**Universidade Estadual de Campinas**
**Faculdade de Engenharia Elétrica e de Computação**
**Departamento de Sistemas de Energia Elétrica**

# Um Simulador do Controle em Tempo Real de Sistemas de Energia Elétrica

Por: **Regina Morishigue Kawakami**
Orientador: **Prof. Dr. Ariovaldo Verandio Garcia**

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da UNICAMP como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica.

**Banca Examinadora:**

| | |
|---|---|
| Prof. Dr. Ariovaldo Verandio Garcia ( Presidente ) | FEEC/UNICAMP |
| Prof. Dr. Antonio Padilha Feltrin | FEIS/UNESP |
| Prof. Dr. Carlos Alberto de Castro Jr. | FEEC/UNICAMP |
| Prof. Dr. Fujio Sato | FEEC/UNICAMP |

Campinas, dezembro de 2000.

## RESUMO

A complexidade do controle em tempo real de sistemas de potência é tão grande que muitas ferramentas devem estar disponíveis de modo a ajudar no treinamento de operadores de sistemas de gerenciamento elétrico e também no ensino de estudantes de engenharia. Sistemas de treinamento baseados em computador têm exercido um papel importante ao fornecer aos usuários um melhor entendimento da operação do sistema de potência. O presente trabalho tem por objetivo apresentar um simulador do controle em tempo real de sistemas de energia elétrica, com interface gráfica ao usuário baseada em Tcl/Tk (Tool Command Language/ Tool Kit). Os usuários podem interagir dinamicamente com o sistema em estudo através do diagrama unifilar, alterando as medidas do sistema (MW, MVAr e kV) e também mudando os taps de transformadores e os estados dos dispositivos seccionadores (chaves e disjuntores).



## ABSTRACT

The complexity of the real time control of power system operation is so high that several tools must be available in order to help training electric management systems (EMS) operators and also teaching engineering students. Computer based training systems have played an important role in providing users with a better understanding of power system operation. In this work, the objective is to present a Tcl/Tk (Tool Command Language/ Tool Kit) based GUI (Graphical User Interface) real time power system operation simulator program. Users can dynamically interact with the target system by using an one-line diagram to change system measurements (MW, MVAr and kV) and also change transformer taps and circuit breaker status.

"Não nos perguntamos qual o propósito útil dos pássaros cantarem, pois o canto é o seu prazer, uma vez que foram criados para cantar.
Similarmente, não devemos perguntar por que a mente humana se inquieta com a extensão dos segredos dos céus...
A diversidade do fenômeno da natureza é tão vasta e os tesouros escondidos nos céus tão ricos, precisamente para que a mente humana nunca tenha falta de alimento."

Johannes Kepler
mysterium Cosmographicum

À minha filha Sabrina

# AGRADECIMENTOS

Desejo expressar meus sinceros agradecimentos,

ao Prof. Ariovaldo Verandio Garcia não só pelo trabalho de orientação, mas também pela amizade e estímulo,

aos meus pais Hiroshi e Thereza (in memoriam) e minha irmãs, por todo apoio e incentivo que sempre me deram,

ao meu marido Paulo Kenji, pelo incentivo, paciência, dedicação e amor,

a todos os colegas do DSEE, professores, Edna secretária e Miriam analista de sistemas, que sempre me ajudaram no decorrer desta jornada,

aos amigos Asada, Grilo, Haffner, Walmir, Madson, Heder, prof. Sato e prof. Castro, pela paciência e ajuda nos momentos necessários,

a FUNDAÇÃO CAPES (Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), pelo suporte financeiro

e a todos que contribuíram para a realização deste trabalho.

# Sumário

# Lista de Figuras

# Capítulo 1

# Introdução

A área de sistemas de potência foi uma das primeiras da engenharia a utilizar o computador como ferramenta para a solução de problemas e implantação de sistemas de supervisão e controle em tempo real. Ao longo dos anos, essa dependência vem se estreitando e tomando nova forma. Como conseqüência, além de utilizar os recursos oferecidos pela tecnologia de informática, os projetistas desses sistemas desenvolvem técnicas e sistemas de computação complexos, necessários para o atendimento dos requisitos exigidos pela área.

Atualmente, o planejamento, a operação e o controle de sistemas de potência enfrentam desafios impostos por: nova conjuntura econômica e tecnológica, utilização de dispositivos eletrônicos inteligentes rápidos e interligados em rede, sistemas integrados de supervisão e controle em tempo real, restrições ambientais, regimes de operação mais estressantes, entre outros. Para superar tais desafios, novos desenvolvimentos de *hardware* e *software* como, por exemplo, em processamento paralelo e distribuído, computação visual e inteligência artificial têm sido largamente utilizados. Tudo isso leva ao tratamento de problemas multidisciplinares cuja solução se baseia no uso intensivo da tecnologia de computação.

Dada a complexidade da operação em tempo real de sistemas de energia elétrica, uma ferramenta muito útil tanto no treinamento de operadores quanto no ensino de engenharia elétrica é o simulador. Por anos os programas de simulação por computador desempenharam um papel importante ao permitir um melhor entendimento da operação de um sistema de potência. Como resultado dos rápidos avanços em *hardware* e *software*, as ferramentas educacionais de sistemas de potência baseadas em computador passaram de implementações muito simples, fornecendo ao usuário pouco mais que uma série de saídas numéricas até representações muito detalhadas do sistema de potência em uma complexa interface gráfica com o usuário (GUI) [Over 95]. Exemplos de tais programas incluem simuladores para treinamento de despachantes (DTS) para instrução do operador.

Diversos sistemas baseados em computadores têm sido desenvolvidos ao longo dos anos. O presente trabalho tem por objetivo apresentar o desenvolvimento de um simulador

de sistema de potência com funções avançadas de análise de rede em tempo real para treinamento e análise, com interface gráfica com o usuário baseada em Tcl/Tk (Tool Command Language/Tool Kit).

No Capítulo 2, inicialmente são feitas algumas considerações sobre a supervisão e controle em tempo real de sistemas elétricos de potência, o esquema geral de um centro de controle, a estrutura de sistemas para análise de redes em tempo real, as bases de dados e sua estrutura. A seguir são descritas as funções de análise de rede em tempo real: configurador de rede e estimador de estado.

No Capítulo 3 é feita uma introdução sobre simuladores e a seguir são apresentados alguns exemplos de simuladores para treinamento desenvolvidos ao longo dos anos e citados em trabalhos publicados na área, com suas características, objetivos e exigências.

O Capítulo 4 apresenta o desenvolvimento do simulador proposto, com as definições feitas, arquivos de dados, características e algoritmo de solução. São descritos os passos trilhados na elaboração do programa que deu origem ao simulador.

No Capítulo 5 são apresentados exemplos de simulações efetuadas com o programa desenvolvido. As telas de saída são mostradas para ilustrar as simulações. As opções existentes de simulação são também apresentadas.

No Capítulo 6 são apresentadas as conclusões gerais.

Ao final do trabalho encontram-se três apêndices: Apêndice A, contendo uma breve introdução à linguagem de programação Tcl/Tk; Apêndice B, detalhamento sobre como foram elaborados os arquivos de dados com alguns exemplos e o Apêndice C, cópia do artigo aceito no IEEE-PES / CSEE Intenational Conference on Power System Technology - PowerCon 2000, Austrália.

# Capítulo 2

# Controle em Tempo Real de Sistemas Elétricos de Potência

## 2.1 Visão Geral

### 2.1.1 Histórico

A necessidade da introdução de funções de análise de segurança em sistemas elétricos de potência recebeu impulso já em meados de 1960 devido em grande parte ao blecaute na região de New York em 1965.

Com o crescimento na dimensão e complexidade dos sistemas devido à contínua interligação de sistemas antes isolados, houve melhorias no que se refere à garantia de continuidade de fornecimento, perfil de tensões, etc., resultando porém, em um sistema de grande porte, distribuído e de operação complexa [Garc 93].

Os primeiros centros computadorizados surgiram basicamente realizando apenas uma função de controle que é o controle da geração. Esse controle envolve o controle automático da geração e o despacho econômico. Para a execução dessas funções os requisitos de informações do sistema não são muito elevados. Bastam praticamente sinais proporcionais à freqüência, ao intercâmbio entre áreas (empresas) e informações sobre os custos de operação das usinas. Inicialmente essa função foi realizada através de computadores analógicos passando mais tarde a ser realizada via computadores digitais.

Em seguida surgiram os centros de supervisão, que realizam o controle supervisório do sistema. Nesse caso já se necessita de um volume muito maior de informações, tais como estado (aberto/fechado) de disjuntores/chaves no sistema, medidas analógicas (MW, MVAr, kV, A) em ramos (linhas e transformadores) e em barramentos, etc. Esse sistema é conhecido como SCADA ("Supervisory Control And Data Acquisition" [Gaus 87]). Trata-

se de um sistema caro, com requisitos de confiabilidade bem definidos, e a cada certo intervalo de tempo, da ordem de segundos, permite que as informações estejam disponíveis no Centro de Controle. Permite ainda o telecomando de disjuntores, reguladores de taps de transformadores, bancos de reatores/capacitores, etc.

A inclusão das funções de controle em tempo real (Análise de Segurança), uma das justificativas para a implantação de um centro de controle, é uma extensão do controle supervisório. O processamento e a qualidade das informações (medidas) estão em um nível bem mais elevado. Envolvem todo um trabalho de modelagem do sistema, de determinação do *estado* atual do sistema e de possíveis ações de controle que permitam que, caso ocorram eventos inesperados (mas previstos) – contingências, o sistema ainda opere respeitando determinadas restrições.

### 2.1.2 Esquema geral de um Centro de Controle

Um sistema moderno de supervisão e controle de sistemas elétricos de potência é composto basicamente de:

- Sistema de aquisição de dados;
- Sistema de computação;
- *Software* básico e avançado.

O sistema de aquisição de dados é composto de estações remotas (UTR), responsáveis pela captação da medida, da conversão analógica/digital (A/D), e pelo envio ao centro de controle via canal de comunicação.

O sistema de computação evoluiu muito. Inicialmente era composto basicamente de dois computadores centrais, operando de maneira "dual", um deles sendo responsável pelo processamento e o outro ficando como *backup*. Nos sistemas atuais é composto por uma rede de estações de trabalho de alto desempenho (sistema distribuído).

## 2.2 Estrutura de sistemas para análise de redes em tempo real

Como já mencionado anteriormente, realizar a análise de segurança é um dos principais objetivos de um centro de operação. Nessa análise procura-se determinar se o sistema, em um dado instante, está operando de modo seguro ou não. Além da análise de segurança existem outras funções, não menos importantes, de supervisão do COS.

As funções de análise de redes são uma parte do *software* de aplicação do COS. São as responsáveis pela construção do modelo da rede (em tempo real) que envolve, basicamente,

a determinação do *estado* do sistema. O *estado* é composto pelos *taps* de transformadores e pelas magnitudes e fases (ângulos) das tensões complexas dos nós elétricos. Com o *estado* disponível, todas as demais informações do sistema podem ser obtidas através de cálculos. Os dados disponíveis são:

- informações sobre a topologia de todas as subestações monitoradas;
- dados dos modelos de todos os dispositivos dessas subestações e de linhas de transmissão;
- estado dos disjuntores (aberto/fechado) – medidas lógicas;
- medidas analógicas (injeções e fluxos de potência ativa e reativa em barras e linhas/transformadores, e magnitudes de tensões de barras, fluxos de corrente nas linhas, taps de transformadores).

Em geral há redundância de informações, ou seja, essas medidas são realizadas em um número bem maior que o mínimo necessário para se calcular o estado. Isso permite que se obtenha o estado mais provável correspondente a um determinado instante (o da realização das medidas) de maneira mais precisa. Nesse processo de obtenção do estado, erros grosseiros são eliminados do conjunto de medidas.

De uma forma muito simplificada, pode-se separar a obtenção do modelo nas seguintes etapas:

- configuração da rede elétrica;
- estimação de estado (que na verdade é composta por análise de observabilidade e estimação de estado propriamente dita);
- obtenção do modelo da rede.

Obtém-se o estado apenas da parte da rede em que isso é possível (parte *observável* do sistema). O sistema observável pode se alterar de um momento para outro, dado que tanto o conjunto de medidas como a topologia do sistema pode se alterar. Isso pode representar um problema se não for tratado de forma adequada. É comum existir uma região de interesse para o qual se deseja obter o modelo (sistema *modelado*, sendo que o restante do sistema é o sistema *externo* – ver Fig. 2.1).

O estado do sistema *modelado* é sempre obtido, seja utilizando dados de tempo real (medidas realmente realizadas) ou então utilizando também outros dados (como de previsão de carga, ou da varredura anterior, etc.).

Podemos ainda considerar o sistema modelado como sendo composto de dois subsistemas: interno e de fronteira. O subsistema de fronteira é composto pelas barras que

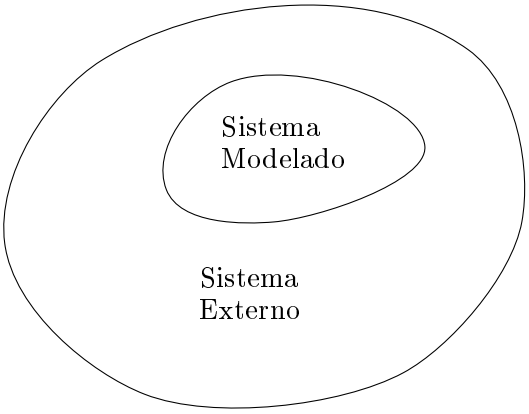


Figura 2.1: Sistemas modelado e externo.

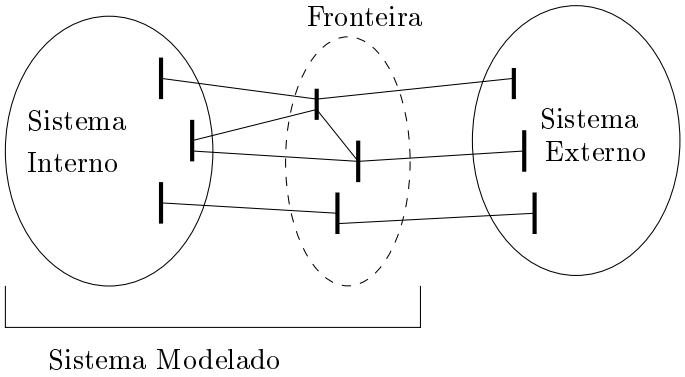


Figura 2.2: Sistemas interno, fronteira e externo.

se ligam ao sistema externo e (se existirem) as ligações entre elas. Com esta divisão o sistema ficaria representado como mostrado na Fig. 2.2 [Mont 79].

De maneira geral, as funções de análise de redes de um centro de controle são:

- Configurador
- Estimador de estado
- Modelagem da rede
- Análise de segurança
- Fluxo de potência ótimo

Estas funções utilizam dados do SCADA e da base de dados, bem como da previsão de carga.

O esquema padrão de execução dessas funções é o seguinte [Garc 93]:

**Configurador:** a cada alteração da topologia do sistema (abertura ou fechamento de disjuntores e chaves), ou através de solicitação do operador do centro de controle, o processo se inicia pelo configurador, que é responsável pela construção do modelo da rede (quantos e quais são os nós elétricos do sistema; onde estão localizadas as medidas, etc.).

**Estimador:** o resultado do configurador é utilizado pelo estimador de estado, que, na realidade é composto (na maioria das implementações) de 3 fases:

- Observador: determina qual a parte do sistema é observável, ou seja, para qual é possível a obtenção do estado (magnitudes e ângulos das tensões das barras);
- Estimador: obtém o estado para a parte observável do sistema;
- Processamento de erros grosseiros: verifica se há e identifica erros grosseiros no sistema. Em caso afirmativo o estado é reestimado.

**Modelagem da Rede:** usando o estado estimado obtido pelo estimador, e utilizando dados da rede *externa* – parte não observável do sistema, constrói-se um modelo para o sistema externo, que reproduza seu comportamento para, por exemplo, o caso de simulação de contingências. Tendo todos os dados fornecidos pelas funções anteriores, ajustam-se as injeções de potência, nas barras (nós) de fronteira. O resultado final é um modelo da rede em tempo real, ajustado para a realização de estudos, análises, etc.

**Análise de Segurança:** a análise de segurança consiste basicamente na simulação de um elenco de contingências e a verificação de limites operativos. Normalmente é produzida uma lista (em tempo real ou em modo *off-line*) de contingências mais críticas, dadas as restrições de tempo de simulação.

**Fluxo de Potência Ótimo:** tendo o modelo da rede para o instante no qual foram realizadas as medidas (basicamente um modelo de fluxo de carga), podem ser realizados estudos visando obter um ponto de operação para o qual uma função objetivo seja minimizada. São determinadas quais ações de controle devem ser tomadas de modo que o sistema seja levado ao ponto de operação desejado. As restrições que esse problema trata são tanto de operação (limites operativos), quanto restrições de segurança.

## 2.3 As bases de dados e sua estrutura

As funções de aquisição de dados atualizam a parte em tempo real da base de dados, enquanto os aplicativos recuperam dados da base e armazenam resultados obtidos a partir deles. Tais bases de dados podem ser facilmente expandidas, implicando em aumento da

importância de sua manutenção. Esta função é responsável pela formatação, carregamento e atualização de todos os arquivos de dados estáticos necessários a todas as outras funções [Gaus 87].

Durante a fase de projeto da base de dados deve-se ter, principalmente, como objetivo, os seguintes pontos [Russ 79]:

- organizar a base de forma a otimizar o acesso aos dados pelos programas que a utilizam;
- otimizar sua edição para mudanças que ocorram rotineiramente no ambiente de operação;
- otimizar sua geração e manutenção para que novos equipamentos, bem como novas funções possam ser acomodados.

Uma das considerações mais importantes no projeto da base de dados é que ela deve prover meios eficientes para armazenamento e recuperação de informações.

Os sistemas de computadores hoje em dia empregados nos centros de controle já resolvem em parte vários problemas relativos ao projeto das bases de dados, mas ainda são severas as limitações impostas pelo tempo de acesso aos dados. Assim, deve-se considerar com atenção o número de acessos a disco necessários à realização de uma dada tarefa.

Outro fator relevante de projeto é o compromisso que sempre existirá entre o grau de generalização da base de dados e a velocidade de operação, isto é, bases de dados altamente generalizadas frequentemente apresentam excessivo custo computacional adicional resultante do desempenho de uma operação que não é diretamente produtiva no processo em que aparece, o que as tornam inadequadas para utilização em tempo real crítico [Zaga 95].

A aquisição de dados é feita de forma periódica. A maioria dos sistemas usados para aquisição de dados em concessionárias de energia, consistem na transmissão de dados das UTRs para o computador mestre, também chamado de *master station*, somente quando ocorre uma requisição do mestre para a UTR. Existem duas maneiras com as quais as UTRs podem responder às solicitações do computador mestre: uma seria devolvendo ao mestre os valores reais ou o estado do ponto ou grupo de pontos desejados, outra seria devolvendo ao computador mestre somente o ponto ou grupo de pontos onde tiverem ocorrido mudança de estado ou variação no seu valor superior a um limite pré-definido desde a última requisição. Esta opção é conhecida como "transmissão por exceção". A transmissão por exceção tem como grande vantagem a redução no custo computacional resultante do desempenho de uma operação de processamento no computador mestre. Pode ser citada também a diminuição da média de carregamento no circuito de comunicação em relação à primeira opção.

Segundo [Gaus 87], o processo de aquisição de dados pode ser considerado como um processo coletivo formado por vários subprocessos especializados e altamente relacionados. Esses subprocessos podem incluir:

- varredura interna e atualização rápida da base de dados interna da UTR;
- consulta periódica do computador mestre à UTR;
- transmissão do conjunto de dados requeridos pela UTR para a estação mestre;
- detecção de erros de transmissão através de conferência de dados;
- conversão dos dados em unidades de engenharia;
- sobreposição de estados novos ou valores anteriores na base de dados.

## 2.4 Modelagem da rede supervisionada – Configurador

### 2.4.1 Introdução

A modelagem da rede deve ser tal que permita a representação completa das condições atuais de operação da rede, ou seja, inclui conhecimento dos parâmetros, da topologia e do estado (magnitudes e ângulos das tensões) da rede tanto da parte supervisionada como da rede externa (ou de seu equivalente).

#### Dados utilizados

- Estado (aberto/fechado) de disjuntores e chaves
- Medidas analógicas
- Dados estáticos da base de dados (parâmetros de linhas de transmissão e transformadores).

### 2.4.2 Componentes do sistema elétrico

O sistema de medição normalmente se restringe a uma parte do sistema. Isso força a divisão do sistema em duas partes: supervisionada e não supervisionada.

**Representação do sistema em tempo real**

Os dados para o problema de supervisão e controle de sistemas de potência são:

- parâmetros das linhas de transmissão e transformadores;
- características dos medidores;
- dados topológicos (forma de conexão entre os diversos elementos do sistema).

São consideradas conhecidas (ou então é necessário o conhecimento de) todas as ligações de disjuntores, esquemas de conexão, localização de medidores, etc. Esses dados devem constar de um Banco de Dados (ou de mais de um Banco) de forma tal que o acesso a ele seja rápido e confiável.

O banco de dados pode, a grosso modo, ser dividido em dois: o *Estático* e o *Dinâmico*. No banco estático devem constar todas as informações anteriormente referidas e cuja alteração/atualização se dá de maneira esporádica (rara) e somente pela atuação do operador do sistema. Do banco dinâmico constam as informações que são atualizadas pelo SCADA ou a cada varredura ou a cada certo número de varreduras (depende do sistema instalado). Normalmente o estado dos disjuntores/chaves (aberto/fechado) são atualizados a cada varredura (de 2 a 5 segundos) e as medidas analógicas (MW, MVAr, kV, A) são atualizadas com um período maior.

Os dados constantes do Banco de Dados são atualizados via operador (estático) ou via sistema de aquisição de dados (dinâmico). Uma parte do banco de dados deve ser reservada também a dados que não são medidos diretamente no sistema mas sim *calculados, estimados* através da função estimação de estado.

Normalmente, na construção do banco de dados devem ser tomados os cuidados de praxe: cuidar para que a consistência dos dados possa ser verificada de maneira fácil e rápida, e a interface com o usuário deve ser a mais amigável possível.

Os elementos do sistema são divididos em três tipos:

*disjuntores, circuitos e medidores.*

*Disjuntores* são todas as chaves existentes, que podem operar tanto de forma manual, automática ou telecomandada. São denominados *circuitos* todos os demais elementos do sistema: linhas de transmissão, barramentos, cargas, capacitores *shunt*, reatores, geradores, transformadores e segmentos de interligação. Um disjuntor só pode interligar dois circuitos. Os *medidores* (ou as medidas) são associados a um circuito (normalmente barramento ou linha de transmissão, dependendo do tipo da medida).

As únicas informações dinâmicas que o Configurador utiliza em seu algoritmo são os estados (aberto/fechado) dos disjuntores, que são atualizados constantemente. As demais medidas são apenas agrupadas e fornecidas ao estimador (observador) – fase seguinte ao configurador, de uma maneira conveniente.

Só há necessidade de acionar o configurador quando houver uma alteração do estado de algum disjuntor da rede, ou seja, se a topologia da rede se alterar. Ainda, só há a necessidade de reconfigurar as subestações relacionadas com a alteração topológica ocorrida [Frei 92].

### 2.4.3 Organização do banco de dados

Normalmente os bancos de dados são orientados por subestação (S/E). Isso permite que se proceda a Configuração apenas da S/E onde houve a alteração na topologia. A seguir, propõe-se uma estrutura do banco de dados (hipotética) para permitir apresentar um possível algoritmo do configurador.

O banco de dados acessado pelo configurador é composto de arquivos. Um esquema possível é a adoção de arquivos inter-relacionados por apontadores. Para efeito didático vamos imaginar o banco de dados dado a seguir, onde se tem 6 arquivos:

- Arquivo de subestações
- Arquivo de circuitos
- Arquivo de disjuntores
- Arquivo de medidores
- Arquivo de parâmetros de circuitos
- Arquivo de parâmetros de medidores

O acesso aos dados desses arquivos é realizado por apontadores e um esquema bem simplificado está mostrado na Figura 2.3.

Para acessarmos todos os circuitos de uma S/E qualquer, por exemplo a S/E "$i$", deveremos seguir os passos dados a seguir:

$ip = apontador\_de\_circuitos\_da\_S/E$"$i$"

DO WHILE $(ip > 0)$

$ici = circuito\_da\_S/E$

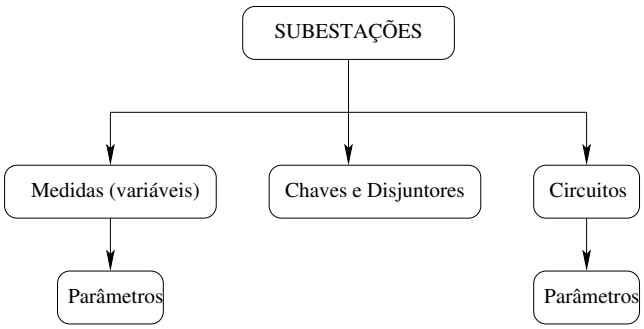


Figura 2.3: Arquivos do Banco de Dados

...
$ip = proximo\_circuito$

ENDDO

onde a variável *ici* assume os valores dos circuitos da S/E (assume-se que o último circuito é identificado por $proximo\_circuito = 0$).

Como exemplo tomemos os circuitos de uma subestação de um sistema interligado, mostrados na Figura 2.4. Os números entre parênteses representam os circuitos. Os quadrados com os números inseridos mostram a representação dos disjuntores. Os retângulos tracejados indicam as outras subestações que compõem o sistema.

### 2.4.4 Algoritmo do Configurador

A configuração do sistema é realizada por subestações.

O esquema geral (bem simplificado) é o seguinte [Garc 93]:

*i* escolhe-se uma S/E a ser configurada;

*ii* cria-se um vetor temporário contendo os números de todos os circuitos da S/E;

*iii* percorre-se todos os disjuntores da S/E, e para aqueles cujo estado se apresenta *fechado*, altera-se o número associado aos circuitos, de acordo com uma determinada regra. Por exemplo, o de número maior fica com o mesmo número do menor;

*iv* repete-se a regra dada acima até que não haja mais alteração de números associado a circuitos;

*v* Os circuitos que tiverem o mesmo número associado constituirão um *nó* elétrico. Cuidados especiais devem ser tomados quanto a linhas de transmissão e transformadores pertencentes ou não a mesma S/E;

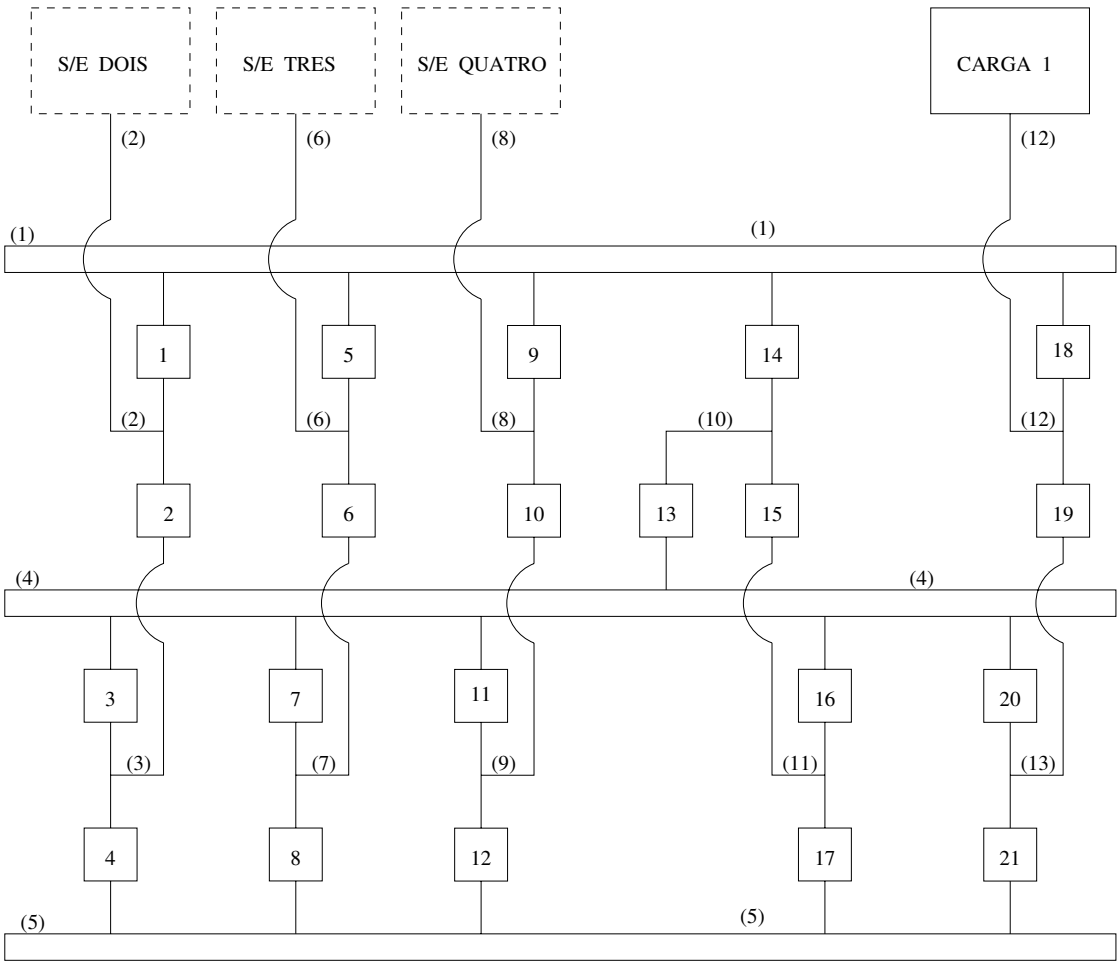


Figura 2.4: Exemplo de Circuitos em uma subestação

*vi* se existirem mais S/E a serem configuradas, escolhe-se outra e retorna-se a *ii*

Como resultado do algoritmo do configurador, tem-se um conjunto de circuitos formando um nó elétrico. Para as funções seguintes (estimação de estado, fluxo de potência, etc.), as informações necessárias são: quantos são os nós elétricos e como estão ligados entre si (topologia).

