas. Nestes casos, é de fundamental importância conhecer quais são os impactos de perturbações no sistema para estudar as forma de mitigar os riscos inerentes a sua ocorrência.

Estudar a estabilidade de sistemas elétricos de potência consiste em conhecer como se comportam as máquinas síncronas durante e após uma perturbação no sistema. Se a perturbação for pequena e de curta duração o sistema poderá voltar ao sincronismo sem maiores problemas. Já no caso de faltas mais severas (pela duração ou por uma grande e súbita alteração nos parâmetros) é possível que o sistema se torne instável, de modo que podem ser necessárias medidas preventivas para eliminar o problema e manter a freqüência do sistema constante.

Usualmente, os estudos de estabilidade são divididos em:

- Estabilidade Dinâmica;
- Estabilidade em Regime Permanente;
- Estabilidade Transitória.

Nos estudos de estabilidade dinâmica e de regime permanente as perturbações consideradas são lentas ou de pequena magnitude. Nestes estudos as equações algébricas e diferencias são substituídas por equações lineares e se procura determinar as respostas do sistema a variações relativamente pequenas em torno do ponto de operação normal das máquinas.

Já os estudos de estabilidade transitória de primeira oscilação analisam o comportamento do sistema para os primeiros segundos (um ou dois segundos) após a ocorrência de uma falta e apresentam grande importância prática. Os problemas de estabilidade transitória envolvem grandes perturbações que não permitem procedimentos de linearização e as equações necessárias (algébricas e diferenciais) devem ser resolvidas através de métodos apropriados. Considerando que o período de análise em estudos de estabilidade transitória é curto, podem-se adotar premissas simplificadoras que são válidas paras os primeiros instantes de oscilação:

- As correntes contínuas de ajuste e as correntes harmônicas que possam fluir nos enrolamentos dos estatores das máquinas síncronas são desprezadas;
- Para representar faltas assimétricas (desequilibradas) são utilizadas componentes simétricas e;
- As perturbações do sistema não afetam a tensão gerada (tensão interna de cada gerador).

A metodologia clássica para a análise de estabilidade em sistemas elétricos de potência consiste na resolução numérica das equações diferenciais associadas ao movimento dos geradores. No passado, a grande desvantagem desta metodologia advinda do grande esforço computacional necessário para resolver o sistema de equações diferenciais associado a cada máquina do sistema, de forma a conhecer o comportamento das máquinas durante o período de falta e algum tempo após a perturbação ter sido eliminada (período pós-falta). O peso computacional resultava não só da resolução de um elevado número de equações diferenciais, como da necessidade de simular diferentes localizações e tempos de eliminação dos defeitos, com diferentes configurações dos sistemas de geração e carga.

Com a grande capacidade de processamento que dispomos nos dias de hoje, resolver os problemas de estabilidade considerando a metodologia clássica torna-se

tarefa mais simples. Além das três premissas acima descritas para caracterizar os estudos de estabilidade transitória, o modelo clássico ainda adota as seguintes premissas:

- A potência mecânica das máquinas é suposta constante;
- Os torques de amortecimento são considerados através de um termo proporcional à velocidade das máquinas e incluído nas equações de oscilação;
- As máquinas são representadas por uma tensão constante atrás da reatância transitória;
- As cargas são representadas por impedâncias constantes.

As implementações computacionais do aplicativo proposto utilizam a metodologia clássica para resolver as equações diferenciais envolvidas no problema da estabilidade transitória de sistemas elétricos multimáquinas. No tópico seguinte será apresentada a base algébrica que constitui o modelo clássico para estudo da estabilidade transitória.

5.1 Análise de estabilidade de transitórios

Segundo Elgerd (1978), as maiores preocupações da análise de estabilidade devem ser em torno de um tipo mais lento, porém mais importante, de transitório, a oscilação eletromecânica do gerador, que segue a uma perturbação de maiores proporções. Em Elgerd (1978) é apresentado um modelo análogo mecânico utilizando-se diversas massas presas a cordões, onde pode-se ter uma boa visão do problema de estabilidade. Comparado ao modelo elétrico as várias massas representando os geradores do sistema elétrico são suspensas por uma “rede” consistindo de cordões elásticos, esses últimos representando as linhas de transmissão elétrica. O sistema está em regime permanente estático, com cada cordão carregado abaixo de seu ponto de ruptura, o que corresponde a estar cada linha de transmissão funcionando abaixo de seu limite de estabilidade.

Se num dado instando, um dos cordões é subitamente cortado, o que corresponde à perda de uma das linhas de transmissão, como resultado as massas

terão movimentos acoplados e as suas forças nos cordões irão variar. Segundo Elgerd (1978), a perturbação súbita pode causar os seguintes efeitos finais:

1. O sistema atingirá um novo estado de equilíbrio, caracterizado por um novo conjunto de forças nos cordões, ou nas linhas de transmissão como no caso do sistema elétrico.
2. Devido às forças transitórias, um outro cordão poderá se romper, provocando o enfraquecimento da rede e resultando no rompimento em cadeia de cordões e eventual colapso total do sistema.

Quando um sistema pode sobreviver à perturbação e atinge um novo regime permanente, pode-se dizer que ele possui “estabilidade de transitório para a falta em questão” (ELGERD, 1978), porém deve-se notar que, logicamente, o sistema pode ser estável para o transitório que segue à perde de uma linha em particular e instável para a perda de outras linhas.

Para o modelo proposto em Elgerd (1978), quanto à sua estabilidade em transitório, deve-se proceder como segue:

1. Determine o estado inicial pré-falta.
2. Inicie a falta.
3. Calcule o movimento transitório pós-falta das massas e as forças resultantes nos cordões.
4. Se essas forças não excederem os ponto de ruptura de cordões, o sistema será considerado estável para a falta em questão.

Um estudo de estabilidade em transitório de um sistema de energia elétrica segue um roteiro análogo. Após a perturbação, as posições angulares dos rotores sofrerão desvios transitórios. Como a falta é admitida de grandes proporções, essas oscilações serão de larga escala. Se puder ser determinado pela análise que todos os ângulos individuais dos rotores irão fixar-se em novos valores de regime pós-falta, correspondendo a um novo estado de equilíbrio síncrono, então pode-se concluir que o sistema é estável em regime transitório, para a perturbação considerada.

5.2 Modelagem Matemática para Estudo Clássico de Estabilidade Transitória

As rotinas computacionais desenvolvidas para os estudos de estabilidade transitória consideraram o modelo clássico para as máquinas síncronas (KUNDUR, 1994). Uma vez que as equações são não lineares, não há solução analítica, e somente soluções numéricas podem ser obtidas. Para resolver estas equações, foi implementada integração numérica pelo método de Runge-Kutta de 4ª ordem.

Segundo Stevenson (1986), para estudar a dinâmica das máquinas síncronas durante as faltas, precisamos relacionar as grandezas elétricas e mecânicas de forma coerente. Na Figura 4 é representada a máquina síncrona, onde pelas relações físicas, sabe-se que o momento de inércia "J" multiplicado pela aceleração angular é igual ao torque aplicado ao rotor da máquina síncrona, isto é:

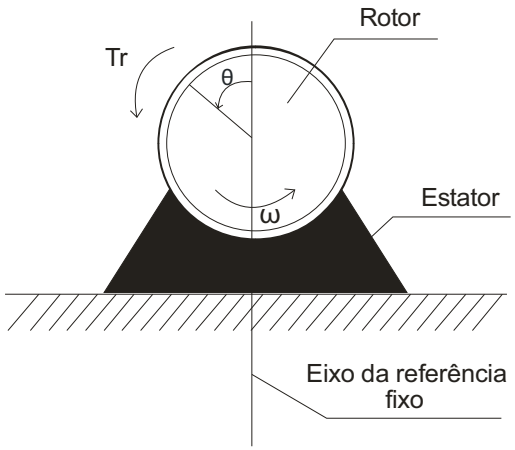


Figura 4: Máquina Síncrona.

$$J\ddot{\theta} = Tr \tag{5.1}$$

onde:

J - momento de inércia [$Kg\ m^2$];

θ_m - deslocamento angular do rotor em relação à referência;

T_r - torque resultante da diferença entre torque mecânico e elétrico, $T_r = T_m - T_e$ [$N\ M$]

O movimento do rotor de uma máquinas síncrona pode ser descrito através da seguinte equação:

$$J\frac{d^2\theta_m}{dt^2} = T_a = T_m - T_e \tag{5.2}$$

onde:

J - momento de inércia do rotor da máquina;

θ_m - deslocamento angular do rotor em relação à referência;

t - tempo;

T_m - torque mecânico fornecido pela turbina;

T_e - torque elétrico;

T_a - torque de aceleração.

Quando a máquina síncrona está em regime permanente, o torque elétrico é igual ao torque mecânico e o torque resultante (de aceleração) é nulo, ou seja, a máquina opera a uma velocidade constante e igual a velocidade síncrona. Nesta situação a máquina está em sincronismo com as demais máquinas do sistema.

O deslocamento angular da máquina (θ_m) é medido em relação a uma referência estática, ou seja, é uma medida absoluta do ângulo do rotor e cresce continuamente com o passar do tempo. Como este deslocamento angular crescente não é muito conveniente para analisar a evolução da estabilidade do rotor, pode-se simplesmente mudar a referência para uma mais conveniente. Para mediar a posição do rotor em relação a um eixo de referência que se move a velocidade síncrona, basta definir que:

$$\theta_m = \omega_{sm}t + \delta_m$$

onde:

ω_{sm} - velocidade síncrona (radianos mecânico/s);

δ_m - deslocamento angular do rotor (radianos mecânicos);

Derivando a equação anterior em relação ao tempo, tem-se:

$$\frac{d\theta_m}{dt} = \omega_{sm} + \frac{d\delta_m}{dt}$$

e derivando-se novamente em relação ao tempo, tem-se:

$$\frac{d^2\theta_m}{dt^2} = \frac{d^2\delta_m}{dt^2} \tag{5.3}$$

As equações acima indicam que a velocidade angular do rotor da máquina será constante e igual à velocidade síncrona somente quando $\frac{d\delta_m}{dt}$ for zero. Portanto, esta derivada indica o desvio do rotor em relação à velocidade síncrona. Substituindo a equação (5.3) na equação (5.2) tem-se:

$$J\frac{d^2\delta_m}{dt^2} = T_a = T_m - T_e \tag{5.4}$$

Considerando que $\omega_m = \frac{d\theta_m}{dt}$ e que a potência pode ser expressa pelo torque vezes a velocidade angular, pode-se facilmente expressar a equação (5.4) de forma bem mais interessante:

$$J\omega_m\frac{d^2\delta_m}{dt^2} = P_a = P_m - P_e$$

onde:

P_a - potência de aceleração;

P_m - potência mecânica de entrada no eixo da máquina;

P_e - potência elétrica no entreferro da máquina;

$J\omega_m$ - momento angular do rotor.

Freqüentemente o momento angular do rotor é expresso, quando considerada a velocidade síncrona, pela constante M . Ou seja:

$$M\frac{d^2\delta_m}{dt^2} = P_a = P_m - P_e \tag{5.5}$$

Outra constante, muito utilizada nos estudos de estabilidade transitória e relacionada à inércia das massa girantes das máquinas síncronas, é a constante H que pode ser expressa como:

$$H = \frac{\frac{1}{2} J \omega_{sm}^2}{S_{maq}} = \frac{\frac{1}{2} M \omega_{sm}}{S_{maq}}$$

onde:

S_{maq} - potência aparente nominal da máquina;

ou seja, a constante M pode ser reescrita como:

$$M = \frac{2H}{\omega_{sm}} S_{maq}$$

que aplicada a equação (5.5) nos fornecerá:

$$\frac{2H}{\omega_{sm}} \frac{d^2 \delta_m}{dt^2} = \frac{P_a}{S_{maq}} = \frac{P_m}{S_{maq}} - \frac{P_e}{S_{maq}}$$

Esta equação conduz a um resultado muito apropriado para a implementação computacional: como têm-se potências divididas pela potência nominal do gerador (potência base), têm-se os dados em por unidade (p.u.) dos valores nominais da máquina. Ainda é possível realizar mais uma simplificação na equação acima, o que facilitará sobremaneira a modelagem dos algoritmos:

$$\frac{2H}{\omega_{sm}} \frac{d^2 \delta}{dt^2} = P_a = P_m - P_e \tag{5.6}$$

ou ainda:

- Considerando δ em radianos elétricos:

$$\frac{H}{\pi f} \frac{d^2 \delta}{dt^2} = P_a = P_m - P_e \tag{5.7}$$

e

- Considerando δ em radianos mecânicos:

A equação (5.6) é chamada de Equação de Oscilação de uma máquina síncrona do sistema (STEVENSON, 1986). Esta é a equação que representa a dinâmica das máquinas nos estudos de estabilidade e deve ser resolvida para cada máquina e para cada período em estudo (durante o acontecimento de faltas e após a eliminação das mesmas).

Como as Equações de Oscilação são diferenciais de segunda ordem, torna-se muito conveniente reescrevê-las como duas equações diferencias de primeira ordem. Este passo é fundamental para a utilização do método de Runge-Kutta 4ª ordem que foi utilizado na implementação computacional do aplicativo proposto. Reescrevendo a (5.6) como duas equações diferenciais de primeira ordem, tem-se, finalmente:

$$\frac{2H}{\omega_s}\frac{d\omega}{dt} = P_m - P_e$$

e

$$\frac{d\delta}{dt} = \omega - \omega_s$$

A resolução de uma Equação de Oscilação consiste em encontrar a curva do deslocamento angular do rotor em relação ao tempo. Esta curva é chamada de Curva de Oscilação da máquina e permitirá determinar se as máquinas permanecem ou não em sincronismo.

5.3 Métodos Numéricos para Resolução de Equações Diferencias de 2ª ordem

De modo geral, não é possível determinar analiticamente a solução de uma equação diferencial não linear. Por este motivo, para resolver as equações que regem os movimentos das máquinas síncronas, que, como vimos anteriormente, são equações diferenciais de 2ª ordem, deve-se utilizar métodos numéricos apropriados.

Um método numérico para resolução de equações é chamado como “de passo único” quando o valor de Y_{n+1}, que representa uma melhor aproximação para a solução da equação, pode ser calculado somente se o valor imediatamente

anterior (isto é, Y_n) for previamente conhecido. Os métodos de Euler, Euler Modificado, Adams e de Runge-Kutta são métodos de passo único.

Estes métodos de integração numérica empregam a técnica "passo-a-passo" para a determinação de valores da variável dependente para um conjunto de valores pré-determinados da variável independente, que no caso da equação de oscilação da máquina síncrona, é o tempo. O processo mais comumente utilizado consiste na seleção dos valores da variável independente como múltiplos de um intervalo fixo. A precisão da solução dependerá então do método numérico usado e da amplitude do intervalo escolhido.

Cada método numérico apresenta formulação matemática distinta e quanto maior a complexidade desta formulação maiores serão os requisitos computacionais para o processamento do método. Como será apresentado adiante, existem métodos mais simples em que os cálculos podem ser elaborados manualmente se desejado.

Os métodos numéricos mais utilizados para a resolução de equações diferenciais ordinárias são:

- método de Euler (Leonhard Euler);
- método de Euler modificado e
- método de Runge-Kutta (Carl Runge e Wilhelm Kutta).

As equações diferenciais ordinárias de primeira ordem que se devem resolver nos estudos de estabilidade transitória têm a forma:

$$\frac{dy}{dx} = f(x, y(x)), y_0(x_0) = y_0$$

onde:

x - variável independente;

y - variável dependente;

y_0 - valor inicial da variável dependente (quando $x = x_0$);

Ou seja, os métodos computacionais que deve-se utilizar para a resolução de sistemas de equações diferenciais não lineares, pertencem à classe dos métodos de

variável discreta considerando condições iniciais conhecidas. Estes métodos têm como objetivo calcular aproximações para a solução, em um conjunto de pontos discretos $X_0, X_1, X_2, ...,$ da variável independente.

A solução aproximada obtida será definida por uma tabela de valores (X_n, Y_n) com $n = 1,2,...,$ etc. Os métodos de passo único requerem em cada iteração apenas o conhecimento do valor das variáveis dependentes das iterações anteriores, enquanto que os métodos de passo múltiplo requerem o conhecimento dos valores das variáveis dependentes de várias das iterações precedentes, dependendo da ordem do método escolhido.

A principal vantagem dos métodos de passo único em relação aos métodos de passo múltiplo é que é suficiente conhecer o valor da função no ponto inicial, para se poder determinar os seus valores nos pontos seguintes (STEVENSON, 1986). A precisão da solução encontrada vai depender do método e do passo de integração utilizados.

Associados à resolução numérica de uma equação diferencial existem erros de arredondamento e erros relacionados à discretização do problema. Os erros de discretização são caracterizados pela diferença entre o valor exato de uma função $Y(X_n)$ no ponto X_n e o valor Y_n calculado pelo método adotado para a resolução. Tais erros dependem do tipo, ordem e passo do método de integração utilizado. Os erros de arredondamento surgem na execução das operações aritméticas e são causados pela incapacidade do computador em representar os números de forma exata. Em lugar de Y_n, obtém-se um valor $Y_{calculado}$ que difere de Y_n devido a erros de arredondamento. O erro de arredondamento local é propagado ao longo do processo iterativo.

Para reduzir os erros de discretização normalmente escolhe-se um passo de integração pequeno, mas quanto menor for o passo de integração maior é o número de iterações e consequentemente os erros de arredondamento acumulados. Haverá assim um passo de integração ótimo para cada problema, que na prática é muito difícil de se determinar.

Na resolução de equações diferenciais ordinárias, se os erros não aumentam de iteração para iteração diz-se que o procedimento é estável. Caso contrário, poderá surgir instabilidade numérica inerente ou induzida. A instabilidade inerente ocorre quando os erros se propagam através dos cálculos com efeito crescente, de

tal forma que a solução obtida pode distanciar-se drasticamente da solução exata. A instabilidade induzida está relacionada com o método utilizado na resolução numérica das equações diferenciais.

A estabilidade do método é basicamente uma medida da diferença entre a solução aproximada e a solução esperada à medida que o número de passos de integração aumenta.

Os métodos de integração numérica também podem ser classificados como explícitos ou implícitos. Nos métodos explícitos o valor da variável dependente Y, para qualquer valor da variável independente X, é calculado a partir do conhecimento do valor anterior da variável independente. Uma limitação dos métodos de integração explícitos é a de que não são numericamente estáveis. Os métodos de integração explícitos mais usados são os métodos de: Euler, Euler Modificado e Runge–Kutta.

Os métodos de integração implícitos usam funções de interpolação e permitem passos de integração relativamente elevados, tornando-os melhores para simulações de média e longa duração. Os métodos de integração implícitos mais usados são o método de Gear e o método de Adams–Bashford.

No desenvolvimento computacional deste trabalho, foi utilizado o método de Runge Kutta 4ª Ordem.

Método de Runge-Kutta

Pelo método de Runge-Kutta os incrementos nos valores das variáveis dependentes são calculados a partir de um conjunto de fórmulas. Uma vez que cada valor de y é determinado pelas fórmulas de uma maneira unívoca, este método não requer repetidas aproximações, como nos demais métodos numéricos para integração de equações diferenciais.

Um inconveniente do método de Runge-Kutta é que, ao longo do processo de cálculo, não é possível conhecer os valores dos erros. A principal vantagem deste método é que, conhecendo apenas o valor da função no ponto inicial, se pode determinar os seus valores nos pontos seguintes.

As fórmulas utilizadas no método de Runge-Kutta são obtidas utilizando uma aproximação para o desenvolvimento em série de Taylor da função. Conforme Stagg

e EL-abiad (1968), pode-se considerar que uma equação diferencial a resolver tenha a forma:

$$\frac{dy}{dx} = f(x, y)$$

e os valores iniciais sejam representados pelo par ordenado (x_0, y_0).

Desenvolvendo a equação diferencial em série de Taylor em torno do ponto inicial, tem-se:

$$y_1 = y_0 + \left(\frac{dy}{dx}\right)_0 h + \left(\frac{d^2y}{dx^2}\right)\frac{h^2}{2!} + \ldots$$

Para tal expressão ser válida, o ponto inicial não deve ser singular e h deverá ser suficientemente pequeno para que a série seja convergente.

