UNIVERSIDADE DE SÃO PAULO - USP
ESCOLA DE ENGENHARIA DE SÃO CARLOS - EESC
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
LABORATÓRIO DE SISTEMAS DE ENERGIA ELÉTRICA

Monografia – Trabalho de Conclusão de Curso

WAPS: *Software* para visualização e análise de faltas a partir de arquivos do padrão IEEE COMTRADE

Aluno: Athila Quaresma Santos

Orientador: Prof. Tit. Denis Vinicius Coury

São Carlos
Junho – 2011

ATHILA QUARESMA SANTOS

WAPS: SOFTWARE PARA VISUALIZAÇÃO E ANÁLISE DE FALTAS A PARTIR DE ARQUIVOS DO PADRÃO IEEE COMTRADE

Trabalho de Conclusão de Curso apresentado à
Escola de Engenharia de São Carlos, da
Universidade de São Paulo

Curso de Engenharia de Computação com
ênfase em Sistemas Embarcados

ORIENTADOR: Prof. Tit. Denis Vinicius Coury

São Carlos
2011

Ficha catalográfica preparada pela Seção de Tratamento
da Informação do Serviço de Biblioteca – EESC/USP

Santos, Athila Quaresma
S237w WAPS : software para visualização e análise de faltas a partir de arquivos do padrão IEE COMTRADE / Athila Quaresma Santos ; orientador Denis Vinicius Coury -- São Carlos, 2011.

Monografia (Graduação em Engenharia de Computação com ênfase em Sistemas Embarcados) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2011.

1. Sistemas Elétricos de Potência. 2. WAPS. 3. COMTRADE. 4. Oscilografia. 5. Proteção. I. Titulo.

"Deus de fato joga dados. E o problema é que muitas vezes Ele os lança em lugares que não enxergamos."

Stephen Hawking, astrofísico e matemático inglês

Resumo

Equipamentos de proteção como relés digitais ou ferramentas de simulação contribuem para o aumento massivo de dados relacionados à área de Proteção de Sistemas Elétricos de Potência. Com isso, um formato comum, como o padrão COMTRADE normalizado pelo IEEE, é necessário para a troca de informações entre diversos fabricantes. Desta forma, foi desenvolvida uma ferramenta de código livre denominada WAPS (*Wave Analyser for Power System*) voltada ao estudo e análise de perturbações e transitórios de redes de energia elétrica. O Laboratório de Sistemas de Energia Elétrica – LSEE, da Universidade de São Paulo – USP disponibilizou suas dependências e recursos para construção do software de manipulação de arquivos COMTRADE. Como resultado, obteve-se uma ferramenta com interface bastante intuitiva e um sistema de ajuda amigável para orientação do usuário final. As amostras podem ser visualizadas nos diversos tipos de gráficos que possuem representações distintas para cada tipo de canal provendo maior legibilidade das informações.

Palavras chaves: WAPS, COMTRADE, Oscilografia, Proteção, Sistemas Elétricos de Potência.

Abstract

Protection equipment such as digital relays or simulation tools contribute to the massive increase of data related to the protection of power systems. Then a common format, like COMTRADE pattern normalized by IEEE, is necessary to exchange information between different manufacturers. Thus, an open source tool called WAPS (*Wave Analyser for Power System*), dedicated to the study and analysis of disturbances and transient of electric power grid, was developed in this work. The Department of Electrical Engineering of Engineering School of São Carlos (EESC), University of São Paulo (USP), provided resources and facilities to build a piece of software to deal with COMTRADE files. As a result, a tool with very intuitive interface and a user-friendly help system to guide the end user was obtained. The samples can be viewed in different types of graphs that have different representations for each channel providing greater readability of the information.

Key words: WAPS, COMTRADE, Oscillography, Protection, Electrical Power Systems.

Sumário

Lista de Abreviaturas

API: *Java Application Programming Interface*. Interface de Programação de Aplicações.

COMTRADE: *Common Format for Transient Data Exchang.* Formato comum de arquivo de dados digitais e meio de troca para intercâmbio de vários tipos de faltas, testes e simulações padronizado pelo IEEE.

EOF: *End Of File*. Caractere especial ASCII para definição de fim de arquivo.

FFT: Transformada Rápida de Fourier (do inglês *Fast Fourier Transform*).

IDFT: Transformada Inversa de Fourier (do inglês *Inverse Discrete Fourier Transform*).