Percorrendo os nós e identificando os circuitos pelo *tipo* (linha de transmissão ou transformador), monta-se a estrutura de dados que se requer. Além disso, é necessário também configurar o sistema de medição. Uma vez configurada a S/E, as medidas realizadas nessa S/E devem ser associadas ou aos nós (no caso de medidas de magnitude de tensão, ou injeção de potência ativa e reativa) ou aos ramos (fluxos de potência ativa e reativa e magnitude de corrente).

## 2.5 Estimação de Estado

### 2.5.1 Introdução

A estimação de estado tem como objetivo a obtenção do estado (magnitudes e ângulos das tensões das barras) do sistema a partir de medidas realizadas em todo o sistema e a partir do modelo (circuito equivalente) montado pelo Configurador.

Algumas das grandezas (fluxos, injeções e tensões) não são medidas diretamente no sistema e mesmo as que o são podem conter erros inaceitáveis (erros grosseiros –*bad data*).

O número de medidas é maior que o mínimo necessário para se obter o estado do sistema (redundância), o que pode permitir, a detecção, a identificação e a eliminação de medidas com erros. Grandezas estimadas (calculadas com o estado estimado) podem ser mais confiáveis que as medidas.

O modelo da rede é composto de circuitos equivalentes das linhas de transmissão e transformadores.

A topologia da rede é determinada pelo configurador (que deve ser executado a cada alteração de topologia).

Na formulação convencional, assume-se que não há erros nem de topologia nem de parâmetros, ou seja, assume-se que o circuito elétrico equivalente do sistema seja perfeito.

As medidas realizadas no sistema são:

- magnitudes de tensão (valor eficaz) nas barras das subestações;

- fluxos de potência ativa e reativa nos ramos;

- injeções de potência ativa e reativa nas barras;

- magnitude das correntes nos ramos.

Deste conjunto de medidas, a imensa maioria dos estimadores de estado não utiliza medidas de corrente.

### 2.5.2 Método dos mínimos quadrados ponderados [Mont 83b, Garc 93]

O estado da rede, o modelo, os erros aleatórios e as medidas estão relacionados por:

$$z = h(x_v) + w \tag{2.1}$$

onde,

$z$ - vetor de medidas ($m \times 1$);

$h(.)$ - vetor de funções não lineares ($m \times 1$);

$x_v$ - vetor do estado verdadeiro ($n \times 1$) – (inacessível);

$w$ - vetor de erros de medidas ($m \times 1$);

$m$ - número de medidas;

$n$ - número de variáveis de estado.

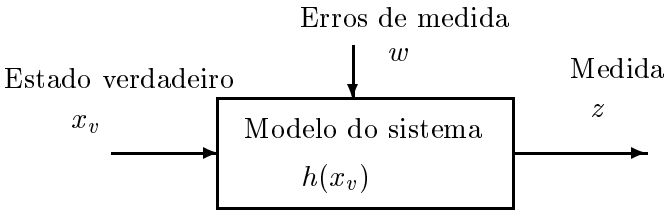


Figura 2.5: Modelo do sistema de medição

O vetor de estado é:

$$x = \begin{bmatrix} t \\ \theta \\ V \end{bmatrix} \quad \begin{array}{l} \text{- taps de transformadores} \\ \text{- ângulos} \\ \text{- magnitude das tensões} \end{array}$$

O conjunto de medidas pode ser particionado:

$$z = \begin{bmatrix} P_{kl}^{med} \\ P_{k}^{med} \\ Q_{kl}^{med} \\ Q_{k}^{med} \\ V^{med} \end{bmatrix} \begin{array}{l} \text{- fluxo de potência ativa} \\ \text{- injeção de potência ativa} \\ \text{- fluxo de potência reativa} \\ \text{- injeção de potência reativa} \\ \text{- magnitude de tensão} \end{array} \tag{2.2}$$

onde *med* representa medido.

O grau de redundância global do sistema ($\eta$) é:

$$\eta = \frac{m}{n} = \frac{m}{2N_B - 1} \tag{2.3}$$

onde $N_B$ é número de barras do sistema.

Obtém-se uma estimativa de $x_v$ encontrando um $\hat{x}$ tal que uma determinada função objetivo seja minimizada. A função objetivo mais utilizada é a de mínimos quadrados ponderados:

$$J(x) = [z - h(x)]' W [z - h(x)] \tag{2.4}$$

onde $W$ é uma matriz de ponderações e ($'$) representa transposição. $W$ é considerada diagonal:

$$W = \begin{bmatrix} 1/\sigma_1^2 & 0 & 0 & ... & 0 \\ 0 & 1/\sigma_2^2 & 0 & ... & 0 \\ 0 & 0 & 1/\sigma_3^2 & ... & 0 \\ ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & 1/\sigma_m^2 \end{bmatrix} \tag{2.5}$$

onde $\sigma_k^2$ é a variância da medida $k$.

O mínimo de $J(x)$ – equação (2.4), é obtido encontrando $\hat{x}$ tal que:

$$\nabla J(\hat{x}) = -2H'(\hat{x}) W [z - h(\hat{x})] = 0 \tag{2.6}$$

onde $H'(\hat{x})$ é a matriz jacobiana de $h(x)$ calculada no ponto $\hat{x}$:

$$H(\hat{x}) = \partial h(\hat{x}) / \partial x \tag{2.7}$$

A solução de (2.6) só é possível de ser encontrada através de um processo iterativo. Linearizando $h(x)$ em torno de um ponto $x^k$ temos:

$$h(x) \approx h(x^k) + H(x^k) \Delta x^k \tag{2.8}$$

Substituindo em (2.6) tem-se a recorrência:

$$[H(x^k)' W H(x^k)] \Delta x^k = H(x^k) W [z - h(x^k)] \tag{2.9}$$

Definindo a matriz ganho $G(x^k)$:

$$G(x^k) = [H(x^k)' W H(x^k)] \tag{2.10}$$

tem-se:

$$\begin{cases} G(x^k)\Delta x^k = H(x^k)W[z - h(x^k)] \\ \\ x^{k+1} = x^k + \Delta x^k \end{cases} \tag{2.11}$$

O processo de obtenção da solução se inicia com $k = 0$, considerando conhecido um valor inicial para o estado ($x^0$) e é repetido até que $|\Delta x_i| \leq \epsilon|$ para $i = 1, .., n$. $\epsilon$ é a tolerância e é fixada a priori.

### 2.5.3 Processamento de erros grosseiros [Garc 93]

Fontes de erros grosseiros: a falha ou descalibração (polarização) de medidores, falhas na transmissão de dados, falhas nos transdutores, etc.

Parte dos erros grosseiros são eliminados em uma fase anterior ao estimador, chamada *pré-processamento* ou *pré-filtragem* , onde medidas com erros absurdos são eliminados. Por exemplo, podem ser feitos testes do tipo:

- comparar valores nominais com as medidas;
- comparar com dados precedentes (da última amostra);
- testes de consistência – Lei de Kirchhoff;
- outros.

Parte desse processamento pode ser realizado no local da medida (nas estações remotas), significando um alívio na carga computacional do centro de operação.

### 2.5.4 Observabilidade do sistema

A parte *observável* de um sistema é aquela na qual é possível obter (calcular) todos os fluxos nos ramos (linhas e transformadores), a partir das telemedidas disponíveis. A determinação do sistema observável é realizada pela função observador, que é realizada logo após a execução do configurador. Enquanto o configurador determina a topologia da rede elétrica, o observador manipula tanto essa topologia como informações do sistema de medição (número e localização das medidas).

São comuns casos em que o sistema se divide em diversas *Ilhas Observáveis* devido a perdas temporárias de medidas, ou seja, o sistema observável pode ser constituído de ilhas mesmo que o sistema elétrico tenha mantido sua topologia original.

**Métodos de análise de observabilidade [Garc 93]**

Uma definição simples de Análise de Observabilidade é: se conseguirmos obter o estado de uma rede, então ela é observável.

Normalmente os métodos de análise de observabilidade são divididos em duas categorias: *numéricos* e *topológicos*. No caso dos métodos topológicos, trabalha-se apenas com informações como existência ou não da medida em um determinado local. Não se trabalha com parâmetros do sistema. Procura-se determinar o *rank* da matriz $H$ (e por conseguinte da matriz ganho $G$) através de análise baseada em teoria de conjuntos, grafos, etc.

Os métodos ditos *numéricos* partem da definição dada no início deste item: Se a matriz $G$ tiver inversa o sistema é observável. Mostrou-se [Mont 92] que mesmo sistemas que são *topologicamente observáveis*, podem ter matrizes ganhos *singulares*. Uma determinada combinação dos *valores* dos parâmetros das matrizes levam à singularidade. Além disso, uma outra grande vantagem de um método numérico é que todo o processamento realizado na fatoração das matrizes, para determinar se o sistema é ou não observável, é aproveitado no processo de obtenção do estado estimado. Isso não acontece nos métodos topológicos.

**Ilhas observáveis**

Como resultado da análise de observabilidade, tem-se o sistema observável (que pode estar dividido em ilhas) e o sistema não observável. No caso de o sistema ser apenas uma ilha, então a parte não observável é tratada como o sistema não monitorado (sistema externo). No caso de o sistema ser formado por Ilhas Observáveis, separados por Ilhas Não Observáveis, então há a necessidade de se combinar essas Ilhas para formar um único sistema. Essa tarefa pode ser feita por uma função específica ou então pode ser mais uma tarefa do fluxo de potência *on-line*. Na combinação dessas ilhas normalmente se utilizam medidas descartadas no processo de estimação de estado, medidas previstas (previsão de carga), valores históricos, etc.

# Capítulo 3

# Simuladores de Operação em Tempo Real

## 3.1 Introdução

Desde os anos 90, a indústria de energia elétrica tem encarado desafios na confiabilidade da operação e na economia da operação que são muito mais exigentes do que quaisquer desafios do passado. Além da complexidade dos sistemas de potência, mudanças diárias que ocorrem na rede tornam os tradicionais estudos de planejamento da operação quase sem utilidade, uma vez que eles são baseados em uma configuração específica do sistema de potência, geradores e dos equipamentos da rede [Vadar 95].

Como resultado das razões mencionadas acima, as concessionárias necessitam de ferramentas que irão ajudá-las a treinar seus despachantes a:

- Realizar melhor suas tarefas normais. Isto pode ser acompanhado através de:
  - Prática repetitiva
  - Reforço nos procedimentos operacionais
- Saber como o seu sistema reagiria caso ocorresse alguma emergência no sistema de potência. Alguns dos objetivos aqui são:
  - Como detectar se ocorreu uma emergência
  - Como tirar o sistema de potência da situação de emergência
  - Como prevenir a ocorrência de uma emergência

Uma outra motivação para desenvolvimento de programas computacionais de simulação de sistema de potência além da simulação da operação em tempo real é o de treinar

estudantes de engenharia, engenheiros e analistas de sistema de potência, fornecendo a eles uma poderosa ferramenta para um melhor entendimento da operação de sistema de potência [Over 95].

Dentro do universo dos simuladores de sistema de potência, um grupo considerado bastante significativo é o chamado simuladores de treinamento. O número de trabalhos publicados nesta área tem crescido ao longo dos anos. Várias propostas de desenvolvimentos de simuladores para treinamento vêm sendo apresentadas, tornando-se extremamente difícil realizar um estudo completo das propostas de forma a definir as mais adequadas e eficientes. Entretanto, pretende-se traçar um esboço das principais existentes vistas na literatura, de forma a dar uma idéia geral de como estão as pesquisas já desenvolvidas na área.

## 3.2 Simuladores de Treinamento - Uma breve introdução

A operação dos sistemas de potência atuais se tornou tão complexa que é praticamente impossível para um despachante/operador prever com exatidão os efeitos de possíveis ocorrências (por exemplo, contigências). A situação se apresenta sempre que a rede não está na sua configuração "normal", devido a chaveamentos de manutenção ou falha de algum equipamento. Por esse motivo simuladores como o Simulador de Treinamento de Despachante (DTS - *Dispatcher Training Simulator*) ou o Simulador de Treinamento de Operador (OTS - *Operator Training Simulator*) são considerados excelentes ferramentas que podem ser usadas para ensinar os despachantes/operadores a reagir sob essas condições, além de outras aplicações.

Para maior facilidade convencionou-se, neste trabalho, que os DTS's e OTS's serão chamados simplesmente de simuladores.

O fenômeno físico e dados do sistema de potência podem ser mais facilmente entendidos caso as informações sejam apresentadas aos usuários em janelas gráficas ao invés de formulários com tabelas numéricas. Muitos pacotes gráficos com a apresentação das saídas baseadas em sistemas de janelas (por exemplo Windows), foram desenvolvidas com o propósito educativo e de treinamento na área de sistema de potência [Chow 92, Li 93, Yu 95, Shin 99].

O simulador para treinamento é composto normalmente por três subsistemas maiores seguintes [Mill 93]:

- Modelo do centro de controle (MCC): É uma réplica do ambiente do centro de controle de energia. É a parte do simulador com a qual o *trainee* interage. O MCC é composto por várias áreas relacionadas de simulação. De modo a satisfazer o critério de exatidão do simulador, é necessário que uma parte substancial do MCC seja composta pelos mesmos componentes encontrados em um Computador de Centro

de Controle (CCC), definido como um conjunto inteiro de sistemas de *hardware* e *software* que juntos formam o sistema real de centro de controle do *trainee*. Inclui primariamente o sistema SCADA, a interface homem-máquina, controle automático de geração e aplicações de segurança de rede.

- Modelo do sistema de potência (MSP): Este subsistema simula os componentes da rede elétrica e as respostas dinâmicas do equipamento do sistema de potência. O MSP é composto por várias áreas relacionadas de simulação. A primeira e mais óbvia área concerne os modelos matemáticos da rede elétrica física, incluindo suas características estáticas e dinâmicas. Esta rede física inclui o sistema de potência controlado diretamente assim como os sistemas externos afetados indiretamente. O MSP deve ter a capacidade de agir tão bem quanto reagir com o *trainee*. É esta capacidade de reação que altera fundamentalmente as características dos modelos matemáticos de seus métodos normais de uso. As técnicas de modelagem devem ser expandidas para processar em um modo interativo através do monitor com o *trainee*. Na segunda área de simulação o MSP deve prover as necessidades de simulação do pessoal da operação que interage com o operador. Este pessoal inclui não somente o pessoal de campo que dá assistência na operação do sistema elétrico, mas também o pessoal envolvido com o sistema externo. Esta representação é parte da porção da posição do instrutor do MSP. A terceira área incluída no MSP é a simulação de dados coletados da rede elétrica e então apresentados ao *trainee* através de seu sistema de centro de controle. Esta área inclui a recepção e processamento de comandos de controle como iniciado através do sistema de centro de controle.

- Posição do instrutor (PI): Este subsistema usa programas e mostradores para monitorar e controlar as sessões de treinamento. Esta é a posição primária do instrutor que o permite ajustar os cenários de treinamento.

### 3.2.1 Objetivos do simulador

[Vadar 95] cita os seguintes objetivos para justificar o desenvolvimento de um simulador de treinamento:

- Objetivos operacionais

  Com a existência de um novo COS o despachante tem à disposição um novo conjunto de ferramentas para ajudá-lo na monitoração e controle da rede de potência. Contudo, o que o despachante também necessita é o treinamento correto para operar este novo sistema. As novas ferramentas devem permitir também ao despachante,

  1. controlar a situação para evitar um problema ou
  2. caso o problema ocorra, ajudar o despachante a controlar melhor a situação durante a duração do problema

- Objetivos de treinamento

  Uma vez que as novas capacidades das ferramentas analíticas de programação estejam entendidas e dominadas, é necessário treinar e periodicamente retreinar o despachante em possíveis cenários do equipamento afetado.

  Este tipo de treinamento não pode ser executado durante o trabalho diário com o COS operando. Contingências de porte elevado,

  - não ocorrem com freqüência suficiente para haver treinamento regular para os despachantes e
  - podem ser vistos por um despachante em um turno somente. Os outros despachantes podem precisar ser treinados neles.

  O simulador deve ter a capacidade de recriar os eventos do COS e permitir o treinamento dos despachantes.

- Objetivos de engenharia/análise

  No curso normal das atividades de operação existem papéis de suporte significativos onde o pessoal de engenharia atua para ajudar o despachante do sistema executar bem suas tarefas. São elas:

  - planejar a saída de equipamento. Usualmente ocorre um dos seguintes:
    * pré-planejamento do trabalho no sistema de transmissão que está para ocorrer no próximo dia ou
    * ajuda na avaliação da severidade de uma contingência que esteja ocorrendo no momento.

    O simulador é uma excelente plataforma para este propósito. Esta análise de engenharia fornece melhores resultados quando o estudo é feito utilizando-se um modelo de sistema de transmissão que inclua mudanças na topologia atual.
  - analisar a precisão do modelo do sistema de potência: esta tarefa tenta basicamente responder a questão - quão bom é o modelo que está sendo usado para a simulação? Precisão do modelo é fundamental para se obter resultados aceitáveis da simulação. Para responder isto, as concessionárias tentam planejar meios de "testar ou checar razoavelmente" o simulador contra eventos reais gravados.
    Se o COS pode gravar eventos reais quando acionado para tal fim, o simulador pode então ser inicializado para recriar o estado do sistema e então executar adiante no tempo para "simular" o evento real. Os resultados da simulação poderiam ser comparados então com os eventos reais gravados e os bancos de dados do simulador e modelagem da rede poderiam ser modificados caso necessário.

### 3.2.2 Aplicações e Benefícios

Aplicações que também podem ser incluídas no treinamento de despachantes/operadores:

- Restauração do sistema
- Serviços normais
- Situações emergenciais
- Novos equipamentos/procedimentos
- Estudo próprio do operador
- Outros estudos
- Introdução do COS a novos operadores
- Demonstração

Benefícios associados com o treinamento utilizando simulador:

- Treinamento da operação sob situações de emergência. O simulador propicia conhecimento prático durante condições restaurativas e de bleclaute; expõe *trainees* e operadores experientes a uma variedade maior de situações críticas.
- Ajuda novos operadores a adquirir conhecimento e habilidade.
- Treinamento pode ser feito de maneira regular dentro das atividades desenvolvidas pelo operador; desenvolvimento de comportamento e se necessário efetuar correção; demonstração de atuação como indivíduo e como equipe.
- O operador pode observar diferentes soluções para o mesmo problema e os riscos e benefícios de cada solução; testar os métodos para aliviar conseqüências de eventos.
- O simulador é uma ferramenta de reforço.
- Avalia a atuação do pessoal operador.
- Pode testar novos programas de computadores (*softwares*) e banco de dados.
- Treinamento para o uso de novo COS.
- Efetua demonstrações do sistema de potência para pessoal técnico assim como para o público.

### 3.2.3 Requisitos do Simulador

Uma vez que um sistema elétrico de potência comumente opera no seu estado normal seguro, algumas contingências simples podem levá-lo a operar numa região insegura, porém, controles preventivos adequados fazem com que o sistema retorne à região segura com certa tranquilidade. São raras as ocorrências que levam o sistema ao estado de

emergência, geralmente são causadas por contingências múltiplas graves. Neste estado, o sistema pode sofrer um colapso, onde grande parte do sistema interligado pode ser afetado, necessitando de controles de emergência e de restauração pelas ações integradas dos centros de controle das empresas afetadas, sendo primordial a intervenção correta por parte dos operadores. Por se tratarem de eventos que raramente ocorrem e pelas mudanças que estão acontecendo no setor elétrico, onde o enfoque principal é o rejuvenecimento de seu quadro funcional, pode-se afirmar que atualmente os operadores possuem pouca ou nenhuma experiência para enfrentar os estados de emergência e de restauração.

Uma vez que é inviável utilizar o sistema de potência real para criar esses problemas operacionais, o simulador se torna o meio mais eficaz para prover as oportunidades de treinamento necessárias. O uso primário do simulador é acelerar e aumentar a experiência do operador na operação do seu próprio sistema.

Uma vez que a faixa de exigência que podem ser especificadas para um simulador é numerosa e complexa, em [DyLi 83], por exemplo, os autores escolheram especificar duas exigências gerais:

- O simulador deve ser realístico e representar as propriedades estáticas e dinâmicas do sistema de potência do *trainee*.
- A interface homem-máquina deve ser exata da mesma maneira que o mostrador do monitor, os mesmos consoles e os mesmos controles que são usados num sistema real.

Essas exigências devem ser satisfeitas de modo a fornecer o treinamento necessário para preparar adequadamente o despachante para uma operação efetiva e eficiente. O simulador deve ser capaz de apresentar cenários para o ensino do básico de operação de sistema de potência, assistir o *trainee* em aprender como seu sistema responde a cada tipo de contingência e por quê ele responde de tal maneira. O simulador deve prover os meios de apresentar novos problemas ao operador, mesmo que não seja possível simular toda contingência e combinação de contingências que possam ocorrer no sistema de potência. O simulador deve ser capaz de operação realística a ponto de representar comunicações com o pessoal externo para o centro. Isto implica na necessidade de instalações para o instrutor para permitir a ele trabalhar com o *trainee*, servindo como um controlador do cenário e fazer uma avaliação posterior da sessão de treinamento. Finalmente, o simulador deve apresentar o mesmo sistema de potência. Se o *trainee* aprende procedimentos de operação de emergência em um sistema diferente ou genérico, não há garantias que ele irá reconhecer o mesmo problema quando apresentado em seu próprio sistema.

## 3.3 Exemplos de Simuladores

Ao longo dos anos os mais diversos tipos e modelos de programas computacionais de simuladores para treinamento foram desenvolvidos. Diversas foram as motivações e

objetivos. Diversos artigos têm sido publicados na literatura especializada, relatando experiências no uso de simuladores para treinamento. Alguns trabalhos significativos neste contexto serão brevemente comentados a seguir.

[Prais 89a] e [Prais 89b] são referências constantemente citadas em trabalhos publicados, em que são feitas considerações sobre os algoritmos de solução, sobre os modelos representativos dos componentes (modelo de carga, modelo de relé de proteção, modelo de transformador, modelo de gerador e modelo HVDC multiterminal) e são comentados resultados de testes realizados.

Em [Bucc 91] são apresentados resumos de considerações, cujo foco de atenção são as áreas de problemas que as concessionárias estão encontrando na utilização de simuladores de treinamento de despachantes como parte de programas de treinamento de despachantes. Cada resumo enfoca um aspecto diferente do treinamento de despachante e utilização de simulação de sistema de potência no currículo de treinamento. São feitas considerações sobre a justificativa do uso de um simulador de treinamento, são descritos problemas experimentados com o uso de um simulador, é desenvolvido uma série de critérios para estabelecer um conjunto de treinamento adequado para dar suporte no treinamento usando uma ferramenta de simulação de sistema de potência e finalmente tem-se uma explicação sobre os desafios em se estabelecer um meio que seja adequado para o treinamento de despachantes.

Em [Mill 93] tem-se a descrição do uso de um simulador, as configurações mais comuns de um DTS, as experiências no uso do DTS e a possível intensificação no uso do DTS de modo a facilitar o aprendizado. São apresentadas as estratégias e procedimentos de treinamento utilizadas, configuração utilizada, descrição do sistema modelado, características do DTS desenvolvido, considerações sobre o treinamento e experiência adquirida.

Em [Raja 94] tem-se o resumo do projeto e implementação do simulador avançado de treinamento de operador (OTS) baseado no OTS desenvolvido para o Electric Power Research Institute (EPRI). São feitas várias considerações sobre o OTS, os principais componentes que constituem um simulador para treinamento, sobre o modelamento do sistema de potência, os requisitos para entrada de dados, são feitas comparações com o OTS desenvolvido para o EPRI, a inicialização com os dados em tempo real é comentada, assim como a determinação do caso base a partir dos dados de inicialização. São feitas considerações também sobre a comunicação entre os subsistemas que compõem o simulador. Finalmente são relatadas as experiências que foram adquiridas com a implementação e utilização do simulador, tanto pelos responsáveis pela aplicação do treinamento, como pelos participantes das sessões de treinamento.

A motivação para o desenvolvimento do programa descrito em [Over 95] foi a de se ter um programa de simulação de sistema de potência próprio para instruir alunos de graduação e pessoal sem conhecimento técnico específico na área. Isto necessitou o uso de uma interface gráfica amigável com o usuário tal que os estudantes pudessem dispor de seu tempo ganhando um sentimento intuitivo para operar um sistema elétrico melhor do que

apenas aprendendo como usar o programa. Conceitos básicos precisaram ser apresentados de maneira simples, porém o programa necessitou detalhes suficientes para manter o interesse e proporcionar desafio ao estudante de engenharia avançado. A principal investida do programa inclui fluxo de potência básico em uma rede, como controles do sistema (tais como ponto de ajuste MW/tensão do gerador, transformadores LTC, saída de linhas e transações de potência) afetam o fluxo de potência e conceitos de controle de área tais como erro de controle de área, controle automático de geração e despacho econômico.

Em [Okap 96] são descritos o projeto e a operação de um protótipo de um pacote de programa de computador baseado em realidade virtual usado para treinar operadores. Demonstra também a exeqüibilidade do sistema de treinamento de operador baseado em realidade virtual na tentativa de direcionar esses resultados no ambiente do operador do sistema.

Em [Shin 99] os autores apresentam um pacote gráfico interativo baseado em Windows, desenvolvido para fins educativos e de treinamento de operação, análise e modelagem de sistema de potência. Possui interface gráfica amigável com o usário e sistema visual de gerenciamento de banco de dados. Podem ser citadas as seguintes características do pacote: editor gráfico para editar visualmente o diagrama do sistema de potência, banco de dados do sistema de potência baseado em Windows para gerenciar interativamente os dados, representações gráficas, esquemas de interrupção e animação. Os programas de aplicação incluídos no pacote desenvolvido são: cálculo de fluxo de potência, análise de estabilidade transitória, análise de faltas, despacho econômico e controle automático de freqüência e carga.

# Capítulo 4

# Simulador Proposto

## 4.1 Desenvolvimento

O simulador proposto apresenta duas partes distintas: uma que apresenta a interface gráfica, desenvolvida com a linguagem Tcl/Tk e outra que engloba as funções avançadas (aplicativos) de análise de rede com os programas desenvolvidos na linguagem Fortran. A interação entre as partes é feita através de arquivos de dados que são compartilhados e frequentemente atualizados. Para o desenvolvimento do programa principal, escrito em Tcl/Tk no ambiente Unix, das estações de trabalho Ultra 1, da Sun Microsystems, que deu origem ao simulador de treinamento para sistemas de potência pretendido, foram seguidas várias etapas que são descritas a seguir.

- Definição do sistema a ser estudado

  Optou-se por um sistema fictício composto por quatro subestações interligadas de forma a compor um sistema fechado. O sistema exemplo foi escolhido de modo a poder expor didaticamente os objetivos propostos. Embora o sistema base para o desenvolvimento seja um sistema fictício, de dimensões reduzidas, todo o programa foi escrito de forma a ser facilmente expandido ou adaptado para outro de dimensões diferentes, bastando a criação de arquivos em formato adequado e atualizar o número de subestações.

- Determinação da topologia física de cada subestação

  Cada subestação foi especificada com um nível de detalhamento elevado, com as barras existentes (operação, manutenção e transferência, dependendo do caso), os disjuntores, as cargas, representação de transformadores e geradores e ainda, sua topologia.