Considerando que:

$$\left(\frac{dy}{dx}\right)_0 = f(x_0, y_0)$$

e

$$\left(\frac{d^2y}{dx^2}\right)_0 = \left(\frac{\delta f}{\delta x}\right)_0 + \left(\frac{\delta f}{\delta y}\right)_0 f(x_0, y_0)$$

tem-se que:

$$y_1 = y_0 + f(x_0, y_0)h + \left(\frac{\delta f}{\delta x}\right)h^2 + \left(\frac{\delta f}{\delta y}\right)f(x_0, y_0)\frac{h^2}{2}$$

pode-se reescrever a equação acima da seguinte forma:

$$y_1 = y_0 + a_1k_1 + a_2k_2 \qquad (5.8)$$

onde:

$$k_1 = f(x_0, y_o)h$$

e

$$k_2 = f(x_0 + b_1 h, y_o + b_2 k_1)h$$

Os coeficientes a_1, a_2, b_1 e b_2 devem ser calculados considerando a expansão em Séries de Taylor da equação abaixo em torno do ponto (x_0, y_0).

$$f(x_0 + b_1 h, y_0 + b_2 k_1)$$

realizando este procedimento, obtém-se:

$$k_2 = \left\{ f(x_0, y_0) + b_1 h \left(\frac{\delta f}{\delta x} \right)_0 + b_2 k_1 \left(\frac{\delta f}{\delta y} \right)_0 + \ldots \right\} h$$

Geralmente, apenas os dois primeiros termos da expansão em série de Taylor são utilizados. Ou seja:

$$y_1 = y_0 + (a_1 + a_2) f(x_0, y_0)h + a_2 b_1 \left(\frac{\delta f}{\delta x} \right)_0 h^2 + a_2 b_2 f(x_0, y_0) \left(\frac{\delta f}{\delta y} \right)_0 h^2$$

Em torno de (x_0, y_0):

$$y_1 = y_0 + \left(\frac{dy}{dx} \right)_0 h + \left(\frac{d^2 y}{dx^2} \right)_0 \frac{h^2}{2} + \ldots \qquad (5.9)$$

e

$$\left(\frac{dy}{dx} \right)_0 = f(x_0, y_o)$$

$$\left(\frac{d^2 y}{dx^2} \right)_0 = \left(\frac{\delta f}{\delta x} \right) + \left(\frac{\delta f}{\delta y} \right)_0 f(x_0, y_0)$$

substituindo na equação (5.8), vem:

$$y_1 = y_0 + f(x_0, y_0)h + \left(\frac{\delta f}{\delta x}\right)_0 \frac{h^2}{2} + f(x_0, y_0)\left(\frac{\delta f}{\delta y}\right)_0 \frac{h^2}{2}$$

pode-se, finalmente, concluir que:

$$a_1 + a_2 = 1 \ , \ a_2 b_1 = \frac{1}{2} \text{ e } a_2 b_2 = \frac{1}{2}$$

Estes resultados caracterizam um sistema de equações com três equações e quatro incógnitas. Determinando um valor qualquer para a_1, pode-se escrever a solução deste sistema em função desta variável:

Se $a_1 = \frac{1}{2}$ então:

$$a_2 = \frac{1}{2} \qquad \text{e} \qquad b_1 = b_2 = 1$$

voltando a equação (5.8):

$$y_1 = y_0 + \frac{1}{2}k_1 + \frac{1}{2}k_2 \tag{5.10}$$

onde:

$$k_1 = f(x_0, y_0)h$$
$$k_2 = f(x_0 + h, y_0 + k_1)h$$

A equação (5.10) é a aproximação de Runge-Kutta de 2ª ordem para a solução da equação diferencial dada. O erro em relação à solução exata é da ordem de h^3 (h é o passo de integração escolhido).

A aproximação da quarta ordem do método de Runge-Kutta é a mais comumente utilizada, pois seu erro é muito pequeno (h^5). Analogamente a

aproximação de 2ª ordem, conforme Stagg e El-Abiad (1968), pode-se obter a aproximação de 4ª ordem:

$$y_1 = y_0 + a_1 k_1 + a_2 k_2 + a_3 k_3 + a_4 k_4 \tag{5.11}$$

onde:

$$k_1 = f(x_0, y_0)h$$
$$k_2 = f(x_0 + b_1 h, y_0 + b_2 k_1)h$$
$$k_3 = f(x_0 + b_3 h, y_0 + b_4 k_2)h$$
$$k_4 = f(x_0 + b_5 h, y_0 + b_6 k_3)h$$

utilizando processo análogo ao utilizado para a aproximação de 2ª ordem, tem-se os coeficientes:

$$a_1 = a_4 = \frac{1}{6},\ a_2 = a_3 = \frac{2}{6},\ b_1 = b_2 = b_3 = b_4 = \frac{1}{2} \text{ e } b_5 = b_6 = 1$$

substituindo os coeficientes acima na equação (5.12), ficará:

$$y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

que corresponde à aproximação de 4ª ordem da solução da equação diferencial.

Conforme Stagg e El-Abiad (1968), os valores de k_1, k_2, k_3 e k_4 podem ser obtidos por:

$$k_1 = f(x_0, y_0)h$$
$$k_2 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{k_1}{2}\right)h$$
$$k_3 = f\left(x_0 + \frac{h}{2}, y_0 + \frac{k_2}{2}\right)h$$
$$k_4 = f(x_0 + h, y_0 + k_3)h$$

5.4 Aplicação na Resolução das Equações de Oscilação das Máquinas Síncronas

Na seção anterior abordou-se a aplicação do método de Runge-Kutta às equações diferenciais de 1ª ordem. Para utilizar o método nas equações de oscilação das máquinas síncronas, que são de 2ª ordem, precisam-se introduzir variáveis auxiliares. Considerando a equação diferencial de segunda ordem genérica abaixo:

$$c_1 \frac{d^2 y}{dx^2} + c_2 \frac{dy}{dx} + c_3 y = 0$$

pode-se reescrevê-la como duas equações diferencias de 1ª ordem:

$$\frac{dy}{dx} = y'$$

$$\frac{d^2 y}{dx^2} = \frac{dy'}{dx} = -c_2 \frac{\left(y' + c_3 y\right)}{c_1}$$

Utilizando este artifício para as equações de oscilação, tem-se:

$$M \frac{d^2 \delta_m}{dt^2} = P_a = P_m - P_e \quad \text{(equação original de 2ª ordem)}$$

Reescrevendo-a como duas equações de 1ª ordem, tem-se:

$$\frac{d\delta_m}{dt} = \omega_t - \omega_o \tag{5.12}$$

e

$$\frac{d\omega}{dt} = \frac{P_a}{M} = \frac{P_m - P_e}{M} \tag{5.13}$$

Como a constante H é comumente utilizada para representar a inércia dos rotores das máquinas, cabe relembrar que:

$$M = \frac{2H}{\omega_0} S_{maq}$$

No algoritmo desenvolvido a equação (5.12) é representada pela função f e a equação (5.13) pela função g, ou seja:

$$f(\omega, \delta, P_a) = \frac{d\delta_m}{dt} = \omega - \omega_o$$

$$g(\omega, \delta, P_a) = \frac{d\omega}{dt} = \frac{P_a}{\frac{2H}{\omega_0} S_{maq}}$$

Para cada instante de tempo pode ser obtido um par ordenado (ângulo; velocidade) utilizando o aproximado de Runge-Kutta:

$$\delta_1 = \delta_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$\omega_1 = \omega_0 + \frac{1}{6}(l_1 + 2l_2 + 2l_3 + l_4)$$

onde:

$$k_1 = f(\omega_0, \delta_0, P_{a0})h$$

$$k_2 = f\left(\omega_0 + \frac{l_1}{2}, \delta_0 + \frac{k_1}{2}, P_{a0}\right)h$$

$$k_3 = f\left(\omega_0 + \frac{l_2}{2}, \delta_0 + \frac{k_2}{2}, P_{a0}\right)h$$

$$k_4 = f(\omega_0 + l_3, \delta_0 + k_3, P_{a0})h$$

$$l_1 = g(\omega_0, \delta_0, P_{a0})h$$

$$l_2 = g\left(\omega_0 + \frac{l_1}{2}, \delta_0 + \frac{k_1}{2}, P_{a0}\right)h$$

$$l_3 = g\left(\omega_0 + \frac{l_2}{2}, \delta_0 + \frac{k_2}{2}, P_{a0}\right)h$$

$$l_4 = g(\omega_0 + l_3, \delta_0 + k_3, P_{a0})h$$

Desta forma, são resolvidas para cada instante de tempo, as duas equações diferenciais de 1ª ordem que representam a equação de oscilação. Têm-se dois regimes a serem considerados: o de falta e o de pós-falta, sendo em cada um deles o sistema representado pelo sistema de equações diferenciais correspondente.

5.5 Obtenção das tensões de rede durante um transitório

O sistema desenvolvido considera o modelo multimáquinas apresentado em (COLVARA, 2005). Esse modelo é constituído por um conjunto de **n** máquinas que fornecem potência a uma rede de **m** barras, como na Figura 5.

As injeções de correntes nas barras são dadas por:

$$I = YV \tag{5.14}$$

onde:

I – vetor de injeção de corrente de barra;

V - vetor de tensões de barra; e

Y – matriz admitância de barra.

assim pode-se definir a i-ésima injeção de corrente de barra:

$$\dot{I}_i = \sum_{j=1}^{m} \overline{Y}_{ij}\, \dot{V}_j \tag{5.15}$$

Se as cargas de admitância constante são mantidas e adicionando-se à rede as barras internas das máquinas, será formada a rede aumentada conforme a Figura 5 (COLVARA, 2005).

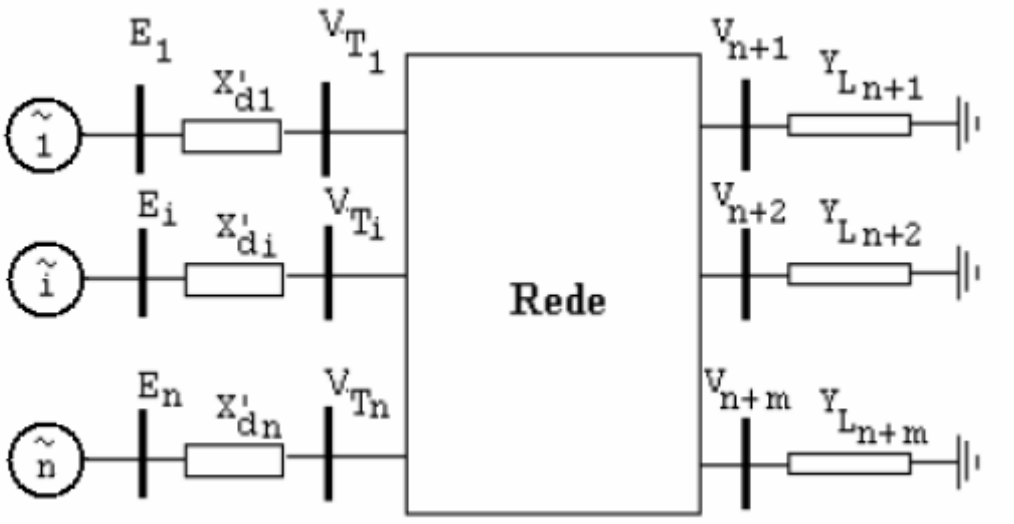


Figura 5: Sistema multimáquina com barras internas e cargas (COLVARA, 2005).

Desta forma, a equação nodal da rede aumentada pode ser conferida na equação (5.16).

$$\begin{bmatrix} \dot{I}g_1 \\ \dot{I}g_2 \\ \vdots \\ \dot{I}g_n \\ \hline \dot{I}_1 \\ \dot{I}_2 \\ \vdots \\ \dot{I}_m \end{bmatrix} = \left[\begin{array}{cccc|ccc} -j\frac{1}{X'd_1} & & & & j\frac{1}{X'd_1} & & \\ & -j\frac{1}{X'd_2} & & & & j\frac{1}{X'd_2} & \\ & & \ddots & & & & \ddots \\ & & & -j\frac{1}{X'd_n} & & & j\frac{1}{X'd_n} \\ \hline j\frac{1}{X'd_1} & & & & Y_{11}-j\frac{1}{X'd_1} & \cdots & Y_{1n} \\ & j\frac{1}{X'd_2} & & & \vdots & \ddots & \vdots \\ & & \ddots & & & & \\ & & & j\frac{1}{X'd_n} & Y_{n1} & \cdots & Y_n-j\frac{1}{X'd_n} \end{array}\right] \cdot \begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_n \\ \hline V_1 \\ V_2 \\ \vdots \\ V_m \end{bmatrix} \quad (5.16)$$

Se for considerado que as cargas estão incorporadas à rede, as injeções de corrente nas barras de carga são nulas, uma vez que toda corrente que chega pelos elementos da rede à barra de carga atende a carga, agora inclusa à rede, e as correntes injetadas são $\dot{I}_1 = \dot{I}_2 = ... = \dot{I}_m = 0$

Uma vez que se fez a inclusão das barras internas dos geradores à rede e também as cargas que são representadas por admitâncias constantes, a rede aumentada fica descrita pela equação matricial

$$\begin{bmatrix} \mathbf{I_g} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{Y_{gg}} & \mathbf{Y_{gr}} \\ \mathbf{Y_{rg}} & \mathbf{Y_{rr}} \end{bmatrix} \begin{bmatrix} \mathbf{E} \\ \mathbf{V} \end{bmatrix}$$

Onde o índice **g** refere-se às barras internas dos geradores e **r** às barras da rede. Como a corrente injetada nas barras da rede é nula, como descrito antes, então as equações correspondente às injeções de corrente são expressas na equação matricial:

$$\mathbf{0} = \mathbf{Y_{rg}}\mathbf{E} + \mathbf{Y_{rr}}\mathbf{V}$$

de onde:

$$\mathbf{V} = -\mathbf{Y_{rr}^{-1}}\mathbf{Y_{rg}}\mathbf{E}$$

$$\mathbf{V} = \mathbf{Y_{V/E}}\mathbf{E}$$

com:

$$\mathbf{Y_{V/E}} = -\mathbf{Y_{rr}^{-1}}\mathbf{Y_{rg}}$$

assim conforme Anderson e Fouad (1994), a rede reduzida é:

$$\mathbf{I_g} = \left[\mathbf{Y_{gg}} - \mathbf{Y_{gr}}\mathbf{Y_{rr}^{-1}}\mathbf{Y_{rg}}\right]\mathbf{E}$$

Como esta redução é efetuada para o período de falta e para o pós-falta, basta armazenar os resultados

$$\mathbf{Y}_{V/E}^{falta} \; e \; \mathbf{Y}_{V/E}^{pós-falta}$$

A matriz $\mathbf{Y_{V/E}} = -\mathbf{Y_{rr}^{-1}}\mathbf{Y_{rg}}$ é uma matriz complexa que tem número de linhas igual a *m*, o número de barras da rede original (antes de acrescentar as barras internas dos geradores), e número de colunas igual a n_g, o número de geradores.

Cálculo das tensões na rede.

A cada passo de integração se tem o instante *t* e os ângulos $\delta_i(t),\ i = 1,2,...n_g$ e, as tensões internas fasoriais são obtidas como:

$$\dot{E}_i(t) = E_i \angle \underline{\delta_i(t)} = E_i \cos(\delta_i(t)) + jE_i sen(\delta_i(t))$$

e o vetor das tensões internas no instante *t* é constituído como

$$\mathbf{E}(t) = \begin{bmatrix} \dot{E}_1(t) \\ \dot{E}_2(t) \\ \dot{E}_3(t) \\ \vdots \\ \dot{E}_{n_g}(t) \end{bmatrix}$$

e as tensões nas barras da rede,

$$\mathbf{V}(t) = \begin{bmatrix} \dot{V}_1(t) \\ \dot{V}_2(t) \\ \dot{V}_3(t) \\ \vdots \\ \dot{V}_m(t) \end{bmatrix} \tag{5.17}$$

são então obtidas por meio da equação

$$\mathbf{V}(t) = \mathbf{Y}_{\mathbf{V/E}}\mathbf{E}(t)\,.$$

Para se obter os valores das tensões de barra da rede a cada instante no regime de falta e de pós-falta, faz-se:

$$\mathbf{V}^{falta} = \mathbf{Y}_{V/E}^{falta}\mathbf{E}$$
$$\mathbf{V}^{pós-falta} = \mathbf{Y}_{V/E}^{pós-falta}\mathbf{E}$$

As tensões $\mathbf{V}(t)$ podem ser obtidas a cada passo t de integração ou, uma vez que os valores de $\theta(t)$ estão armazenados em banco de dados, o cálculo pode ser efetuado “off-line”.

CAPITULO 6

Programação Estruturada e Orientada a Objetos

A Programação Estruturada é baseada em linguagens de programação procedurais. Essas linguagens possuem estruturas de controle que promovem teste de condições, repetição de blocos de códigos e seleção de alternativas, além de dividir código do programa em módulos, chamados de funções ou procedimentos. As linguagens procedurais são caracterizadas pela existência de algoritmos, que determinam à seqüência de chamadas de procedimentos que constituem o programa.

A Programação Orientada a Objetos manteve algumas características da programação estruturada e acrescentou novos aspectos para uma nova abordagem de programação. Uma das principais características é basear-se em objetos do mundo real e não em procedimentos de algoritmos. Os procedimentos mudam com muita freqüência e os objetos do mundo real são mais estáveis.

A orientação a objetos apresenta-se favorável à modularização e reutilização de componentes, facilita a transição entre análise e projeto, exibe melhores resultados em qualidade, produtividade e apresenta-se mais flexível a mudanças e adaptações. A análise e projeto orientados a objetos são voltados para a construção de modelos melhores, mais próximos da realidade através do uso dos principais conceitos da orientação a objetos como abstração, encapsulamento, herança e polimorfismo, criando um vocabulário e entendimentos comuns entre usuários do sistema e os desenvolvedores (AGOSTINI, DECKER E SILVA, 2002).

A motivação para esse trabalho foi a necessidade de buscar novas alternativas para as soluções de sistemas de energia. À que medida que os sistemas crescem, também cresce a complexidade e torna-se mais difícil satisfazer a um número cada vez maior de requisitos desses sistemas, muitas vezes conflitantes. A programação orientada a objetos tem papel fundamental, para ser empregada no desenvolvimento de sistemas de software complexos e de grande porte. A programação orientada a objetos foi desenvolvida devido às limitações encontradas na programação estruturada, ditas procedimentais.

Pascal, C, Basic e Fortran são linguagens de programação procedimentais. Isto é, cada declaração na linguagem informa que o computador deve realizar alguma tarefa como, por exemplo, ler um dado de entrada, adicionar uma constante, dividir por algum número e exibir o resultado. Um programa em uma linguagem procedimental é visto como uma lista de instruções, executadas de forma estruturada.

Para pequenos programas, nenhum princípio organizacional (ou paradigma) é necessário. O programador cria uma lista de instruções e o computador as executa. Entretanto, à medida que os programas se tornam maiores, entender e tratar uma única lista de instruções fica mais difícil.

Para amenizar a dificuldade do programador em entender códigos extensos, é possível dividir esse código em unidades menores, denominadas funções ou procedimentos. Cada função tem uma proposta claramente definida, bem como uma interface bem definida com as outras funções do programa (RUMBAUGH, BLAHA, PREMERLANI, et al., 1994).

Dividir um programa em funções pode ser efetuado através do agrupamento de várias funções em uma entidade maior, denominada módulo. O princípio é similar, um módulo nada mais é que um grupo de componentes executando tarefas específicas.

Dividir um programa em funções e módulos é um dos fundamentos da programação estruturada. No entanto, à medida que os programas se tornam maiores e mais complexos, também a programação estruturada mostra-se limitada.

Um exemplo clássico acontece quando muitas funções têm acesso a um conjunto de dados. A organização dos dados não pode ser modificada sem que todas as funções que têm acesso a eles também sejam modificadas. Se novos dados são adicionados, então será necessário modificar todas as funções que têm acesso a esses dados para que elas também possam ter acesso aos novos dados. Tornar-se-á difícil achar tais funções e modificá-las corretamente.

Neste sentido, considera-se importante a introdução de programação orientada a objetos (POO). A idéia por trás das linguagens de programação orientadas a objetos, ou linguagens orientadas a objetos, é combinar em uma única entidade tanto os dados quanto as funções que operam sobre estes dados. Tal entidade é denominada objeto. As funções de um objeto, chamadas funções membro em C++, tipicamente, oferecem uma única forma de acesso a seus dados.

Assim, caso torne-se necessário ler dados em um objeto, então basta chamar a função membro desse objeto. Ela lerá os dados e retornará o valor a quem chamou a função. Os dados não podem ser acessados diretamente, são ocultados, ficando protegidos contra alterações acidentais (RUMBAUGH, BLAHA, PREMERLANI, et al., 1994).

Dados e funções são ditos ser encapsulados em uma única entidade, denominada objeto. O encapsulamento e ocultação de dados são aspectos importantes na descrição de linguagens orientada a objetos. Além disso, se o programador precisar modificar os dados de um objeto, ele terá de saber exatamente quais funções interagem com aquele objeto (suas funções membro). Outras funções não podem ter acesso ao(s) dado(s). Como resultado, obtém-se simplificação na escrita, bem como na depuração e manutenção do programa. Neste contexto, um programa C++, tipicamente, consiste de muitos objetos comunicando-se entre si através da chamada de funções membro dos outros objetos. Chamar a função membro de um objeto é como enviar uma mensagem para o objeto.

Pensar em termos de objetos é muito parecido a como faríamos na vida real. Por exemplo, vamos pensar em um automóvel como exemplo de um modelo de um esquema de POO. Diríamos que o carro é o elemento principal que tem uma série de características, como poderiam ser a cor, o modelo ou a marca e também uma série de funcionalidades associadas, como podem ser andar, parar ou estacionar.

Então em um esquema POO o automóvel seria o objeto, as propriedades seriam as características como a cor ou o modelo e os métodos seriam as funcionalidades associadas como andar ou parar.

Em outra situação, considere-se a definição de um modelo de uma fração em um esquema POO, ou seja, essa estrutura matemática que tem um numerador e um denominador que divide o numerador. Por exemplo, seja a fração 3/2.

A fração será o objeto e terá duas propriedades, o numerador e o denominador. Poder-se-ia, de imediato, definir vários métodos, como simplificar, somar com outra fração ou número, subtrair com outra fração.