IEEE: *Institute of Electrical and Electronics Engineers.*

LSEE: Laboratório de Sistemas de Energia Elétrica. Laboratório alocado junto ao Departamento de Engenharia Elétrica da Escola de Engenharia de São Carlos pertencente à Universidade de São Paulo.

MDI: *Multiple Document Interfaces*. Conceito que define o uso de vários ambientes de trabalho, geralmente delimitados por janelas, em uma mesma aplicação.

POSIX, *Portable Operating System Interface for Unix*. Padrão de interface especificada pelo IEEE para garantir compatibilidade entre as diversas versões do sistema operacional Unix.

PQDIF: *Power Quality Data Interchange Format.* Formato para troca de arquivos referentes à qualidade de energia, mas que se insere no mesmo contexto do padrão COMTRADE.

RMS: valor quadrático médio (do inglês *Root Mean Square*).

TOP: The Output Processor. Ferramenta de análise de transientes assim como o WAPS.

VJM: *Virtual Java Machine*. Máquina Virtual Java. É a base da plataforma Java. Foi portada para várias plataformas de *hardware* a fim de se conseguir compatibilidade.

WAPS: *Wave Analyser for Power System.* É o *software* desenvolvido por este trabalho de

conclusão de curso e que permite a leitura e visualização de canais provenientes de arquivos COMTRADE.

Lista de Tabelas

Lista de Figuras

1 Introdução

Este capítulo é responsável por introduzir o contexto e os conceitos iniciais que levaram ao desenvolvimento da ferramenta WAPS, *Wave Analyser for Power System*. A origem e necessidade do padrão COMTRADE (*Common Format for Transient Data Exchange*), definido pelo *Institute of Electrical and Electronics Engineers* (IEEE), são mostradas assim como os requisitos necessários para implementação dessa ferramenta. Os dois módulos básicos são apresentados ressaltando-se a importância do módulo 1, foco deste trabalho de conclusão de curso. Por fim explana-se a divisão de capítulos em que foi organizado este texto.

1.1 Contexto

O aumento do uso de tecnologia digital em dispositivos de proteção, oscilografia, medição e controle, principalmente em subestações de energia elétrica, resultou na acumulação de grandes quantidades de registros digitais de eventos transitórios. Simuladores analógicos e digitais de sistemas de energia também contribuem como fontes digitais de criação, armazenamento e transmissão desses registros gerando uma grande variedade de formatos sem qualquer tipo de padronização [1].

A rápida evolução e implementação de dispositivos para registro digital de faltas e testes gera a necessidade de um formato padrão para troca de dados. Tais dados podem ser utilizados em vários dispositivos para melhorar ou automatizar análises, testes, avaliações e simulações de sistemas de energia bem como esquemas de proteção durante faltas ou situações de distúrbios. Uma vez que cada origem de dados pode usar um formato proprietário diferente, um formato comum torna-se necessário para facilitar a troca de dados entre aplicações permitindo o intercâmbio entre sistemas [2].

Neste contexto, insere-se o padrão IEEE COMTRADE que define um formato comum para arquivo de dados digitais e meio de troca para intercâmbio de vários tipos de faltas, testes e simulações. Um formato comum padronizado é de vital importância para a flexibilização e adaptação entre sistemas heterogêneos.

1.2 Definição do problema

Existe a necessidade de criação de um produto em código livre que seja funcional em diversas plataformas e que seja capaz de tornar os dados referentes aos fenômenos ligados à geração e distribuição de energia elétrica manipuláveis computacionalmente. Dentre os diversos padrões de registros de oscilografias existentes escolheu-se o padrão IEEE COMTRADE que propõe um formato comum para troca e armazenamento de dados provenientes de amostras e simulações de transientes como o formato inicial para o registro dos dados.

Embora esse formato tenha sido adotado para a implementação das diversas funcionalidades é importante ressaltar que o *software* deverá ser capaz de interpretar dados independentemente do padrão de entrada utilizando para isso uma estrutura padrão que converterá os arquivos para uma base comum. Para o reconhecimento de outros padrões será necessário apenas o desenvolvimento de módulos capazes de transformar o formato de entrada para o formato base.

O aplicativo deverá ser capaz também de trabalhar com mais de um arquivo ao mesmo tempo, ou seja, deverá ser capaz de manipular vários canais de diferentes estações. E deverá exibir a quantidade massiva de dados de uma maneira clara muitas vezes adaptando representações para gráficos de origens distintas.

Além disso, o aplicativo deverá ser capaz de realizar operações essenciais para o ambiente de proteção. É importante lembrar das quatro operações algébricas básicas (adição, subtração, divisão e multiplicação) além de operações auxiliares como inversão e negativação ($f(x) = 1/x$ e $g(x) = -x$, respectivamente).