- Identificação dos *circuitos* de cada subestação

  A identificação dos *circuitos* em cada subestação foi o passo seguinte. Definiu-se

como *circuito* todo elemento que não seja chave seccionadora ou disjuntor. Fazem parte dessa denominação de *circuito*: as barras, as linhas de transmissão e os taps de transformadores de cada subestação. A definição dos circuitos de cada subestação é feita uma única vez e não foi seguida regra alguma de formação, com exceção dos números dos circuitos que foram seqüenciais para cada subestação. Caso haja alguma modificação posterior em cada uma das subestações, deve ser efetuada a revisão dos circuitos existentes, acrescentando ou retirando circuitos.

- Elaboração dos arquivos de dados

  A elaboração dos arquivos de dados foi a etapa seguinte. Foi feita a opção de que para cada subestação existiria arquivos de dados correspondentes, com as informações necessárias.

## 4.2 Ferramenta de auxílio

Durante o desenvolvimento do diagrama unifilar das subestações e do sistema geral, foi utilizada a ferramenta gráfica Visual Tcl [Alle 97] para auxiliar na correta alocação dos elementos na tela de saída. vTcl é um ambiente de desenvolvimento de aplicação, para plataformas Unix, Windows e Macintosh, disponível aos usuários sem custo. Foi escrito inteiramente em Tcl/Tk e gera código puramente em Tcl/Tk. Suas características são[Macd 97]:

- não necessita bibliotecas externas, utilizando somente Tcl/Tk;
- importa códigos Tcl/Tk pré-existentes;
- interface gráfica para a maioria dos aspectos do desenvolvimento do Tcl/Tk;
- suporte extensivo para gerenciador de geometria e *widget*.

## 4.3 Arquivos de Dados

Os bancos de dados são as bases onde ficam armazenadas as informações sobre o sistema elétrico, requeridas pelas funções de análise de rede. No presente trabalho, tem-se também arquivos de dados necessários para a parte de interface gráfica ao usuário com as informações exigidas pela linguagem Tcl/Tk. Alguns arquivos são compartilhados pelas funções avançadas escritas em Fortran e pelo programa desenvolvido em Tcl/Tk. A elaboração do banco de dados é uma etapa importante que requer que sejam tomados certos cuidados. Para as funções de análise de rede, são necessários basicamente dois tipos de banco de dados: estático e dinâmico. O banco de dados estático recebe este nome porque armazena informações que possuem baixa freqüência de atualização, como

é o caso dos parâmetros de linhas e de transformadores. Os dados de alta freqüência de atualização são armazenados em um banco de dados dinâmico, como é o caso de valores de magnitude de tensão, fluxo de potência em linhas de transmissão, estados de chaves e disjuntores (aberto/fechado).

Os bancos de dados em tempo real são constituídos pelos dados descritivos do sistema, em conjunto com os modelos da rede e os resultados obtidos pelas várias funções e têm por objetivo final atender os requisitos de operação do sistema.

No presente trabalho, os arquivos de dados foram divididos nas seguintes categorias:

- arquivos com dados topológicos
- arquivos com dados dinâmicos
- arquivos com dados auxiliares
- arquivos com dados temporários

Os arquivos topológicos foram por sua vez subdivididos em:

- arquivos com os dados topológicos de configuração do sistema e das subestações que fazem parte do sistema
- arquivos com os parâmetros elétricos dos componentes das subestações

Na figura 4.1 pode ser vista de maneira simplificada o fluxo dos dados entre os programas de interface gráfica (Tcl) e as funções de análise de rede escritas na linguagem Fortran. As setas indicam o caminho permitido para o fluxo dos dados.

Tem-se arquivos de dados exclusivos para as rotinas Tcl, assim como arquivos exclusivos para os aplicativos de rede. Tem-se também arquivos de dados que são compartilhados. As rotinas Tcl geram arquivos de dados que servirão como dados de entrada para as rotinas de análise de rede. Os resultados das rotinas de análise de rede, por sua vez são armazenados em arquivos de dados que após as devidas manipulações para adequação de formato, servirão de dados de entrada para o programa de interface gráfica Tcl.

A figura mostra também que o usuário tem permissão de atuar nos conjuntos de arquivos de dados denominados: dados topológicos, dados dinâmicos e dados auxiliares. Essa atuação pode ser de maneira direta, ou seja, nos próprios arquivos de dados, ou então indiretamente, isto é, o usuário atua através da interface gráfica. Deve-se observar que o próprio programa reconhece quando as modificações são efetuadas via teclado, encarregando-se de atualizar devidamente os arquivos de dados modificados.

A seguir é feita a descrição dos arquivos de dados criados para cada subestação.

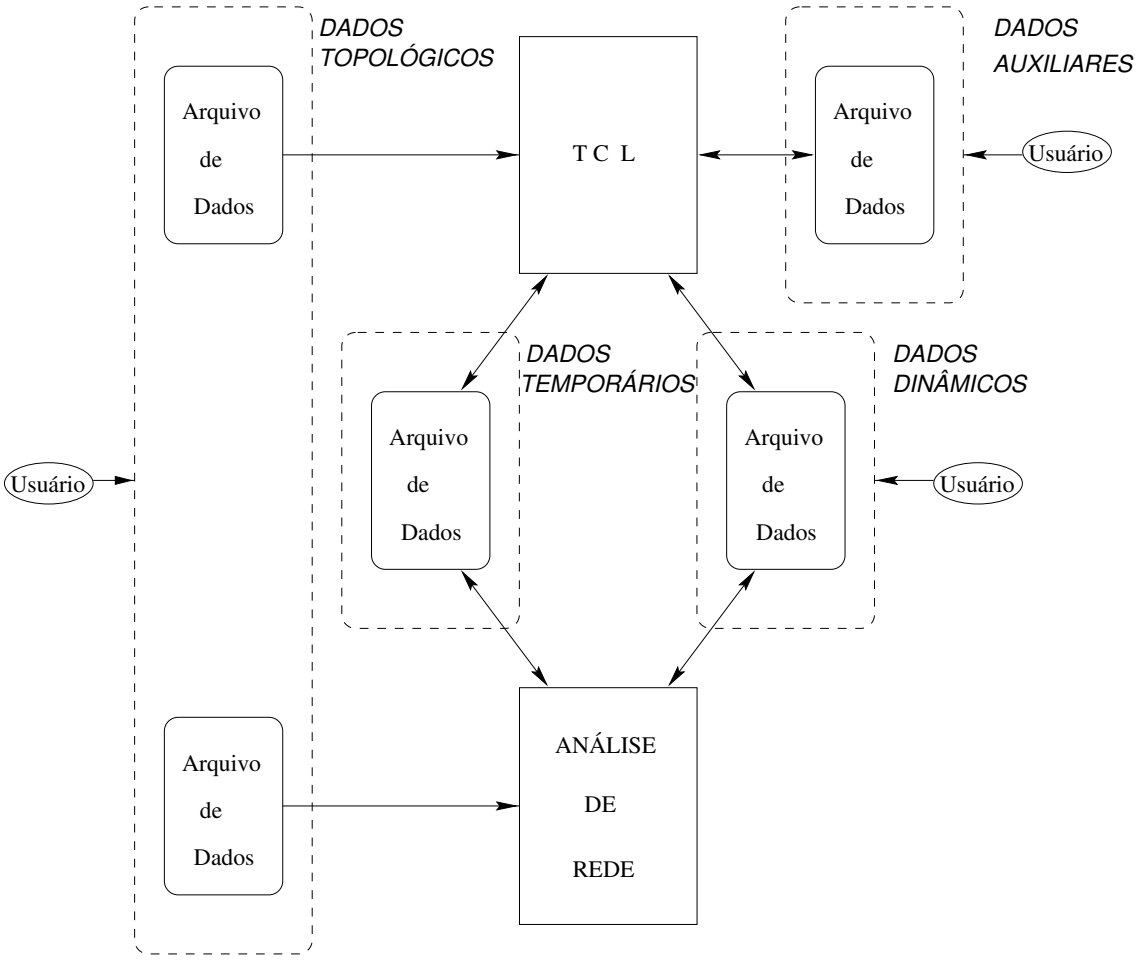


Figura 4.1: Fluxo dos Dados

### 4.3.1 Arquivos com Dados Topológicos

Neste conjunto de dados se encontram os dados necessários para a configuração física dos diagramas unifilares das subestações, do sistema geral e dos elementos que compõem as subestações. De acordo com o tipo do elemento a ser representado na tela de saída, são fornecidos os dados adequados. Para maior detalhamento na elaboração dos arquivos de dados ver Apêndice B. Os tipos de elementos podem ser:

- Elementos que são desenhados:
  - linha de transmissão
  - barramento
  - textos em geral
  - gerador
  - transformador
- Elementos com variáveis associadas:
  - tensão
  - fluxo ativo
  - fluxo reativo
  - geração de potência ativa
  - geração de potência reativa
  - carga ativa
  - carga reativa
  - tap de transformador
  - disjuntor
- Arquivos com especificações dos parâmetros elétricos dos elementos

  Várias informações que serão utilizadas tanto pelos aplicativos quanto pelo programa Tcl/Tk estão contidas nestes arquivos. Os grupos de informações são separados por indicadores pré-definidos. Os grupos são:
  - circuitos e elementos associados

    Os circuitos definidos estão aqui especificados. Cada circuito está associado a um elemento que de acordo com o tipo deve possuir determinadas especificações, que descrevem os parâmetros necessários para obter o desempenho elétrico da rede. Os tipos possíveis e as informações necessárias são:
    - linha de transmissão:
      - nível de tensão

        - resistência da linha
        - reatância da linha
        - admitância shunt da linha
        - limite máximo
    * barramento:
        - nível de tensão
        - identificação da variável associada
    * segmento:
        - nível de tensão
    * transformador:
        - resistência
        - reatância
        - tap nominal
        - tap mínimo
        - tap máximo
        - limite ativo máximo
    * gerador:
        - nível de tensão
        - reatância do gerador
        - potência ativa máxima
        - potência reativa mínima
        - potência reativa máxima
        - identificação do gerador
    * carga:
        - nível de tensão
        - carga ativa
        - carga reativa

- disjuntores e circuitos associados

  Cada disjuntor deve possuir:

    * os números dos circuitos associados antes e após o disjuntor
    * o nome da variável que representa cada disjuntor
    * os números dos circuitos

- variáveis do sistema

  Neste grupo, de acordo com o tipo da variável existe um indicador associado. As informações necessárias são:

    * tipo (V, MW ou MVAr)
    * valor medido ou valor calculado
    * nome da variável
    * nome da janela de saída do valor da variável

* número do circuito associado
* valor da variável

Uma vez que os arquivos contendo os dados acima descritos são do tipo texto, cuja formação detalhada pode ser verificada no Apêndice B, nada impede que seja desenvolvida no futuro uma rotina que crie automaticamente esses bancos de dados de maneira desejada pelo usuário. A preocupação com a modularização dos arquivos, de modo a facilitar a posterior alteração de arquivos, foi uma constante durante o desenvolvimento do programa proposto.

### 4.3.2 Arquivos com Dados Dinâmicos

Os arquivos que contém dados dinâmicos do sistema são aqueles que são criados ao longo da simulação, pelas próprias rotinas do programa Tcl/Tk. A formação desses arquivos pode ser feita de duas maneiras:

- os dados são obtidos a partir das telas de saída de cada subestação

  Neste caso os dados representam o estado atual de cada variável que podem ter sido modificados ou não pelo usuário.

- os dados são resultantes dos cálculos efetuados pelos programas de cálculo do fluxo de carga ou estimador de estado.

Durante a elaboração dos arquivos de dados, optou-se pela criação de arquivos para cada subestação que compõe o sistema em estudo, visando facilitar novas inclusões ou retiradas futuras, ver Apêndice B.

Os arquivos com os dados dinâmicos que representam o estado atual das variáveis de cada subestação é composto pelas seguintes informações:

- estado dos disjuntores (aberto ou fechado)
- tensão nas barras
- fluxo de potência ativa nos linhas de transmissão
- fluxo de potência reativa nas linhas de transmissão
- potência ativa dos geradores
- potência reativa dos geradores
- consumo ativo das cargas

- consumo reativo das cargas
- posicionamento dos taps dos transformadores

No caso onde os arquivos de dados são formados a partir dos resultados dos cálculos efetuados, é feita uma seleção dos dados através de rotinas (*procedures*) do próprio programa principal escrito na linguagem Tcl, para cada subestação, os dados necessários são devidamente tratados, podendo ser mostrados diretamente na tela de saída ou copiados em arquivos correspondente a cada subestação para uso posterior.

A principal característica dos arquivos de dados chamados dinâmicos é a de que a cada nova simulação esses arquivos são criados e apagados, podendo então ter seus valores modificados durante a execução da simulação.

### 4.3.3 Arquivos com Dados Auxiliares

Neste caso estão incluídos os arquivos que são criados durante a simulação, onde o usuário efetua o salvamento de tantas quantas forem as configurações que julgar necessárias para estudo posterior. No momento que o usuário desejar o caso em estudo pode ser guardado, podendo ser acessado novamente depois a qualquer instante.

Neste tipo de arquivo tem-se também os arquivos que representam as curvas de cargas das injeções que fazem parte do sistema em estudo. Esses arquivos devem ser devidamente preparados e adequados para a utilização. Podendo, portanto ser modificados a qualquer momento pelo usuário. A figura 4.2 mostra um exemplo de curvas de carga utilizadas para compor o banco de dados. Tomou-se como exemplo as curvas de cargas das injeções ativas e reativas da subestação 2 em uma quarta-feira, detalhes ver Apêndice B.

### 4.3.4 Arquivos com Dados Temporários

Durante a simulação alguns arquivos de dados são criados temporariamente para utilização em determinado momento, não sendo mais necessários após uso. Como a criação deste tipo de arquivo está diretamente relacionada com a execução do programa de simulação, o usuário não tem permissão de modificar esses dados, que são criados/destruídos ao longo da simulação, como mostra a figura 4.1.

A cada novo início de uma determinada simulação esses arquivos são devidamente preparados. Neste tipo de arquivo podem ser citados como exemplos os arquivos que fazem a interface entre o programa escrito em Tcl e os programas de análise de rede escritos na linguagem Fortran.

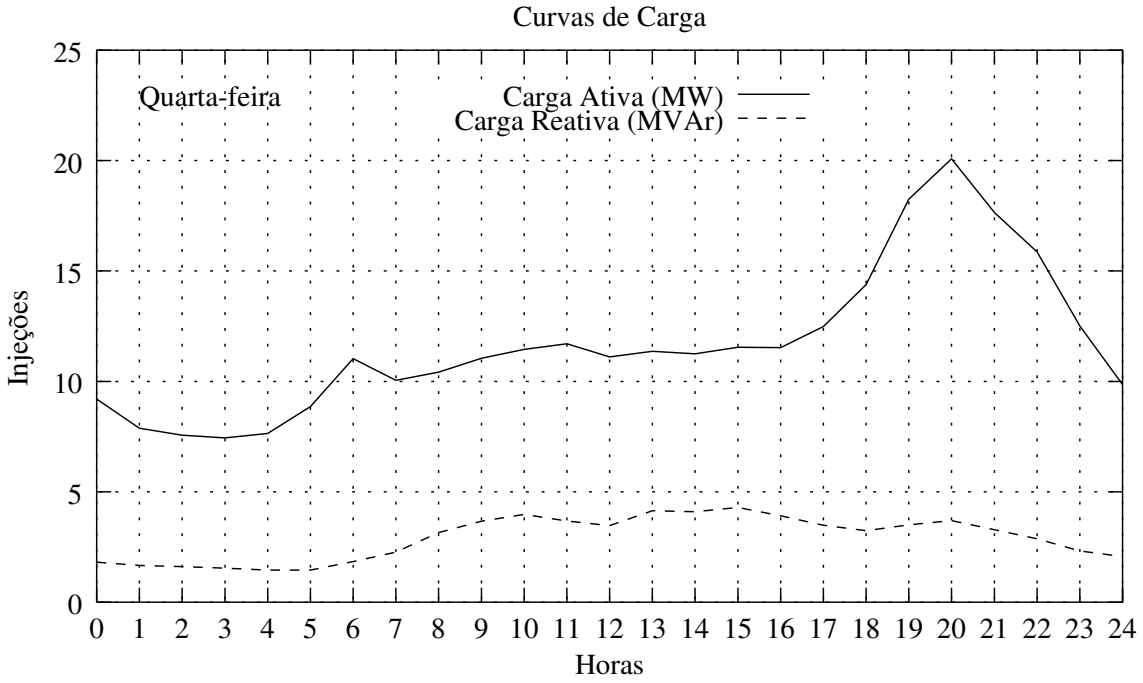


Figura 4.2: Curvas de Carga - Subestação 2 - Quarta-feira

## 4.4 Algoritmo Básico de Solução do Simulador

A figura 4.3 mostra de maneira simplificada o fluxograma de solução do simulador que foi desenvolvido. Cada bloco é detalhado nas seções seguintes.

O simulador desenvolvido segue os seguintes passos:

- Parte 1: Inicialização
  - são inicializadas as variáveis que fazem parte do sistema - assim como qualquer linguagem de programação, Tcl exige que as variáveis sejam inicializadas para um correto andamento da simulação.
  - preparação dos arquivos de dados - os arquivos de dados dinâmicos e os arquivos temporários remanescentes de simulações anteriores caso existam são apagados e a seguir criados, porém vazios, ficando prontos para a nova simulação.
  - chamada da rotina com a configuração inicial - escolheu-se como configuração (estado dos disjuntores) inicial, uma dada configuração qualquer, onde todas as subestações que compõem o sistema em estudo têm definidos os estados de todos os seus disjuntores. Servindo, portanto como ponto de partida para a simulação.
  - ajuste do tempo $t$ para as curvas de carga - cada injeção do sistema, tanto ativa como reativa foi representada no presente trabalho, através de funções que

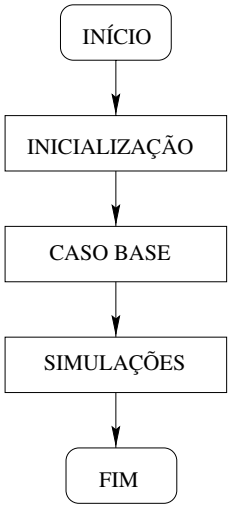


Figura 4.3: Fluxograma de solução do simulador

representam o comportamento das cargas em função do tempo. Esses dados foram obtidos junto à concessionária de energia elétrica local e representam dados reais obtidos ao longo do ano. O usuário determina nesta fase em qual instante $t$ deseja iniciar os valores das injeções.

- Parte 2: Definição do Caso Base

  É necessário que o usuário efetue:

  - Definição das injeções de potência em geradores e cargas. Neste ponto as injeções de todos os geradores e cargas devem ser definidas, com exceção do gerador escolhido para efetuar o balanço de potências.
  - Escolha da topologia da rede. Define-se o estado (aberto ou fechado) de todas as chaves/disjuntores das subestações que fazem parte do sistema em estudo.

  De posse das informações acima, a função de análise de redes configurador de topologia de rede é executada. Tem-se como resultado a rede elétrica atual. A seguir utilizando-se como medidas as tensões conhecidas e as injeções de potência que foram fornecidas pelo configurador, executa-se o analisador de observabilidade que vai permitir a obtenção de uma rede (e também os seus dados), que pode estar dividida em ilhas. Uma vez que em cada uma dessas ilhas exista geração e carga, um fluxo de potência pode ser executado para cada uma delas. Os valores gerados (fluxos ativos e reativos nas linhas e tensões nas barras) servirão de *medidas exatas* para a Parte 3 seguinte.

- Parte 3: Simulações

  Tomando como ponto de partida os dados obtidos na Parte 2, o usuário pode modificar as medidas efetuadas, adicionando erros aleatórios ou simulando erros grosseiros (erros de medição ou de topologia). Feitas as modificações, é acionada a função

estimador de estado que fornece os valores calculados do sistema. O usuário pode escolher entre visualizar os dados medidos ou os resultados do cálculo da estimação de estado.

## 4.5 Inicialização do simulador

No momento em que o usuário inicia o programa simulador, várias rotinas são executadas antes que a primeira tela de saída seja apresentada ao usuário. Essas rotinas são as de configuração inicial e envolvem vários comandos pré-estabelecidos.

A figura 4.4 mostra o fluxograma da primeira etapa do simulador, onde é feita a inicialização do programa. Assim que o usuário aciona o programa (INÍCIO), os passos seguintes são os descritos abaixo.

Tem-se:

- a inicialização das variáveis definidas para o programa Tcl
- a chamada das rotinas que desenham na tela de saída a representação esquemática do sistema em estudo
- determinação da configuração topológica inicial de cada subestação, isto é, estado dos disjuntores na posição aberta ou fechada
- preparação dos arquivos de dados dinâmicos

  Esses arquivos devem ser criados caso, ou caso já existam devem ter seus valores remanescentes zerados para não acarretar erros no decorrer da simulação.
- determinação do ponto de inicialização nas curvas de carga das injeções do sistema

  Optou-se por adequar o horário da estação de trabalho (dia da semana e hora) com o instante inicial da simulação. Os arquivos de dados contendo as informações das curvas de cargas estão separadas por dia da semana para cada conjunto de injeções. Para cada injeção foi determinado o fator multiplicador que rege o comportamento das curvas. Os valores foram então separados em valores percentuais. Isso visando facilitar alterações nesses valores, isto é, alterando somente o fator multiplicador para um valor desejado, com isso toda a curva sofrerá a modificação, sem contudo modificar o comportamento da curva. Um programa que efetua a interpolação linear entre dois pontos determina o instante desejado na curva de carga. Como na linguagem Tcl os números que começam com o digito zero são interpretados como valores octais (ou hexadecimais, caso o zero seja seguido imediatamente por um valor "x"), houve a necessidade de se criar uma rotina que fizesse a conversão de dados que porventura fossem iniciados por zero (hora, minuto e segundo).

  – determinação do dia da semana para escolha adequada do arquivo de dados

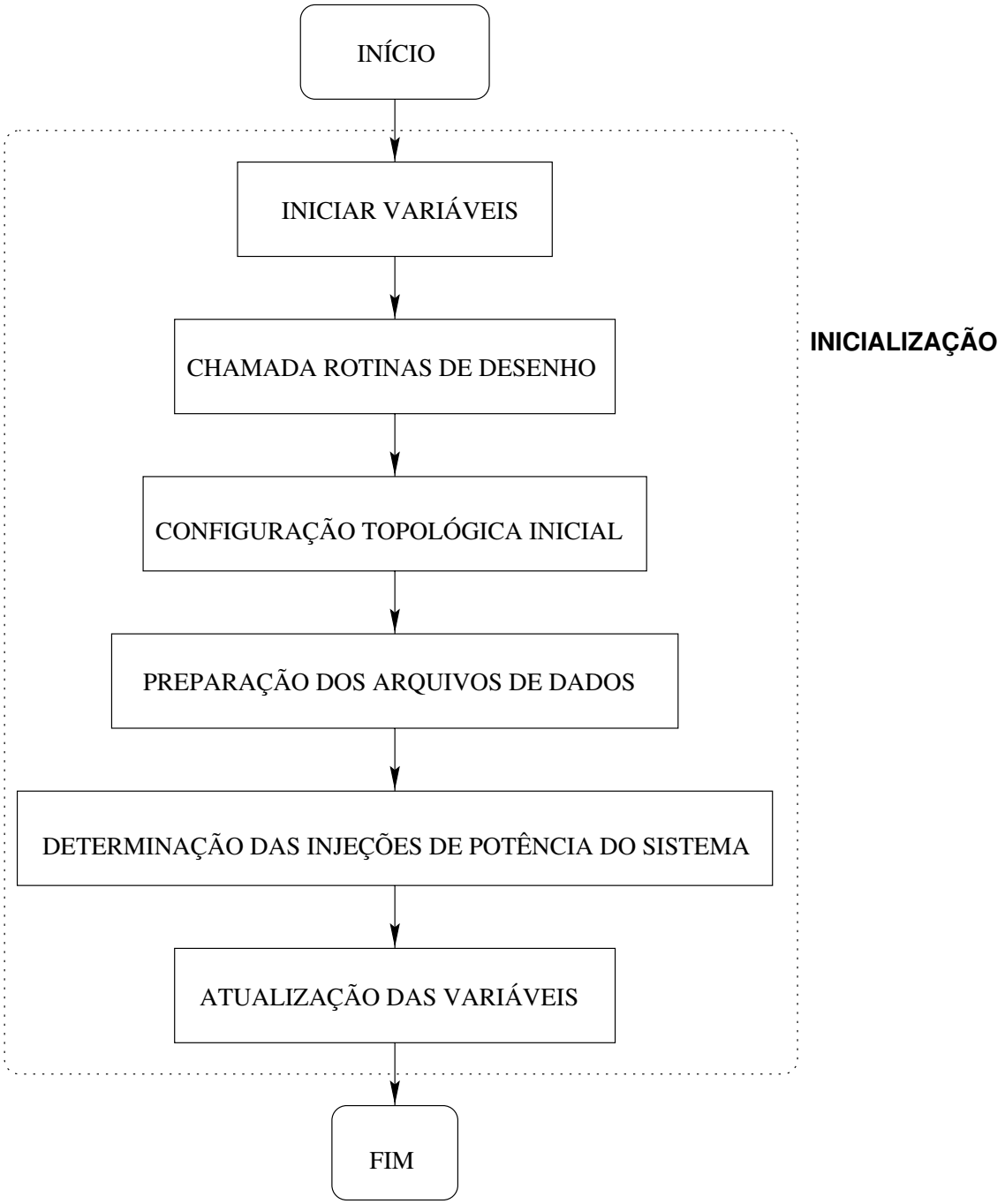


Figura 4.4: Etapa 1: Inicialização do simulador

  - determinação da hora (hora, minuto e segundo) local da estação de trabalho
  - conversão da hora em décimos de hora.

    Por exemplo: 10h 25m 08s equivale a 10,42 h. Esse será o instante t que será utilizado para determinar a partir das curvas de carga, os valores das injeções. A rotina que efetua a interpolação linear determina qual injeção equivale ao instante t desejado.

- atualização dos valores das variáveis correspondentes às injeções do sistema

  A atualização dos valores das variáveis correspondentes às injeções do sistema que foram determinados na rotina anterior é feita em duas etapas: primeiro um arquivo temporário é atualizado (arquivo *recupera*) com os valores determinados; a seguir esse arquivo é lido e as telas de saída são atualizadas. Finalmente os arquivos de configuração (arquivos *config*) são atualizados com esses valores de tela.

A seguir tem-se então mostrada ao usuário a tela geral do sistema com os valores iniciais de todas as injeções iniciais adequadas de acordo com a hora local. Este ponto na figura 4.4 corresponde ao bloco FIM do fluxograma, ou seja, final da etapa de inicialização e no programa de simulação corresponde à primeira tela de saída que é apresentada ao usuário, quando o programa é iniciado através do comando *wish nome_do_programa_principal* digitado no teclado da estação de trabalho.

## 4.6 Caso Base

Após a etapa Inicialização, executada pelo usuário ao se iniciar o programa de simulação, tem-se agora a etapa denomidada Caso Base. O comando Calcular Caso Base associado ao botão correspondente, é composto por um conjunto de rotinas envolvendo tanto procedimentos Tcl quanto acionamento de rotinas de análise de rede.

Inicialmente ao se ter acesso à primeira tela de saída do programa simulador, que no presente caso, mostra o diagrama unifilar do sistema geral em estudo, composto por todas as subestações interligadas, o usuário pode caso desejar, efetuar modificações na configuração topológica das subestações, ou seja, abrir e/ou fechar disjuntores em uma ou mais subestações. Lembrando-se que na etapa Inicialização, o sistema partiu de uma configuração pré-determinada para cada subestação. Todos os outros valores das variáveis disponíveis (injeções de potência ativa e reativa, magnitudes de tensão nas barras e valores de taps de transformadores) também podem ser modificados pelo usuário. Esta sub-etapa foi denominada Modificações (topológicas e de valores). Essas modificações são efetuadas diretamente na tela de saída através da atuação conjunta no *mouse* e teclado. Na figura 4.5 essa sub-etapa pode ser vista no primeiro bloco tracejado do fluxograma.

Deve-se notar que a figura apresentada relaciona-se somente com a etapa de cálculo do caso base, sendo que o bloco INÍCIO é na verdade a saída da etapa INICIALIZAÇÃO,

descrita anteriormente. Por conseguinte FIM representa o final desta etapa, mas não significa o final da simulação.