Estes objetos poderão ser utilizados nos programas, por exemplo, em um programa de matemática seria feito o uso de objetos fração e em um programa relacionado a uma oficina de carros, seria utilizado o uso de objetos carro. Os programas orientados a objetos utilizam muitos objetos para realizar as ações a que se destinam e eles mesmos também são objetos, ou seja, a oficina de carros será

uma classe (definida nos tópicos a seguir) que poderá utilizar como objetos automóvel, ferramenta, mecânico, trocas.

Programação orientada a objetos

O conceito de programação orientada por objetos não é novo. No final da década de 60, a linguagem Simula67, desenvolvida na Noruega, introduzia conceitos hoje encontrados nas linguagens orientadas a objetos. Em meados dos anos 70, o Centro de Pesquisa da Xerox (PARC) desenvolveu a linguagem Smalltalk, a primeira totalmente orientada a objetos. No início da década de 80, a AT&T lançaria a Linguagem C++, uma evolução da linguagem C em direção à orientação a objetos.

O surgimento de novas técnicas e linguagens de programação, principalmente a partir dos anos 80, permitiu a implementação de programas com um alto grau de complexidade, sem perda de legibilidade do código. Dentre estas novas metodologias de programação, a POO surgiu como uma alternativa para os problemas associados ao desenvolvimento, manutenção e atualização de programas computacionais de grande porte reduzindo a complexidade no desenvolvimento de software e aumentando sua produtividade.

A linguagem de programação C++ tem se destacado como ferramenta para aplicações da POO e apresenta tanto as características da POO quanto facilidades relativas a eficiência e portabilidade do código escrito.

A programação orientada a objetos não tem a intenção de substituir a programação estruturada tradicional. Pode-se considerar que a POO é uma evolução de práticas que são recomendadas na programação estruturada, mas não formalizadas, como o uso de variáveis locais e modularização do código. O modelo de objetos permite a criação de bibliotecas que tornam efetivos o compartilhamento e a reutilização de código, reduzindo o tempo de desenvolvimento e, principalmente, simplificando o processo de manutenção das aplicações (RUMBAUGH, BLAHA, PREMERLANI, et al., 1994).

O surgimento das linguagens de concepção moderna (C, C++, Pascal, ADA, etc.), juntamente com a POO, introduziu novos conceitos e nomenclaturas no ambiente de programação.

A seguir são apresentados os principais conceitos introduzidos com as linguagens de última geração e com a POO. Estes conceitos são de fundamental importância para o entendimento da estrutura orientada a objetos definida para representar o SEE. É interessante salientar que uma abordagem rigorosa do tema não é o objetivo deste trabalho, podendo ser encontrada nas referências.

De um modo geral, um software desenvolvido com base na programação orientada a objetos, é um software desenvolvido com uma coleção de objetos separados que incorporam tanto a estrutura quanto o comportamento dos dados. Isso contrasta com a programação convencional, segundo a qual a estrutura e o comportamento dos dados têm pouca vinculação entre si.

A grande dificuldade para compreender a POO é a diferença de entendimento do problema. Enquanto a programação estruturada tem como principal foco as ações (procedimentos e funções), a POO se preocupa com os objetos e seus relacionamentos. Além do conceito de objeto, a POO tem como base os conceitos de objetos, classes, encapsulamento, herança, abstração e polimorfismo, definidos na seção 6.1 a seguir.

Quando surgiram os primeiros computadores, a preocupação dos programadores era a busca da maior eficiência com o pouco uso de memória devido às limitações do hardware da época. Os programas consistiam em um único bloco, pois a divisão em vários blocos consumia mais recursos. Com a evolução do hardware a preocupação está na eficiência do desenvolvimento, isto é o tempo de trabalho dos programadores (RUMBAUGH, BLAHA, PREMERLANI, et al., 1994).

A modularização do sistema aumentou a potencialidade do reuso de código e facilitou o entendimento de códigos complexos. A abstração é um conceito fundamental para conseguir uma boa modularização, excluindo do contexto do problema aquilo que não interessa para a sua solução. É fundamental para o raciocínio e resolução de problemas, focar em seus aspectos relevantes. Em programas bem modularizados, cada módulo representa uma abstração existente no contexto do problema. Essa técnica é empregada em sua forma mais simples por funções e procedimentos parametrizados, criando funções genéricas que resolvem subproblemas.

6.1 Conceitos Básicos de Programação Orientada a Objetos

Modelagem e projetos baseados em objetos é um novo modo de estudar os problemas com a utilização de modelos fundamentados em conceitos do mundo real. A estrutura básica é o objeto, que combina a estrutura e o comportamento dos dados de uma única entidade. Um objeto pode ser concreto como um gerador ou linha de transmissão em um sistema de energia elétrica ou conceitual como a aplicação de estudos de potência.

Um grupo de objetos é descrito por uma classe. Uma classe agrupa objetos com propriedades e comportamentos semelhantes. As propriedades são os atributos dos objetos. Os comportamentos são as ações realizadas pelo objeto.

Pensando em uma linha de transmissão como um objeto, seus atributos são: *r:* resistência; *x:* impedância; *o:* barra de origem; *d:* barra de destino; *Bsh:* admitância shunt da linha. Esse objeto exerce ações como, por exemplo, a coleta dos dados de um arquivo texto. Neste trabalho foi usada a formatação padrão de arquivos fornecida pelo IEEE. Neste trabalho **ler_dados** foi usado para representar o método que faz a entrada dos dados de linha. O método c*alc_adm* foi utilizado para calcular os valores das admitâncias séries de cada linha de transmissão.

Assim para representar esse objeto linha é necessária uma classe que agrupe todas as características do objeto (atributos e métodos), de modo que se possam criar infinitos objetos do tipo linha (classe linha). Os métodos de uma classe também são chamados de funções-membro. Essas funções membro pertencem a uma determinada classe e existem somente dentro dela.

As classes são divididas em seções: privada, pública e protegida (*private, public e protect).* Tanto os atributos quanto as funções-membro podem ser declaradas em qualquer uma dessas seções desde que respeitem suas regras de acesso.

A seção *private* de uma classe determina as implementações internas, ou seja, atributos restritos a classe em questão. Isso significa que somente funções-membro escritas nessa classe podem acessá-los (RUMBAUGH, BLAHA, PREMERLANI, et al., 1994).

Esse artifício de proteger os dados internos de uma classe das ações de funções-membro não pertencentes a ela denomina-se *Encapsulamento.*

A seção *public* de uma classe permite acesso a todas as outras funções da aplicação, ou seja, em qualquer parte do programa é possível se criar um objeto da classe e chamar essa função membro.

A seção *protect,* assim como a seção *private,* não pode permitir acesso a usuários, porém pode ser acessada por meio de classes derivadas desta classe, permitindo que novos elementos públicos sejam adicionados.

As classes são estruturas que determinam tipo de dados. Não se pode confundir a declaração de uma classe com a declaração de uma *struct.* Uma *struct* agrupa várias variáveis numa só, formando um conjunto de dados não similares, que determinará o tipo de uma nova variável ou atributo de um objeto. A diferença é que as classes determinam tipos de objetos e structs tipos de atributos (variáveis).

Assim como um atributo pode ser declarado como tipo float, pode-se declarar um atributo do tipo *struct*. Na declaração da classe linha utilizada nesse trabalho foi utilizada uma *struct* para representar dados complexos, pois no compilador utilizado (Borland C++ Builder) não se dispõe de tipos de dados complexos; assim foi necessário manipular os dados complexos na sua forma pura (parte real e parte imaginária) com a *struct complexo*, mostrada a seguir:

```
struct complexo
{
  float real, img;
};
```

Abaixo está um exemplo da declaração da classe linha de transmissão utilizando C++ e sua respectiva representação gráfica.

```
lass c_linha
{
      private:
            int para,de;
            float r,x,bsh;
            struct complexo dados;
       public:
          void ler_dados(int o1, int d1, float r1,float x1, float bsh1)
          {
           de=o1;
          para=d1;
```

```
        r=r1;
        x=x1;
        bsh=bsh1;
        dados.img = x;
        dados.real = r;
       }

        void calc_adm()
        {
        float r,x;
        r = dados.real / ((dados.real)*(dados.real)+(dados.img)*(dados.img));
        x = -dados.img / ((dados.real)*(dados.real)+(dados.img)*(dados.img));
        dados.real=r;
        dados.img=x;
    }
};
```

Linha
int: para[nl] int: de[nl] int: r[nl] int: x[nl] int: bshunt[nl]
ler_dados() cal_adm()

Figura 6: Classe Linha.

Na Figura 6 pode-se notar que na declaração da classe **Linha**, as funções públicas **ler_linhas**, **Calc_admitancias**, **Bshunt_div_2**, são declaradas do tipo *void*, porque não retornam valores, apenas executam as operações e alteram os valores dos atributos internos da classe linhas de transmissão.

Em C++ para se declarar e criar um objeto usa-se o seguinte código:

C_linha *linhas_ob = new C_linha

Para se alterar os valores do objeto linhas_ob chama-se o método ler_dados:

linhas_ob->ler_dados(1,2,0.1,-.1,0.05);

onde são passados como parâmetro os valores lidos para executar a função-membro calcular_admitancia:

Linhas_ob->calcular_admitancia();

Enxergar as classes como características de objetos do mundo real caracteriza a idéia de abstração, que significa a habilidade da linguagem em modelar características reais do problema a que o programa se aplica. Por exemplo, para desenvolver um sistema que resolva um fluxo de potência é muito mais fácil lidar com uma linguagem onde pode-se criar algo como *linha de transmissão*, um objeto do mundo real, do que lidar com estruturas de dados que não tem qualquer significado inerente. A abstração se refere à capacidade da linguagem de modelar o mundo real, exatamente o objetivo do desenvolvimento de ferramentas computacionais: modelar no computador eventos do mundo físico.

Outra característica importante da Programação Orientada a Objetos é a *herança.* Herança se descreve como a habilidade que um objeto tem de herdar as características gerais de outro objeto, adicionando-lhe características próprias. Isso, além de permitir sua reutilização, diminui a quantidade de código a ser escrito. A existência desta habilidade justifica-se na necessidade dos objetos do mundo real se ajustarem em hierarquias. Em SEE esse conceito pode ser observado nesse trabalho quando criada as classes Linhas de Transmissão e Transformadores sendo que ambas herdam características da classe linha, apresentada na Figura 6.

Existem outras características da programação orientada a objetos como, por exemplo: polimorfismo e sobrecarga de operadores. Não foram usadas nesse trabalho, mas é possível descrevê-las de forma prática utilizando outros exemplos do mundo real.

O polimorfismo é a habilidade que os objetos possuem de responder de forma semelhante a um determinado comando. Também se refere a capacidade de uma mesma ação ser realizada de formas diferentes, resultando na mesma resposta. Isto normalmente resulta na criação de duas classes relacionadas, que têm o mesmo identificador de método, porém, implementadas de forma diferente.

Tomado-se como exemplo em um sistema de controle financeiro, onde tem

entidades funcionários, clientes, fornecedores que se diferem entre si pelos atributos, mas possuem características em comum como endereço, telefone. Assim pode-se tratar o objeto Funcionário como sendo o objeto Pessoa, pois o objeto Funcionário herda todos as propriedades e métodos do objeto pessoa. O contrário não pode ser feito, pois o objeto Funcionário possui características que o objeto Pessoa não tem.

Borland C++ Builder

O Boland C++ Builder é um ambiente visual de desenvolvimento de aplicações orientada a objetos que permite desenvolver, de forma rápida, aplicações para os sistemas operacionais Windows e Linux. Com ele pode-se criar eficientes aplicações com o mínimo de codificação manual. Assim pode-se com mínimo de conhecimento em C e C++ desenvolver programas relativamente complexos em pouco tempo, usando conceitos de programação visual e reutilização de códigos. Essas aplicações podem rodar em um computador especifico ou serem cliente-servidor.

O ambiente de programação do Boland C++ Builder oferece bibliotecas de componentes reutilizáveis como as VCL *(visual components library),* que contém os objetos que encapsulam as APIs *(Application Programming Interface)* e várias técnicas de programação necessárias para o desenvolvimento de aplicações Win32; CLX (*Components Library for Cross-Plataform)*, biblioteca de componentes para plataforma X, contendo objetos que encapsulam as técnicas utilizadas para o desenvolvimento de aplicações que executem tanto na plataforma Windows como em Linux.

C++ Builder oferece uma interface amigável para o desenvolvimento de sistemas. Essa ferramenta é utilizada comercialmente para desenvolvimento de sistemas robustos utilizando ou não a programação orientada a objeto. É uma ferramenta portátil de fácil entendimento, com total integração a diversos bancos de dados (Paradox, Firebird, SQL). É uma ferramenta orientada primeiramente a evento, onde através de seus componentes, o programador consegue interagir com o usuário de forma rápida e eficiente.

Suas ferramentas já estão voltadas para as principais exigências de aplicativos acadêmicos e comerciais. Possui ferramentas visuais para entrada de

dados, interação com banco de dados, interação com o sistema operacional, tabulação dos dados, interação com o usuário, acesso remoto e muitas outras ferramentas que tornam o sistema mais agradável para o usuário e mais fácil de trabalhar para o programador.

Não há nada de errado em se programar em modo texto com aplicações em console, mas no ponto de vista comercial esta abordagem está ultrapassada. É claro que em termos de eficiência um software visual e software em modo texto são equivalentes, mas atualmente o mercado busca cada vez mais softwares que interagem mais com usuário e que sejam portáteis ao sistema operacional.

Toda essa inovação em recursos visuais exige um custo. Aplicações em console na maior parte das vezes consomem menos memória e desgastam menos o processador, mas são menos vendidas e menos aceitas comercialmente.

O C++ Builder oferece recursos visuais que, se fossem trabalhados no modo texto, exigiriam muito mais linhas de programação, como recursos para entrada de dados através de um arquivo texto e formatação de dados tabulados.

A manutenção do sistema é muito mais fácil e mais limpa. O programa pode ser dividido em diversos módulos e diversas telas, o que possibilita uma rápida depuração dos erros resultando em correções mais rápidas e eficazes. O próprio C++ Builder possui ótimas ferramentas de depuração e auxílio ao desenvolvedor oferecendo uma grande biblioteca para consulta de seus recursos e ferramentas.

Um sistema desenvolvido com essa ferramenta gera vários arquivos para sua compilação: “.bpr” que lista todos os seus elementos e o código de inicialização do aplicativo; “.dfm” com as informações gráficas do aplicativo, cada arquivo desse é associado a uma “unit” (arquivo fonte dos formulários); “.cpp” contendo as ações do aplicativo. Os arquivos “unit” podem estar associadas ou não a formulários; “.h” que contém as informações sobre cada arquivo “.cpp” criado, podendo conter funções, procedimentos, classes, structs. Além desses o compilador utiliza outros arquivos para sua execução como os de extensão “.obj” e “.res”.

Quando é desenvolvida uma aplicação orientada a objetos utilizando o C++ Builder é necessário um cuidado para separar o código fonte das classes relacionadas a interface gráfica da aplicação (classes visuais criadas automaticamente pelo compilador) das classes dos sistema. Para ajudar essa separação e facilitar a manutenção das classes do sistema é aconselhável separar essas classes em arquivos fora da aplicação, nos arquivos “.h”.

CAPITULO 7

OBJETO DE APRENDIZAGEM PARA SISTEMAS DE ENERGIA

Este trabalho tem como objetivo o desenvolvimento de uma ferramenta para apoio a Ensino/Aprendizagem em disciplinas de análise de sistemas de energia elétrica nos cursos de graduação e pós graduação em engenharia elétrica.

O software se divide em dois grandes módulos: Fluxo de Potência e Análise de estabilidade. No módulo de Fluxo de Potência pode-se destacar as seguintes funcionalidades:

1. Entrados de dados através de sistemas criados pelo usuário ou lidos de arquivos pré-formatados segundo normas do IEEE, (1973);
2. Calculo do fluxo de potência pelos métodos de Newton-Raphson e Newton-Desacoplado;
3. Relatório de resultados de fluxo de potência que pode ser visualizado em tela ou impresso.

No módulo de Análise de estabilidade foram implementadas as seguintes funcionalidades:

1. Escolha, pelo usuário, da barra onde acontece um curto-circuito trifásico e da linha de transmissão desconectada para eliminar a falta;
2. Cálculo do fluxo de potência pós-falta;
3. Análise da estabilidade estática tendo como base os ângulos pré e pós falta;
4. Análise da estabilidade transitória implementada pelo método de integração numérica de Runge Kutta de 4ª orderm;
5. Escolha da máquina de referência;
6. Gráficos de resultados para ângulo versus tempo, velocidade versus tempo, diferença angular das máquinas em relação à máquina de referência, tensões das barras;

7. Animação gráfica dos efeitos nos ângulos dos geradores.

7.1 Aplicação de paradigmas de POO em sistemas de energia elétrica

Analisando a estrutura dos SEE, nota-se que a aplicação da POO é benéfica em muitos aspectos. Os SEE têm uma estrutura física bastante adaptável a uma estrutura hierárquica de classes. Nos trabalhos de Rumbaugh, Blaha, Premerlani, et al. (1994) pode ser encontrado que uma operação que tem características próprias deve ser modelada como classe. As formas como os componentes se conectam constituindo a rede elétrica, e também as funcionalidades desses componentes, sugerem o formato geral das estruturas de classes.

Seguindo a idéia proposta por Agostini, Decker e Silva (2002), os elementos do sistema elétrico serão representados de forma independente, oferecendo flexibilidade à modelagem.

Foi adotada a programação orientada a objetos neste trabalho, pois uma de suas vantagens é a possibilidade de aumentar as funcionalidades a aplicação sem grandes alterações no código do sistema. Assim futuramente poderão ser acrescentados a este trabalho novos dispositivos e conceitos de análise de Sistemas de Energia. Os conceitos da POO citados anteriormente resultarão numa análise mais eficiente dos problemas dos Sistemas de Energia.

As motivações para o estudo são contornar as atuais limitações impostas pelos métodos e linguagens tradicionalmente utilizados no desenvolvimento dos softwares na área de SEE, tais como confiabilidade e, principalmente, clareza e objetividade na exibição dos resultados com vistas à utilização em Ensino/Aprendizagem.

Algumas características de POO como encapsulamento, abstração e herança foram utilizadas no desenvolvimento do software e foram citados os seus conceitos no capitulo 6 desse trabalho.

7.2 Descrição das classes utilizadas no módulo de fluxo de potência

Classe Barra

A classe barra modela as barras do sistema elétrico. A partir dessa classe são declarados os objetos do tipo Barra, para execução do fluxo de potência. A

quantidade desses objetos é determinada no método **entrada_dados()** encontrado na classe **fluxo de potência.** Essa classe foi criada utilizando apenas 2 métodos: l**er_barras** e **Pesp_Qesp.** Os atributos dos objetos **barras,** podem ser vistos na Figura 7.

Classe Linha

A classe Linha modela as linhas de transmissão do sistema elétrico. A partir dessa classe são declarados os objetos do tipo linha, para execução do cálculo de fluxo de potência. A quantidade desses objetos é determinada no método **entrada_dados()** encontrado na classe **fluxo de potência.** Essa classe foi criada utilizando 3 métodos: **ler_linhas,** Calc_admitancias, Bshunt_div_2. Os atributos dos objetos **linhas,** podem ser vistos na Figura 7.

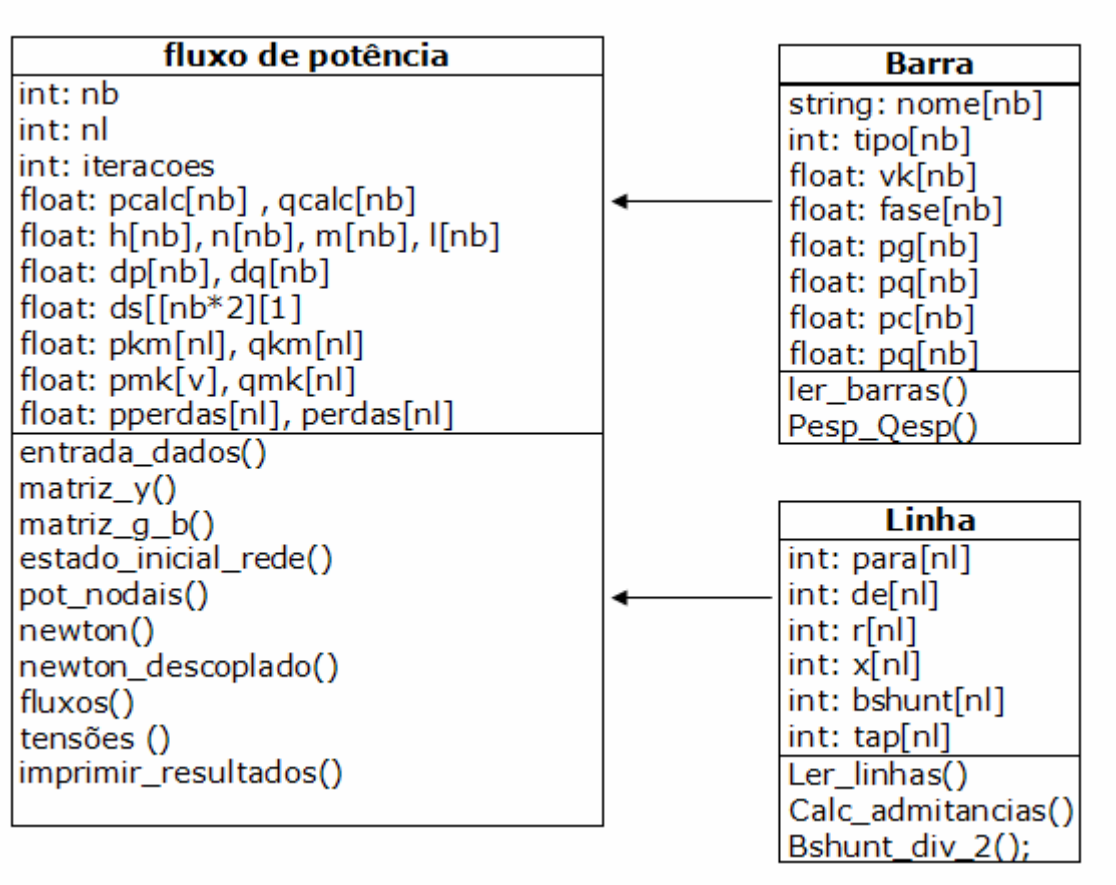


Figura 7: Estruturas de classe do módulo fluxo de potência.