Outras operações importantes dizem respeito à realização da Transformada Rápida de Fourier (do inglês *Fast Fourier Transform*, ou FFT) nos sinais importados para a obtenção dos espectros de freqüência, bem como a sua inversa. Também deverá ser possível a visualização do valor quadrático médio (do inglês *Root Mean Square*, ou RMS) e também o fasor de cada ponto amostrado.

Neste contexto, para atingir as características citadas anteriormente, foi desenvolvido um aplicativo de *software* que será dividido em dois módulos conforme mostra a Figura 1.

1.3 Objetivo

Este trabalho tem como objetivo o desenvolvimento de um aplicativo de *software* denominado WAPS, para o estudo e a análise de perturbações e transitórios de redes de energia elétrica.

O *software* deverá ser desenvolvido em dois módulos conforme mostrada na figura 1.

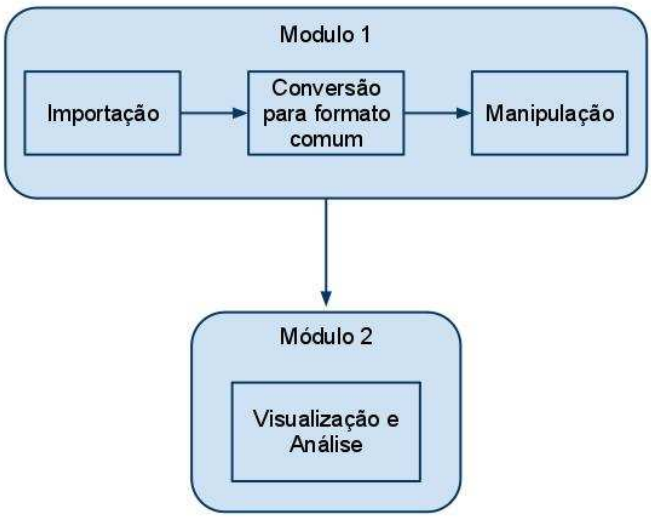

Figura 1 - Módulos do *software* WAPS

O módulo 1 deverá ser responsável por importar um arquivo oscilográfico escolhido pelo usuário para a conversão dos dados em um formato comum ao *software*. Esse formato comum deverá conter as mínimas informações necessárias para a realização das operações algébricas em cada instante de tempo da amostra, além de valores necessários para o cálculo da Transformada Rápida de Fourier, Transformada inversa de Fourier, dos valores RMS e fasoriais do sinal.

Já o módulo 2 será responsável pela exibição dos sinais importados e criados através das manipulações algébricas realizadas no módulo 1. Permitir ao usuário a visualização de maneira clara dos valores das amostras, dos fasores, dos valores RMS e dos decorrentes da

aplicação da FFT em cada instante de tempo, permitindo ao mesmo, inclusive, realizar comparações entre sinais.

O foco do desenvolvimento deste trabalho de conclusão de curso está no Módulo 1. A aquisição de dados e formulação de um formato comum independente do tipo de arquivo de entrada bem como as diversas manipulação algébricas, lógicas e de transformadas, além de ferramentas para geração dos valores RMS e fasoriais serão implementadas construindo o núcleo do *software* para posterior visualização e análise a partir do Módulo 2 que está sendo desenvolvido paralelamente a este trabalho por outra equipe de desenvolvimento.

1.4 Organização do Texto

Os capítulos 2, 3 e 4 são uma revisão bibliográfica. O capítulo 2 descreve os principais *softwares* existentes no mercado com funcionalidades distintas, mas que têm o mesmo objetivo: a visualização e análise de transientes. O WAPS tenta agregar as diversas características dessas ferramentas para extrair o melhor de cada uma.

O capítulo 3 é responsável por exibir um panorama do paradigma de orientação a objetos e uma breve descrição da linguagem Java para justificar o seu uso neste trabalho.

O capítulo 4 é responsável por descrever o padrão COMTRADE, amplamente utilizado pelo mercado e grandes instituições e que é o formato foco empregado no WAPS.

O capítulo 5 descreve as propriedades e diversas funcionalidades existentes na ferramenta desenvolvida WAPS. É dada uma visão geral de cada módulo justificando o seu desenvolvimento.

O capítulo 6 exemplifica o uso e analisa a confiabilidade da ferramenta desenvolvida através de um teste projetado para agregar as diversas funcionalidades do sistema.