Feitas as modificações na topologia da rede e nos valores das variáveis,tem-se a seguir as ações que são executadas após o acionamento do Botão de comando: Calcular Caso Base. Para que os resultados do caso base sejam exibidos na tela de saída para o usuário, a seqüência de passos que o programa executa são:

- ajuste de variáveis sinalizadoras (*flags*), que definem importantes pontos do programa desenvolvido. Pode-se citar como exemplo a variável *tipo*, que é um localizador podendo assumir os valores "m" ou "c". Tudo relacionado com o cálculo do caso base convencionou-se definir como tipo "m" e no momento em que está se tratando do cálculo do estimador de estado, como tipo "c".
- chamada da rotina que converte os arquivos de dados de configuração topológica, para os arquivos dinâmicos que serão passados como arquivos de entrada de dados para as rotinas de análise de rede.
- atualização dos arquivos de dados com os valores atuais de tela de todas as subestações.
- com o objetivo de apagar os valores remanescentes das variáveis, evitando assim a inclusão de erros nos cálculos, foi desenvolvida uma rotina que apaga os valores, chamados de "flutuantes" das variáveis, ou seja, não os valores dos arquivos de dados, mas os que estão em uso corrente durante a simulação.
- chamada das rotinas de análise de rede, escritas na linguagem Fortran (já no formato executável) que efetuarão os cálculos propriamente ditos. Na figura 4.5 é representada pelo segundo bloco tracejado, análise de rede.
  - chamada da rotina configurador de rede, que efetua a configuração elétrica do sistema a partir das informações fornecidas dos estados das chaves/disjuntores.
  - chamada da rotina cálculo do fluxo de carga para o sistema desejado.
- armazenamento dos resultados dos cálculos em arquivos de saída.
- conversão das informações para um formato adequado em arquivos de dados, para que sejam acessados a seguir pelo programa Tcl.
- leitura das informações dos arquivos de dados pelo programa Tcl e saída desses dados nas telas de interface ao usuário.

## 4.7 Cálculo do Estimador de Estado

Assim como no caso do cálculo do caso base descrito acima, também no caso do cálculo do estimador de estado, tem-se vários comandos associados para que a execução

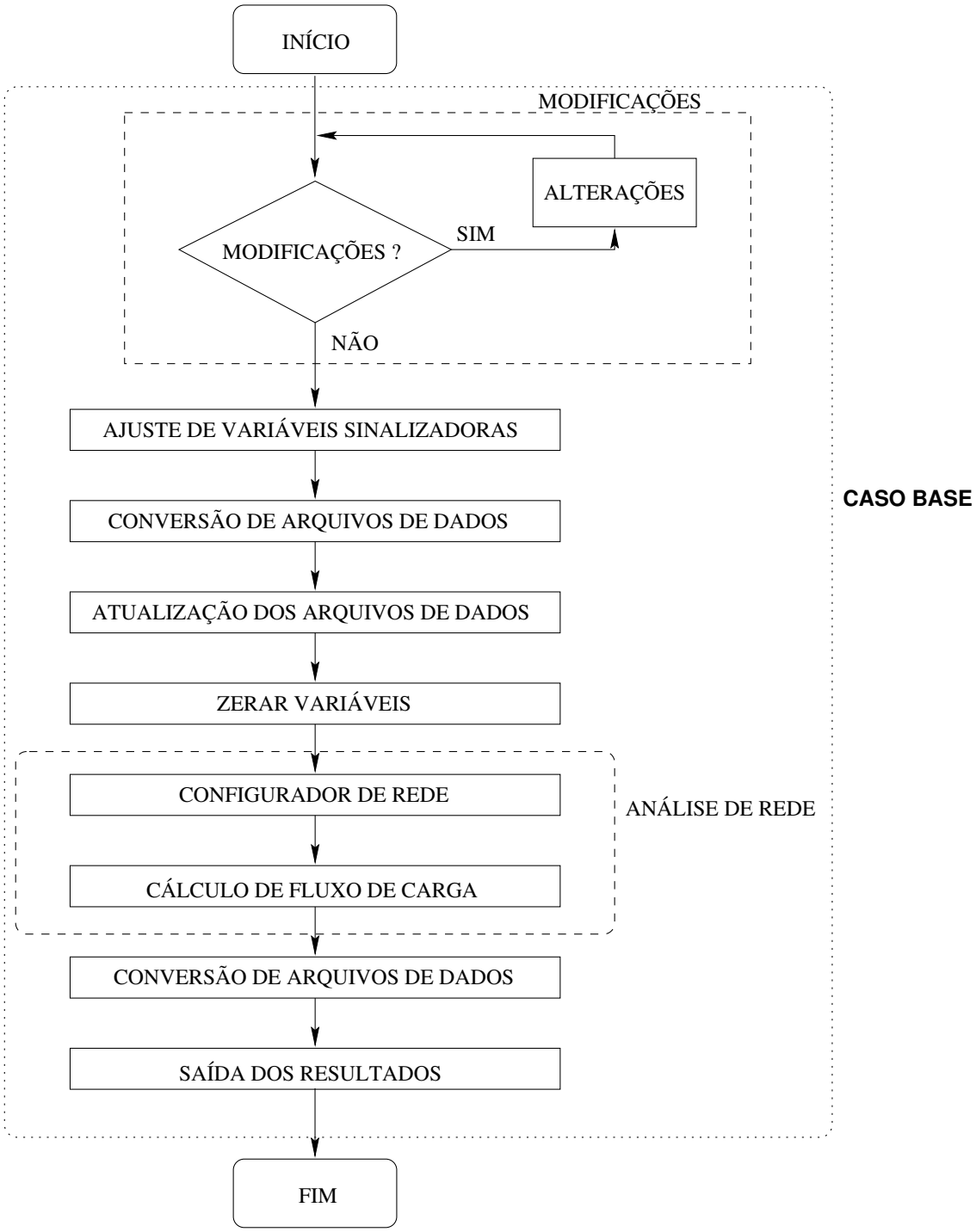


Figura 4.5: Etapa 2: Cálculo do caso base

do comando possa ser realizada.

A figura 4.6 mostra um fluxograma simplificado com a seqüência de comandos e ações que são executados na etapa denominada de cálculo do estimador de estado. Cabe lembrar novamente que o fluxograma é parcial referindo-se somente a esta etapa do programa. Portanto INÍCIO na figura indica na realidade o fim da atapa anterior, no caso cálculo do caso base.

A seguir tem-se a descrição da seqüência de passos que é executada nesta etapa:

- Verificação de condições

  Em primeiro lugar é feita uma verificação de uma variável sinalizadora. Caso o valor assumido por essa variável seja tal que não satisfaça a condição requerida, o usuário é informado através de uma caixa de diálogo que a operação que deseja realizar é inválida neste momento devendo, portanto, realizar antes as ações corretivas necessárias. Caso satisfaça a condição requerida prossegue-se.

- É feito um ajuste das variáveis sinalizadoras. São checadas duas variáveis sinalizadoras. Dependendo do valor assumido os procedimentos diferem.

  - Variável sinalizadora *tipo*

    Se a variável sinalizadora "tipo" tem seu valor ajustado em "m", o procedimento a ser seguido é o mesmo que quando assume o valor "c" até o momento onde ocorre a SAÍDA DOS RESULTADOS, parte inferior do fluxograma da figura 4.6. A mudança acontece no momento em que após calculados os resultados do estimador de estado, não são chamadas as telas das subestações e sim os valores das variáveis são somente atualizados, ou seja, não haverá a troca de tela de saída mas somente os valores modificados serão atualizados. O objetivo disso foi o de diminuir o tempo de simulação e tornar um pouco mais agradável a visualização final das telas de saída.

  - Variável sinalizadora *paginaatual*

    Esta variável sinalizadora determina qual foi a última tela acessada pelo usuário durante a simulação. Isto para que os resultados a serem mostrados correspondam a essa mesma tela, ou seja, quando o botão para cálculo do estimador foi acionado a partir da tela dos valores medidos, essa mesma subestação ou a tela com o sistema geral é a que será mostrada, agora com os valores estimados calculados.

- É feita a conversão dos arquivos de saída do cálculo do caso base para os arquivos dinâmicos que servirão como dados de entrada para o estimador.

- Atualização dos arquivos de dados de acordo com os valores atuais de tela das variáveis do sistema.

- Os arquivos temporários são apagados, para que as variáveis não possuam valores remanescentes que possam causar incorreções nos cálculos a serem efetuados.

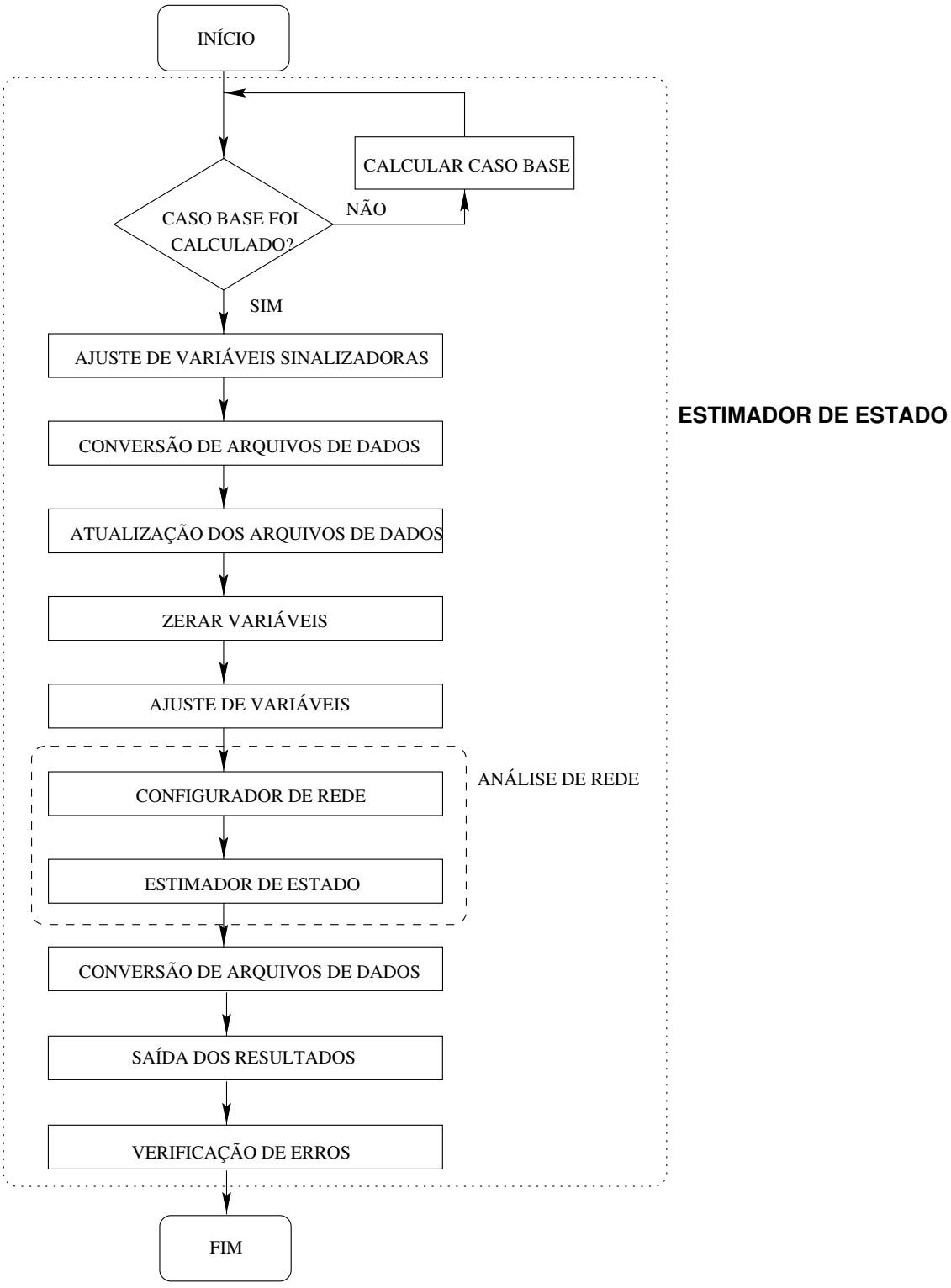


Figura 4.6: Etapa 3: Cálculo do estimador de estado

- É feito ajuste de variáveis que serão dados de entrada das funções de análise de rede.
- Chamada das funções de análise de rede
  - Configurador de Rede, efetua a configuração elétrica do sistema a partir das informações fornecidas dos estados das chaves/disjuntores.
  - Estimador de Estado, rotina que efetua os cálculos da estimação de estado do sistema em estudo.
- É feita a adequação dos arquivos de saída resultantes dos cálculos efetuados pelas funções de análise de rede para os arquivos que serão lidos e manipulados pelo programa Tcl.
- Apresentação dos resultados estimados nas telas de saída ao usuário.
- Verificação se nos resultados obtidos pelo estimador não houve a ocorrência de erros grosseiros ou aleatórios. Caso sim, no momento em que esse fato ocorreu, as variáveis cujos valores estão fora da faixa aceitável, terão seus valores destacados com a cor vermelha nas telas de saída.

A última verificação efetuada é com a variável sinalizadora que bloqueia ou permite o aparecimento do botão cujo comando associado é o de execução automática.

## 4.8 Opções de Simulação

O usuário tem a seu dispor no programa desenvolvido as seguintes opções de simulação: Módulo Estudo e Módulo Execução Automática.

As figuras 4.7 e 4.8 mostram os fluxogramas simplificados dos módulos Estudo e Execução Automática que serão descritos detalhadamente a seguir.

### 4.8.1 Módulo Estudo

A principal característica deste módulo é o de possibilitar ao usuário efetuar diversos tipos de acionamentos no sistema. Neste módulo, com as condições desenvolvidas e apresentadas, o usuário tem a possibilidade de se exercitar no que diz respeito ao item modificações na configuração topológica do sistema. Este módulo pode ser classificado também como um módulo de aprendizagem. Os passos a serem seguidos neste módulo são:

- Inicialização:

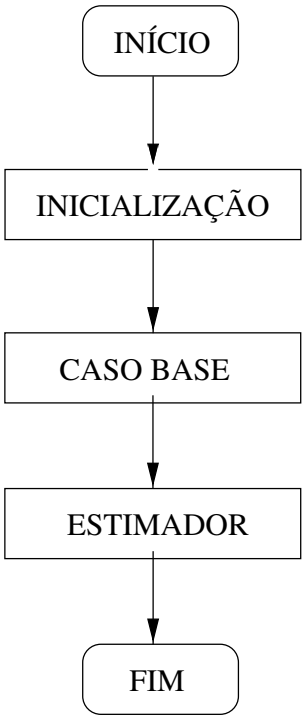


Figura 4.7: Módulo Estudo

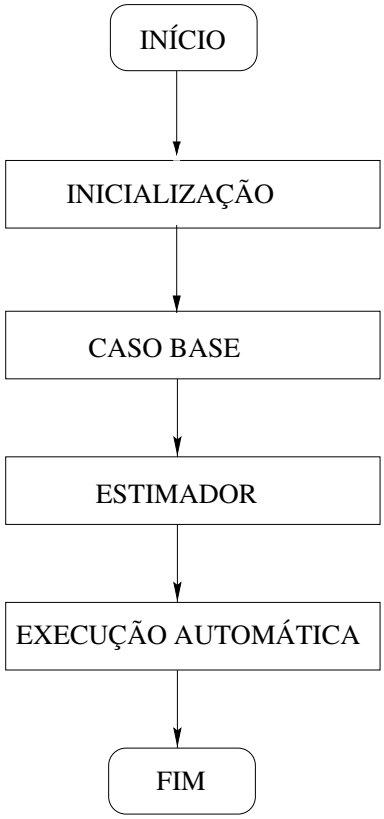


Figura 4.8: Módulo Execução Automática

Esta etapa é a mesma para todos os módulos. O sistema se inicializa com as condições iniciais previamente determinadas. O sistema mostrado na tela de saída ao usuário é o sistema unifilar geral com todas as subestações interligadas.

- Modificações na topologia e valores das variáveis:

  Neste ponto o usuário pode efetuar todas as modificações desejadas, tanto de configuração topológica (abrindo ou fechando chaves/disjuntores), quanto de valores de injeções de potência ativa e reativa, valores de taps de transformadores e níveis de tensão nas barras.

- Cálculo do caso base:

  A seguir o usuário aciona o botão correspondente à execução do caso base. Com as informações contidas nas telas de cada subestação: estado das chaves/disjuntores, injeções ativas e reativas, níveis de tensão e taps de transformadores, são acionadas as rotinas: Configurador da Rede Elétrica e Cálculo do Fluxo de Carga da situação atual. Todas as informações atuais de tela são devidamente armazenadas em arquivos com denominações e formatos adequados, que servirão como arquivos de entrada para os cálculos.

  Os resultados calculados são devidamente mostrados ao usuário nas telas de saída. Neste ponto vale ressaltar que os valores mostrados nas telas de saída são aqueles

que foram previamente definidos como *valores medidos*, ou seja, aqueles onde no equivalente real, haveriam medidores instalados, que estariam disponibilizando as medidas realizadas para o centro de controle.

O usuário pode efetuar tantas modificações quanto desejar, lembrando sempre que após cada modificação na topologia do sistema, deve sempre ser acionado o botão correspondente ao Cálculo do Caso Base.

- Cálculo do estimador de estado:

  No momento em que o usuário determinar que a configuração básica escolhida está devidamente adequada para as suas necessidades, o passo seguinte é o de acionar o botão de cálculo do estimador de estado. Novamente as rotinas de análise de rede, escritas na linguagem Fortran são acionadas e os cálculos executados, sendo então fornecidos ao usuário os resultados obtidos, agora nas telas de saída correspondentes aos valores estimados. O usuário pode então visualizar as telas com os resultados medidos (que serviram de base para os cálculos do estimador de estado) e os resultados estimados. Tem acesso a qualquer subestação pertencente ao sistema em estudo, assim como ao sistema geral.

- Introdução de erros:

  Após determinados os valores medidos e os valores estimados, o usuário pode verificar a influência que a introdução de erros (grosseiros ou aleatórios) nas medidas causa no sistema em estudo. Isso é feito modificando-se qualquer valor medido e a seguir acionando-se o botão para cálculo do estimador de estado. Os valores que porventura aparecerem com a cor vermelha indicam a presença de erros nas medidas. Assim que os valores retornarem a valores aceitáveis a cor retorna ao valor original.

- Salvando uma configuração

  Durante a simulação, a qualquer momento que desejar, o usuário pode salvar uma determinada configuração do sistema para estudo posterior. Para isso basta ir na Barra de Menu, item Arquivo e acionar o comando Salvar. Com isso uma janela se abrirá para que o usuário informe o nome do arquivo onde gostaria que fosse armazenadas as informações que serão salvas. Com isso a situação atual do sistema (estado dos disjuntores, injeções de potência, valores dos taps de transformadores e magnitudes de tensões nas barras) é salva, podendo ser recuperada a qualquer instante, bastando acionar o comando Abrir, na Barra de Menu, item Arquivo, fornecendo o nome do arquivo desejado, que a seguir os valores e configuração topológicas são imediatamente atualizados na tela de saída.

- Fim:

  Caso o usuário não desejar efetuar o passo anterior, ou já estar satisfeito com as simulações efetuadas, tem-se o término do programa.

### 4.8.2 Módulo Execução Automática

A idéia básica ao se criar este módulo foi o de poder apresentar ao usuário a simulação automatizada, ou seja, a verificação de como o sistema reagiria com as injeções de potência seguindo as curvas de carga, isso com o instante de tempo t de simulação variando de uma maneira pré-definida.

A parte inicial da simulação segue os mesmos passos do módulo estudo descrito anteriormente. Após o início da simulação, tem-se o bloco inicialização, a seguir o usuário aciona o botão correspondente ao cálculo do caso base. Com os resultados dos valores medidos agora disponíveis, o passo seguinte é o de calcular os valores estimados acionando o botão correspondente. Neste ponto o usuário está apto a acionar o botão que faz com que o programa entre no módulo Execução Automática. Para que a utilização deste módulo seja possível, diversas condições devem ser respeitadas. Essas condições são:

- Os valores das injeções de potência tanto ativa como reativa das cargas existentes nas subestações não podem ser modificadas. Devem ser os valores que foram fornecidos nas curvas de carga existentes nos arquivos de dados.
- A seqüência para o acionamento do botão Execução Automática deve seguir a seguinte ordem: Inicialização (detalhado em 4.5), Calcular Caso Base (detalhado em 4.6), Calcular Valores Estimados (detalhado em 4.7) e então o acionamento do botão Iniciar Execução Automática está liberado.

As ações que são executadas quando o programa entra no módulo execução automática são descritos a seguir. A figura 4.9 mostra a fluxograma relativo a esse módulo. Lembrando que o bloco INÍCIO do fluxograma refere-se ao momento anterior ao acionamento do botão Iniciar Execução Automática.

- Inicialmente ocorre o ajuste do instante de tempo t, ou seja, qual a hora inicial da simulação. Determinou-se que esse instante inicial seria o mesmo da hora local da estação de trabalho, isto é, se a simulação está ocorrendo em uma quarta-feira da semana e às 14h e 32m, esses serão os dados iniciais.
- Determinação das injeções do sistema. Com os dados anteriores (dia da semana e instante t inicial) uma das rotinas do programa determina a partir dos arquivos de dados com as curvas de cargas, as injeções do sistema que representam esse dia da semana e essa hora do dia.
- tem-se uma rotina que faz com que os botões que no momento não são mais necessários sejam apagados da barra de menu, ficando somente aquele cujo comando ainda seja necessário, no caso o botão cujo comando associado é o de Parar a Execução Automática.

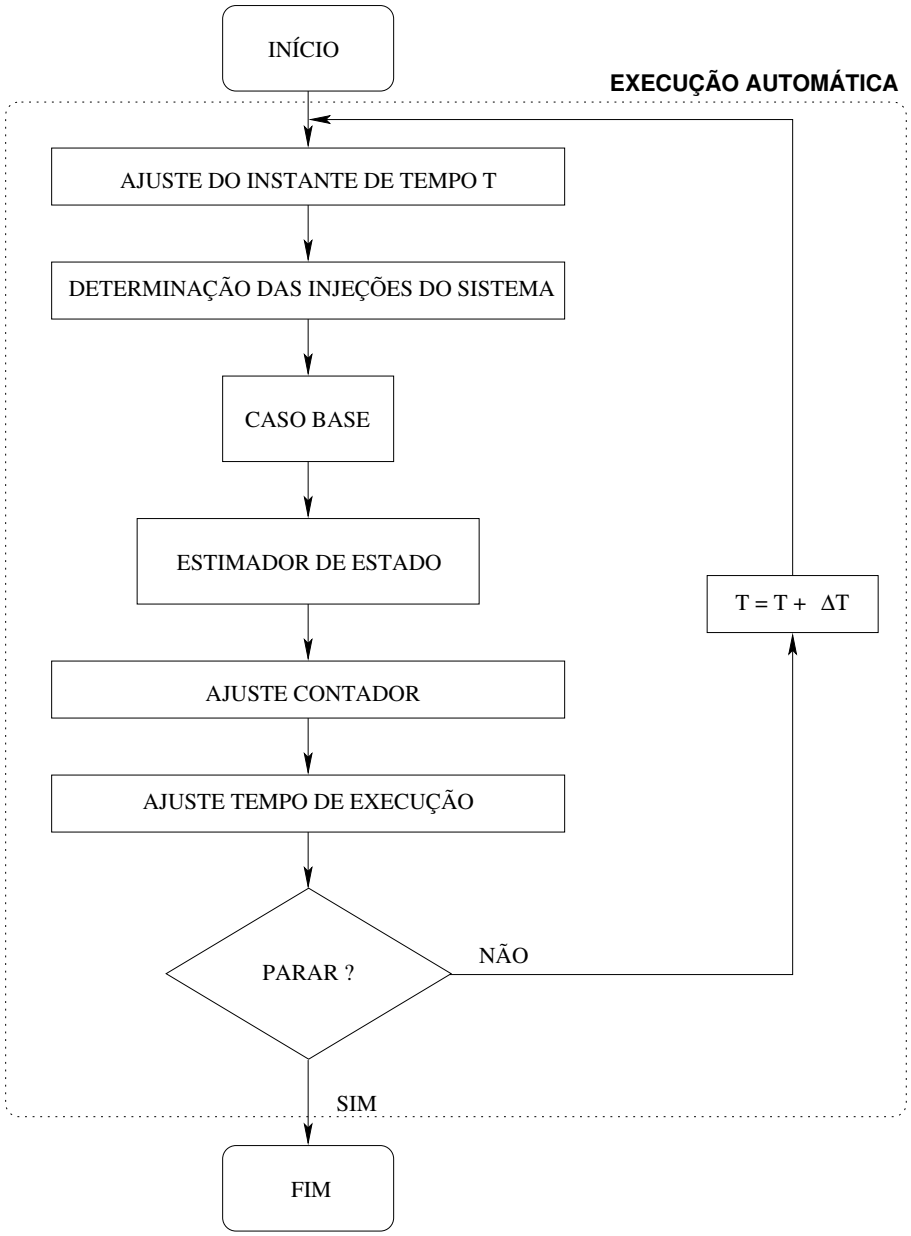


Figura 4.9: Execução Automática

- O bloco cálculo do caso base (descrito em detalhes em 4.6) é acionado e logo a seguir o bloco estimador de estado (detalhado em 4.7).
- É feito o ajuste do contador do número de execuções.
- É feito a atualização do tempo de execução (indicando a hora da execução).
- Está disponível ao usuário o botão Parar Execução Automática. O acionamento desse botão implica em:
  - Uma caixa de diálogo é aberta e é perguntado ao usuário se deseja prosseguir com a simulação. Se a resposta for SIM, o programa retorna ao ponto de parada, sendo que o usuário deve, se desejar continuar com a simulação acionar o botão Continuar Execução Automática. Caso a resposta seja NÃO, nova caixa de diálogo é aberta, perguntando agora se o usuário deseja finalizar a simulação. Se SIM, todo o programa de simulação é finalizado. Se NÃO, retorna-se ao programa no ponto da interrupção para a continuação da execução automática.
- Caso o usuário não acione o botão Parar Execução Automática, é feito um incremento de tempo e o programa retorna no ponto em que os cálculos são executados e o processo é repetido.

As telas de saída que estão disponíveis ao usuário são aquelas com os valores definidos como medidos (resultados do cálculo do caso base) e as telas com os valores estimados (resultados do cálculo do estimador de estado). Durante a simulação da Execução Automática, o usuário pode efetuar modificações nos valores dos fluxos, assim como mudar a topologia do sistema abrindo e fechando disjuntores, verificando a influência dessas ações nos resultados dos cálculos que estão sendo efetuados.

## 4.9 Esquema de Acionamento dos Botões de Comando

Com o intuito de minimizar a possibilidade de acionamento errôneo por parte do usuário, optou-se pelo não-aparecimento na tela de saída dos botões, que são os elementos que quando acionados executam os comandos pré-estabelecidos, nos momentos em que não são solicitados. Isto é, somente quando a execução do comando associado ao botão já pode ser efetuada é que o botão aparece na tela, evitando portanto, erros na simulação.

Cabe lembrar também que o simulador contempla dois tipos de botões de acionamento. Um tipo onde o comando associado está relacionado com ações de controle da parte gráfica (por exemplo: acessar diferentes telas de subestações; salvar e abrir arquivos de configuração) e um outro tipo onde os comando associados são os acionamentos propriamente ditos das funções de análise de rede.

A seguir é apresentado um esquema onde se tem quais os botões mostrados de acordo com as ações associadas. *Início* indica e inicialização do programa. Os outros itens indicam que quando acionados, são mostrados na tela de saída, os botões correspondentes.

No caso de acionamento de determinado botão, caso exista um botão, agora não mais necessário, imediatamente esse botão desaparece da tela, ficando somente aqueles cujos comandos podem ser acionados neste momento.

- **Início:**
  - Arquivo
  - Sistema
  - Zerar Variáveis
  - Calcular Caso Base
  - Acessar outras Subestações (Valores Medidos)

  Este item indica o momento em que a primeira tela de saída é apresentada ao usuário após o início da simulação com o programa desenvolvido.

- **Arquivo:**
  - Salvar
  - Abrir
  - Sair

  Este item se refere a comandos que podem ser executados a qualquer momento pelo usuário. Salvar indica que uma determinada configuração, com os estados dos disjuntores e valores das variáveis, pode ser guardada e acessada posteriormnte através do comando Abrir. Deve-se notar neste ponto que após o acionamento do comando Abrir, aparecerá uma caixa de diálogo alertando o usuário que as informações anteriores serão perdidas perguntando ao usuário se confirma a ação.

- **Calcular caso Base:**
  - Arquivo
  - Sistema
  - Estimador
  - Zerar Variáveis
  - Calcular Caso Base
  - Acessar outras Subestações (Valores Medidos)

- **Zerar Variáveis:**
  - Arquivo

- Sistema
- Zerar Variáveis
- Calcular Caso Base
- Acessar outras Subestações (Valores Medidos)

Caso o botão Zerar Variáveis seja acionado, aparecerá uma mensagem de alerta ao usuário informando que todas as informações anteriores serão perdidas não podendo mais ser recuperadas e o usuário deve confirmar a operação que deseja efetuar. A seguir um indicador será ajustado para que algumas funções não sejam executadas na seqüência. Neste caso o usuário estará apto a efetuar somente simulações passo a passo, não podendo mais ter acesso a simulação com execução automática.