Nesse módulo foram utilizadas funções auxiliares, inversão de matrizes e cálculos básicos com grandezas complexas. Essas funções foram desenvolvidas dentro do software em linguagem C++. Não foram utilizadas bibliotecas prontas

disponíveis na internet em vista do fato de algumas não serem de livre acesso e outras não se adaptarem aos cálculos propostos pelo software.

7.3 Descrição das classes utilizadas no módulo de Análise de estabilidade

Classe Análise de Estabilidade

Essa classe colhe informações importantes para execução do módulo de estabilidade como: tempo de duração da falta, tempo total de simulação, passo de integração, freqüência do sistema, linha de falta e barra de falta. Em seus métodos estão implementados as seguintes funcionalidades: Determinar se o usuário irá usar configurações do período pré ou pós-falta; escolher a barra de referência; montar as matrizes aumentadas e reduzidas para o período pré e pós-falta; criar os gráficos: ângulo versus tempo, velocidade versus tempo, diferença angular das máquinas para máquina de referência, tensões das barras e criar animação gráfica para demonstração do comportamento da diferença angular entre as máquinas.

Classe Estabilidade Estática

Esta classe realiza os cálculos dos fluxos pós-falta, calcula as tensões pós-falta e calcula diferença angular entre os ângulos pré e pós falta. Os atributos da classe estabilidade estática estão descritos na Figura 8.

Classe Estabilidade Transitória

Esta classe contém métodos que determinam se o sistema irá utilizar configurações do período de pré ou de pós-falta, elege a barra de referência, monta as matrizes aumentadas e reduzidas de pré e pós-falta, calcula a potência elétrica e potência mecânica, efetua a integração numérica utilizando o método de Runge-kutta para o período de falta e pós-falta. Os atributos da classe estabilidade transitória estão descritos na Figura 8.

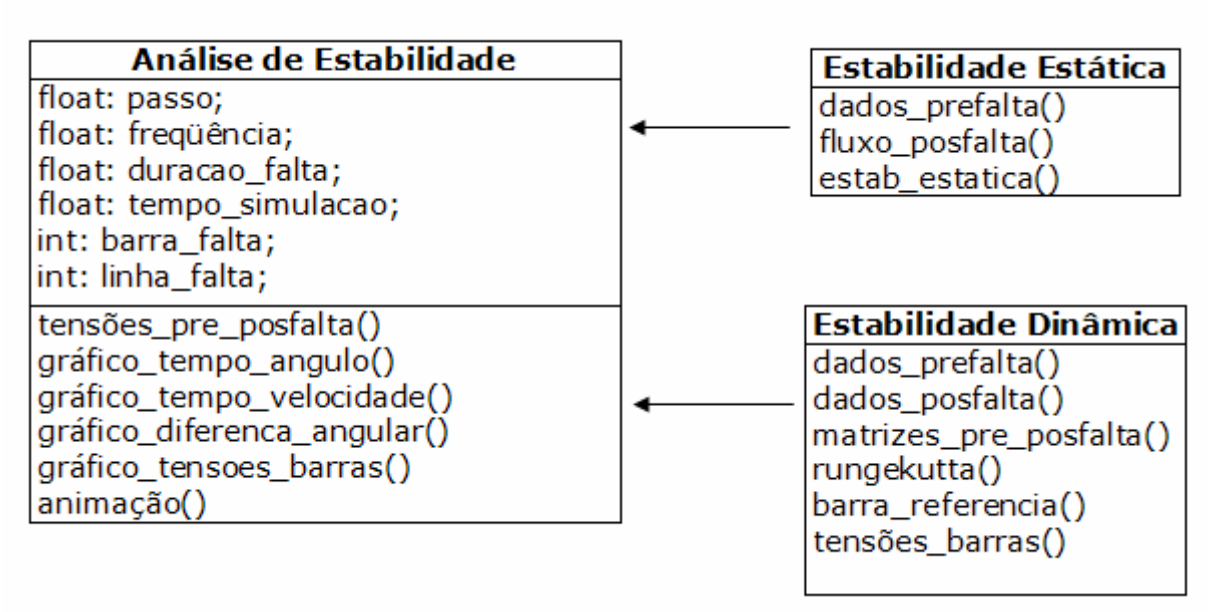


Figura 8: Estruturas de classe do módulo análise de estabilidade.

Nos próximos capítulos serão apresentados conceitos do desenvolvimento do *software* resultado do presente trabalho e também exposto algumas de suas aplicações dentro do contexto da disciplina de análise de sistemas de energia elétrica.

CAPITULO 8

RESOLUÇÃO DO FLUXO DE POTÊNCIA

Para comprovar o efetivo funcionamento dos algoritmos desenvolvidos, foram realizadas simulações com um sistema de nove barras extraído de Anderson e Fouad (1994). O diagrama unifilar do sistema é apresentado na Figura 9. O detalhamento do problema de fluxo de potência foi feito no Capitulo 4 deste trabalho.

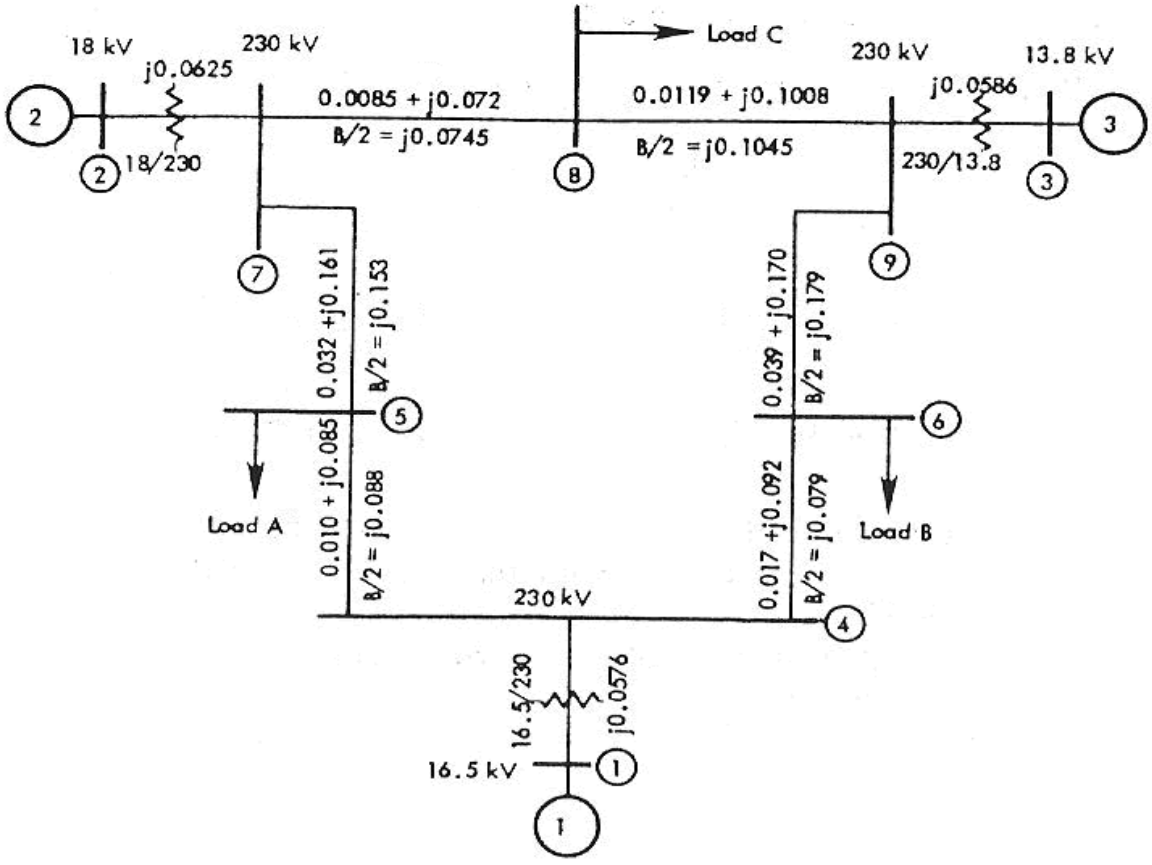
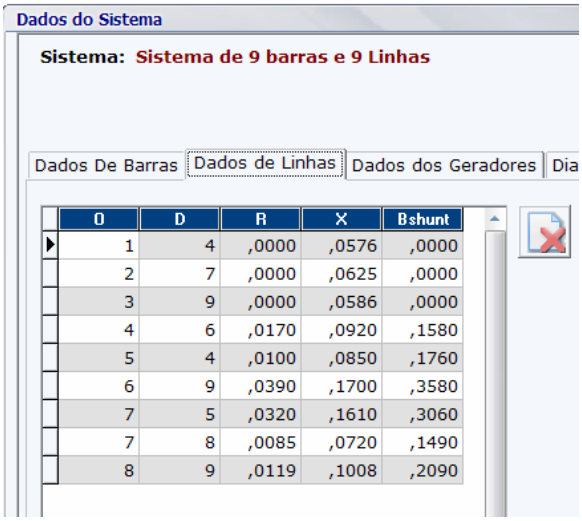


Figura 9: Diagrama do Sistema de 9 Barras e 9 Linhas.

No sistema computacional desenvolvido, os dados dos barramentos são introduzidos de forma tabular, podendo ser lidos através de arquivos textos no formato conhecido do IEEE (1973) ou digitados pelo usuário. Os resultados da leitura dos dados de barras e linhas podem ser vistos na Figuras 10 e 11.

Num	Nome	Tipo	Vnom	Fase	Pg	Qg	Pc	Qc	Shunt
1	BUS-1	3	1,04	,0000	,0000	,0000	,0000	,0000	,0000
2	BUS-2	2	1,025	,0000	1,6300	,0000	,0000	,0000	,0000
3	BUS-3	2	1,025	,0000	,8500	,0000	,0000	,0000	,0000
4	BUS-4	0	1	,0000	,0000	,0000	,0000	,0000	,0000
5	BUS-5	0	1	,0000	,0000	,0000	1,2500	,5000	,0000
6	BUS-6	0	1	,0000	,0000	,0000	,9000	,3000	,0000
7	BUS-7	0	1	,0000	,0000	,0000	,0000	,0000	,0000
8	BUS-8	0	1	,0000	,0000	,0000	1,0000	,3500	,0000
9	BUS-9	0	1	,0000	,0000	,0000	,0000	,0000	,0000

Figura 10: Dados de Barras lidos do arquivo padrão IEEE.



O	D	R	X	Bshunt
1	4	,0000	,0576	,0000
2	7	,0000	,0625	,0000
3	9	,0000	,0586	,0000
4	6	,0170	,0920	,1580
5	4	,0100	,0850	,1760
6	9	,0390	,1700	,3580
7	5	,0320	,1610	,3060
7	8	,0085	,0720	,1490
8	9	,0119	,1008	,2090

Figura 11: Dados de Linhas lidos do arquivo padrão IEEE.

Os dados lidos podem ser alterados conforme a necessidade do usuário. Para execução do módulo de estabilidade devem ser preenchidos pelo menos a constante de inércia (H) e a reatância subtransitória (x'_d) nos dados dos geradores, como visto na Figura 12. Para o teste realizado, foram usados dados de geradores encontrados em Anderson e Fouad (1994).

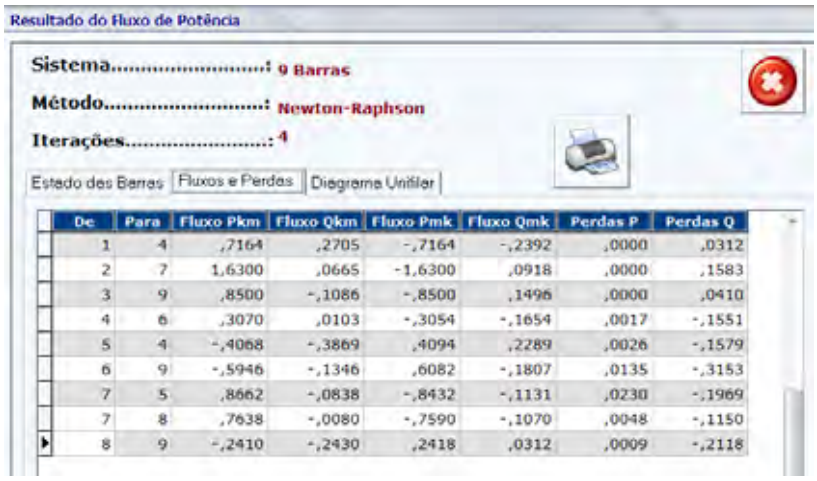
Gerador	Reatância Sub-Transitória X'_d	Constante H (MJ/MVA)
1	0,0608	23,64
2	0,1198	6,40
3	0,1813	3,01

Figura 12: Dados de Geradores para o sistema de 9 barras e 9 linhas.

Considerando o método de Newton-Raphson e uma tolerância de 0,000001 para a convergência e número de iterações máximas igual a 30, os resultados obtidos para o fluxo de potência pelo software desenvolvido podem ser vistos nas Figuras 13 e 14.

Barra	Tipo	V	Ang	Pot. Ativa	Pot. Reativa
1	3	1,0400	,0000	,7164	,2705
2	2	1,0250	9,2800	1,6300	,0665
3	2	1,0250	4,6648	,8500	-,1086
4	0	1,0258	-2,2168	,0000	,0000
5	0	,9956	-3,9888	-1,2500	-,5000
6	0	1,0127	-3,6874	-,9000	-,3000
7	0	1,0258	3,7197	,0000	,0000
8	0	1,0159	,7275	-1,0000	-,3500
9	0	1,0324	1,9667	,0000	,0000

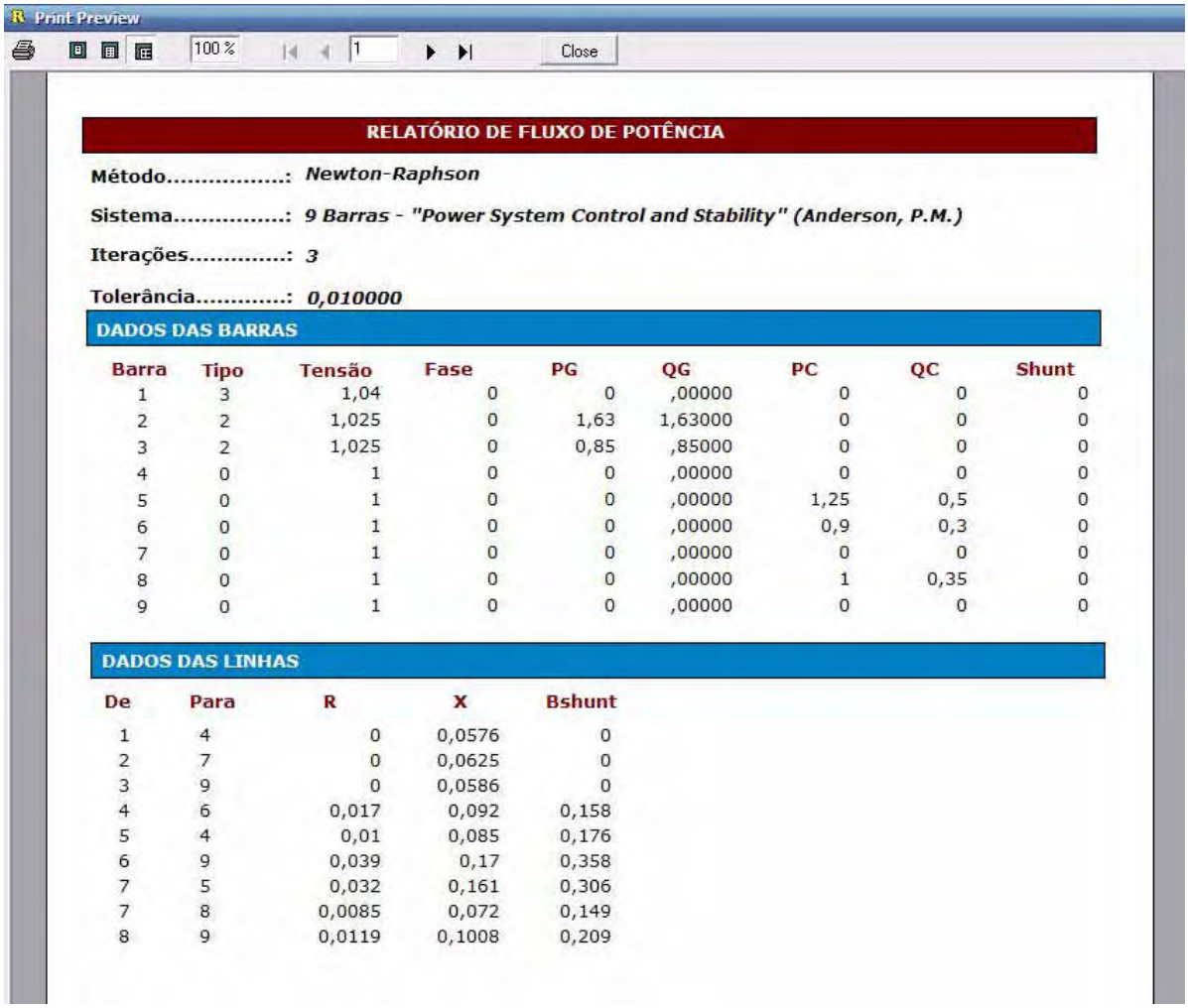
Figura 13: Resultado de Estados das Barras para o método Newton-Raphson.



De	Para	Fluxo Pkm	Fluxo Qkm	Fluxo Pmk	Fluxo Qmk	Perdas P	Perdas Q
1	4	,7164	,2705	-,7164	-,2392	,0000	,0312
2	7	1,6300	,0665	-1,6300	,0918	,0000	,1583
3	9	,8500	-,1086	-,8500	,1496	,0000	,0410
4	6	,3070	,0103	-,3054	-,1654	,0017	-,1551
5	4	-,4068	-,3869	,4094	,2289	,0026	-,1579
6	9	-,5946	-,1346	,6082	-,1807	,0135	-,3153
7	5	,8662	-,0838	-,8432	-,1131	,0230	-,1969
7	8	,7638	-,0080	-,7590	-,1070	,0048	-,1150
8	9	-,2410	-,2430	,2418	,0312	,0009	-,2118

Figura 14: Resultado de fluxo e perdas nas linhas para o método Newton-Raphson.

O usuário pode conferir os resultados do fluxo de potência e também os dados de entrada consultando o relatório que será gerado ao clicar no botão impressora na tela de resultados de fluxo de potência. O relatório de fluxo de potência gerado pelo sistema pode ser visto na Figura 15.



RELATÓRIO DE FLUXO DE POTÊNCIA

Método.....: *Newton-Raphson*

Sistema.....: *9 Barras - "Power System Control and Stability" (Anderson, P.M.)*

Iterações.....: *3*

Tolerância.....: *0,010000*

DADOS DAS BARRAS

Barra	Tipo	Tensão	Fase	PG	QG	PC	QC	Shunt
1	3	1,04	0	0	,00000	0	0	0
2	2	1,025	0	1,63	1,63000	0	0	0
3	2	1,025	0	0,85	,85000	0	0	0
4	0	1	0	0	,00000	0	0	0
5	0	1	0	0	,00000	1,25	0,5	0
6	0	1	0	0	,00000	0,9	0,3	0
7	0	1	0	0	,00000	0	0	0
8	0	1	0	0	,00000	1	0,35	0
9	0	1	0	0	,00000	0	0	0

DADOS DAS LINHAS

De	Para	R	X	Bshunt
1	4	0	0,0576	0
2	7	0	0,0625	0
3	9	0	0,0586	0
4	6	0,017	0,092	0,158
5	4	0,01	0,085	0,176
6	9	0,039	0,17	0,358
7	5	0,032	0,161	0,306
7	8	0,0085	0,072	0,149
8	9	0,0119	0,1008	0,209

Figura 15: Relatório gerado pelo software para o módulo fluxo de potência.

CAPITULO 9

MÓDULO DE ANÁLISE DE ESTABILIDADE

Análise de Estabilidade Estática para o caso de 9 barras e 9 linhas

Os conceitos de estabilidade discutidos no Capítulo 5 foram aplicados no módulo de análise de estabilidade do software desenvolvido. Considerando os resultados obtidos pela execução do fluxo de potência já apresentados nas Figuras 13 e 14, pode-se continuar didaticamente a análise iniciada no módulo de fluxo de potência executando o módulo de estabilidade.

Figura 16: Tela principal do módulo de Estudo de Estabilidade.