Por fim o capítulo 7 descreve as experiências e conclusões inferidas a partir deste projeto de conclusão de curso.

2 Levantamento Bibliográfico

Este capítulo é responsável por levantar os principais aplicativos existentes no mercado que possuem o mesmo escopo da ferramenta WAPS ressaltando suas diferenças e destacando suas características principais.

2.1 TOP – The Output Processor

TOP (*The Output Processor*) é um *software* para visualização de dados de acompanhamento e resultados de simulação. Esse aplicativo foi desenvolvido pela Electrotek Concepts® lendo uma grande variedade de fontes e transformando-as em gráficos de alta qualidade. Possui uma grande diversidade de gráficos, utiliza MDI, *Multiple Document Interfaces*, permite a visualização de várias curvas em um único gráfico, permite a realização de diversas operações matemáticas como às operações básicas, a FFT e a IDFT (*Inverse Discrete Fourier Transform*), raiz quadrada, entre outras. Contudo, não possui um cursor integrado entre os gráficos, o que dificulta a comparação dos valores entre eles. É disponível de forma gratuita e atualmente está na versão 6.02 [3]. A Figura 2 mostra a interface gráfica do aplicativo citado.

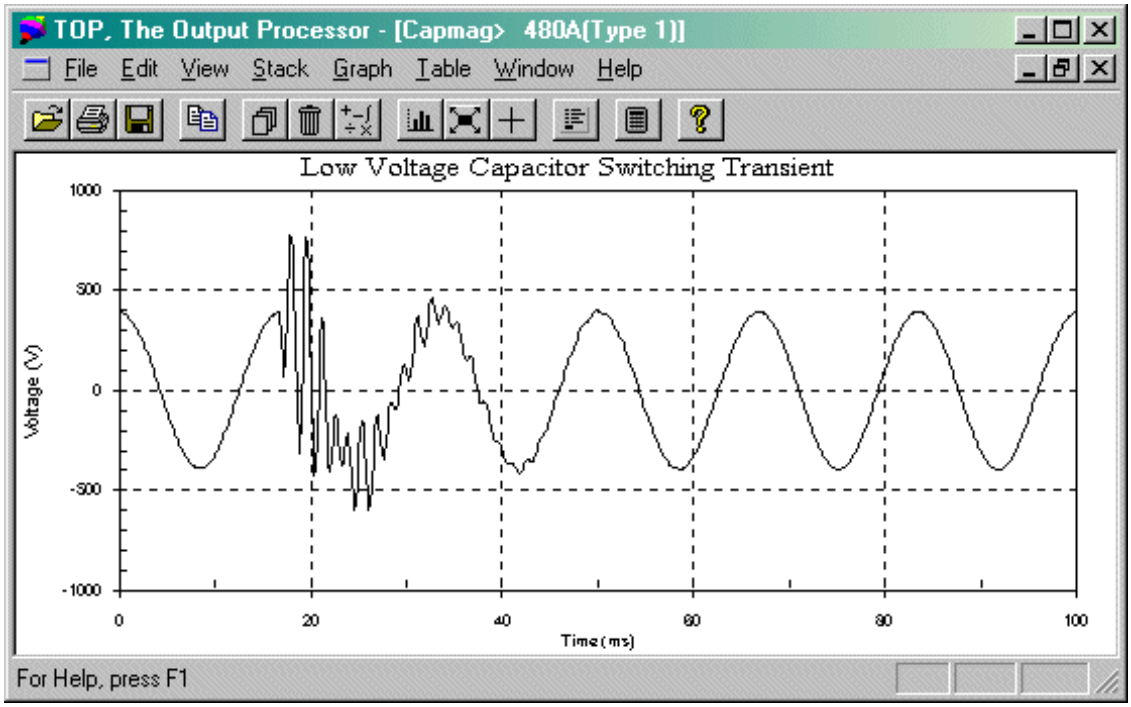

Figura 2 - TOP The Output Processor[3]

2.2 SIGRA

SIGRA - *Powerful Analysis of all Protection Fault Records*: Esse produto foi aprovado por pessoas que têm bastante experiência prática na área de avaliação de falhas. Ele exibe os valores dos sinais no tempo, diagramas fasoriais, diagrama de barras para mostrar os componentes harmônicos após o cálculo da FFT. Esse produto calcula as componentes simétricas, impedâncias, saídas, valor RMS, além de possuir dois cursores para avaliação de falta em um determinado registro. Uma característica muito interessante é que o *software* apresenta uma interface com a capacidade de trabalhar com várias janelas internas (MDI), e os cursores são sincronizados entre todas as janelas. O *software* possui uma poderosa ferramenta de zoom, uma configuração bastante amigável com mecanismos de arrastar e soltar (*drag & drop*) além de ser muito rápido. É uma ferramenta disponível apenas para Windows e sua licença não é gratuita, lê arquivos apenas no formato COMTRADE [4]. A Figura 3 mostra a interface gráfica do aplicativo citado.