- **Estimador:**
  - Arquivo
  - Sistema
  - Iniciar Execução Automática
  - Estimador
  - Zerar Variáveis
  - Calcular Caso Base
  - Acessar outras Subestações (Valores Medidos e Valores Estimados)
- **Iniciar Execução Automática:**
  - Arquivo
  - Sistema
  - Parar Execução Automática
  - Acessar outras Subestações (Valores Medidos e Valores Estimados)
- **Parar Execução Automática:**

  Neste caso, ao se acionar este botão, aparecerá na tela uma caixa de diálogo, perguntando ao usuário se deseja prosseguir com a simulação. Caso a resposta seja "Sim", os botões abaixo estarão disponíveis.
  - Arquivo
  - Sistema
  - Continuar Execução Automática
  - Parar Execução Automática
  - Acessar outras Subestações (Valores Medidos e Valores Estimados)

  Caso a resposta seja NÃO, outra caixa de diálogo surge, perguntando ao usuário se deseja finalizar a simulação. Caso a resposta seja SIM, o programa é finalizado completamente. Caso a resposta seja NÃO, retorna-se ao programa no ponto onde houve a primeira interrupção na Execução Automática e a simulação pode prosseguir normalmente.

- **Continuar Execução Automática:**
  - Arquivo
  - Sistema
  - Parar Execução Automática
  - Acessar outras Subestações (Valores Medidos e Valores Estimados)

# Capítulo 5

# Exemplos de Aplicação

A seguir serão apresentados exemplos de aplicação do simulador proposto. Foi dada ênfase em mostrar as telas de saída com a aparência geral, que são apresentadas ao usuário com os valores resultantes dos cálculos efetuados durante a simulação. Basicamente existem três conjuntos de telas de saída: as telas de inicialização, as telas com os valores do caso base e as telas com os valores estimados, resultantes do cálculo do estimador de estado.

## 5.1 Inicialização

A figura 5.1 mostra o diagrama unifilar do sistema utilizado como exemplo. É composto por quatro subestações interligadas entre si formando um sistema fechado. Esta é a tela inicial do programa de simulação. As cargas ativas e reativas indicam valores que foram obtidos a partir das curvas de carga de cada uma das variáveis. Foi considerado o dia da semana e horário da ocorrência da simulação.

A figura 5.2 mostra o detalhe da figura 5.1 onde são mostradas ao usuário informações sobre a simulação. Na figura, *Função* indica qual a função que está sendo executada no momento. No caso do exemplo, nada indica, pois é a tela inicial do programa simulador. *Execução n.:* está relacionado ao módulo Execução Automática, não tendo, portanto utilização neste momento, indicando 0 (zero).

Cabe lembrar neste ponto que as caixas que contém os fluxos ativos e reativos nas linhas que aparecem na tela de saída representam que naquele local fisicamente existem medidores que captam as informações e as transferem para o centro de controle.

Outro importante ponto a ser observado é que todos os valores que aparecem em caixas de saída podem ser acessados e modificados da maneira que o usuário achar mais conveniente. Os valores representados pelos botões de escala também são de livre acesso,

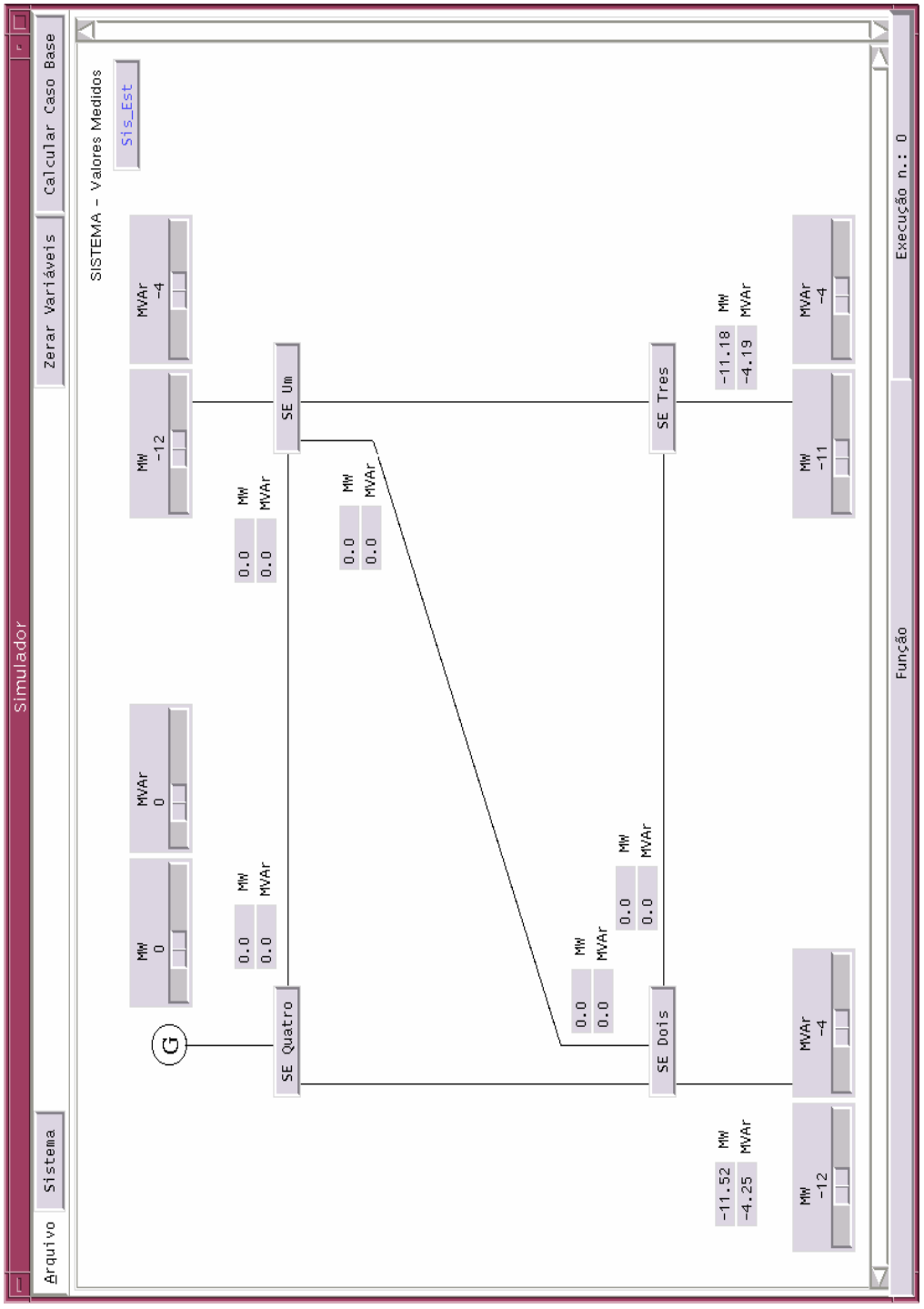


Figura 5.1: Diagrama Unifilar do Sistema


Figura 5.2: Detalhe do Diagrama Unifilar do Sistema

sofrendo somente como restrição os limites (mínimo e máximo) já impostos pela própria escala.

A seguir o usuário pode acessar cada uma das subestações desejadas para efetuar modificações nos disjuntores e nos valores das variáveis, clicando-se o botão correspondente da subestação. A figura 5.3 mostra o diagrama unifilar da subestação 4. Os disjuntores aqui representados podem ser acionados, passando da posição fechado para aberto ou vice-versa. Como notação foi especificado para os disjuntores, que os botões acionados representam disjuntores na posição fechada e botões não acionados, disjuntores na posição aberta. Todos os valores de fluxos ativos e reativos, tensões nas barras e os valores de taps de transformadores representam as variáveis do sistema, podendo portanto, ser modificadas pelo usuário.

## 5.2 Cálculo do Caso Base

Após a inicialização do programa simulador, o usuário efetua todas as modificações desejadas na topologia. Caso prossiga a simulação com os valores das injeções previamente fornecidos, o passo seguinte é acionar o botão cujo comando associado é o de efetuar o cálculo do caso base para a situação (topologia) indicada. Os resultados calculados pela rotina Fortran retornam em forma de arquivos de dados que são devidamente tratados e a seguir esses resultados são mostrados na tela de saída denominada tela com os Valores Medidos. Caso o usuário não desejar efetuar mais nenhuma alteração, o passo seguinte é acionar o botão que efetua o cálculo do estimador de estado. Os valores que servirão de base para o cálculo do estimador são os apresentados como sendo os valores medidos.

A figura 5.4 mostra o sistema geral após o cálculo do caso base, agora já com os resultados calculados mostrados na tela de saída. Já a figura 5.5 mostra a subestação 4 com os estados calculados após acionamento do caso base.

A figura 5.6 mostra o detalhe inferior da tela de saída da figura 5.4. Pode-se ver agora que as funções de análise de rede que foram executadas foram *Configurador e Fluxo de Potência.*

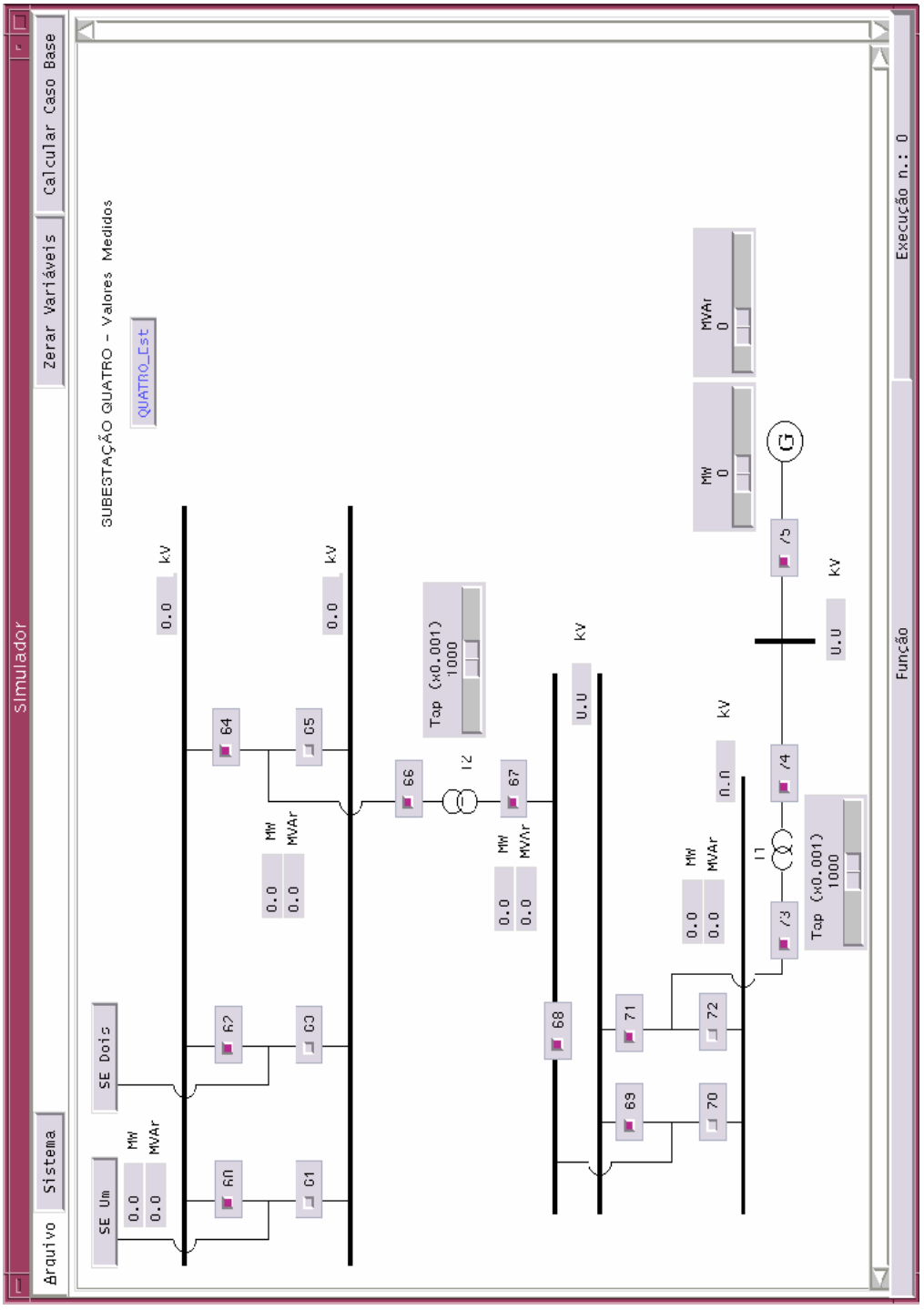


Figura 5.3: Diagrama Unifilar da Subestação 4

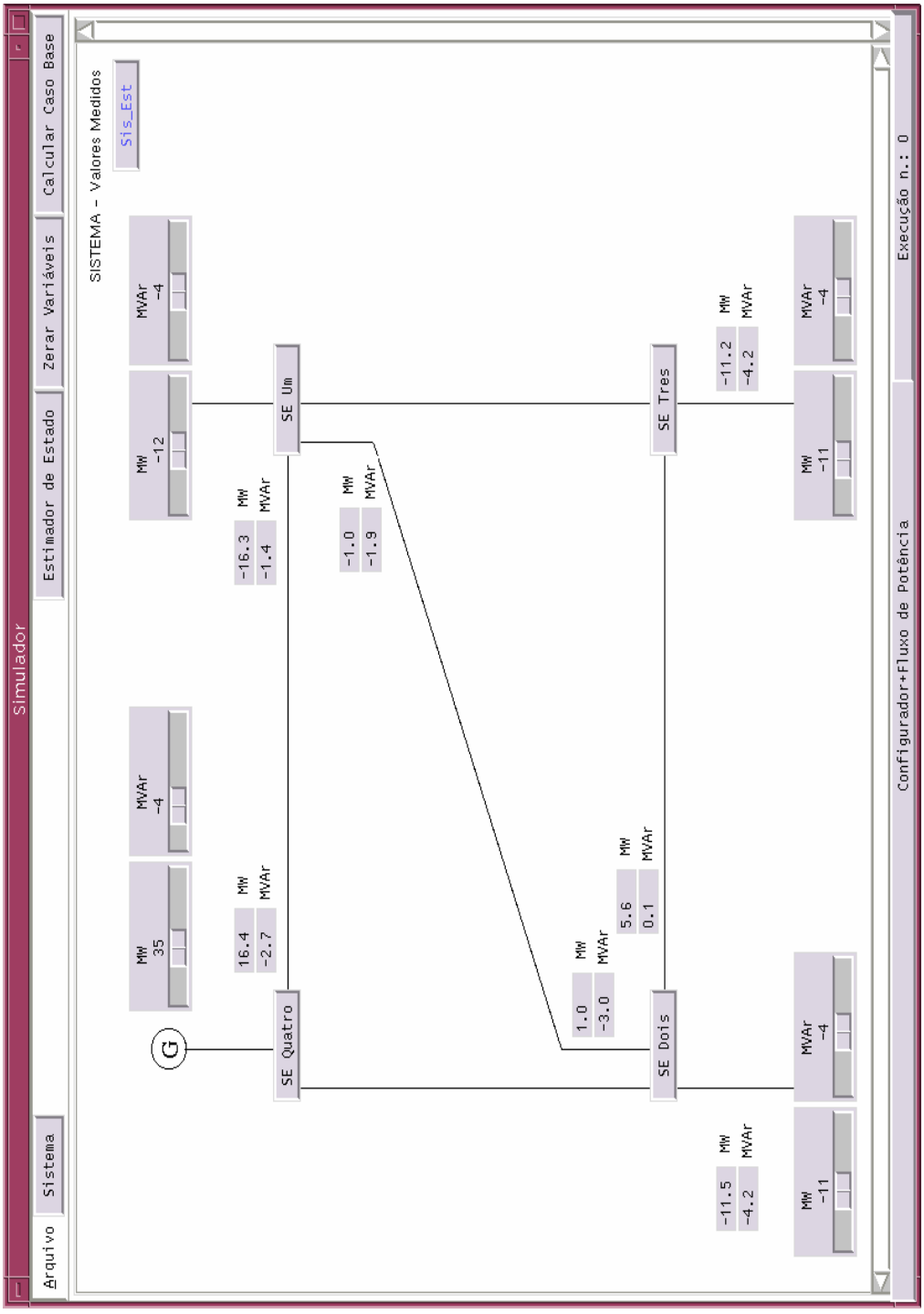


Figura 5.4: Resultados do Caso Base - Sistema Geral

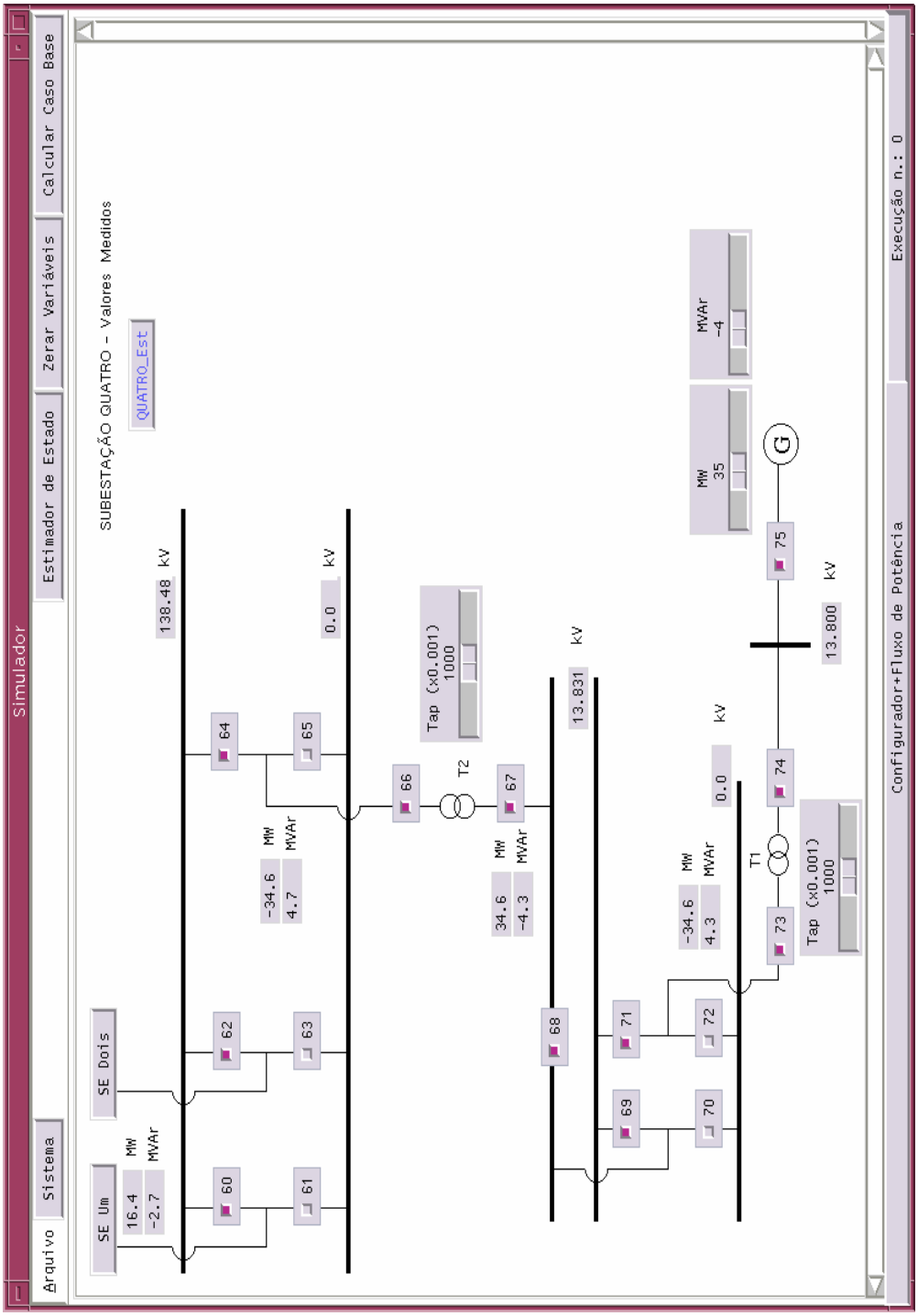


Figura 5.5: Resultados do Caso Base - Subestação 4


Figura 5.6: Detalhe da Tela: Resultados do Caso Base - Sistema Geral

## 5.3 Cálculo do Estimador

Similarmente ao passo anterior, uma rotina Fortran efetua os cálculos e novamente os resultados, após devido tratamento, são apresentados ao usuário. Porém, agora a tela de saída é outra e recebeu a denominação de tela com os Valores Estimados. A figura 5.7 mostra os valores estimados para o sistema geral anterior. Nota-se agora que todos os valores dos fluxos estão disponíveis ao usuário para leitura. Os valores que porventura aparecerem com a cor vermelha sinalizam que no valor da medida em questão está ocorrendo algum tipo de erro. Assim que o problema for sanado a cor de saída retorna ao valor original.

A figura 5.8 mostra o detalhe da figura 5.7 com a indicação que a função *Estimador* foi executada.

Na figura 5.9 tem-se os valores estimados referentes a subestação 4. Lembrando-se que os valores de entrada foram os valores resultantes do cálculo anterior do caso base para a situação topológica definida.

Caso seja do interesse do usuário, é possível neste ponto a inclusão manual de erros nos valores medidos. Isto é, supondo que o usuário deseje saber qual o efeito, que a inclusão de um determinado erro em uma dada medida, tem sobre o sistema em estudo em termos de valores estimados, essa verificação é perfeitamente possível. O procedimento para se verificar o efeito do erro agregado nas medidas é o seguinte: na tela dos valores medidos calculados após a execução do caso base, o usuário efetua as modificações desejadas e a seguir aciona o botão para calcular os valores estimados. Caso o erro introduzido seja detectado pelo programa de cálculo do estimador de estado, na saída dos dados os valores são mostrados em destaque (cor de saída diferente). Corrigindo-se o erro, as saídas retornam ao seu estado normal, significando que o problema foi solucionado.

## 5.4 Execução Automática

A execução automática efetua a variação automática dos valores das injeções de acordo com um intervalo de tempo especificado pelo usuário. Podendo variar de segundos a horas. No momento em que a rotina apropriada determina o instante de tempo de

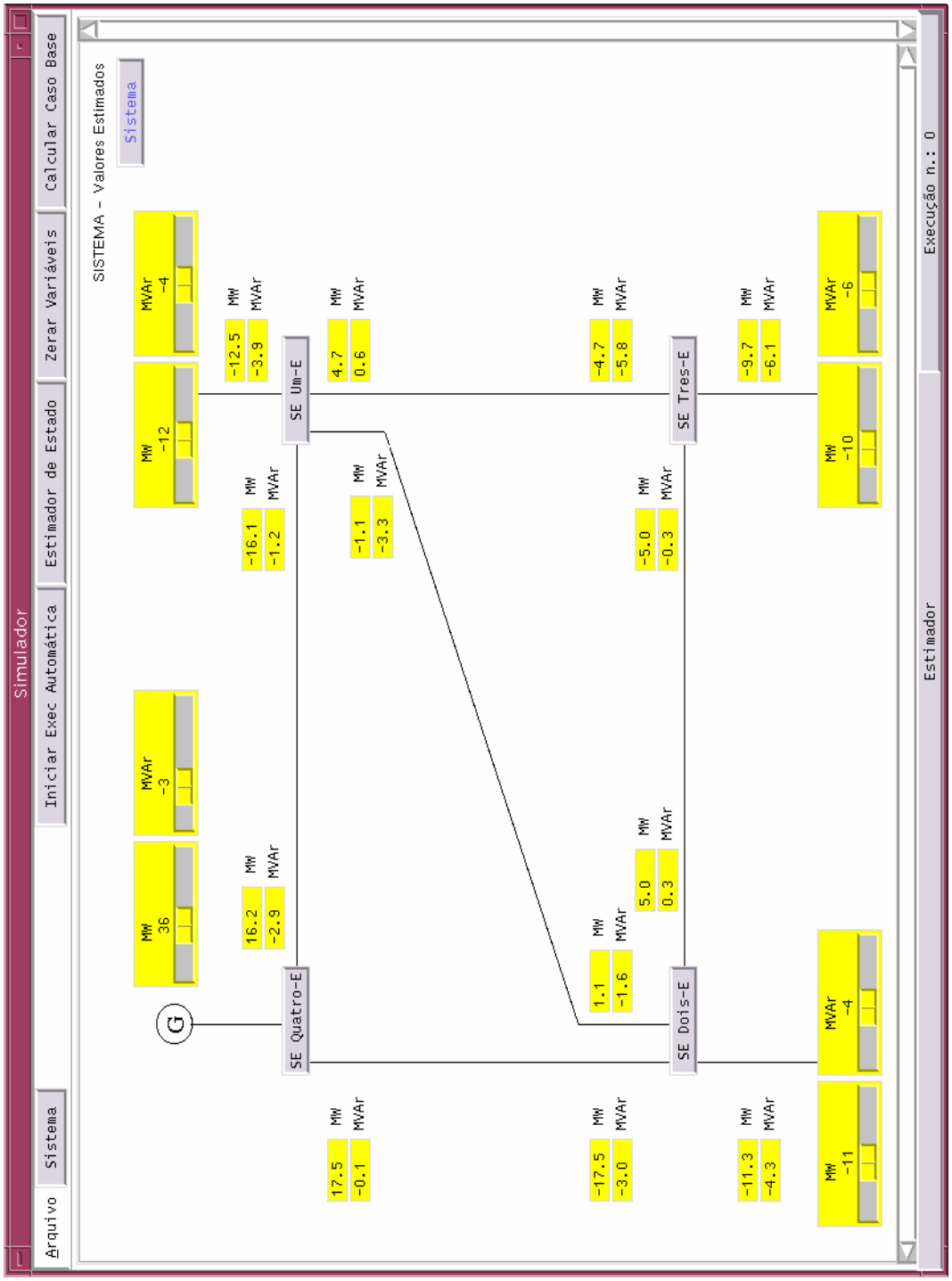


Figura 5.7: Resultados do Estimador - Sistema Geral

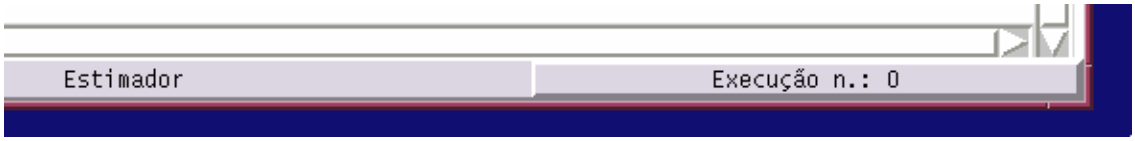


Figura 5.8: Detalhe da Tela: Resultados do Estimador - Sistema Geral

simulação, tem-se disponível o conjunto de dados que representam todas as injeções do sistema. Antes que se tenha a devida autorização para executar esta função (execução automática), é necessário que pelo menos uma vez tenha sido calculado um caso base para o sistema e um cálculo do estimador de estado. De posse dos resultados, o programa executa então a rotina que efetua o acréscimo no intervalo de tempo, determina as novas injeções e alternadamente realiza os cálculos do caso base e do estimador para se obter os valores medidos e os valores calculados. A saída desses resultados obedece a seguinte norma: se o início da execução automática ocorreu quando o usuário tinha a seu dispor a tela com valores medidos, serão esses valores que sofrerão a atualização e serão mostradas ao usuário. Caso tenha ocorrido nas telas com os valores estimados, os valores das variáveis dessas telas é que serão atualizados. Deve-se observar que tanto os valores medidos como os valores calculados estão disponíveis ao usuário e que somente por uma questão de comodidade foi estabelecido que somente uma das duas telas teria os seus valores atualizados *on-line*. Para ter acesso aos outros valores basta acionar o botão correspondente a tela desejada. A figura 5.10 mostra um exemplo de saída de tela durante a execução automática. Pode-se observar na parte inferior da figura qual o número da execução e o tempo (hora) da execução.

A figura 5.11 mostra que o programa simulador está no módulo Execução Automática, no detalhe da figura 5.10. As outras informações disponíveis são: *Execução n.: 9*, indicando que é a nona vez que ocorre incremento no tempo t de simulação e o processo está se repetindo com os novos valores das injeções; *Tempo: 15.91*, indicando a hora de processamento desse ciclo (no caso 15.91 equivale a: 15 horas, 54 minutos e 36 segundos).

Durante a execução automática o usuário tem permissão para introduzir erros em algumas medidas para verificar a influência deles no sistema. Isto pode ser verificado nas figuras 5.12 e 5.13. Pode também efetuar acionamentos nos disjuntores simulando saída ou entrada de linha para análise dos resultados.