Na tela principal do módulo de estabilidade, mostrada na Figura 16, o usuário pode visualizar os dados de linhas lidos inicialmente no módulo de fluxo de potência. O usuário pode selecionar a barra que sofrerá o defeito e a linha que será excluída na simulação de falta.

Na simulação de Estabilidade Estática é possível executar o fluxo de potência pós-falta onde o usuário pode observar se o método convergiu ou não

convergiu. Se o fluxo de potência para o regime pós-falta convergir, significa que após a eliminação da falta (retirada da linha), o sistema tem um ponto de equilíbrio. É necessário observar se as defasagens angulares pós-falta entre os geradores não superam 90º para se concluir que o ponto de equilíbrio do sistema é estaticamente estável. Se não convergir, nada se afirma, uma vez que pode significar apenas que o sistema de equações algébricas é mal condicionado, não necessariamente que o sistema elétrico não tenha ponto de equilíbrio estável.

Assim, uma vez executado o fluxo de potência pós-falta, pode-se concluir que as máquinas do sistema não ultrapassam a defasagem limite de 90º. Os valores das tensões internas pré-falta e pós-falta dos geradores podem ser vistos na Figura 17.

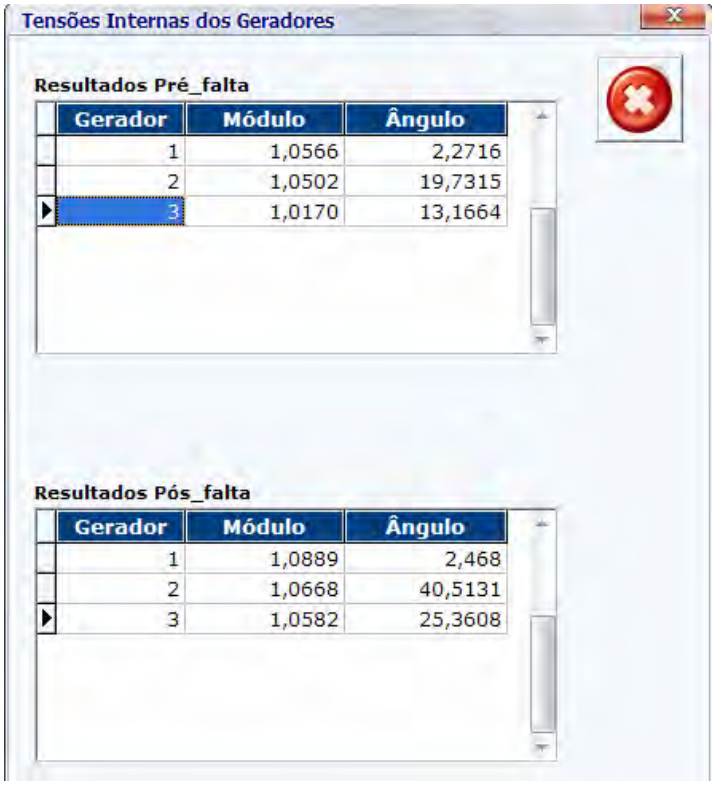


Figura 17: Tensões internas dos Geradores.

Análise de Estabilidade Transitória para o caso de 9 barras e 9 linhas

Situação Inicial

Para o estudo da estabilidade transitória, são necessários os valores das reatâncias sub-transitórias e das constantes de inércia (neste caso representadas

pelas constantes H e x'_d) de cada máquina. Os dados foram lidos no módulo de fluxo de potência e estão apresentados na Figura 12 e Figura 18.

Figura 18: Dados necessários para a análise de estabilidade transitória.

Para as simulações seguintes, será considerado um curto-circuito trifásico próximo ao barramento 7. Tal defeito será eliminado através da saída da linha que interliga os barramentos 7 e 5. O método de integração utilizado será o Runge-Kutta de 4ª ordem, o passo de integração será 0,001 e o tempo de abertura da linha seguido exemplo de Anderson e Fouad (1994) será de 0,0833 s e o tempo total de simulação será de 2 s, como mostrado na Figura 19. Esses dados podem ser alterados pelo usuário conforme a necessidade da simulação.

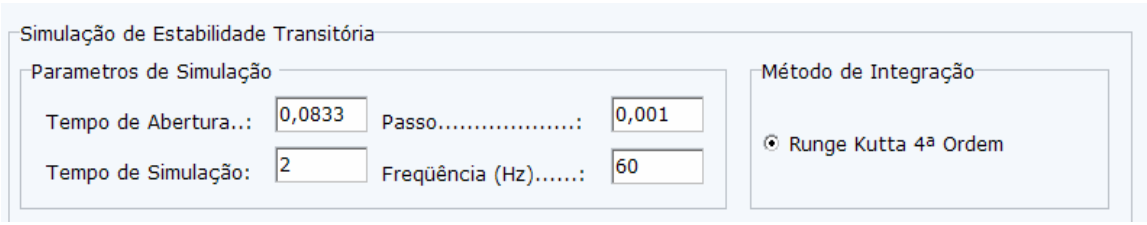


Figura 19: Dados iniciais para execução da simulação de estabilidade transitória.

Após configurar os parâmetros de simulação o usuário deverá clicar no botão “Simulação” onde há um método matemático para definição da barra de referência, cujo critério adotado foi à máquina menos acelerada no instante inicial da falta, na suposição de que é menos afetada pela contingência e portanto menos afetada pela falta, como mostrado na Figura 20.

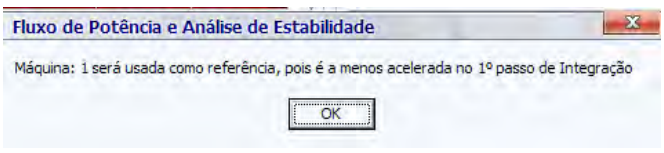


Figura 20: Escolha da barra de referência.

Em poucos segundos o sistema avisa o usuário que a simulação foi completada. Assim pode-se conferir no botão “Relatório” os resultados de simulação de ângulo e velocidade no domínio do tempo resultantes da execução do Runge Kutta 4ª ordem. Nos botões velocidade e ângulo o usuário pode comprovar o comportamento dos geradores para a situação de falta descrita. Os gráficos são representados nas Figuras 21 e 22.

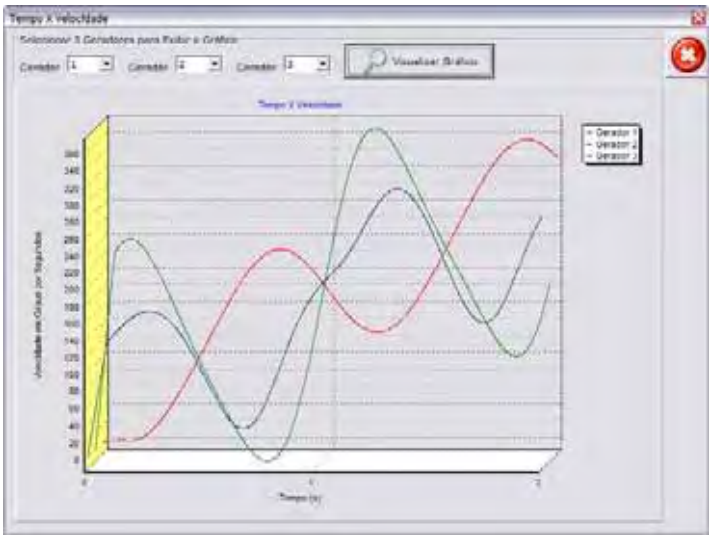


Figura 21: Curva velocidade versus tempo para t = 0,0833 s.

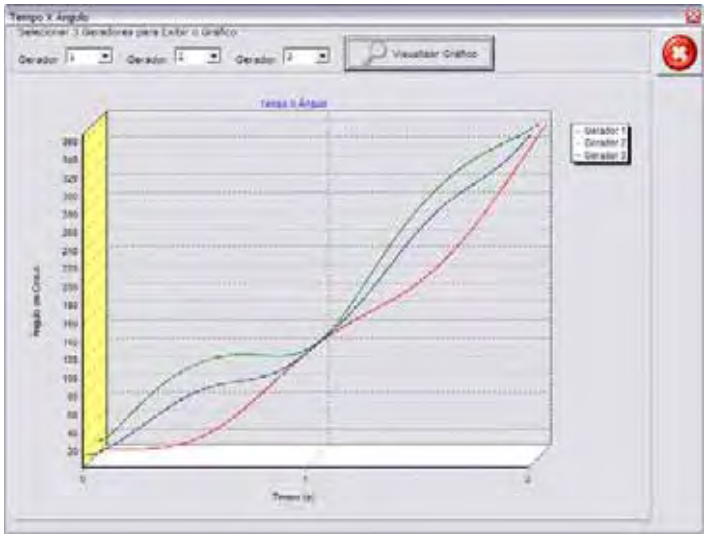


Figura 22: Curva ângulos versus tempo para t = 0,0833 s.

As Figuras 20 e 21 indicam que o sistema é estável para o tempo de abertura da linha com t= 0,0833 s. Para completar a análise, pode-se visualizar o gráfico de defasagens angulares e a animação da dinâmica dos rotores, conforme Figura 23.

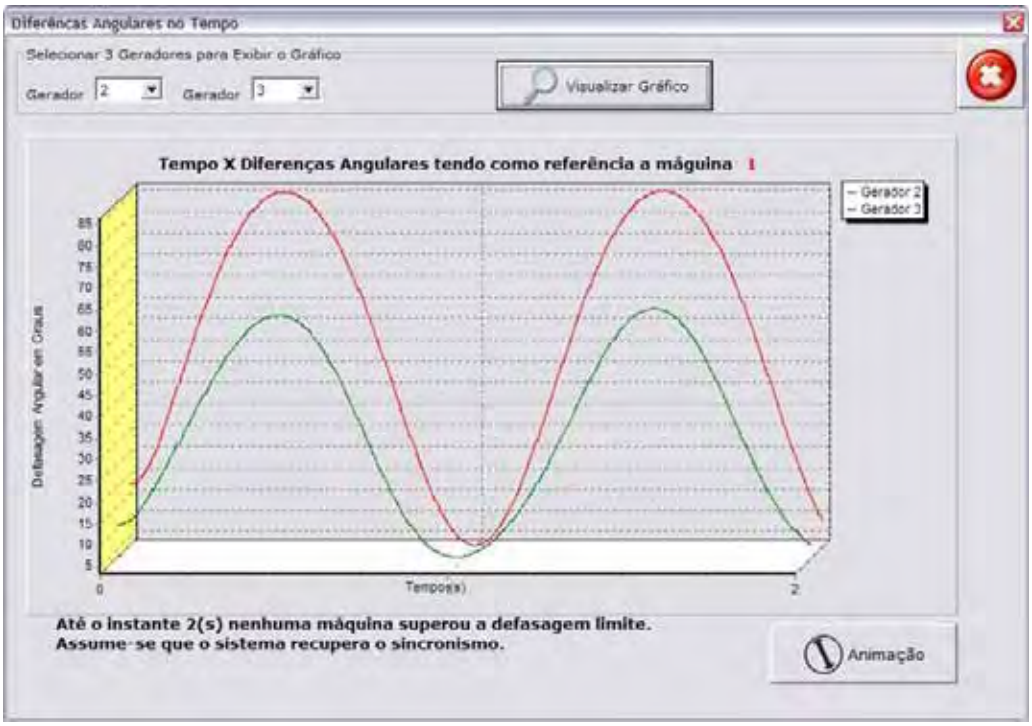


Figura 23: Defasagens angulares para t = 0,0833 s.

A análise das defasagens angulares é feita considerando, empiricamente, como ângulo limite 180^{o} entre as máquinas em estudo e uma máquina de referência. Na simulação em tela, o aplicativo considerou a máquina 1 como referência, ou seja, o gráfico da Figura 23 representa as defasagens angulares entre os geradores 2 e 3 em relação ao gerador 1. A nota “Assume-se que o sistema recupera o sincronismo” é adicionada após constatar que no intervalo considerando não ocorreu violação do critério de máximo ângulo utilizado

Ainda pela Figura 23, podem-se notar que a máquina 2 é mais sensível ao defeito ocorrido no barramento 7. Esta maior sensibilidade era esperada já que tal gerador é o que se encontra mais próximo ao defeito. A maior defasagem angular entre o gerador 2 e o gerador 1 ocorre aproximadamente em 0,43 s.

Uma ilustração bastante didática do transitório é apresentada quando se utiliza a animação que representa o movimento dos rotores dos geradores em estudo, como apresentado na Figura 24.

No instante de tempo igual a 0,23 s o rotor do gerador 2 (representado pela linha verde) e o rotor do gerador 3 (linha azul) estão se movendo mais rápido que o rotor do gerador 1. Os ângulos relativos entre os geradores 2 e 3 e o gerador 1 estão aumentando (a seta indica o sentido do movimento), como mostrado na Figura 24.

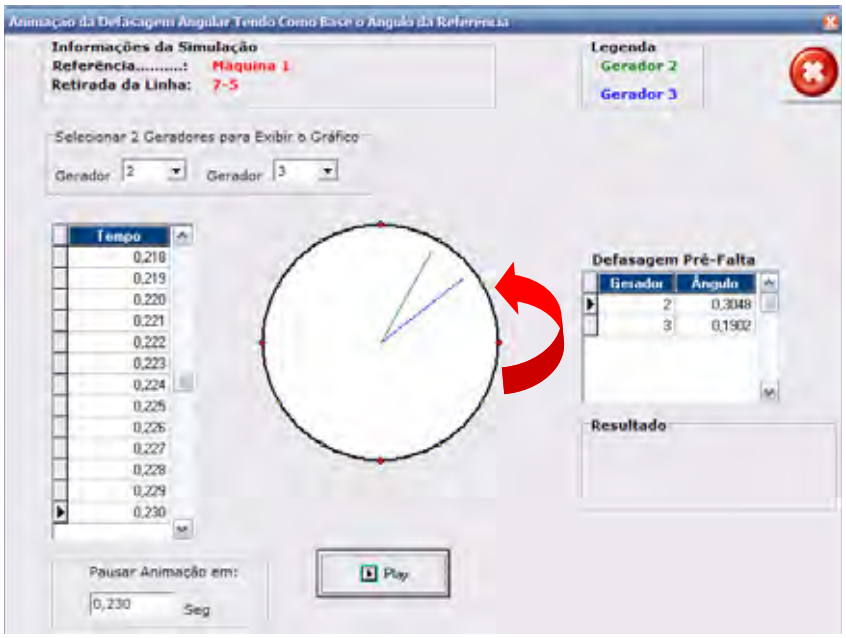


Figura 24: Animação da posição angular dos rotores considerando t = 0,23 s.

No instante de tempo igual a 0,43 s a defasagem entre os geradores é máxima aproximadamente, 85º. A partir deste ponto as defasagens angulares começam a reduzir, como mostrado na Figura 25.

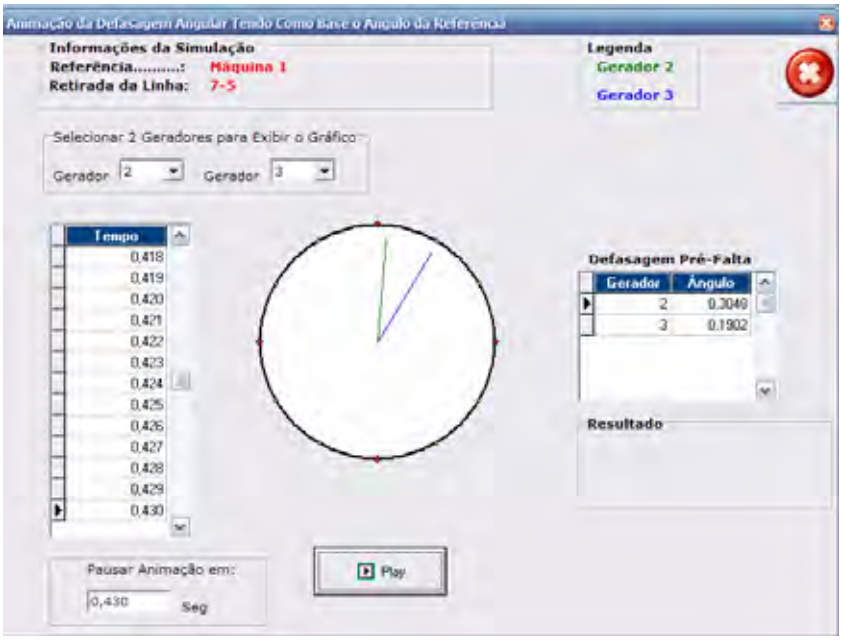


Figura 25: Animação da posição angular dos rotores considerando t = 0,43 s.

No instante de tempo igual a 0,85 s, as defasagens angulares estão próximas a zero graus. As retas que representam tais defasagens estão se movimentando no sentido horário (indicado pela seta vermelha). Para o todo o intervalo de tempo considerado (2 s), o movimento das retas se mantêm entre 0º e 85º, como mostrado na Figura 26.

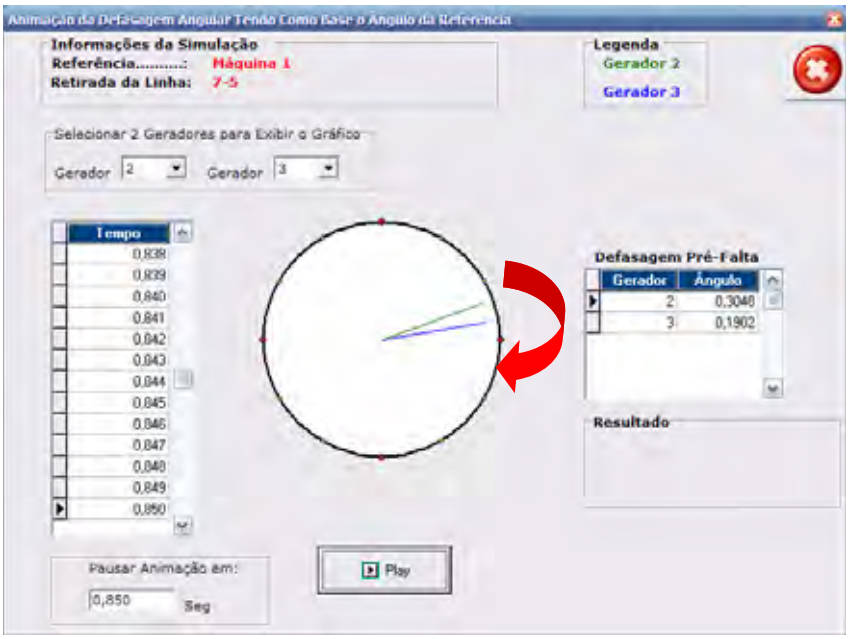


Figura 26: Animação da posição angular dos rotores considerando t = 0,85 segundos

Abertura tardia dos disjuntores

Será analisado a seguir o efeito da abertura tardia dos disjuntores que isolam a linha entre os barramentos 5 e 7 do restante do sistema. Para tanto, utiliza-se um tempo de abertura de 0,16667 s, como mostrado na Figura 27.

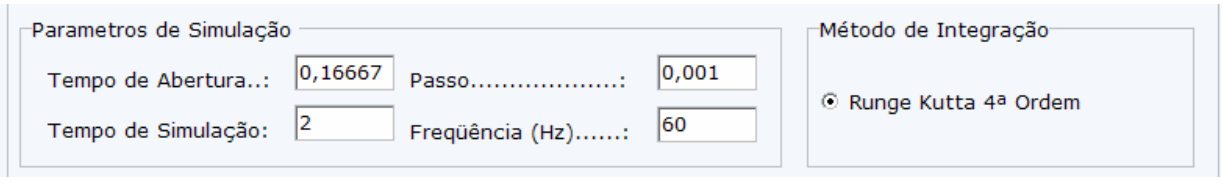


Figura 27: Dados para abertura tardia dos disjuntores.

Para esta situação, o comportamento dos geradores poderá ser descrito pelos gráficos obtidos com o sistema computacional objeto de aprendizagem desenvolvido, mostrados nas Figuras 28 e 29

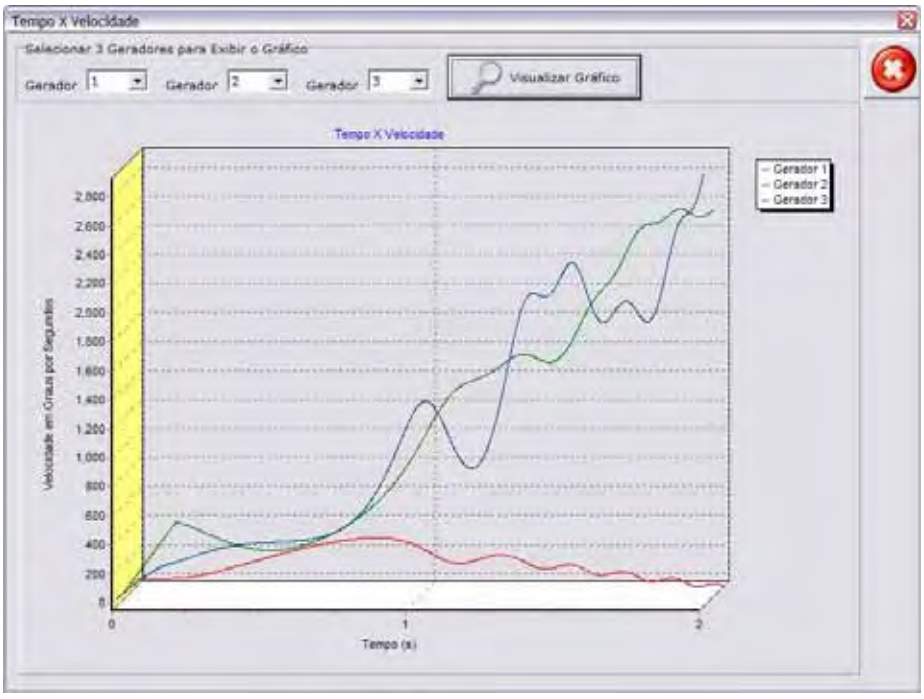


Figura 28 – Curva velocidade versus tempo para t = 0,1667 s.