## 5.5 Outras Simulações

Caso o usuário não queira utilizar os valores das injeções das cargas previamente fornecidos, poderá entrar com valores próprios através da opção *Zerar Variáveis* disponível a qualquer momento. Nesta opção, o usuário deve fornecer todos os valores das variáveis para que os cálculos sejam efetuados. Uma vez feita essa escolha, deve-se ter em mente que a opção *Execução Automática* não pode mais ser acessada, visto que os valores cor-

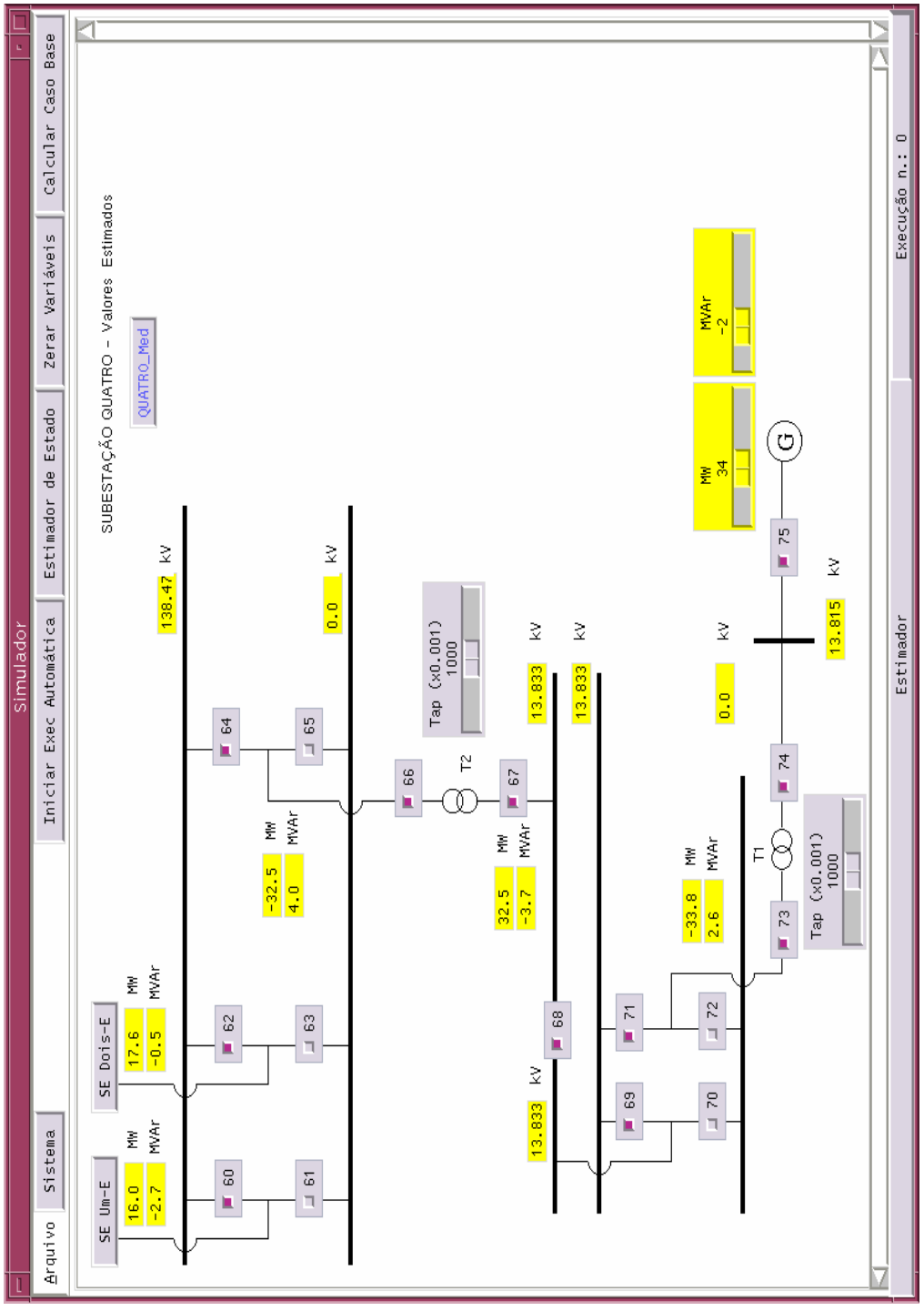


Figura 5.9: Resultados do Estimador - Subestação 4

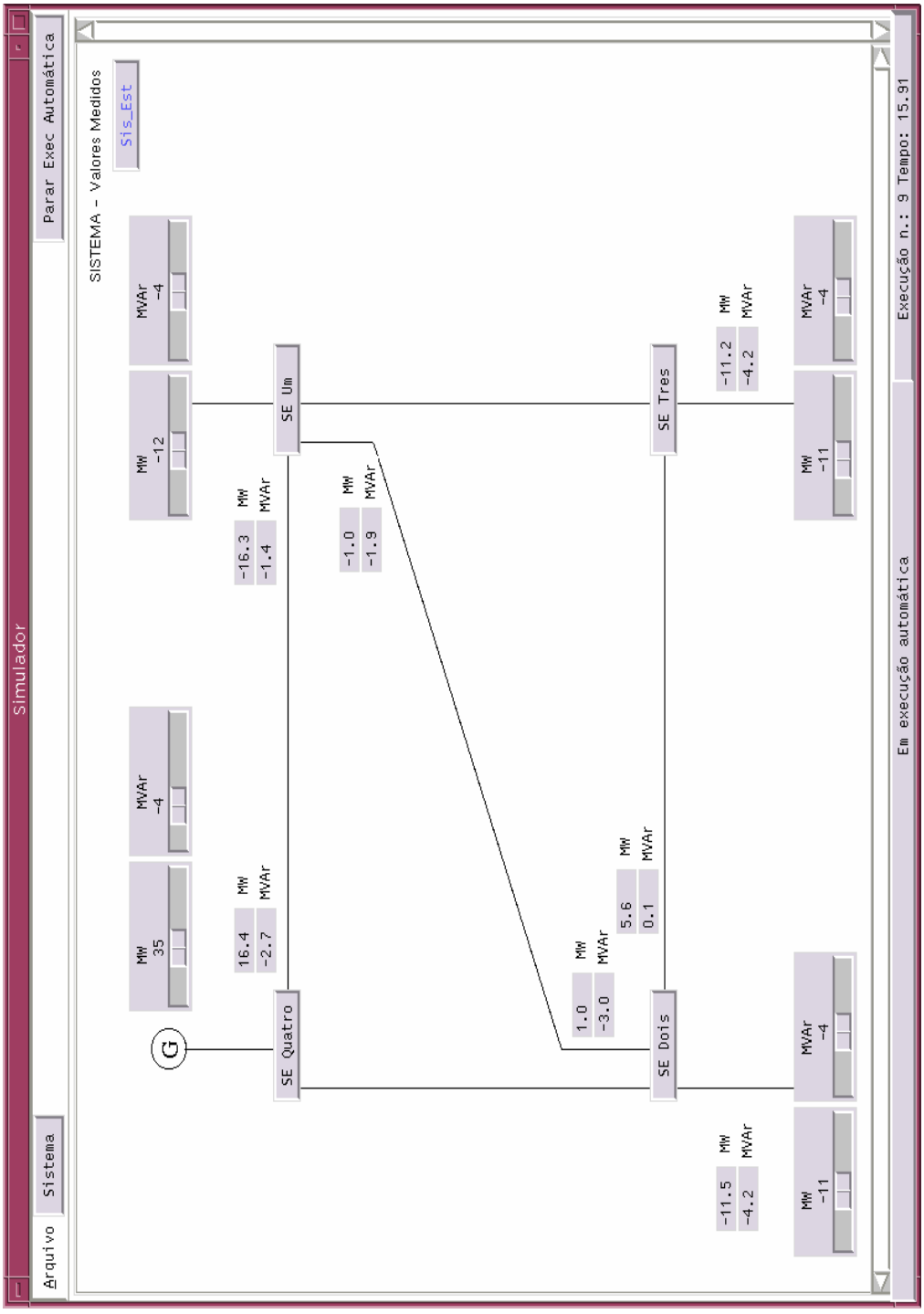


Figura 5.10: Execução Automática


Figura 5.11: Detalhe da Tela: Execução Automática

respondentes às injeções são fixos e para uma determinada situação. Portanto, a análise possível se restringe ao estudo do caso base e a respectiva estimação de estado para a situação desejada.

## 5.6 Simulando no ambiente Windows

Para verificar a portabilidade do simulador desenvolvido, foram feitos testes no ambiente Windows. A figura 5.14 mostra uma saída do programa, que é apresentada na tela do PC. Foi utilizado um PC com a seguinte configuração: processador AMD K62, 500 MHz, 64 MB RAM, placa de vídeo ATI 8 MB RAM, sistema operacional Windows 98.

Verificou-se que o aspecto geral da tela de saída é similar ao encontrado quando a simulação foi feita na ambiente Unix, excetuando-se alguns detalhes referentes à parte gráfica, representação dos disjuntores, por exemplo. Com relação a chamada das funções de análise de rede não houve problema, sendo executada com a mesma rapidez das simulações anteriores.

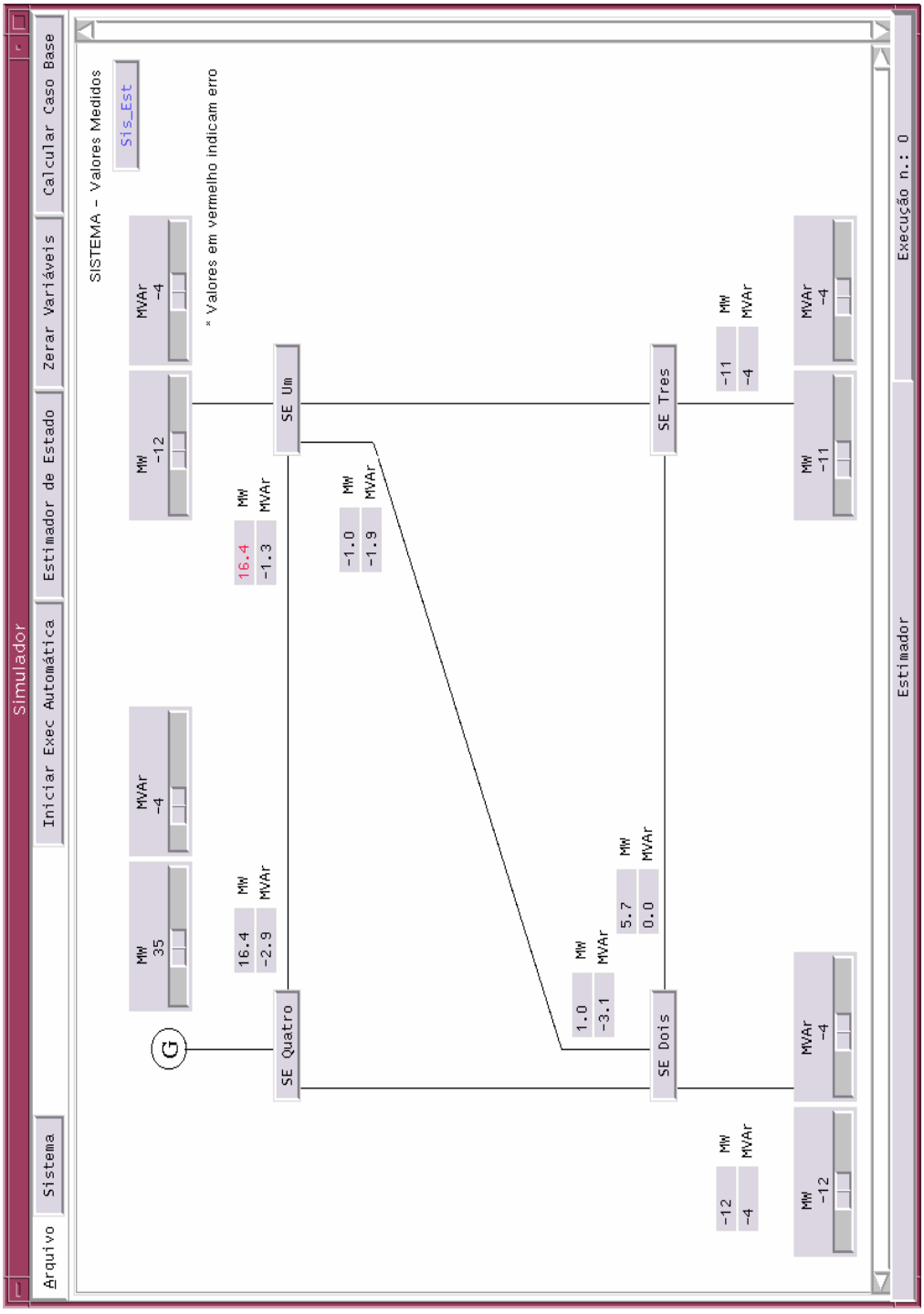


Figura 5.12: Detecção de Erro - Valores Medidos

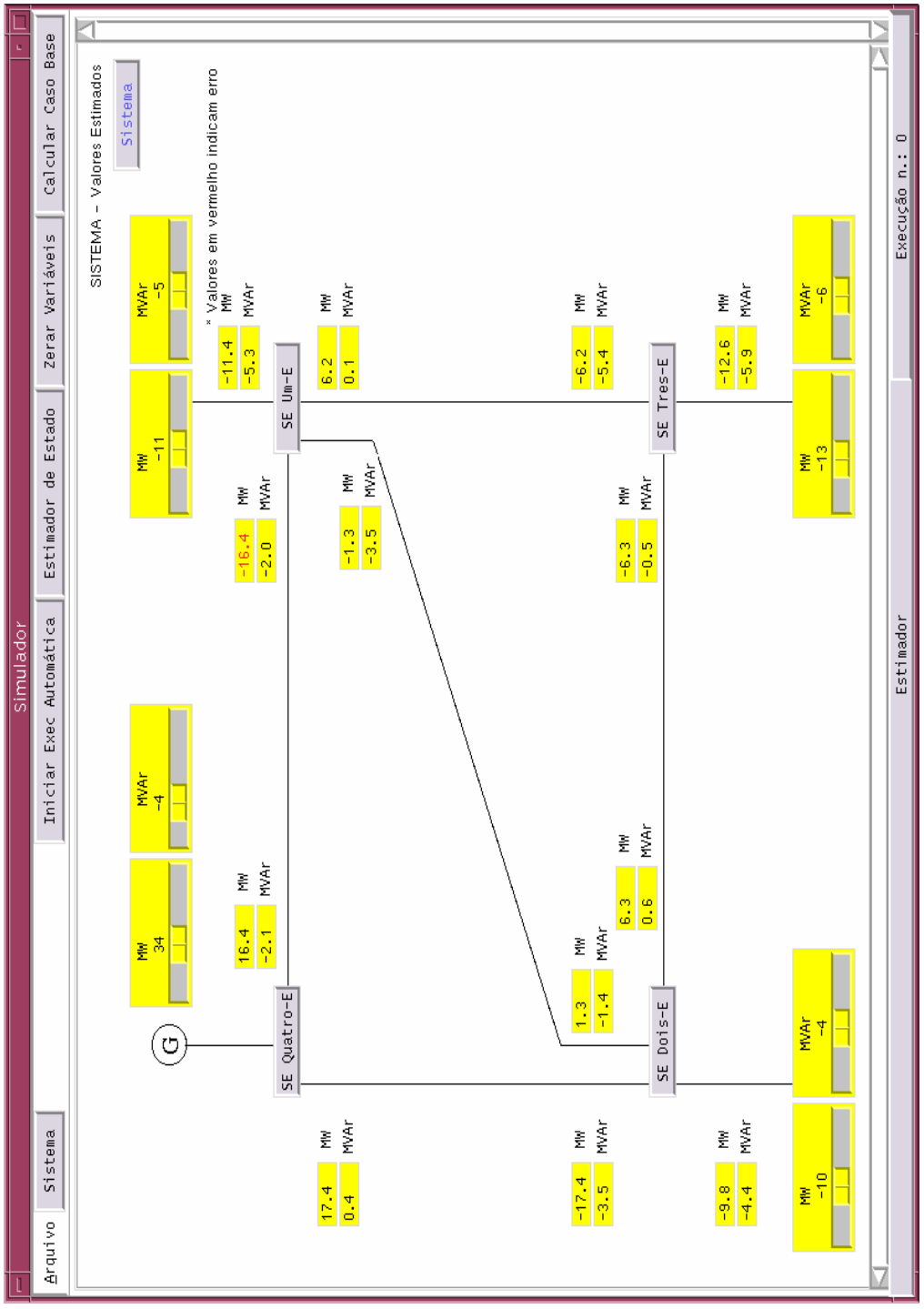


Figura 5.13: Detecção de Erro - Valores Estimados

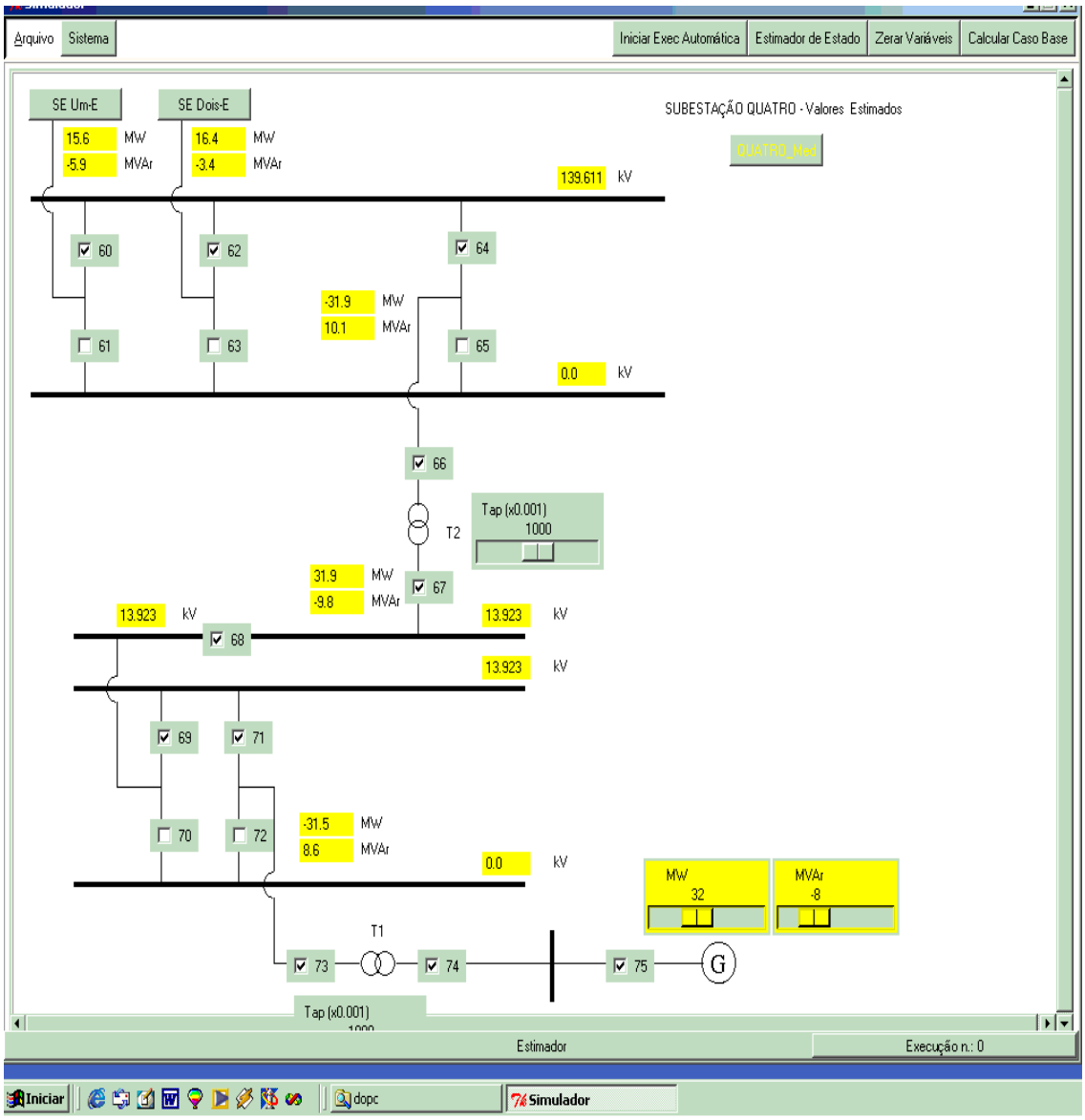


Figura 5.14: Simulando no Windows

# Capítulo 6

# Conclusões

Os programas de simulação por computador desempenharam por anos um papel importante na área de sistemas de potência, sendo uma ferramenta de grande utilidade tanto no treinamento de operadores quanto no ensino de tópicos de engenharia aos alunos de graduação. Com o rápido avanço em *hardware* e *software*, as ferramentas educacionais de sistema de potência baseadas em computador aumentaram, passando de implementações muito simples, fornecendo ao usuário pouco mais que uma série de saídas numéricas até representações muito detalhadas do sistema de potência com uma extensa interface gráfica com o usuário (GUI).

O que se pretendeu neste presente trabalho foi desenvolver um programa computacional de um simulador de sistema de potência para treinamento com interface gráfica ao usuário baseada em Tcl/Tk, incluindo ainda funções avançadas de análise de rede em tempo real. Procurou-se fazer uma interface gráfica amigável onde os resultados fossem apresentados de forma rápida e direta ao usuário.

A incorporação dos aplicativos de análise de rede é considerada uma das principais características dos modernos centros de controle. Os aplicativos de análise de rede elétrica, também conhecidos como funções avançadas de análise de rede, são programas cuja execução ocorre em tempo real visando proporcionar maior segurança e confiabilidade na operação de sistemas elétricos de potência.

A utilização da linguagem Tcl/Tk para o desenvolvimento da interface gráfica ao usuário, teve como objetivo, verificar a eficácia da linguagem em uma aplicação prática, juntamente com outra linguagem, no presente caso a linguagem Fortran. Verificou-se que durante o desenvolvimento do programa a utilização da mesma se mostrou bastante eficaz sem apresentar problemas no momento da implementação. O fato da distribuição do Tcl/Tk ser gratuita e dispor de farto material de auxílio e literatura especializada potencializa sua utilização. Novas atualizações estão constantemente a disposição dos usuários, além de listas de discussões existentes na Internet onde usuários diversos efetuam trocas de informações, podem ser citados também como pontos favoráveis. A portabilidade que

a linguagem Tcl/Tk oferece foi um grande atrativo no desenvolvimento do simulador.

A preocupação com a modularização das rotinas durante o desenvolvimento do programa permite que modificações posteriores sejam efetuadas de maneira mais simples e rápida. Portanto o acréscimo ou retirada de subestações do sistema em estudo torna-se uma tarefa fácil, devendo somente ser observados alguns cuidados no momento das substituições dos arquivos de dados.

A implementação da rotina Execução Automática junto ao simulador desenvolvido permite ao usuário um entendimento maior no que se refere à análise de rede em tempo real. O fato da rotina que executa o incremento de tempo para a captura dos dados relativos às injeções do sistema simular o tempo decorrido de acordo com a definição feita pelo usuário, faz com que o usuário possa verificar o comportamento do sistema ao longo do tempo.

A possibilidade de se poder gravar/salvar um determinado cenário do sistema durante a execução da simulação, permite ao usuário retornar posteriormente a esse caso de modo a efetuar os estudos desejados no momento que for necessário.

Sugestões para trabalhos futuros:

- Desenvolvimento de um *builder*

  Para se poder efetuar simulações de sistemas com diferentes cenários, um ferramenta de grande auxílio seria um *builder*, rotina que *constrói* os diagramas unifilares das subestações de acordo com o desejo do usuário, permitindo a criação de cenários diversos.

- Inclusão de novas funções avançadas de análise de rede

  Outras funções avançadas de análise de rede poderiam ser acrescentadas ao programa simulador, aumentando a faixa de opções de simulações de modo a melhorar o estudo de análise de rede do sistema em estudo.

- Inclusão de eventos aleatórios

  Durante a execução do programa de simulação, poderiam ocorrer eventos aleatórios, como por exemplo, saída de uma linha de transmissão. A ocorrência desses eventos poderia ser programada ou não pelo usuário antes do início da simulação.

# Referências Bibliográficas


[Ajub 98] Tcl Developer Xchange. Em novembro de 2000, disponível em: http://dev.scriptics.com/software/tcltk.

[Alle 97] Allen, S., vTcl - Código para download. Em novembro de 2000, disponível em: http://vtcl.sourceforge.net/dload.

[Bucc 91] Bucciero, J.M.; Dodge, J.A.; Gillespie, R.N.; Hinkel, R.O.; Mann, S.L.; Martin, S.M.; Schulte, R.P., "Dispatcher Training Simulator Lessons Learned", IEEE Transactions on Power Systems, Vol. 6, No. 2, pp. 594-601, May 1991.

[Chow 92] Chowdhury, B.H.; Clark, D.E., "COPERITE - Computer-Aided Tool for Power Engineering Research, Instruction, Training and Education", IEEE Transactions on Power Systems, Vol. 7, No. 4, pp. 1565-1570, Nov 1992.

[DyLi 83] DyLiacco, T.E.; Enns, M.K.; Schoeffler, J.D.; Quada, J.J.; Rosa, D.L.; Jurkoshek, C.W.; Anderson, M.D., "Considerations in Developing and Utilizing Operator Training Simulators", IEEE Transactions on Power Apparatus and Systems, Vol. PAS-102, No. 11, pp. 3672-3679, May 1983.

[Frei 92] Freire, L.M.; Garcia, A.V.; Monticelli, A.J., "Modernização Incremental do Centro de Operação do Sistema da CPFL", 1º Simpósio de Automação de Sistemas Elétricos, 1992, Campinas, SP, Anais ... Campinas: CPFL, 1992, p.318-328

[Garc 93] Garcia, A.V., "Supervision y Control en Tiempo Real de Sistemas de Potencia", Curso tutorial ministrado no X Congreso Chileno de Ingenieria Electrica, Valdívia, Chile, Out 1993.

[Gaus 87] Gaushell, D.J.; Darlington, H.T., "Supervisory Control and Data Acquisition", Proceedings of IEEE, Vol. 75, No. 12, pp. 1645-1658, 1987.

[Grai 94] Grainger, J.J.; Stevenson Jr., W.D., "Power System Analysis", McGraw-Hill International Editions, 1994.

[Li 93] Li, S.; Shahidehpour, S.M., "An Object Oriented Power Systems Graphics Package for Personal Computer Environment", IEEE Transactions on Power Systems, Vol. 8, No. 3, pp.1054-1060, Aug 1993.

[Macd 97] Macdonald, R., Tutorial do vTcl. Em novembro de 2000, disponível em: http://vtcl.sourceforge.net/tutorial.html.

[Mill 93] Miller, G.; Storey, A.; Vadari, S.V.; Brewer, K., "Experiences Using the Dispatcher Training Simulator as a Training Toll", IEEE Transactions on Power Systems, Vol. 8, No. 3, pp. 1126-1132, Aug 1993.

[Mont 79] Monticelli, A.; Deckmann, S.M.; Garcia, A.V.; Stott, B., "Real-time external equivalents for static security analysis " , IEEE Transactions on Power Apparatus and System, Vol. 98, No. 2, pp 498-508, Abr 1979.

[Mont 83] Monticelli, A.J., "Fluxo de Carga em Redes de Energia Elétrica", Editora Edgard Blücher Ltda., São Paulo, Brasil, 1983.

[Mont 83b] Monticelli, A.J.; Garcia, A.V., "Reliable bad data processing for real-time state estimation", IEEE Transactions on Power Apparatus and Systems, Vol PAS 102, pp. 1126-1139, May 1983;

[Mont 90] Monticelli, A.J.; Garcia, A.V., "Fast decoupled state estimators", IEEE Transactions on Power Systems, Vol. 5, No. 2, pp. 556-564, May 1990.

[Mont 92] Monticelli, A.J.; Garcia, A.V.; Slutsker, I., " Handling discardable measurements in power system state estimation". IEEE Transactions on Power Systems, Vol. 7, No. 3, pp. 1333-1340, Aug 1992.

[Okap 96] Okapuu-von Veh, et alii, "Design and operation of a Virtual Reality Operator-Training System", IEEE Transactions on Power Systems, Vol. 11, No. 3, pp. 1585-1591, Aug 1996.

[Oust 94] Ousterhout, J.K., "Tcl and the Tk Toolkit", Addison-Wesley Publishing Company, 458 p., 1994.

[Oust 98] Ousterhout, J.K., "Scripting: Higher Level Programming for the 21st Century", IEEE Computer Magazine, Mar 1998.

[Over 95] Overbye, T.J.; Sauer, P.W.; Marzinzik, C. M.; Gross, G., " A User-Friendly Simulation Program for Teching Power System Operations", IEEE Transactions on Power Systems, Vol. 10, No. 4, pp. 1725-1733, Nov 1995.

[Over 97] Overbye, T.J.; Gross, G.; Laufenberg, M.; Sauer, P.W., "Visualizing power System Operation in an Open Market", IEEE Computer Applications in Power, pp.53-58, Jan 1997.

[Prais 89a] Prais, M.; Zhang, G.; Chen, Y.; Bose, A.; Curtice, D., "Operator training Simulator: Algorithms and Test Results", IEEE Transactions on Power Systems, Vol. 4, No. 3, pp. 1154-1159, Aug 1989.

[Prais 89b] Prais, M.; Johnson, C.; Bose, A.; Curtice, D., "Operator training Simulator: Component Models", IEEE Transactions on Power Systems, Vol. 4, No. 3, pp. 1160-1166, Aug 1989.

[Raja 94] Rajagopal, S.; Rafian-Naini, M.; Lake, J.; Turke, A.; Sigari, P.; Silverman, S.; Henry, L.; Hamlin, R.; Bednarik, R.A., "Workstation Based Advanced Operator Training Simulator for Consolidated Edison", IEEE Transactions on Power Systems, Vol. 9, No. 4, pp.1980-1986, Nov 1994.

[Russ 79] Russell, J.C.; Masiello, R.D.; Bose, A., "Power System Control Center Concepts", Power Industry Computer Applications Conference, pp. 170-176, 1979.

[Shin 99] Shin, J.R; Lee, W.H.; Im, D.H., "A Windows-Based Interactive and Graphic Package for the Education and Training of Power System Analysis and Operation", IEEE Transactions on Power Systems, Vol. 14, No. 4, pp.1193-1199, Nov 1999.

[Stot 74] Stott, B.; Alsaç, O., "Review of Load-Flow Calculation Methods", Proceedings of the IEEE, Vol. 62, No. 7, pp. 916-929, Jul 1974.

[TclTk] The Tcl/Tk Consortium. Em novembro de 2000, disponível em: http://www.tcltk.com/main.html

[Vadar 95] Vadari, S.V.; Montstream, M.J.; Ross Jr., H.B., "An Online Dispatcher Training Simulator Function for Real-Time Analysis and Training", IEEE Transactions on Power Systems, Vol. 10, No. 4, pp. 1798-1804, Nov 1995.

[Welc 97] Welch, B.B., "Practical programming in Tcl and Tk - Second Edition", Prentice Hall, Inc, 630p., 1997.

[Yu 95] Yu, D.C., Chen, D.; Ramasamy, S.; Flinn, D.G., "A Windows Based Graphical Package for Symmetrical Components Analysis", IEEE Transactions on Power Systems, Vol. 10, No. 4, pp.1742-1749, Nov 1995.

[Zaga 95] Zagari, E.N.F., "Escalonamento em Tempo Real das Funções Avançadas de Análise de Rede Elétrica em um moderno Centro de Controle", Dissertação de Mestrado, Faculdade de Engenharia elátrica e de Computação da UNICAMP, 115p., Dez 1995.

# Apêndice A

# Tcl/Tk

## A.1 Introdução

Tcl é a abreviatura de *Tool Command Language* e foi criada, juntamente com o conjunto de ferramentas para interface gráfica Tk (Toolkit), por John Ousterhout da Universidade da Califórnia, Berkeley (EUA), no fim da década de 80 [Oust 94].

Tcl é na realidade uma linguagem para escrever *scripts* [Oust 98], similar a outras linguagens de comando utilizadas em *shells* do UNIX e também um interpretador para uma linguagem projetada para ser facilmente incorporada na sua aplicação.

Tcl foi projetada com a filosofia que se deveriam ser utilizadas duas ou mais linguagens quando se estivesse projetando grandes sistemas de software. Uma para manipular estruturas de dados internas complexas, ou onde o desempenho é chave e outra, tal como Tcl, para escrever *scripts* menores para manter juntas as outras partes.

Como qualquer *shell*, Tcl permite a execução de outros programas e fornece variados recursos de programação, como variáveis, procedimentos e controle de fluxo. Possibilita assim a construção de scripts complexos que podem ser agregados a programas existentes para atender de maneira eficiente às necessidades do usuário.

A principal diferença entre Tcl e linguagens tais como C, é que Tcl é uma *linguagem interpretada* ao invés de uma *linguagem compilada*. Os programas em Tcl são simplesmente *scripts* consistindo de comandos do Tcl que são processados por um interpretador Tcl durante a execução. Uma vantagem que isso oferece é que programas em Tcl podem eles próprios gerar *scripts* em Tcl que podem ser avaliados mais tarde. Isto pode ser útil, por exemplo na criação de interface gráfica ao usuário com um botão de comando que precisa perfazer diferentes ações em diferentes momentos.

Tcl suporta muitas das características das linguagens procedurais convencionais in-

cluindo transferência de variável, chamada de procedimentos, estruturas de controle e além disso tem facilidade de uso no acesso a janelas gráficas. Tcl é menos "restrito" no manuseio de dados do que a maioria das linguagens convencionais. Não é uma linguagem adequada para escrever programas grandes e complexos, onde se requer que este seja altamente confiável e robusto. É, contudo, bastante adequado para programas que requerem uma boa interface com o usuário.

Algumas características chaves do Tcl incluem [Oust 94]:

- Tcl é uma linguagem de *script* de alto nível.
  Se comparado com aplicações Win32 ou Motif, resulta em muito menos código para se realizar o mesmo trabalho.
- Tcl é interpretado.
  Pode-se executar o código diretamente, sem ter que compilar.
- Tcl é extensível.
  É muito fácil acrescentar seus próprios comandos para extender a linguagem Tcl. Pode-se escrever seus próprios comandos em Tcl ou C.
- Tcl pode ser executado em muitas plataformas.
  Existem versões para plataformas Unix, Windows e Macintosh. Exceto por algumas poucas deferenças entre as plataformas, um *script* Tcl é executado da mesma maneira em todos os sistemas.
- A facilidade de auto-carregamento do Tcl é apropriada para aplicações menores.
  Tcl carrega automaticamente em suas bibliotecas de procedimentos Tcl com uma linha de código para ajustar o caminho para a biblioteca.
- Tcl é grátis.
  Pode-se obter os códigos fontes gratuitamente através da Internet [TclTk, Ajub 98].

## A.2 Iniciando o Sistema

Uma vez instaladas as versões de Tcl/Tk, existem basicamente duas *shells* que podem ser utilizadas: tclsh e wish. Tclsh é um *shell* básico que pode ser utilizado como o cshell. Wish (*windowing shell*) é o interpretador de Tcl que foi extendido com comandos do Tk utilizados para criar e manipular *widgets*. Ambas *shells* imprimem um prompt % e executarão comandos do Tcl interativamente, imprimindo na tela o resultado de cada comando.

Para facilitar a entrada de programas longos que estejam em um arquivo editado com seu editor preferido, tem-se duas opções. A primeira é executar esses programas utilizando o comando do Tcl chamado *source*.

A segunda é criar um *script* assim como se cria um *script* do csh ou sh no ambiente Unix.

## A.3 Programando com a linguagem Tcl

Para o aprendizado da linguagem Tcl [Oust 94, Welc 97] existem basicamente duas partes a serem consideradas:

1. Sintaxe e regras de substituição
2. Comandos de construção
   - Podem ser aprendidos independentemente conforme necessidade

Tcl não possui gramática fixa.

- Um script em Tcl equivale a:
  - uma seqüência de comandos.
  - os comandos são separados por novas linhas ou pelo ponto e vírgula.
- Um comando Tcl equivale a:
  - uma ou mais palavras separadas por um espaço em branco.
  - a primeira palavra é o nome do comando, o restante são os argumentos
  - retorna o resultado

O símbolo colchetes [ ] executa um comando aninhado. Por exemplo, quando se deseja passar o resultado de um comando como o argumento de outro, este tipo de sintaxe pode ser usado.

O símbolo aspas duplas agrupa palavras em um único argumento para um comando. O símbolo cifrão $ e colchetes [ ] são interpretados dentro de aspas duplas.

O símbolo chaves {} também agrupa palavras em um único argumento. Neste caso, contudo elementos dentro de parenteses não são interpretados.

O símbolo barra invertida \ é usado para citar caracteres especiais.

### A.3.1 Variáveis

Tcl possui dois tipos de variáveis: variáveis simples e *arrays*.

- Variáveis simples

  O comando *set*, usado para atribuir valores a uma variável, recebe dois argumentos: o nome da variável seguido pelo valor. Não é necessário declarar as variáveis antes de usá-las e elas podem ter qualquer tamanho, sendo sensível a caixa alta e caixa baixa.

- *Arrays*

  A utilização de *arrays* na programação em Tcl é um meio de se agrupar dados. *Arrays* são simplesmente coleções de itens, no qual a cada item é dado um único índice pelo qual ele poderá ser acessado. Assim como todas as outras variáveis Tcl, os *arrays* não precisam ser declarados antes de sua utilização e diferentemente de *arrays* na linguagem C, seus tamanhos tampouco precisam ser especificados.

### A.3.2 Listas

Listas em Tcl são apenas cadeias de caracteres com uma interpretação especial. Fornecem um meio simples de agrupar coleções de itens e tratar essas coleções como uma entidade única. Quando necessário, cada item de um grupo pode ser acessado individualmente. Listas em Tcl são representados como *strings* com um formato específico. Como tal, podem ser usadas em qualquer lugar onde *strings* são normalmente permitidos.

- O comando *list* constrói uma lista de seus argumentos. Se algum argumento tem algum caracter especial, o comando list o delimita por chave para garantir que o elemento seja único na lista.
- O comando *lappend* é usado para adicionar elementos ao final de uma lista.
- O comando *concat* concatena listas.

### A.3.3 Procedimentos

Um procedimento em Tcl é definido com o comando *proc*. O nome do procedimento é sensível a caixa alta e baixa e pode conter qualquer caracter. Uma vez definido um procedimento em Tcl, pode ser utilizado como qualquer outro comando do Tcl. Quando chamado, cada argumento é atribuído ao seu parâmetro correspondente e o corpo é executado. O comando *return* pode ser utilizado para retornar um valor específico resultante de um procedimento.

### A.3.4 Escopo

Existe um único escopo global para nome de procedimentos. Procedimentos definidos dentro de outros procedimentos são visíveis de todos os lugares. Existe um espaço diferente para variáveis e procedimentos, de modo que é possível a existência de um procedimento e uma variável com o mesmo nome sem conflito.

Cada procedimento possui um escopo local para variáveis. Variáveis introduzidas em um procedimento existem somente durante a execução da chamada do procedimento. Após o termino da execução do procedimento as variáveis introduzidas no procedimento se tornam indefinidas.

### A.3.5 Estruturas de Controle

A linguagem Tcl oferece facilidades para que comandos possam ser gerados, sequenciados e condicionalmente executados. Os comandos podem ser: *if, switch, foreach, while, for, catch, error, break e continue.*

### A.3.6 Processos

Tcl fornece também algumas facilidades para o tratamento de processos. Processos podem ser criados, pode haver comunicação entre processos, variáveis de ambiente podem ser lidas e escritas e o processo corrente pode ser finalizado. Os comandos podem ser:

- exec

  O comando *exec* cria um ou mais processos e em caso normal, aguarda que o processo complete a sua execução. Os argumentos do comando *exec* podem especificar redireção de entrada e saída e pode também especificar um pipeline de um processo.

- e/s usando pipeline

  Subprocessos podem ser criados através do comando *open*. Utilizando-se os comandos *gets* e *puts*, padrões podem ser lidos e escritos nas entradas e saídas dos subprocessos.

- variáveis de ambiente

  As variáveis de ambiente podem ser lidas e escritas utilizando o mecanismo padrão de variáveis do Tcl.

- término da execução de um processo

  O comando *exit* pode ser usado para terminar o processo no qual o comando está sendo executado.

## A.4 Fundamentos do Tk

Tk (*toolkit*) é um conjunto de ferramentas usadas para programação de interfaces gráficas ao usuário [Oust 94]. Foi projetado para o sistema de janelas X usado nos sistemas UNIX e agora após atualizações nas versões disponíveis pode ser instalado em sistemas Macintosh e Windows. Tk compartilha muitos conceitos com outros conjuntos de ferramentas envolvendo janelas, mas não é necessário ter muito conhecimento sobre interfaces gráficas ao usuário para iniciar a utilização do mesmo.

Tk define comandos em Tcl que permitem a criação e manipulação de interface *widgets*. Esta ferramenta baseada em *scripts* para programação de interfaces com o usuário apresenta os seguintes benefícios:

- Desenvolvimento rápido, já que não é necessário esperar pelo tempo de compilação
- Uma interface mais alto nível que a maioria das *toolkits* em forma de bibliotecas na linguagem C
- A interface pode ser produzida de forma disjunta com a aplicação. O programador pode se concentrar na implementação da sua aplicação e entaõ construir de forma fácil a interface com o usuário.

### A.4.1 Breve introdução ao Sistema de Janelas

O sistema de janelas X gerencia recursos típicos de um ambiente de janelas tais como eventos, criação e remoção de janelas, dispositivos de entrada e saída (mouse, teclado, vídeo e outros) [Oust 94, Welc 97].

Permite aplicações para criar e destruir janelas, mover e redefiní-las dentro das janelas principais, desenhar saídas gráficas, tais como texto, linhas e *bitmaps* (desenhos pré-feitos).

Envia eventos para notificar aplicações das ações do usuário, tal como pressionar tecla, movimentar ponteiro indicador e pressionar botão.

Pode também gerar eventos para modificações estruturais, tais como redefinição e destruição de janelas.

Um ambiente X contém 3 tipos de processos: servidores X, aplicações e gerenciadores de janelas. Para cada *display* existe um processo servidor X que gerencia o *hardware* do *display* e hierarquia de janelas, desenha gráficos e gera eventos. Aplicações são os processos que fornecem serviços de interesse aos usuários. Essas aplicações se comunicam com os servidores X através de protocolos de rede, tais como TCP/IP (Transmission Control Protocol/ Internet Protocol). Isso significa que uma aplicação pode desenhar na tela de qualquer máquina existente em sua rede (sujeito a vários controles de acesso) e

uma única aplicação pode criar janelas em múltiplos *displays*. O terceiro tipo de processo em um ambiente X é um gerenciador de janelas, um para cada *display*. Permite ao usuário manipular as janelas principais de uma maneira uniforme para todas as aplicações.

Uma aplicação X deve se comunicar com o servidor X e com o gerenciador de janelas. A aplicação se comunica com o servidor X para criar janelas, desenhar nelas e receber eventos. O servidor X se comunica com o gerenciador de janelas para especificar títulos, dimensões desejadas e outras informações para suas janelas principais.

### A.4.2 *Widgets*

Tk utiliza X para implementar um conjunto pré-existente de controles. Esses controles são chamados *widgets*. Cada *widget* é um membro de uma classe que determina sua aparência e comportamento.

*Widgets* do Tk são organizados em uma hierarquia. Para uma aplicação, a hierarquia de janelas significa que existe uma janela primária principal e dentro dessa janela pode haver uma certa quantidade de janelas secundárias. Por sua vez as janelas secundárias podem conter mais janelas e assim por diante.

A hierarquia afeta o esquema de nomeação usado por *widgets* do Tk e isso é usado para ajudar a dispor os *widgets* na tela.

Tk fornece quatro grupos principais de comandos Tcl que:

1. criam e destroem *widgets*
2. arranjam *widgets* na tela
3. se comunicam com *widgets* existentes
4. interconectam *widgets* dentro e entre aplicações

Uma aplicação baseada em Tk possui um fluxo de controle dirigido ao evento, como a maioria dos conjuntos de ferramentas de sistemas de janelas. Os *widgets* do Tk maneja a maioria dos eventos automaticamente. Para comportamentos especializados, é usado o comando *bind* para registrar um comando do Tcl que é acionado quando um evento ocorre. Existem muitos eventos que incluem, movimentação de *mouse*, acionamento de teclas, dimensionamento de janelas e destruição de janelas.

A estrutura básica de um *script* Tk começa criando *widgets* e arranjando-os com um gerenciador de geometria e então vinculando ações aos *widgets*. Após o interpretador processar os comandos que inicializam a interface ao usuário, entra-se no ciclo do evento e então a aplicação é acionada.

### A.4.3 Nomeando *Widgets* do Tk

Tk usa um sistema de nomeamento para os *widgets* que refletem sua posição na hierarquia dos *widgets*. A raiz da hierarquia é a janela principal da aplicação e seu nome é simplesmente um ponto (.).

Tk fornece os seguintes tipos de *widgets*:

- *button* - um botão de acionamento normal. De acordo com a figura A.1, os botões podem ser representados de diversas maneiras. O programador define cores, espessuras das bordas, tipo de fonte do texto identificador do botão, assim como qual o comando a ser associado quando o botão for acionado.


Figura A.1: Button

- *canvas* - uma forma de área de desenho estruturada. Na figura A.2 tem-se exemplos de alguns objetos que podem ser incluídos em uma dada aplicação.

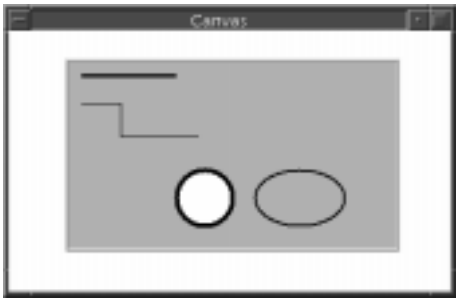


Figura A.2: Canvas

- *checkbutton* - botão do tipo liga-desliga. Na figura A.3 tem-se exemplo de checkbutton em estado normal e no estado acionado.

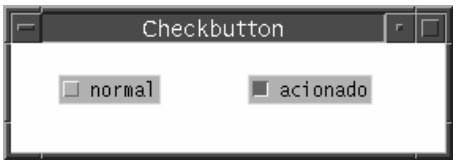


Figura A.3: Checkbutton

- *label* - um rótulo de texto estático.

- *entry* - campo de uma linha para entrada de texto. Na figura A.4 foi feito um exemplo utilizando-se *entry* e *label* para a saída gráfica.

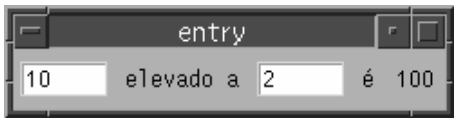


Figura A.4: Entry e Label

- *frame* - um recipiente para outros *widgets*; deve possuir bordas limitadoras. As bordas podem ser do tipo: *flat*, *raised*, *groove*, *ridge* e *sunken* e estão representadas na figura A.5.

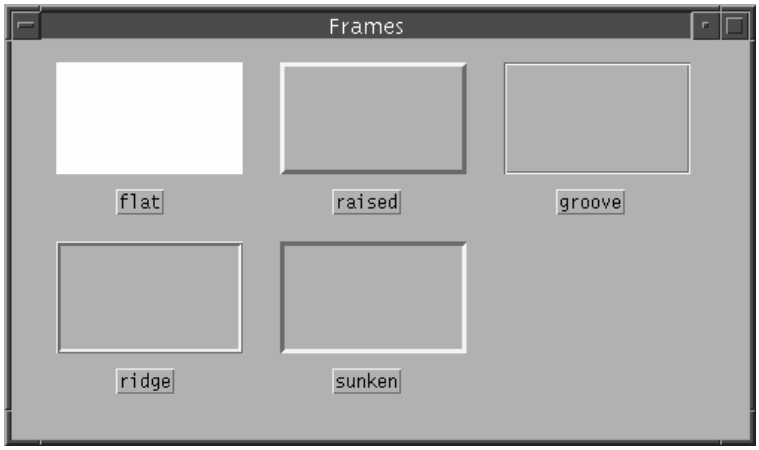


Figura A.5: Frames

- *listbox* - uma lista de escolhas. Dependendo do tamanho da lista armazenada e a forma de saída da lista, existe a necessidade de inclusão da barra de rolagem com o comando correspondente associado. Esse tipo de exemplo pode ser visto na figura A.6.

- *menu* - menu de opções normal.

- *menubutton* - situado na barra de menu. Faz com que o menu seja aberto.

- *message* - rótulo de texto com várias linhas.

- *radiobutton* - botão do tipo liga-desliga; somente um botão desse tipo em um dado frame pode estar ativo de cada vez.

- *scale* - entrada analógica de um mínimo até um máximo. Os valores da escala do intervalo de variação da escala podem ser mostrados, como no exemplo da figura A.7, ou podem ser omitidos, sendo mostrado somente o valor atual onde o cursor da escala se encontra.


Figura A.6: Listbox

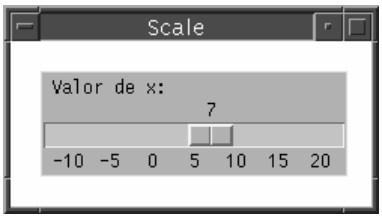


Figura A.7: Scale

- *scrollbar* - barra de rolagem, normalmente atua junto com *listbox* ou uma janela de texto. Na figura A.8 tem-se um exemplo de scrollbar vertical e horizontal atuando em uma janela de texto.

- *text* - janela para edição de textos com várias linhas.

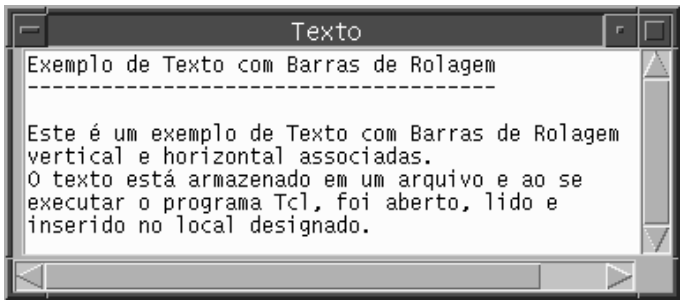


Figura A.8: Text

### A.4.4 Gerenciadores de Geometria

*Widgets* não determinam seu próprio tamanho ou localização em um tela. Esta função é executada pelos gerenciadores de geometria. Cada gerenciador implementa um estilo particular de disposição.

Um gerenciador de geometria utiliza um *widget* como uma tela principal e dispõe as múltiplas telas adicionais dentro da tela principal. A tela principal é na maioria das vezes um *frame*, mas não necessariamente. Um *widget* só pode ser gerenciado por um gerenciador de geometria de cada vez, mas podem ser utilizados gerenciadores diferentes para controlar *widgets* diferentes na interface ao usuário. Se um *widget* não for gerenciado, então ele não aparecerá de modo algum na tela. O empacotador é um poderoso gerenciador de geometria baseado em restrições. Ao invés de especificar em detalhes a localização de cada janela, o programador define algumas restrições sobre como as janelas deveriam ser posicionadas e o empacotador elabora os detalhes.

Tk atualmente possui três gerenciadores de geometria:

- *grid* - permite dispor objetos em forma matricial, como numa planilha com as bordas de células invisíveis.
- *pack* - dispõe uma série de objetos sequencialmente ao redor da margem de um determinado espaço, respeitando contudo o espaçamento solicitado por cada um dos objetos. Pode ser usado para gerar linhas, colunas e outros tipos de arranjos.
- *place* - fornece simples locações fixas com um gerenciador de geometria interno dentro do *widget* canvas, que pode ser usado para mesclar objetos com estruturas gráficas. Este gerenciador implementa posicionamento fixo através de especificação da posição e tamanho de cada janela em relação a janela principal.

Na figura A.9 tem-se exemplos de disposição de objetos em determinado espaço físico na janela de saída de acordo com o tipo de gerenciador de geometria.

### A.4.5 Associando Comandos a Eventos

O termo *binding* em Tcl/Tk diz respeito à associação de comandos Tcl a eventos de um sistema de janelas, como por exemplo o sistema X de janelas. Os eventos podem ser: pressionar uma determinada tecla, soltar alguma tecla, pressionar ou soltar algum dos botões do mouse, entrar ou sair com o indicador de posicionamento em alguma janela, alterar o tamanho de janela, abrir ou fechar alguma janela, entre outros. *Bindings* são ferramentas muito importantes e bastante utilizadas na contrução de interfaces gráficas.

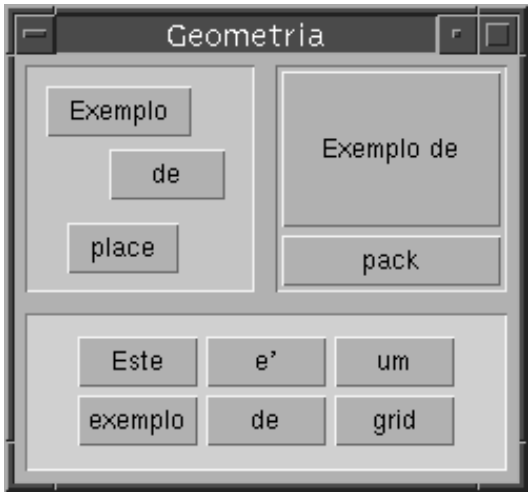


Figura A.9: Gerenciadores de Geometria

# Apêndice B

# Arquivos de Dados

## B.1 Elaboração dos Arquivos de Dados

Neste apêndice são apresentados os arquivos de dados elaborados, assim como o detalhamento para a sua correta formação. Os arquivos de dados necessários para a execução do programa elaborado, foram divididos nas seguintes categorias:

- arquivos com dados topológicos
- arquivos com dados dinâmicos
- arquivos com dados auxiliares
- arquivos com dados temporários

Sendo que os arquivos topológicos foram por sua vez subdivididos em:

- arquivos com os dados topológicos de configuração do sistema e das subestações que fazem parte do sistema
- arquivos com os parâmetros elétricos dos componentes das subestações

Os arquivos de dados criados para cada subestação foram:

### B.1.1 Arquivos com Dados Topológicos

Neste conjunto de dados se encontram os dados necessários para a configuração física dos diagramas unifilares das subestações, do sistema geral e dos elementos que compõem

as subestações. De acordo com o tipo do elemento a ser representado na tela de saída, são fornecidos os dados adequados. Os tipos de elementos podem ser:

- Elementos que são desenhados:
  - linha de transmissão: coordenadas (x e y) iniciais e finais de cada segmento; espessura da linha
  - barramento: coordenadas (x e y) inicial e final; espessura da linha
  - textos em geral: coordenadas (x e y) de localização; texto
  - gerador: um *bitmap* previamente feito a ser inserido na saída gráfica, devem ser fornecidas as coordenadas (x e y) de localização
  - transformador: coordenadas (x e y) de localização do *bitmap* com a representação de transformador
- Elementos com variáveis associadas:
  - tensão: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres); dimensões (largura e altura) da janela de saída do valor da variável; coordenadas (x e y) de localização da unidade de medida da variável; valor medido [1] ou valor calculado; identificação da janela de saída.
  - fluxo ativo: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres); dimensões (largura e altura) da janela de saída do valor da variável; coordenada (x e y) de localização da unidade de medida da variável; valor medido ou valor calculado; identificação da janela de saída.
  - fluxo reativo: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres); dimensões (largura e altura) da janela de saída do valor da variável; coordenada (x e y) de localização da unidade de medida da variável; valor medido ou valor calculado; identificação da janela de saída.
  - geração de potência ativa: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres); limite mínimo de geração ativa; limite máximo de geração ativa; valor medido ou valor calculado; nome da janela de saída do valor da variável.
  - geração de potência reativa: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres); limite mínimo de geração reativa; limite máximo de geração reativa; valor medido ou valor calculado; nome da janela de saída do valor da variável.
  - carga ativa: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres); limite mínimo de carga ativa; limite máximo de carga ativa; valor medido ou valor calculado; nome da janela de saída.

[1] A existência de medidores em pontos diversos no sistema faz com que os valores das variáveis definidas, onde esses medidores se encontram, sejam do tipo "valores medidos" sendo representados de maneira adequada na tela de saída, possibilitando também que sejam efetuadas alterações nos valores.

  - carga reativa: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres) ; limite mínimo de carga reativa; limite máximo de carga reativa; valor medido ou valor calculado; nome da janela de saída.
  - tap de transformador: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres); valor mínimo do tap de transformador; valor máximo do tap de transformador; valor medido ou valor calculado; nome da janela de saída.
  - disjuntor: coordenadas (x e y) de localização da janela de saída; nome da variável (7 caracteres); identificação (número) do disjuntor.

- arquivos com especificações dos parâmetros elétricos dos elementos

  Várias informações que serão utilizadas tanto pelas rotinas Fortran quanto pelo programa Tcl estão contidas nestes arquivos. Os grupos de informações estão separados por indicadores pré-definidos (*FF, fim e fim*). Os grupos são:

  - circuitos e elementos associados

    Os circuitos definidos estão aqui especificados. Cada circuito está associado a um elemento que de acordo com o tipo deve possuir determinadas especificações, que descrevem os parâmentros necessários para obter o desempenho elétrico da rede. Os tipos possíveis e as informações necessárias são:

    * linha de transmissão: nível de tensão (kV); resistência da linha (pu) ; reatância da linha (pu); admitância shunt da linha (pu); limite máximo (MW)
    * barramento: nível de tensão (kV); identificação da variável associada.
    * segmento: nível de tensão (kV)
    * transformador: resistência (pu); reatância (pu); tap nominal (pu); tap mínimo (pu); tap máximo (pu); limite ativo máximo (MW).
    * gerador: nível de tensão (kV); reatância do gerador (pu); potência ativa máxima (MW); potência reativa mínima (MVAr); potência reativa máxima (MVAr); identificação do gerador.
    * carga: nível de tensão (kV); carga ativa (MW); carga reativa (MVAr)

  - disjuntores e circuitos associados

    Cada disjuntor deve possuir os números dos circuitos associados antes e após. Deverá ser fornecido o nome da variável que representa cada disjuntor (7 caracteres) e os números dos circuitos (2 caracteres) separados por um espaço.