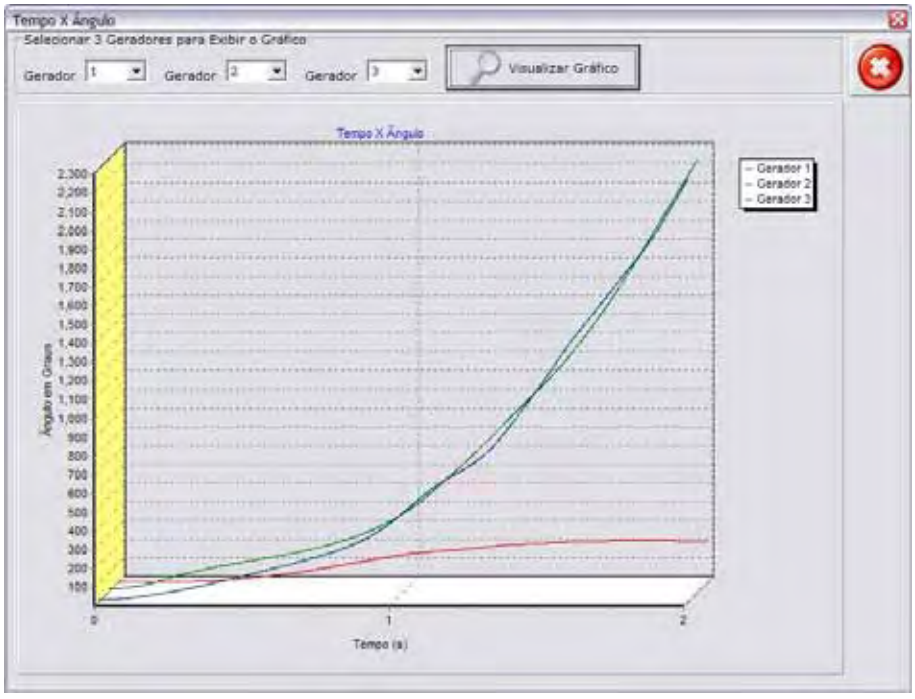


Figura 29: Curva ângulos versus tempo para t = 0,1667 s.

As Figuras 28 e 29 indicam que o sistema é instável para o tempo de abertura de 0,1667 s. O diagrama das defasagens angulares e a animação da dinâmica dos rotores permitem a mesma conclusão.

Figura 30: Defasagens angulares para t= 0,1667 s.

Pela Figura 30 pode-se verificar que as defasagens angulares entre os geradores 2 e 3 e o gerador 1 aumentam indefinidamente, indicando claramente uma situação de perda de sincronismo entre as máquinas do sistema.

Gerando uma nova ilustração do movimento dos rotores dos geradores em estudo pode-se analisar seu sincronismo de acordo com a variação no tempo. No instante de tempo igual a 0,5 s os rotores dos geradores 2 e 3 (representados pelas linhas verde e azul, respectivamente) estão se movendo muito mais rapidamente que o rotor do gerador 1. Para ambos os geradores a defasagem angular é superior a 90º, como mostrado na Figura 31.

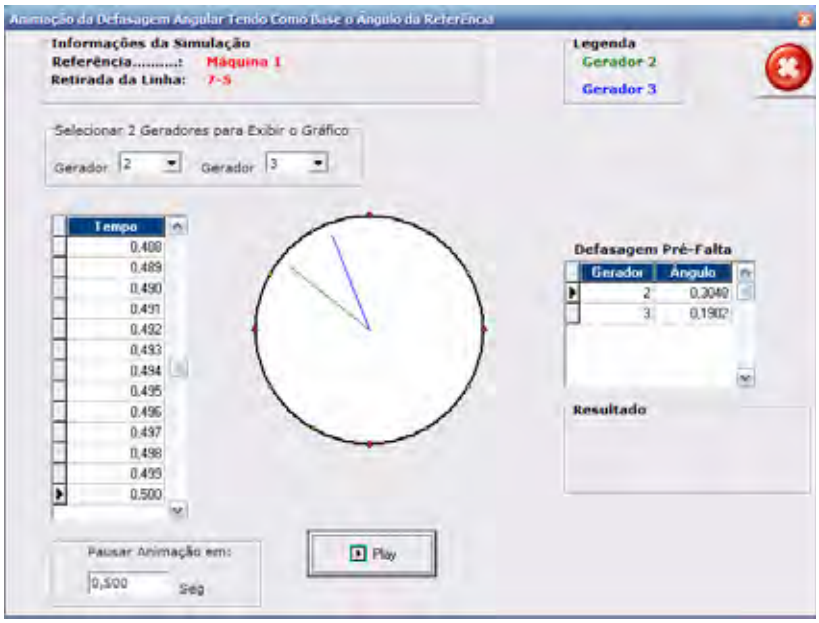


Figura 31: Animação da posição angular dos rotores considerando t = 0,5 s.

No instante de tempo igual a 0,93 s a defasagem entre os geradores 2 e 3 é quase nula, conforme a Figura 32. Esta situação se repetirá em outros instantes do intervalo de tempo considerado.

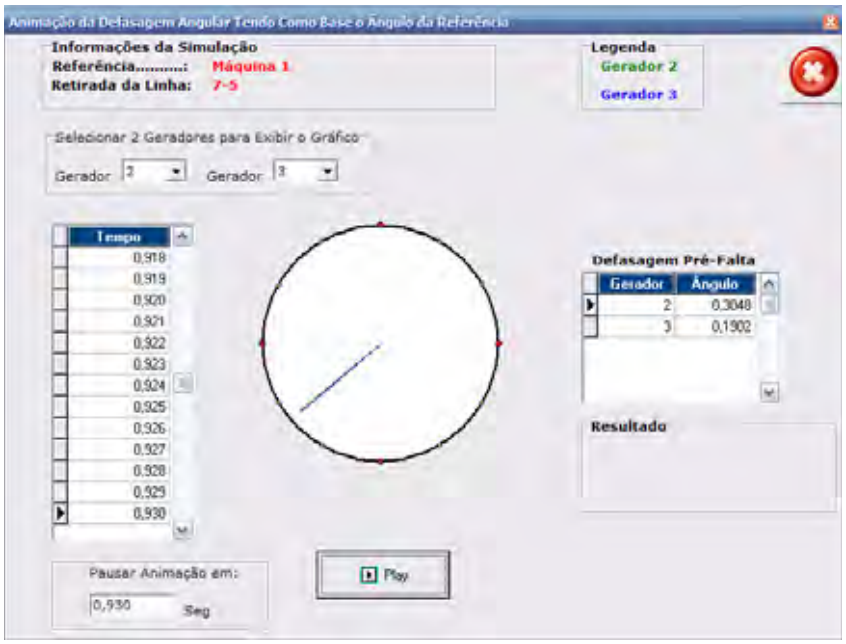


Figura 32: Animação da posição angular dos rotores considerando t = 0,93 s.

Em t = 1,0 s pode ser verificado que a defasagem do gerador 3 é superior à defasagem do gerador 2. A defasagem cresce indefinidamente com o tempo. Na animação este fato é mostrado pelo aumento da velocidade de rotação das linhas que representam os rotores das máquinas, a posição final dos ângulos para pausa em 1 s é apresentada na Figura 33.

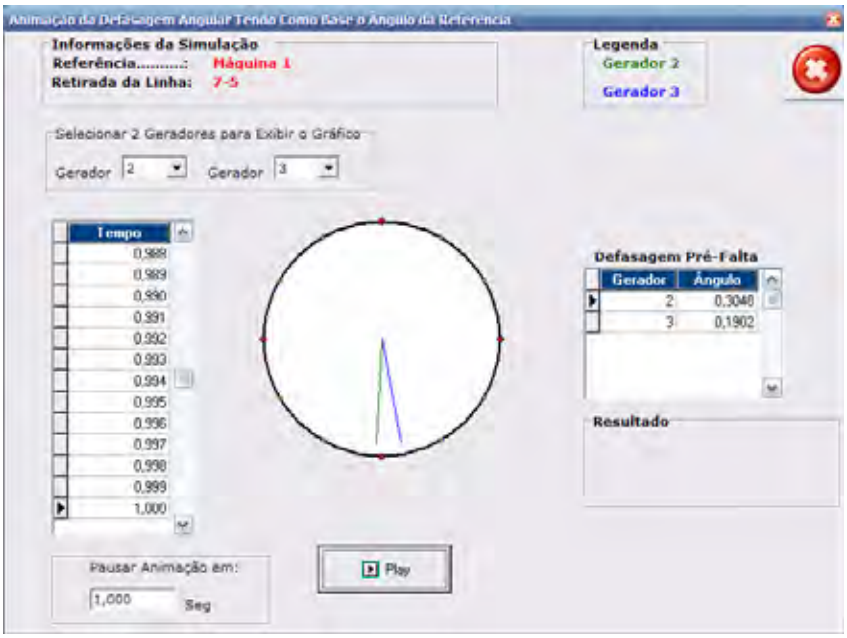


Figura 33: Animação da posição angular dos rotores considerando t = 1,0 s.

Os resultados obtidos através da simulação deste sistema com o objeto de aprendizagem desenvolvido puderam ser comprovados pela comparação direta. Além disso, durante a fase de implementação e testes, os resultados intermediários (tensões internas, matrizes admitância, matrizes reduzidas, etc.) puderam ser verificados com base na literatura. Todos os resultados obtidos atestam a qualidade das rotinas desenvolvidas.

CONCLUSÕES E SUGESTÕES PARA FUTUROS TRABALHOS

Neste trabalho foi desenvolvido um sistema computacional que permitirá aos usuários representar sistemas elétricos de potência, executar fluxos de potência e simular o comportamento transitório dos geradores síncronos quando o sistema é sujeito a defeitos do tipo curto-circuito. Todos os resultados obtidos são exibidos na forma tabular ou gráfica de maneira a permitir uma rápida e intuitiva análise do comportamento do sistema. Todos os resultados das rotinas implementadas foram confrontados com os resultados obtidos na literatura e não foram encontrados desvios significativos. Todo o sistema foi desenvolvido utilizando os conceitos de Programação Orientada a Objetos.

O objeto de aprendizagem desenvolvido oferece facilidade na visualização dos resultados (relatórios, tabelas ou gráficos). Um usuário iniciante necessitará de poucos minutos para se familiarizar com o aplicativo e, como o sistema permite que praticamente todos os parâmetros sejam alterados, conseguirá realizar uma infinidade de simulações. Esta flexibilidade para alteração dos dados de entrada aliada à simplicidade de uso permitirá que estudantes simulem diversas situações em pouco tempo e, além disso, o recurso de animação que permite, de forma lúdica e didática, que o usuário acompanhe o movimento das defasagens angulares entre os rotores das máquinas do sistema.

Para desenvolvimento do *software* foram implementadas diversos algoritmos para resolução de equações com números complexos e matrizes seguindo os padrões de POO. O ambiente C++ Builder diferente de outros como Matlab ou Fortran não possui funções pré-definidas para cálculos com números complexos e matrizes. Assim foram implementados, por exemplo, algoritmos para inversão de matrizes reais e complexas, calculo de determinantes e diversas equações básicas para números complexos como: somar, subtrair, multiplicar, dividir, ângulo de complexo, arco-seno e arco-tangente de complexo, conjugado, inverso, etc.

Para trabalhos futuros, o software poderá incluir novos dispositivos relacionados a sistemas de energia elétrica como reguladores de velocidade e tensão, dispositivos FACTS. Poderá ser incluída também a entrada de dados através do

desenho do diagrama unifilar, assim o aluno poderá construir o diagrama e configurar conforme a sua necessidade.

Para deixar o software totalmente portável aos sistemas operacionais poderá ser desenvolvida uma versão para Linux assim o usuário não ficará preso ao ambiente do Microsoft Windows, ou ainda desenvolver uma versão que rode em plataforma *web* em linguagem Java ou PHP.

Em versões futuras o software poderá atender além da disciplina de análise de estabilidade transitória no curso de engenharia elétrica, outras disciplinas da grade curricular do curso, como a disciplina Estabilidade Dinâmica de Sistemas de Energia Elétrica Multimáquinas e Estabilidade Dinâmica de Sistemas de Energia Elétrica: Abordagem Clássica onde poderão ser desenvolvidas formas de aproximar o aluno ao fenômeno estudado através de diversas simulações e animações.

REFERÊNCIAS

ANDERSON, P. M. ;FOUAD, A.A. **Power system control and stability**. New York: IEEE Press, 1994. (IEEE Power Systems Engineering Series).

AGOSTINI, M. N.; DECKER, I. C.; SILVA, A. S. Nova filosofia para o projeto de softwares na área de sistemas de energia elétrica utilizando modelagem orientada a objetos, In: CONGRESSO BRASILEIRO DE AUTOMÁTICA – CBA. **Anais...** Natal: CBA, 2002. p. 1555-1561.

CLOE, “Co-operative Learning Object Exchange”. Disponível em: **<**http://tlc.uwaterloo.ca/projects/CCCO/cloe_stories.html**>.**
Acesso em: 03 nov. 2007

COLVARA, L. D. (Comp.). **Estabilidade transitória de sistemas de energia elétrica.** Ilha Solteira: UNESP/FEIS/Departamento de Engenharia Elétrica, 2005. (Disciplina do curso de pós-graduação em engenharia elétrica, UNESP, 2005. Notas de aula do curso)

IEEE. Common format for exchange of solved load flow data. **IEEE Transactions Power Apparatus and Systems**, New York, v. 92, n. 6, 1973, p. 1916-1925.

IEEE. Institute of Electrical and Electonics Engineers. Apresenta textos sobre objetos de aprendizagem. Disponível em: <http://ltsc.ieee.org/wg12/index.html>. Acesso em: Acesso em: 30 out. 2007.

ELGERD, O. I. **Introdução a teoria de sistemas de energia elétrica**. São Paulo: McGraw-Hill, 1978.

KUNDUR, P. **Power system stability and control**. New York: Electric Power Research Institute, 1994.

LABVIRT, “Laboratório Didático Virtual”. Disponível em:
<http://www.labvirt.futuro.usp.br/> Acesso em: 03 nov.2007

MILANO, F., VANFRETTI,L., MORATAYA, J. C. An open source power system virtual laboratory: the psat case and experience. **IEEE Transactions on Education,** v.51, n.1, p.17-23, 2008. (Acertar a data no texto para 2008)

MONTICELLI, A. J. **Fluxo de carga em redes de energia elétrica**. São Paulo: Edgard Blücher, 1983.

MONTICELLI, A. J.; GARCIA, A. **Introdução a sistemas de energia elétrica.** Campinas: Ed; Unicamp, 2000. v.1.

“RIVED - Rede Internacional Virtual de Educação. Ministério da Educação, Governo Federal, Brasil”. Disponível em: **<http://www.rived.mec.gov.br/ >**. Acesso em: 30 out. 2007.

RUMBAUGH, J.; BLAHA, M.; PREMERLANI, W. et al. **Modelagem e projetos baseados em objetos**. Rio de Janeiro : Campus, 1994

SÁ FILHO, C. S.; MACHADO, E. C. *O* computador como agente transformador da educação e o papel do objeto de aprendizagem. Disponível em: http://www.universia.com.br/materia/materia.jsp?materia=5939**.** 2004. Acesso em: 26 out. 2007

SCHLÜNZEN, E.T.M. **Mudanças nas práticas pedagógicas do professor: criando um ambiente contrucionista contextutalizado e significativo para crianças com necessidades especiais físicas**. 2000. Tese (Doutorado) – Pontifica Universidade Católica, São Paulo, 2000.

STAGG, G. W.; EL-ABIAD, A. H. **Computer methods in power system analysis.** New York: McGraw-Hill Books, 1968.

STEVENSON JÚNIOR., W.D. **Elementos de análise de sistemas de potência.** 2.ed. São Paulo: McGraw-Hill, 1986.

VALENTE, J. A. Por quê o computador na educação? In: VALENTE, J. A.(Org.). **Computadores e conhecimento**: repensando a educação. Campinas: NIED, 1993. p. 24-44.

VALENTE, J. A. A espiral da aprendizagem e as tecnologias da informação e comunicação: repensando conceitos. In: JOLY, M. C. R. A. **Tecnologia no ensino**: implicações para a aprendizagem. São Paulo: Casa do Psicólogo, 2002.

VIEIRA JUNIOR, N.; COLVARA, L. D. Tecnologia motivacional: aplicação de um software educacional para sistemas elétricos de potência. In: In: TOZZI, M. et al. **Novos paradigmas na educação em engenharia.** Curitiba: ABENGE, 2007

VIEIRA JUNIOR, N.; COLVARA, L. D. **A prática docente e novos recursos de ensino para a estabilidade de sistemas de energia elétrica**. Curitiba: COBENGE, 2007.

WILEY, D. The instructional use of learning objetcts. *On-line version*. Disponível em: <http://www.reusability.org/read/>. 2000. Acesso em: 26 out. 2007.

TRABALHOS PUBLICADOS

PEREIRA, M. B. C., COLVARA, L.D. Desenvolvimento de um objeto de aprendizagem para estudo de análise de sistemas de energia elétrica In: Conferência Internacional de Educação em Engenharia e Tecnologia - INTERTECH 2008, 2008, Santos.

ANEXO 1

Ajuda do Sistema Desenvolvido

Neste trabalho desenvolveu-se um software seguindo os paradigmas de programação orientada a objetos, onde foram modelados os principais elementos de sistemas de potência, objetivando-se a resolução de fluxo de potência e a análise de estabilidade estática e transitória de sistemas multimáquinas. O programa traz incorporados os dados correspondentes a diversos sistemas-exemplo da literatura especializada, e permite que o usuário modifique parâmetros e até mesmo acrescente dados de outros sistemas de uma maneira bastante amigável. Com isto, as possibilidades de criação e análise de casos ficam praticamente ilimitadas, possibilitando estudos acurados do desempenho de Sistemas de Energia Elétrica.

O Sistema possui módulos de fluxo de potência com resoluções pelo método de Newton-Raphson e Newton Desacoplado, módulo de estabilidade com análise da estabilidade estática e estabilidade transitória. Oferece como recursos de análise relatórios, gráficos e animação gráfica para facilitar a compreensão dos resultados obtidos pelo software.

1. Entrada de Dados no Software

1.1 Utilizando Sistemas Carregados

A. Ao abrir o Sistema o primeiro passo é clicar no botão "Fluxo de Potência"

Figura 1 - Tela Inicial

O Software traz carregado os dados de um sistema de 9 barras, 9 linhas e 3 geradores, extraído de "Power System Control and Stability" (ANDERSON e FOUAD, 1994) . Caso o usuário desejar ler um sistema diferente que esteja no

formato "Common Format for Exchange of Solved Load Flow Data" do IEEE, na pasta do sistema existem 3 arquivos nesse formato para casos de 9, 14 e 30 barras.