  - variáveis do sistema

    Neste grupo, de acordo com o tipo da variável existe um indicador associado. As informações necessárias são:

    * tipo (V, MW ou MVAr) - (4 caracteres: volt, mwat ou mvar);
    * valor medido ou valor calculado ("m" ou "c");
    * nome da variável (7 caracteres);
    * nome da janela de saída do valor da variável (4 caracteres);

- número do circuito associado (2 caracteres);
- valor da variável.

As informações acima devem ser separadas por um espaço.

A seguir um exemplo de arquivo de dados da subestação 4.

```
Linha 40 30 40 90 40 30 40 90 1
Linha 40 110 40 175 40 175 70 175 1
Linha 70 100 70 250 70 100 70 250 1
Arco 30 90 50 110 1
Linha 160 30 160 90 160 30 160 90 1
Linha 160 110 160 175 160 175 190 175 1
Linha 190 100 190 250 190 100 190 250 1
Arco 150 90 170 110 1
Segmento 380 240 380 175 380 175 420 175 1
Segmento 420 100 420 250 420 100 420 250 1
Segmento 380 260 380 334 380 260 380 334 1
Segmento 380 365 380 435 380 365 380 435 1
ArcoSeg 370 240 390 260 1
Segmento 100 435 100 465 100 435 100 465 1
Segmento 100 485 100 540 100 540 131 540 1
Segmento 131 475 131 605 131 477 131 605 1
ArcoSeg 90 465 110 485 1
Segmento 203 475 203 605 203 475 203 605 1
Segmento 203 540 246 540 246 540 246 595 1
Segmento 246 615 246 640 246 640 326 640 1
Segmento 360 640 645 640 360 640 645 640 1
ArcoSeg 236 595 256 615 1
Barra 20 100 610 100  4 4007
Barra 20 250 610 250  4 4006
Barra 60 435 200 435 4 4005
Barra 220 435 480 435 4 4004
Barra 60 475 480 475 4 4003
Barra 60 605 400 605 4 4002
Barra 505 615 505 670 4 4001
TENSAO 510  75 v_0407m 48 18 555  75 m 38
TENSAO 510 225 v_0406m 48 18 555 225 m 39
TENSAO 440 410 v_0404m 48 18 495 410 c 40
TENSAO 100 410 v_0405m 48 18 150 410 c 41
TENSAO 440 450 v_0403m 48 18 495 450 m 42
TENSAO 380 580 v_0402m 48 18 435 580 m 43
TENSAO 490 680 v_0401m 48 18 545 680 m 44
FLUXOP  50  45 p_0401m 50 18 100  45 m 45
FLUXOP 170  45 p_0402m 50 18 220  45 c 46
```

```
FLUXOP 290 170 p_0403m 50 18 340 170 m 47
FLUXOP 270 550 p_0404m 50 18 320 550 m 48
FLUXOP 280 380 p_0405m 50 18 330 380 m 49
FLUXOQ  50  65 q_0401m 50 18 100  65 m 50
FLUXOQ 170  65 q_0402m 50 18 220  65 c 51
FLUXOQ 290 190 q_0403m 50 18 340 190 m 52
FLUXOQ 270 570 q_0404m 50 18 320 570 m 53
FLUXOQ 280 400 q_0405m 50 18 330 400 m 54
FLUXOP   0   0 p_0406m  0  0   0   0 c 59
FLUXOQ   0   0 q_0406m  0  0   0   0 c 60
GERACAOP 590 560 pg0401m -250 500 m 55
GERACAOQ 710 560 qg0401m -200 590 m 56
TAP 430 315 tap4_2m 900 1100 m 57
TAP 265 660 tap4_1m 900 1100 m 58
DISJUNTOR  57 127 disj_60 60
DISJUNTOR  57 200 disj_61 61
DISJUNTOR 177 127 disj_62 62
DISJUNTOR 177 200 disj_63 63
DISJUNTOR 408 125 disj_64 64
DISJUNTOR 408 200 disj_65 65
DISJUNTOR 368 290 disj_66 66
DISJUNTOR 368 385 disj_67 67
DISJUNTOR 180 425 disj_68 68
DISJUNTOR 117 490 disj_69 69
DISJUNTOR 117 565 disj_70 70
DISJUNTOR 186 490 disj_71 71
DISJUNTOR 187 565 disj_72 72
DISJUNTOR 258 630 disj_73 73
DISJUNTOR 380 630 disj_74 74
DISJUNTOR 555 630 disj_75 75
DISJ_TRAF   0   0 disj_76 76
DISJ_TRAF   0   0 disj_77 77
Transf 383 352 2
Transfv 346 642 1
Texto 420 355 T2
Texto 350 620 T1
SE 18 15 um Um 1 m
SE 138 15 dois Dois 2 m
BOTAO 670 50 quatro QUATRO_Est 4 c
Gera 662 645
Texto 720 30 SUBESTAO QUATRO - Valores Medidos
SUBESTACAO 4
```

### B.1.2 Arquivos com Dados Dinâmicos

Os arquivos que contém dados dinâmicos do sistema são criados ao longo da simulação, pelas próprias rotinas do programa Tcl/Tk. A formação desses arquivos pode ser de duas maneiras: os dados são obtidos a partir das telas de saída de cada subestação, ou seja, os dados representam o estado atual de cada variável que podem ter sido modificados ou não pelo usuário, ou resultam dos cálculos efetuados pelos programas de cálculo do Fluxo de Carga ou Estimador.

Optou-se pela criação de arquivos relativos a cada subestação que compõe o sistema em estudo. Portanto, no primeiro caso onde os arquivos representam o estado atual das variáveis de cada subestação, a Subestação 1, por exemplo terá associado um arquivo com os dados dinâmicos de suas variáveis (estado dos disjuntores - aberto (0) ou fechado (1)) e os valores atuais (representados na tela de saída) das variáveis: tensão, fluxo ativo e reativo nas linhas de transmissão, potência ativa e reativa dos geradores, consumo ativo e reativo das cargas e o posicionamento dos taps dos transformadores. No caso onde os arquivos de dados são formados a partir dos resultados dos cálculos efetuados, é feita uma seleção dos dados através de rotinas (*procedures*) do próprio programa principal escrito na linguagem Tcl, para cada subestação, os dados necessários são devidamente tratados, podendo ser mostrados diretamente na tela de saída ou copiados em arquivos correspondente a cada subestação para uso posterior.

### B.1.3 Arquivos com Dados Auxiliares

Neste caso estão incluídos os arquivos que são criados durante a simulação, onde o usuário efetua o salvamento de tantas quantas forem as configurações que julgar necessárias para estudo posterior. No momento em que o usuário desejar, o caso em estudo pode ser guardado, podendo ser acessado novamente a qualquer instante.

Para a preparação dos arquivos de dados que representam as curvas de cargas das injeções que fazem parte do sistema, foram seguidos os seguintes passos:

- cada dia da semana (domingo a sábado) possui seu próprio arquivo de dados
- cada injeção ativa e reativa é devidamente nomeada (p. ex.: p1 representando a injeção ativa existente na subestação 1)
- determina-se o fator multiplicador de cada injeção e os demais valores percentuais hora a hora a partir de zero hora

Os valores correspondentes a cada injeção ficam em uma linha do arquivo de dados (formato texto); a primeira informação é a identificação da injeção, após um espaço segue

o valor do fator multiplicador; a seguir, novamente após um espaço tem-se os valores (percentuais) das injeções correspondentes a cada hora do dia (de zero a 23 horas).

O motivo de se criar o fator multiplicador é o de facilitar as modificações que possam vir a ser feitas por parte do usuário, como por exemplo, mudar a característica da curva de carga de determinada injeção.

A seguir tem-se como exemplo as curvas de carga relativas às injeções existentes na subestação 1. São mostradas as curvas referentes ao dia da semana quarta-feira.

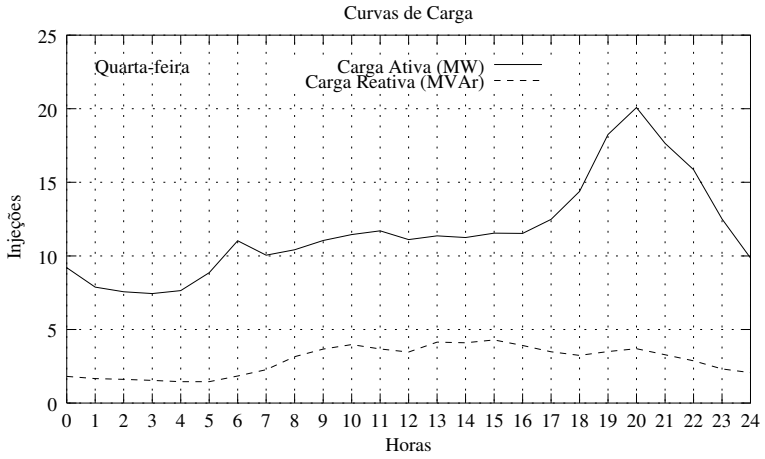


Figura B.1: Curvas de Carga - Quarta-feira

### B.1.4 Arquivos com Dados Temporários

Arquivos com dados temporários são aqueles que são criados temporariamente durante a simulação para utilização em determinado momento, não sendo mais necessários após seu uso. Como a criação deste tipo de arquivo está diretamente relacionada com a execução do programa de simulação, o usuário não tem o poder de modificar esses dados, que são criados/destruídos ao longo da simulação.

Fazem parte deste tipo de arquivo de dados, os arquivos que fazem a interface entre o programa escrito em Tcl e os programas escritos na linguagem Fortran.

# Apêndice C

# Divulgação da Dissertação

## C.1 Artigo aceito no IEEE-PES / CSEE Intenational Conference on Power System Technology - PowerCon 2000 - Austrália

# A Real Time Operation Power System Simulator

Regina M. Kawakami
UNICAMP/FEEC/DSEE - Campinas - SP - Brazil
regina@dsee.fee.unicamp.br

Ariovaldo V. Garcia
UNICAMP/FEEC/DSEE - Campinas - SP - Brazil
ari@dsee.fee.unicamp.br

*Abstract*—**The complexity of the real time power system operation is so high that several tools must be available in order to help training electric management systems (EMS) operators and also teaching engineering graduate and undergraduate students. Computer based training systems have played an important role in providing users with a better understanding of power system operation. They have witnessed steady progress since the early 1980's. In this article the authors' objective is to present the development of a Tcl/Tk (Tool Command Language/Tool Kit) GUI (Graphical User Interface) based real-time power system operation simulator program. Users can dynamically interact with the target system by using an one-line diagram to change system measurements (MW, MVAr and kV) and also change transformers taps and circuit breaker status.**

**Keywords:** Power system simulator, real-time network analysis, Tcl/Tk, Operating Training Simulator

## I. Introduction

Real time power system operation is so complex that there is the need to make available tools that can help for teaching engineering undergraduates and personnel (operators) involved in the electricity industry. As pointed out by [1, 2], computer based training systems have played an important role in providing users with a better understanding of power system operation. In this work the authors' objective is to present a development of a Tool Command Language/Tool Kit (Tcl/Tk) [3] graphical user interface (GUI) based power system simulator program. Users dynamically interact with the target system by using an one-line diagram to change system controls including generator and load outputs, transformer tap positions and circuit breaker status. Since all routines were written in Fortran and the graphical interface is a script written in Tcl/Tk language, the developed simulator can be run in most of the known platforms (UNIX, MS Windows, Linux, etc).

The paper is organized into the following sections: Section II presents a briefly description of real time power system structure including EMS general schema, systems structure for real-time network analysis, databases, and real-time network analysis functions (network configuration and state estimator). Section III introduces power system simulators. In section IV we present the simulator developed by the authors. In section V a brief overview of Tcl/Tk system is presented. Section VI shows the general conclusions.

## II. Real time control of power systems

One of the main features of an energy management system (EMS) is the Training Simulator, which objective is mainly training systems operators. It can be used also to help teaching power systems students. The simulator and the EMS environments must be the same for obvious reasons. Some decades ago power system operation digital simulation was performed simply by running power flow programs. In this kind of simulation, all data (network topology, active and reactive power injections, controlled buses specified voltage, etc.) are prepared by the user and then the problem is submitted to the solution calculation. Even today power flow programs are used to analyze future operating scenarios of the system. Requirements are quite different when we deal with power system real-time operation (or control). In an EMS we have the Supervisory Control And Data Acquisition (SCADA), several control functions (Network Configurator, Network Observer, State Estimator, Load Dispatch, Optimal Power Flow, Security Analysis) if we only mention the advanced software. It is clear that there is much more available information: Circuit breakers status (open/closed), measurements (bus voltages, branch active and reactive power

flows, bus active and reactive power injections, transformer taps, etc.). A simulator must be, therefore, capable of deal with all these resources.

*A. EMS general schema*

A modern EMS has basically

- Data acquisition system
- Computers system
- Basic and advanced software

*B. System structure for real-time network analysis*

The main objective of an EMS is to obtain the secure power system operation. This is maybe the biggest known power system analysis problem and to solve it means to answer the simple question "is the system now operating in a secure mode?". In order to solve this problem we need first to obtain a *real-time model* of the system. To obtain such model is the final objective of several advanced software functions normally called "network analysis functions". This model can be seen as the actual network structure plus the actual system state variables.

Available data are:

- Monitored substations topology.
- Substations devices and transmission lines.
- Circuit breaker status (open/closed).
- Measured values for power injections and magnitude voltages at buses, for power flows in transmission lines and transformers and for line currents and transformers taps

In a simple manner the real-time model construction can be divided into three main phases: Electric network configuration, state estimation (composed by observability analysis, state estimation and bad data processing) and network model determination. Notice that the observable part of the system can change at any time since the measurement set and system topology changes continually.

*C. Databases and their structure*

The data organization plays an important role in an EMS since we have to take decisions in short periods of time. The access to the data must be efficient and secure. Some functions update the real-time databases (dynamic) while the major part of the data is static (topological data, parameters, etc.). The data maintenance must be performed with care and by specialized people.

Some features of the database design are:

- be well organized such that its access is optimized;
- optimize changes that can occur during the operation;
- optimize the creation and maintenance in order to make easy new equipments as well as new functions adding.

*D. Real-time network analysis functions*

- Network Configurator: each system topology modification (circuit breakers operation) or by demand of the Control Center Operator solicitation starts the process. The configurator is responsible for the electric model construction (How many nodes, and how they are interconnected, and also where are the measurements.).
- State Estimator: The Configurator results are used by the state estimator, which is in fact composed by:
  - Network Observer - determines the observable (s) part (s) of the system
  - State Estimator (and Bad Data processor) - obtains the most probable system state

## III. Simulators

A real-time power system operation simulator can provide a realistic environment for trainees to practice operating tasks under normal, emergency and restorative conditions [4]. It may also be used for other purposes, such as, engineering studies, power system model evaluation, off-line testing of energy management system functions and operating procedures. The power system realism, as defined in the modeling requirements, will provide the training capability for the various system states and trainee skill levels.

A simulator can be developed initially with a minimum set of simulation capabilities, and functions can be added at a later time as required [6]. This is what we have done. This building block approach provides the individual utility with the ability to specify the level and detail of simulation and system capability as required to satisfy their present or future needs.

## IV. The developed simulator

In the simulator described in this paper, we assume that all the loads (active and reactive power) are known (these values can be easily changed bye the user at any time). These loads can be obtained from daily load curves (in this case we only set the week day and time) similar to the one shown if Fig 1, or then arbitrarily set by the user. Also the generator outputs (active power dispatch) are known. The user set all these values and also the system

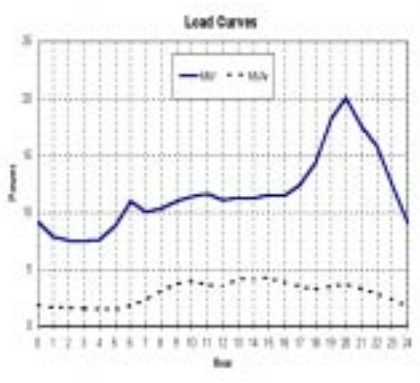


Fig. 1. Load Curves

topology through setting the state of the circuit breakers. With this data the system topology can be obtained (Network Configurator task). Notice that the system may be split in several islands. This means that the load flow and state estimator must be able to deal with this kind of network. To represent the "true values" of the system (active flows, injections voltages, etc.) a load flow is run (or then a state estimator using a non-redundant measurement set). When this step is finished the user can change any measurement, circuit breaker status, transformer tap value and see the effect on the system. Bad data can be inserted in measurements or in the topology, random errors etc.

*A. Simulator execution modes*

In an initial phase a set of starting routines is executed, in order to set initial values of variables, to prepare data files, etc. In the sequence an initial graphical window is displayed to the user who can start the simulation. Two execution modes are implemented in the developed simulator: - Study Mode and Automatic Execution Mode. In the Study Mode, user can set any system data value and simulates the power system operation, state estimation etc.. In the Automatic Execution Mode we try to simulate what really happens in an EMS. Initially the user can set only the initial simulation time (week day, and time). The active and reactive loads at each bus are continuously changed according to their daily load curve. In this mode user can interact with the simulator by changing the system topology or changing some parameters, like transformer taps, voltage level at buses where it is permitted.

*B. System data files*

As already pointed out the simulator we developed uses Tcl/Tk scripts to make the graphical windows (with diagrams) and also to manage the comunication between the user and the simulator. In spite of being a script language, the performance of Tcl/Tk is quite good (as compared with Java language, for instance). All the CPU intensive calculations (Configurator, State Estimator, Load Flow) software were written in Fortran and are implemented in the EMS center of a local utility[5]. The tests were performed in Sun workstations (Ultra 1) and also in PCs. Notice that we only made tests with small systems to test the graphical interface. The data flow scheme is shown in Fig. 2. Arrows indicate the data allowed flow. There are exclusive data files for Tcl routines, as well as for Fortran routines. Tcl routines generate data files that will be input data for the Network Configurator function. The files Tcl reads are those with graphical windows information and, also, results returned from network functions (e.g. State Estimator). Fig. 2 shows also which data files user has permission to modify. Some file modifications can be done by the user directly or indirectly. For example, line transmission parameters can only be changed with a direct access to the corresponding file, while injections (active and reactive power) can be changed through the graphical interface (indirectly).

Each substation has the following associated data files:

- Topologycal data files: they contain the one-line diagram information. These data files are read by the Tcl program to build the diagram with all the objects correctly placed. They also contain the substation electric parameters.
- Dynamic data files: they are created during the simulation cycle by the Tcl and Fortran routines.
- Auxiliary data files: daily load curves are included in this data file type. Fig. 1 shows an example of a daily load curve (these are real data obtained from a local utility - Companhia Paulista de Força e Luz) . Active and reactive power change according to the time and the week day. In the example a Wednesday load curve is presented. When necessary a linear interpolation between two consecutive points is made to obtain data at any time of the day.
- Temporary data files: during the simulation some files are necessary and the user has no access to them.

The graphical interface permits the user interaction with the simulator by simply choosing a device which permits change or its state (circuit breaker as an example) or its value (loads, transformer taps, MW and MVAr measurements). The one-line diagram uses graphical symbols

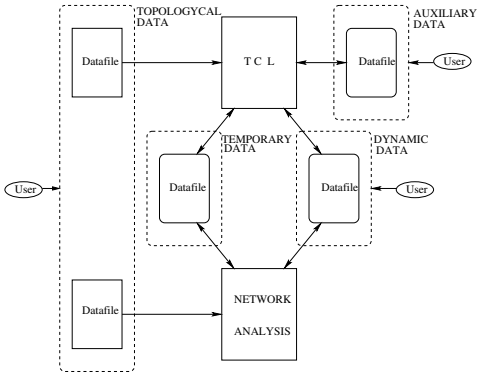


Fig. 2: Data flow

to represent system equipment such as buses, transmission lines, generators, loads, circuit breakers and transformers. Numeric values are used to show system values – see Fig. 3. User can navegate through substation diagrams by simply pressing an icon (*Tcl/Tk push button*) that has its name. Circuit breakers (represented by push buttons) can be opened or closed in a similar manner. The color of the push buttons changes if it is open or closed. The active and reactive power outputs of the generator are modified by positioning the cursor on the *Tcl/Tk scale symbol* placed near the generator symbol and clicking with the left button to increase or decrease the values. Similarly, loads outputs and transformer taps can also be modified in the same way. Bus voltage magnitudes are represented by numeric values in *Tcl/Tk listbox* symbols. Their values can be modified just by positioning the cursor on the listbox and changing the value typing on the computer keyboard the desired value.

When the simulator is started, the target the entire system is shown with all substations, transmission lines, loads and generators Fig. 4. The diagram of a substation can be viewed in details by clicking the substation symbol. Once a substation is selected by the user, it is possible to make modifications on its the network topology by changing the circuit breakers status and power outputs (generator and loads).

Fig. 5 shows command buttons detail. They are placed on the window top and when pressed the associated function is executed. For example, Base Case Calculation button, performs the initial load flow with the actual data. Notice that only the functions that can be executed at that time are displayed in this window. For instance, at the beginning only the Base Case Calculation and Reset

Fig. 5. Command buttons detail

buttons are displayed. In the example of Fig. 5 the available functions are Automatic Execution, State Estimator, Reset and Base Case Calculation. After running a State Estimator (or Automatic Execution), user can choose the displayed window: Measured Values or Estimated Values.

Fig. 6. Further information detail

The bottom of the graphical window is detailed in Fig. 6. This figure shows the actual analysis function that is being performed. (In this case, Automatic Execution.) Other informations are: number of executions and the time used to obtain the data from the daily load (in the example 1h 21' 36" PM that corresponds to 13.36). Notice that Fig. 5 and Fig. 6 have different scales.

It is possible to save cases for future studies from the Menu bar. All the system variables values are stored in a file and later the saved file can be accessed by the user and the simulation can continue without problem.

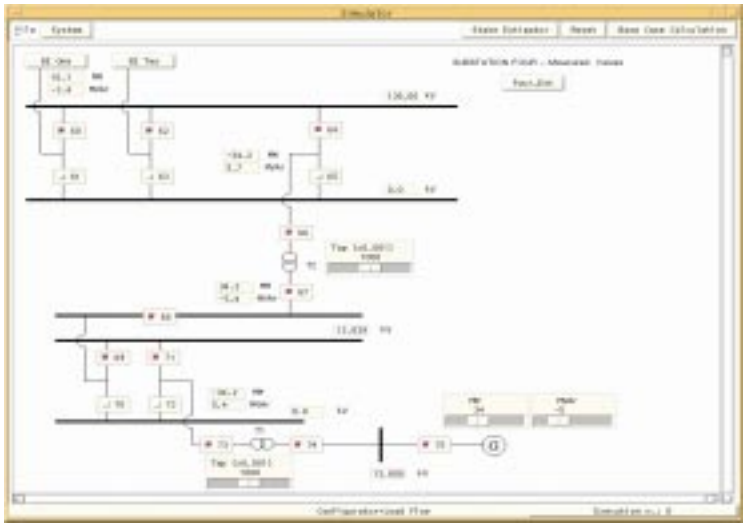

Fig. 3: Example of a substation diagram

## V. Tcl/Tk - Brief overview

Tcl stands for the Tool Command Language [3]. With its associated user interface toolkit, Tk, it is possible to create cross-platform applications. Tcl/Tk is a programming system easy to use, and which has very useful graphical interface facilities. Tcl is the basic programming language, while Tk is a ToolKit of widgets, which are graphical objects similar to those of other GUI toolkits. Unlike many of the other toolkits, it is not necessary to use C or C++ in order to manipulate the widgets, and useful applications can be built very rapidly once some expertise of the Tcl/Tk system has been gained.

Tcl supports many of the features of conventional procedural languages, including variable assignment, procedure calls, control structures, and in addition it has very easy to use access to graphical widgets. Tcl is more like a scripting language than a programming language, so it shares a greater similarity to the C shell or perl than it does to C++ or C.

Scripting Language is a simple interpreted language designed for rapid program development.

Some key features of Tcl include: Tcl is a high-level scripting language; Tcl is interpreted; it is extensible; it is embeddable; Tcl runs on many platforms; Tcl's autoloading facility makes for smaller applications, and finally Tcl is free.

## VI. Conclusions

This paper presents a real-time operation power system simulator which permits both training people involved with power system real-time operations as well as testing advanced functions algorithms, like state estimators, optimal power flow, etc. The developed simulator is portable to the majority of known platforms, since all software is written in Tcl/Tk scripts and the advanced functions in Fortran. The way the simulator was developed, permits insertion of new function in a very easy manner. All data interchange between functions is made through using data files (ASCII files) what permits simple debugging, and also permits the test of the functions in a separated way.

## VII. Acknowledgments

The authors would like to acknowledge the financial support of CAPES.

## VIII. References

[1] T.J. Overbye, P.W. Sauer, C. M. Marzinzik, G. Gross, ``A User-Friendly Simulation Program for Teaching Power System Operations", IEEE Transactions on

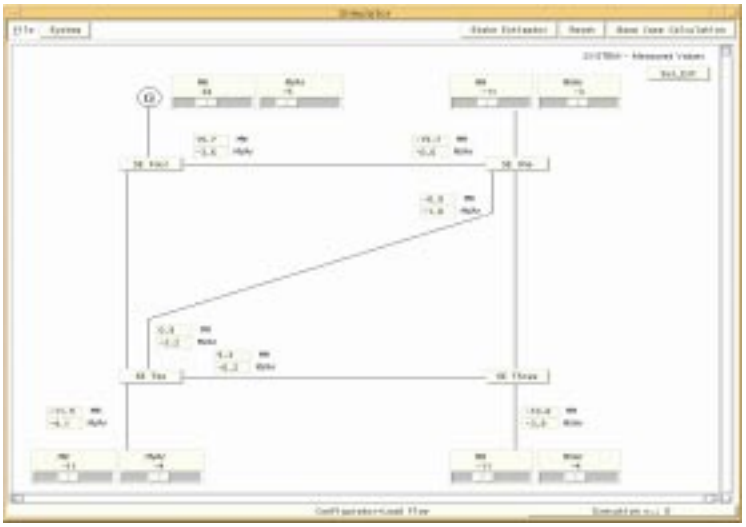

Fig. 4: System one-line diagram

Power Systems, Vol. 10, No. 4, Nov. 1995, pp. 1725-1733.

[2] J.R. Shin, W.H. Lee, D.H. Im, "A Windows-based Interactive and Graphic Package For the Education and Training of Power System Analysis and Operation". IEEE Transactions on Power Systems, Vol. 14, No. 4, Nov. 1999, pp. 1193-1199.

[3] J.K.Ousterhout, ``Tcl and the Tk Toolkit", Addison-Wesley Publishing Company, 458 p., 1994.

[4] S.V.Vadari, M.J.Montstream, and H.B.Ross Jr., "An Online Dispatcher Training Simulator Function for Real-Time Analysis and Training", IEEE Transactions on Power Systems, Vol. 10, No. 4, Nov. 1995, pp.1798-1804.

[5] L.M. Freire, F.B. Mokarzel Jr., A. V. Garcia and A. Monticelli, "Integration and evaluation of the CPFL control center functions", In: Symposium of Eletrical Systems Automation, Belo Horizonte, MG, Brazil, 1994 (in portuguese).

[6] T.E.DyLiacco, M.K.Enns, J.D.Schoeffler, J.J.Quada, D.L.Rosa, C.W.Jurkoshek, M.D.Anderson, "Considerations in Developing and Utilizing Operator Training Simulators", IEEE transactions on Power Apparatus and Systems, Vol. PAS 102, No. 11, Nov. 1983, pp. 3672-3679.

*Regina M. Kawakami* obtained her B.Eng. at EFEI, Itajubá, Brazil in 1989. From 1994 to 1995 worked as electrical engineer trainee at Chubu Electric Power Company, Japan in the Brazil-Japan Technical Interchange Program and is currently working on her M.Eng. in Electrical Engineering at UNICAMP.

*Ariovaldo V. Garcia* received the B.Sc. degree in 1974 and Ph.D. degree in 1981 from UNICAMP, Campinas, Brazil, where he is currently a Professor of Electrical Engineering. He has served as a consultant to a number of organizations. His general research interests are in the areas of planning and control of electrical power systems.