Se o usuário possuir os dados de máquina dos sistemas de 14 e 30 barras poderá utilizar o software com todas as simulações no módulo de estabilidade, senão apenas poderá executar o módulo fluxo de potência. Os passos para se introduzir novo sistema a partir de arquivos ou digitados pelo usuário estão na seção 1.2 dessa Ajuda

Para abrimos o sistema de 9 barras já cadastrado e visualizar seus dados deve-se:

A. Clicar no "nome do sistema" no grid de entrada

B. Clicar no botão "Abrir Selecionado".

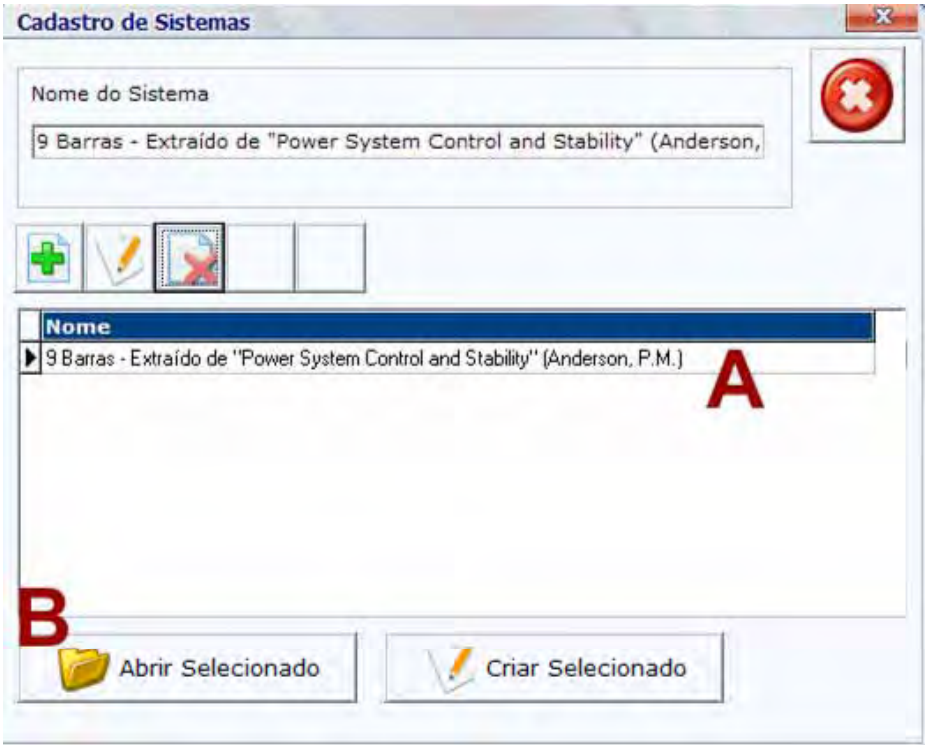
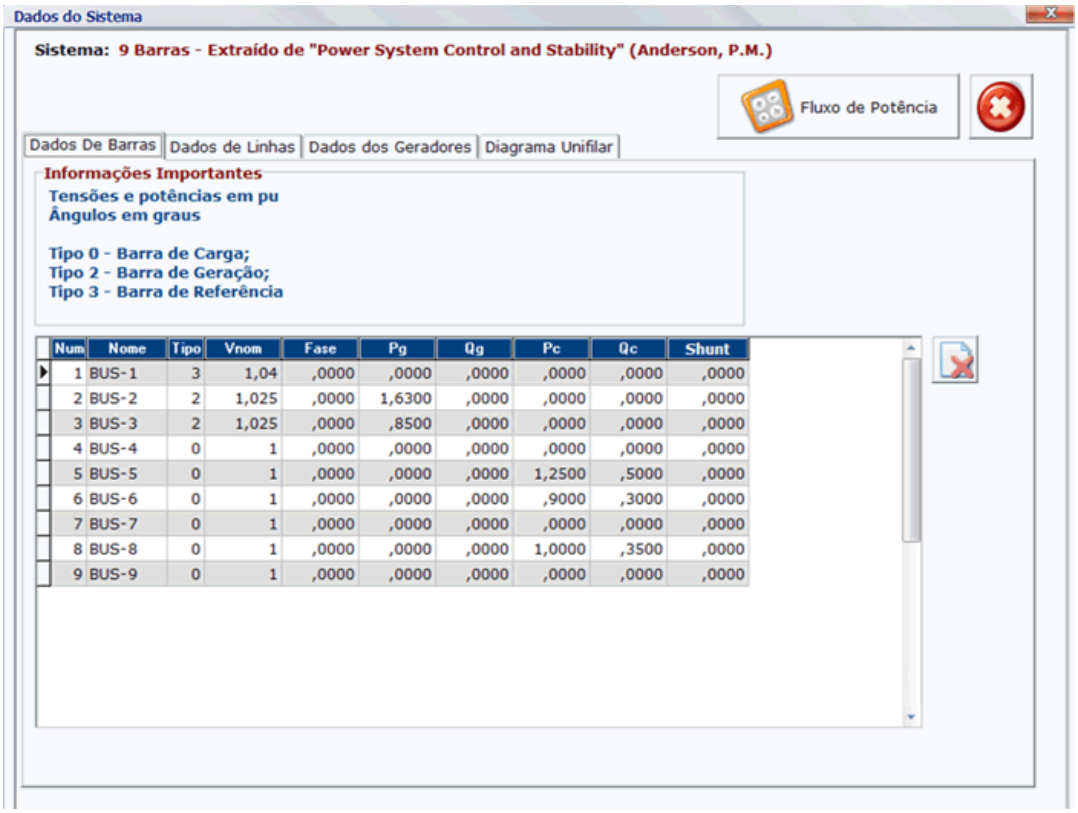


Figura 2 - Escolha do Sistema

Após a seleção do sistema cadastrado, o software abre as telas de resumo dos dados lidos. Em todas as telas é possível que o usuário altere, exclua ou insira novos dados. Pode-se ainda incluir uma imagem no formato JPG do diagrama unifilar do sistema. Esta imagem deve ficar salva na pasta do sistema. Abaixo temos o exemplo dos dados de barras lidos.

Dados de Barras



Num	Nome	Tipo	Vnom	Fase	Pg	Qg	Pc	Qc	Shunt
1	BUS-1	3	1,04	,0000	,0000	,0000	,0000	,0000	,0000
2	BUS-2	2	1,025	,0000	1,6300	,0000	,0000	,0000	,0000
3	BUS-3	2	1,025	,0000	,8500	,0000	,0000	,0000	,0000
4	BUS-4	0	1	,0000	,0000	,0000	,0000	,0000	,0000
5	BUS-5	0	1	,0000	,0000	,0000	1,2500	,5000	,0000
6	BUS-6	0	1	,0000	,0000	,0000	,9000	,3000	,0000
7	BUS-7	0	1	,0000	,0000	,0000	,0000	,0000	,0000
8	BUS-8	0	1	,0000	,0000	,0000	1,0000	,3500	,0000
9	BUS-9	0	1	,0000	,0000	,0000	,0000	,0000	,0000

Figura 3 - Dados de barra

1.2 Carregando Sistemas Através de Arquivos do IEEE

Para fazer leitura de arquivos do IEEE o usuário deve:

C. Clicar no Botão "Inserir" para Cadastrar um novo Sistema
D. Digitar o nome do novo Sistema
E. Clicar no botão "Confirmar" para finalizar o cadastro
F. Clicar no nome do sistema criado
G. Clicar no botão "Criar Selecionado"

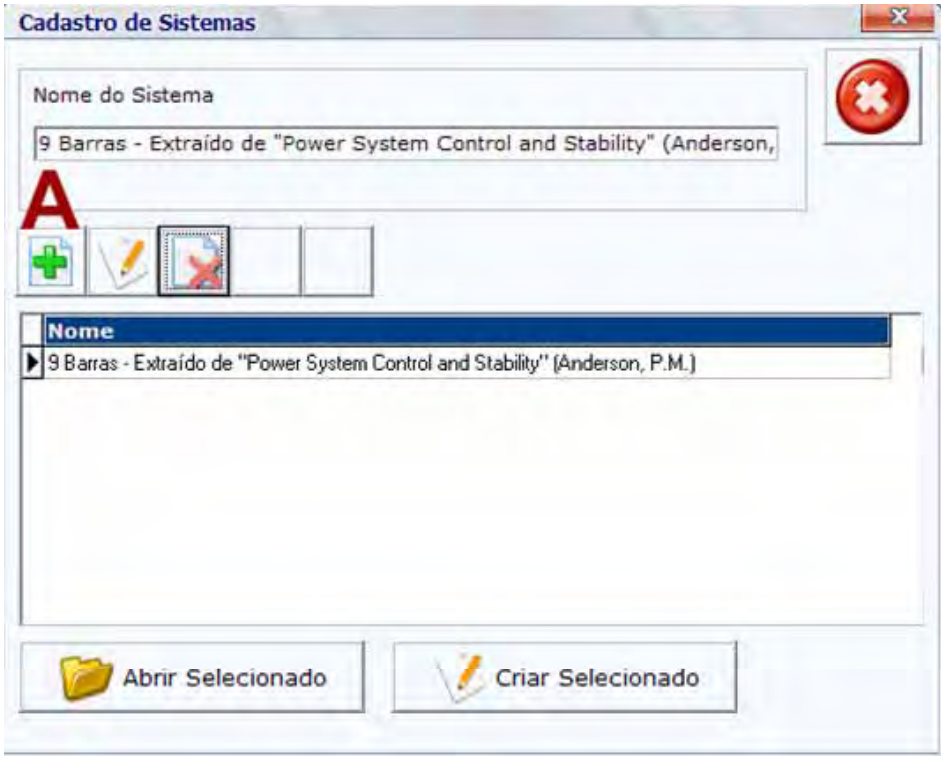


Figura 4 - Inserindo um novo sistema

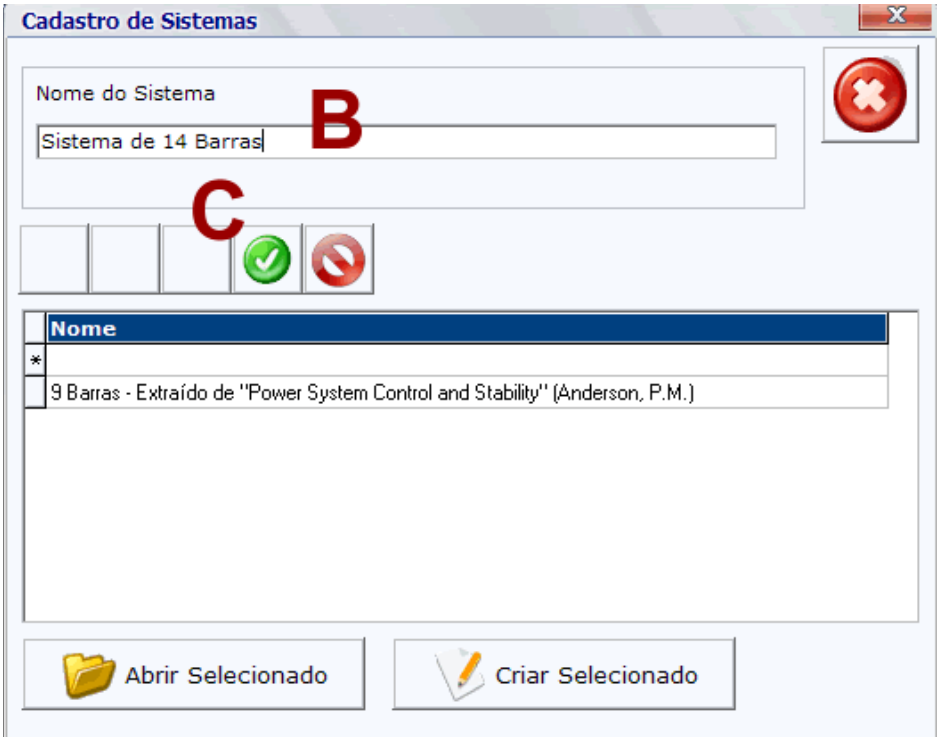


Figura 5 - Digitação do nome do sistema

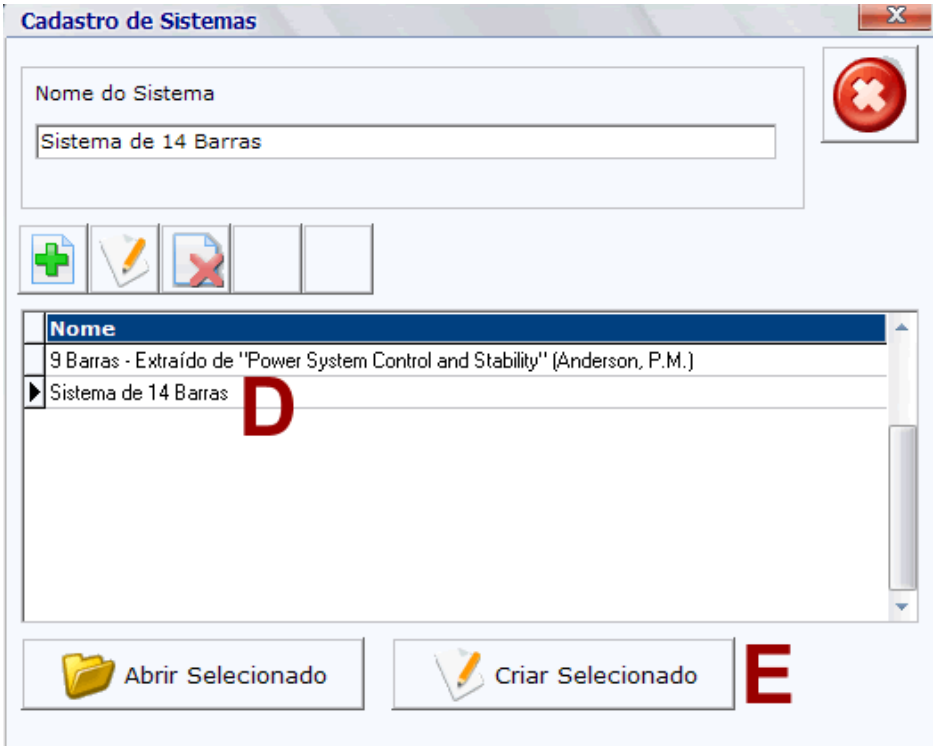


Figura 6 - Escolha do Sistema Criado

Após clicar no botão "Criar Selecionado" para lermos o arquivo do IEEE, na próxima tela deve-se clicar no botão "Ler" para selecionar o arquivo na pasta do sistema.

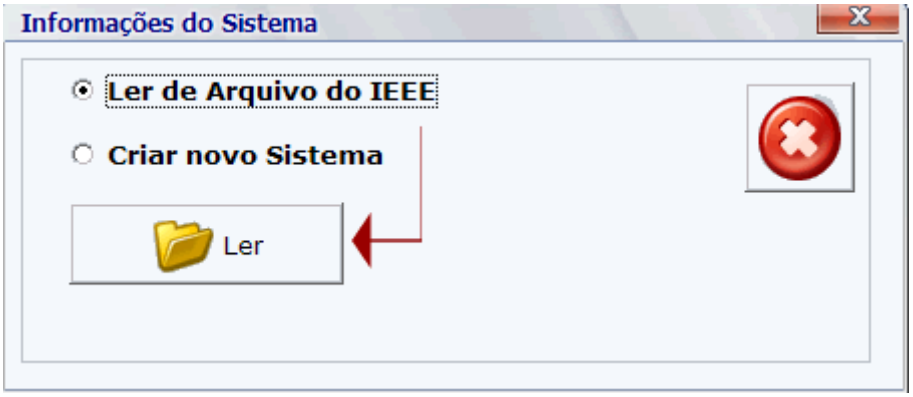


Figura 7 - Leitura de um Arquivo do IEEE

Após a seleção do arquivo, o software abre as telas de resumo dos dados lidos.

1.3 Digitando novos Sistemas

Seguindo os mesmos passos do item 1.1 devemos na figura 7

- **H.** Escolher a opção Criar novo Sistema
- **I.** Digitar o número de barras e linhas do sistema
- **J.** Clicar no botão "Criar"

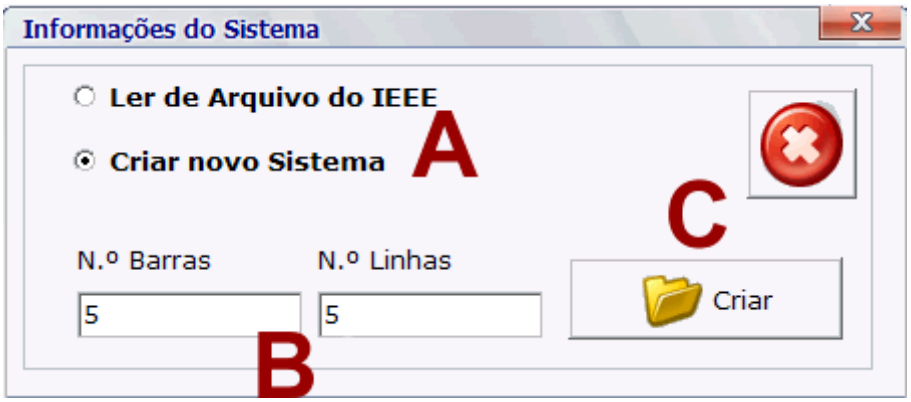


Figura 8 - Passos para digitação de um novo sistema

Na próxima tela o usuário poderá utilizar os "grids" de cada item de informação do sistema para digitação dos dados. Para inserir uma nova linha basta utilizar a "seta para baixo" do teclado. Para excluir uma linha basta utilizar o botão "Excluir" ao lado dos "grids"

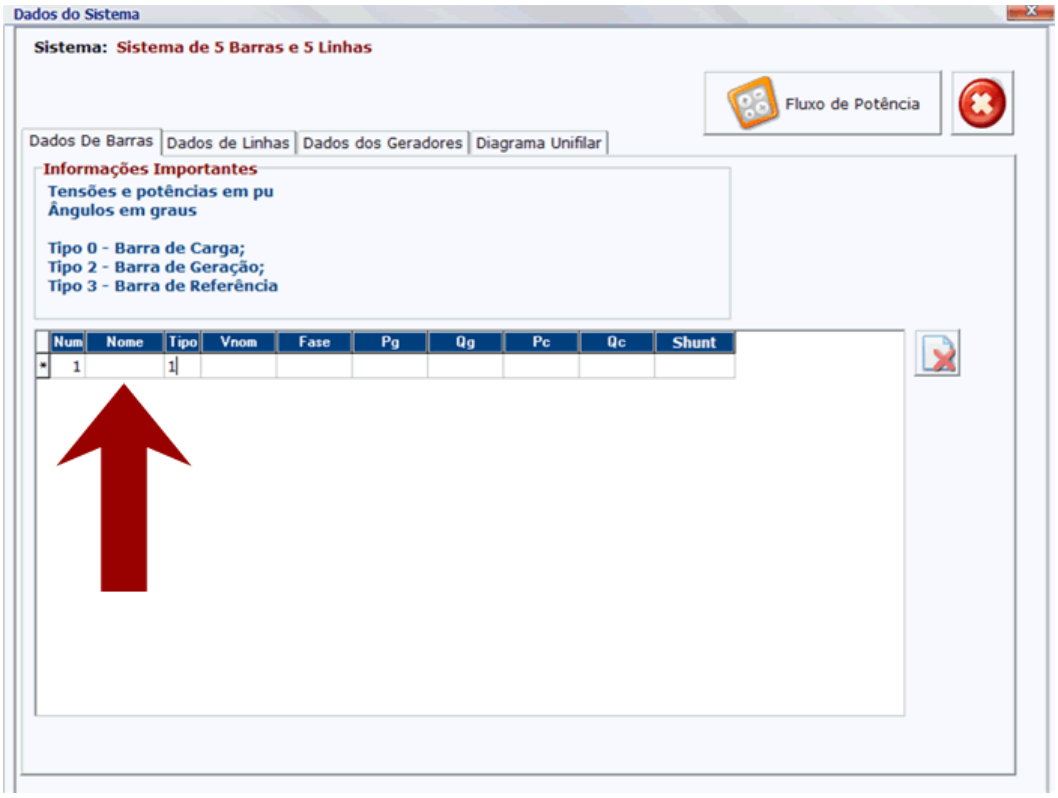


Figura 9 - Digitação do novo sistema

2. Fluxo de Potência

Próximo passo será clicar no botão "Fluxo de potência" da tela "Dados de sistemas".

Os passos para execução do fluxo de potência são:

K. Escolher o método para resolução

L. Digitar número de iterações máximas

M. Digitar a tolerância para convergência do método

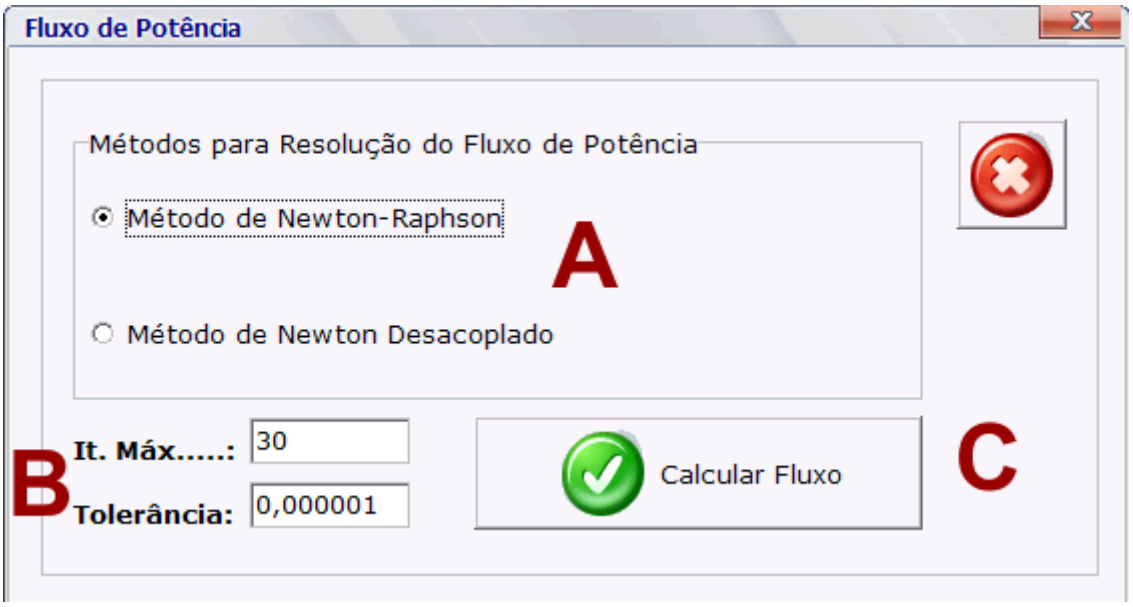
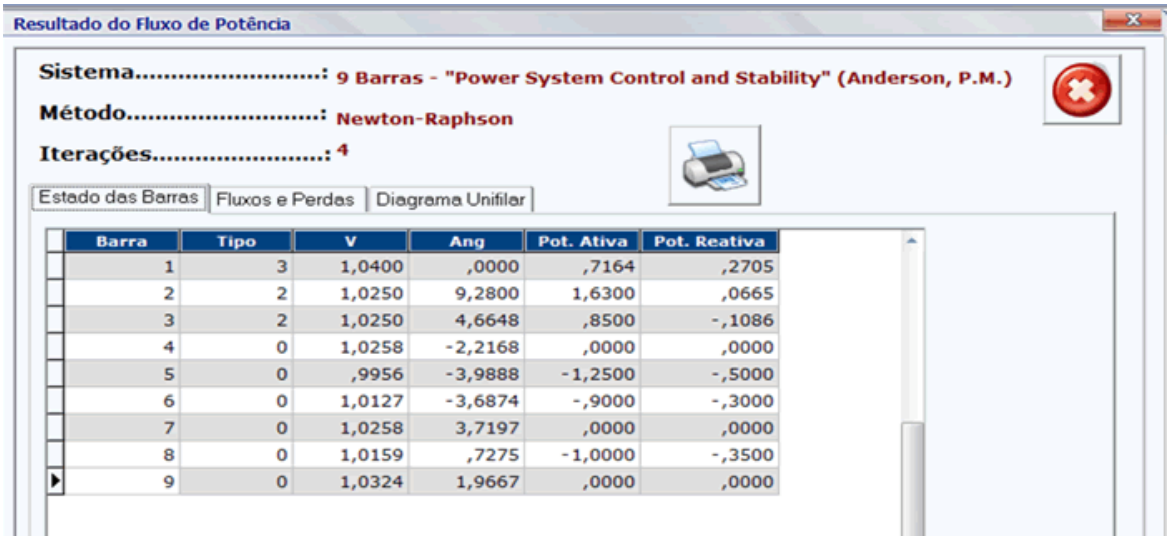


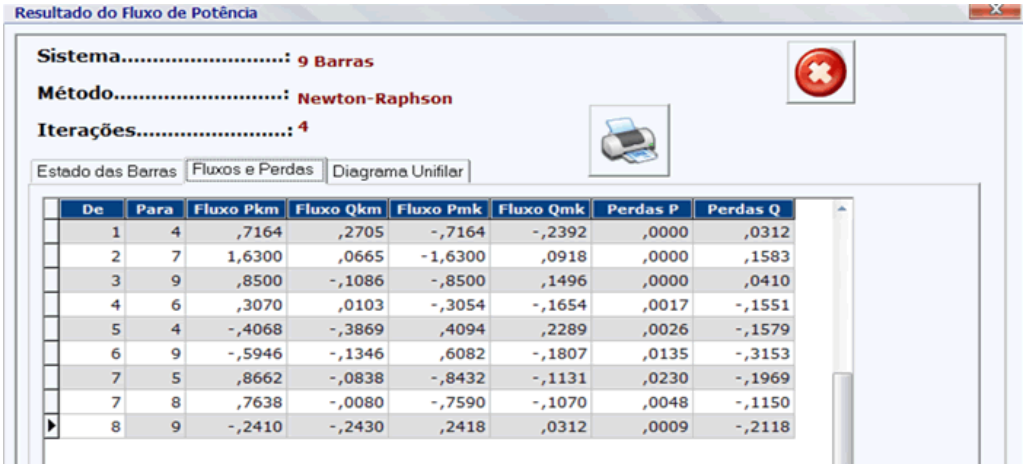
Figura 1 - Fluxo te Potência

Após a execução do método o software mostra os resultados dos "Estados das Barras" e "Fluxo na Linhas" . esses dados podem ser vistos no "grid" da tela de resultados ou em relatório próprio para impressão, para isso basta clicar no botão "Impressora".



Barra	Tipo	V	Ang	Pot. Ativa	Pot. Reativa
1	3	1,0400	,0000	,7164	,2705
2	2	1,0250	9,2800	1,6300	,0665
3	2	1,0250	4,6648	,8500	-,1086
4	0	1,0258	-2,2168	,0000	,0000
5	0	,9956	-3,9888	-1,2500	-,5000
6	0	1,0127	-3,6874	-,9000	-,3000
7	0	1,0258	3,7197	,0000	,0000
8	0	1,0159	,7275	-1,0000	-,3500
9	0	1,0324	1,9667	,0000	,0000

Figura 2 - Resultados de Fluxo te Potência para Estado das Barras



De	Para	Fluxo Pkm	Fluxo Qkm	Fluxo Pmk	Fluxo Qmk	Perdas P	Perdas Q
1	4	,7164	,2705	-,7164	-,2392	,0000	,0312
2	7	1,6300	,0665	-1,6300	,0918	,0000	,1583
3	9	,8500	-,1086	-,8500	,1496	,0000	,0410
4	6	,3070	,0103	-,3054	-,1654	,0017	-,1551
5	4	-,4068	-,3869	,4094	,2289	,0026	-,1579
6	9	-,5946	-,1346	,6082	-,1807	,0135	-,3153
7	5	,8662	-,0838	-,8432	-,1131	,0230	-,1969
7	8	,7638	-,0080	-,7590	-,1070	,0048	-,1150
8	9	-,2410	-,2430	,2418	,0312	,0009	-,2118

Figura 3 - Resultados de Fluxo te Potência para Fluxo nas Linhas

3. Estabilidade de Sistemas de Energia

Para acessar o módulo de Estabilidade de Sistemas, deve-se fechar as telas de resultados do Fluxo de Potência e assim na tela principal, clicar no botão estabilidade. Estão disponíveis os sub-módulos:

3.1 Estabilidade Estática

Nessa ajuda será apresentada a análise de estabilidade para o sistema de 9 barras de (ANDERSON e FOUAD, 1994) , com a eliminação da linha 7-5 e falta na barra 7. Assim os passos para se obter os resultados de estabilidade estática são:

N. Escolher a linha que será excluída

O. Selecionar a barra de falta

P. Executar "Fluxo de Potência Pós-falta"

Q. Verificar os resultados de tensões Pré E Pós-Falta

Figura 1 - Passos para Execução da Estabilidade Estática

Para o cálculo da Estabilidade Estática deve-se executar o "Fluxo de Potência Pós-Falta", onde o usuário pode observar se o método convergiu ou não convergiu. Se o fluxo de potência para o regime pós-falta convergir, significa que, após a eliminação da falta (retirada da linha), o sistema tem um ponto estável. É preciso observar se os valores de ângulo pós-falta dos geradores não superam 90º em defasagem ao ângulo da barra de referência. Se não convergir, nada se afirma, pode significar apenas que o sistema de equações algébricas é mal condicionado, não necessariamente que o sistema elétrico não tenha ponto de equilíbrio estável.

As defasagens pós-falta obtidas do fluxo de potência são estáticas, ou seja, são valores do ponto de operação do sistema após a falta e após passar a perturbação. Como em um pêndulo que balança até parar no ponto de equilíbrio. No sistema de potência, este ponto de equilíbrio só existe se as defasagens entre os eixos das máquinas (todas) forem inferiores a 90 graus.

Após a execução do fluxo de potência pós-falta podemos concluir que o sistema não ultrapassa a defasagem limite de 90º assim existe um ponto de operação estável após a execução do fluxo de potência pós-falta. Temos os valores de tensões internas pré-falta e pós-falta dos geradores no botão "Tensões e ângulos internos Pré e Pós-Falta".

3.2 Estabilidade Transitória

Para o estudo da estabilidade transitória, são necessários os valores das reatâncias sub-transitórias e das constantes de inércia (neste caso representadas pelas constantes H e x'_d) de cada máquina. Os dados foram lidos no módulo de fluxo de potência e podem ser inseridos, ou alterados na tela de dados de entrada do sistema.

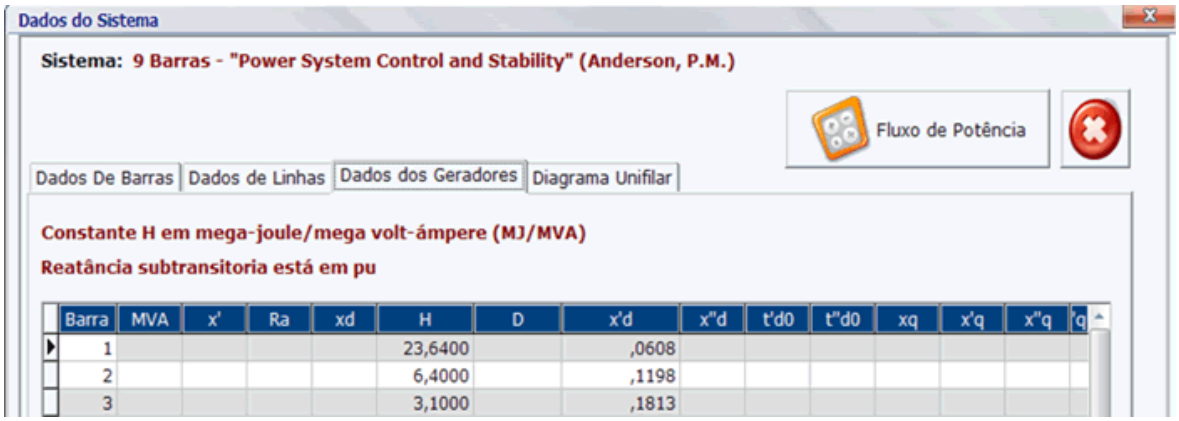


Figura 2 - Dados de Geradores lidos no módulo de Fluxo de Potência

Para as simulações seguintes, será considerado um curto-circuito trifásico próximo ao barramento 7. Tal defeito será eliminado através da saída da linha que interliga os barramentos 7 e 5. O método de integração utilizado será o Runge-Kutta de 4ª ordem, o passo de integração será 0,001 e o tempo de abertura da linha seguido exemplo de (ANDERSON e FOUAD, 1994) será de 0,0833 segundos e tempo total de simulação será de 2 segundos. Esses dados podem ser alterados pelo usuário conforme a necessidade da simulação.

A. Assim os passos para execução da Estabilidade Transitória são:

B. Escolher os parâmetros do sistema: Tempo de abertura, Tempo de Simulação, Passo de Integração e Freqüência.

C. Clicar no botão "Simulação" para que seja realizado o método de integração Runge Kutta 4ª Ordem

D. Verificar os relatórios de resultados do Runge Kutta 4ª Ordem

E. Visualizar o gráfico de Velocidade x Tempo

F. Visualizar o gráfico de Ângulo x Tempo

G. Visualizar o gráfico de Defasagem Angular

Figura 3 - Passos para Execução da Estabilidade Transitória

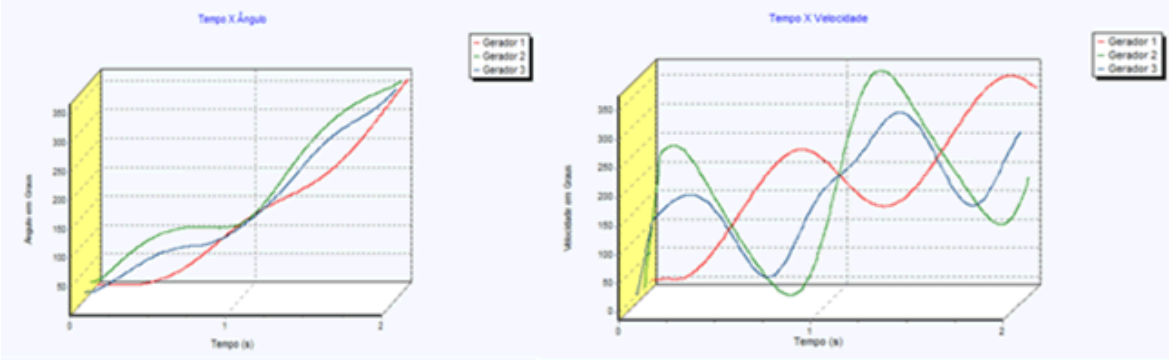


Figura 4 - Gráficos de Tempo x Ângulo e Tempo x Velocidade

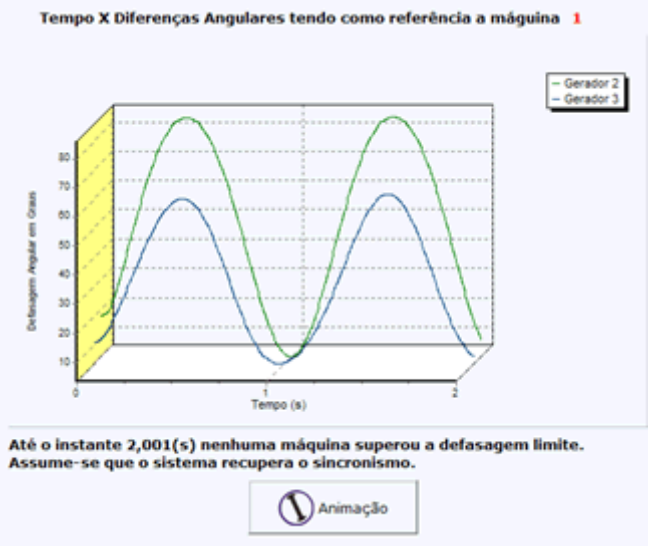


Figura 5 - Gráficos de defasagem angular

A análise das defasagens angulares é feita considerando, empiricamente, como ângulo limite 180º entre as máquinas em estudo e uma máquina de referência. Na simulação em tela, o aplicativo considerou a máquina 1 como referência, ou seja, o gráfico da Figura 5 representa as defasagens angulares entre os geradores 2 e 3 em relação ao gerador 1. A nota “Assume-se que o sistema recupera o sincronismo” é adicionada após constatar que no intervalo considerando não ocorreu violação do critério de máximo ângulo utilizado

Ainda pela figura 5, podemos notar que a máquina 2 é mais sensível ao defeito ocorrido no barramento 7. Esta maior sensibilidade era esperada já que tal gerador é o que se encontra mais próximo ao defeito. A maior defasagem angular entre o gerador 2 e o gerador 1 ocorre aproximadamente em 0,43 segundos.

Uma ilustração bastante didática do transitório é apresentada quando se utiliza a animação que representa o movimento dos rotores dos geradores em estudo. Essa animação deve ser executada clicando-se no botão "Animação" do gráfico de Defasagens Angulares.

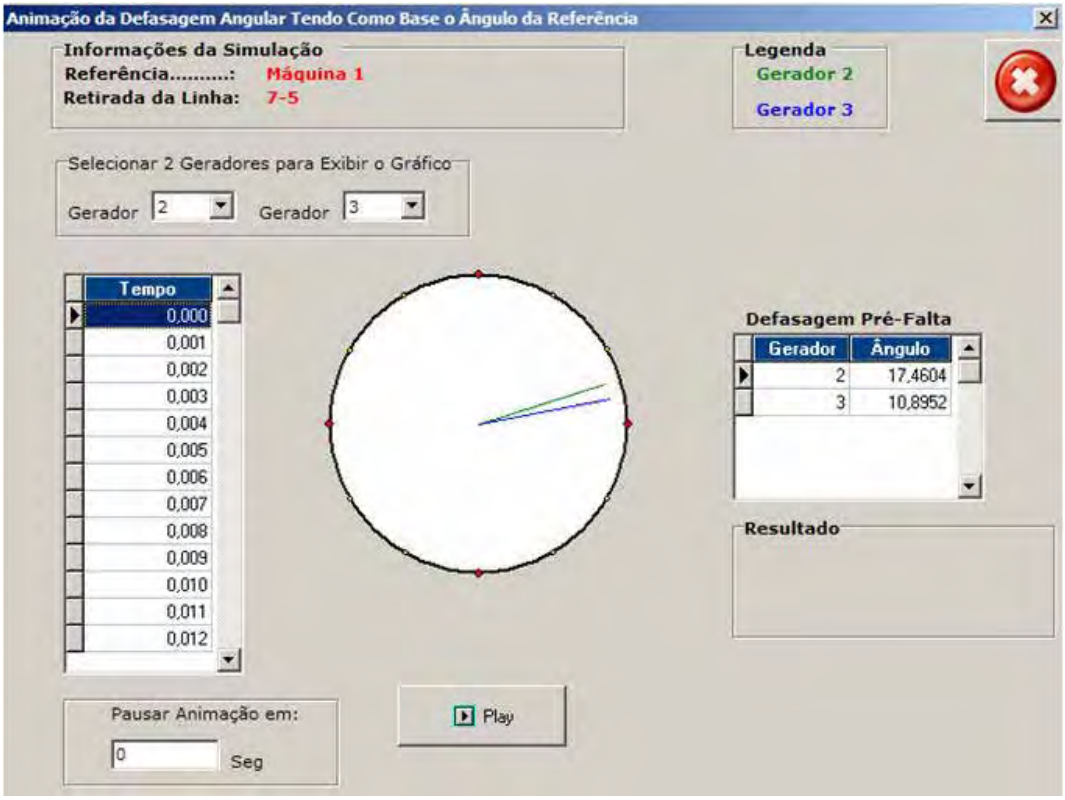


Figura 6 - Animação de defasagem angular

Para se executar a animação das defasagens angulares dos geradores em relação a referência deve-se:

A. Digitar o tempo de pausa da animação (se for zero a animação será executada sem pausa)

B. Clicar no botão "Play"

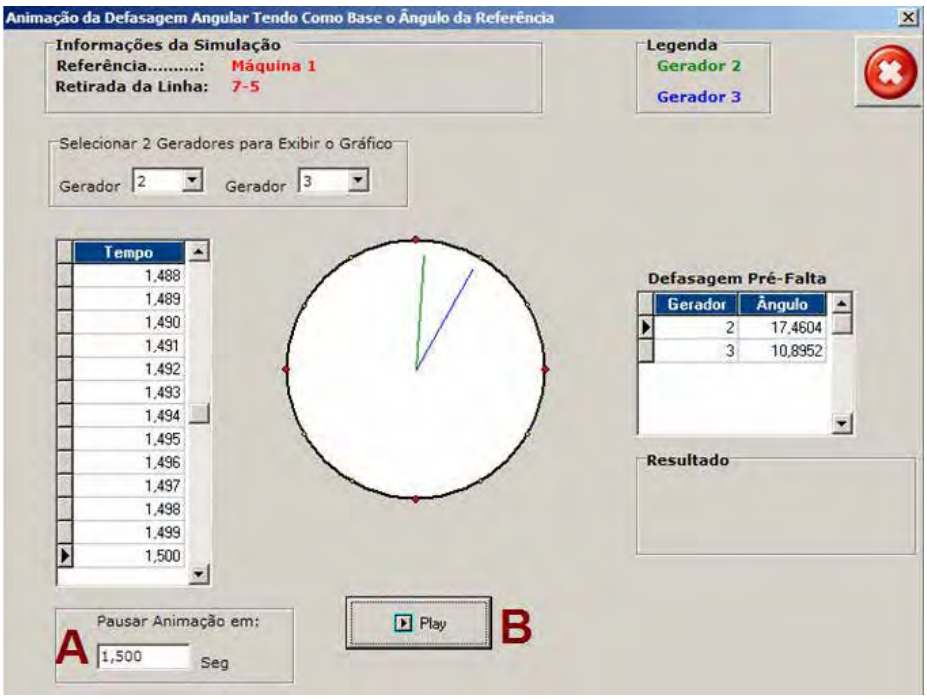


Figura 7 - Animação pausada em 1,5 segundos

A posição inicial da animação está nos ângulos de defasagem pós-falta. Após a execução do sistema será mostrado no espaço "Resultado" se o sistema evoluiu para a situação estável ou instável após o término do tempo de simulação.

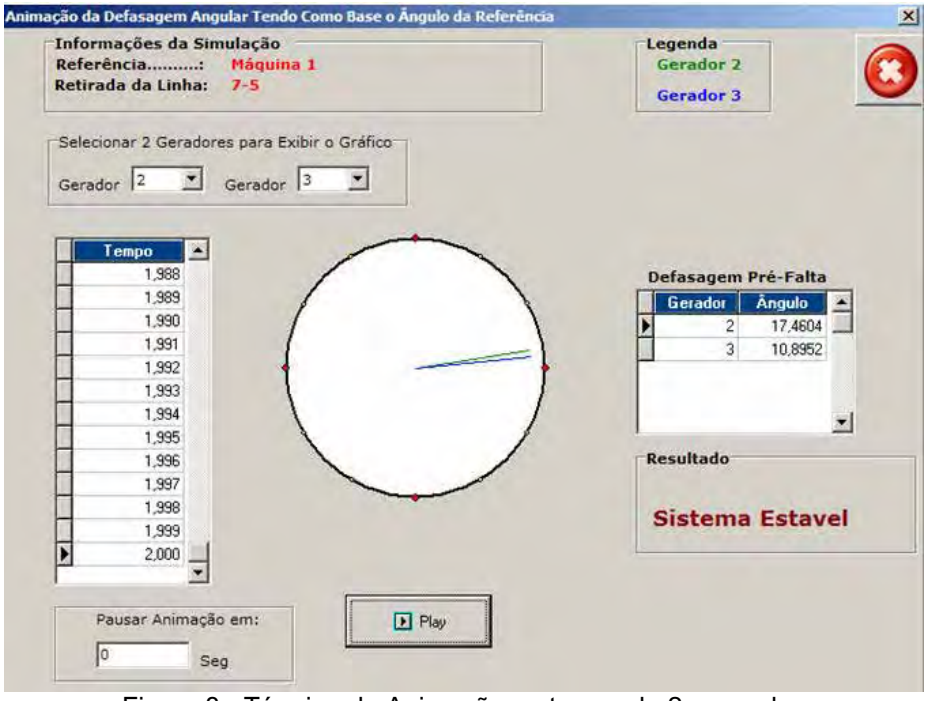


Figura 8 - Término da Animação no tempo de 2 segundos