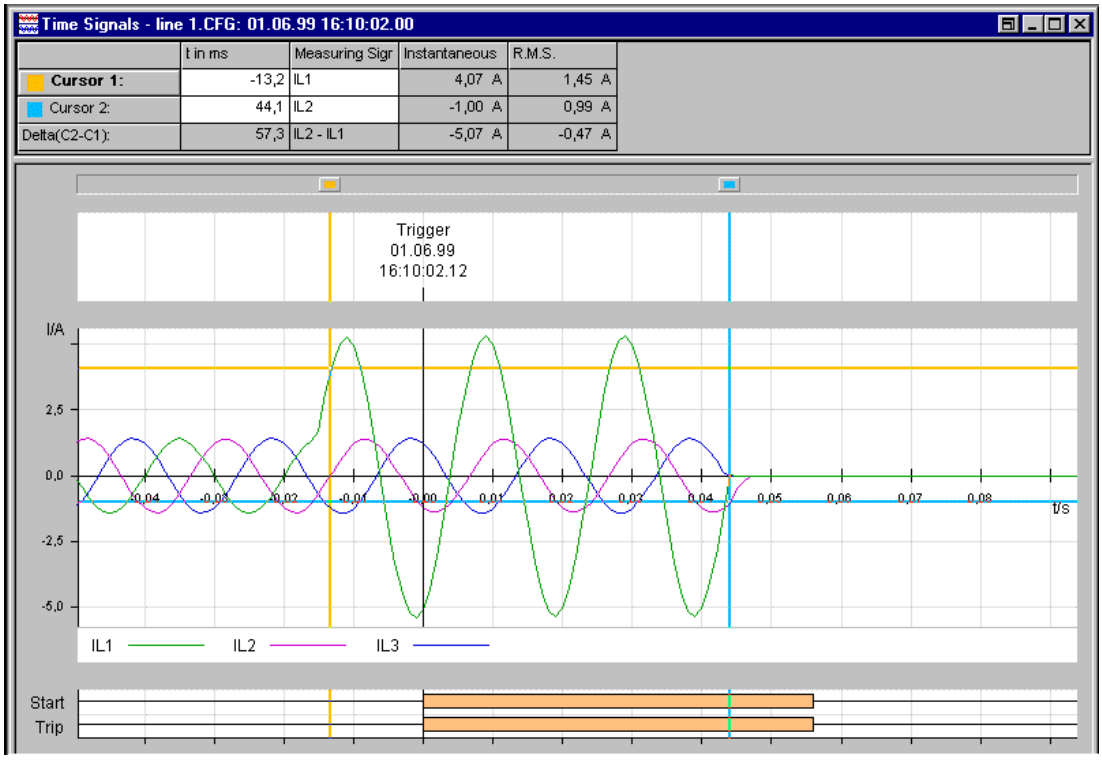

Figura 3 – SIGRA [4]

2.3 .NET Relayer

.NET Relayer é um *software* bastante simples, anteriormente denominado *Waveform Analyzers*, sua principal funcionalidade é a simulação de algoritmos de retransmissão e experimentos, mas também é um poderoso visualizador de arquivos COMTRADE. O *software* exibe os sinais das amostras em função do tempo, possui cursores e um painel bem completo e informativo para configurar as séries exibidas. Os gráficos são separados em abas diferentes, porém vários canais podem ser exibidos em um único gráfico. Esse produto tem uma funcionalidade bastante interessante de exportar para o Excel os dados da amostra. A licença desse *software* é livre, ou seja, o produto pode ser utilizado para estudos em universidades e empresas. Não possui gráficos circulares e gráficos de barra para a exibição dos valores fasoriais e dos harmônicos da FFT. Pelo *software*, os cálculos matemáticos precisam ser programados antes de serem exibidos, o que pode ser uma vantagem para cálculos muito específicos, porém um problema no caso de manipulações muito usuais como, por exemplo, o cálculo do valor eficaz e as operações matemáticas básicas [5]. A figura 4 mostra um exemplo da interface do aplicativo .NET *Relayer*

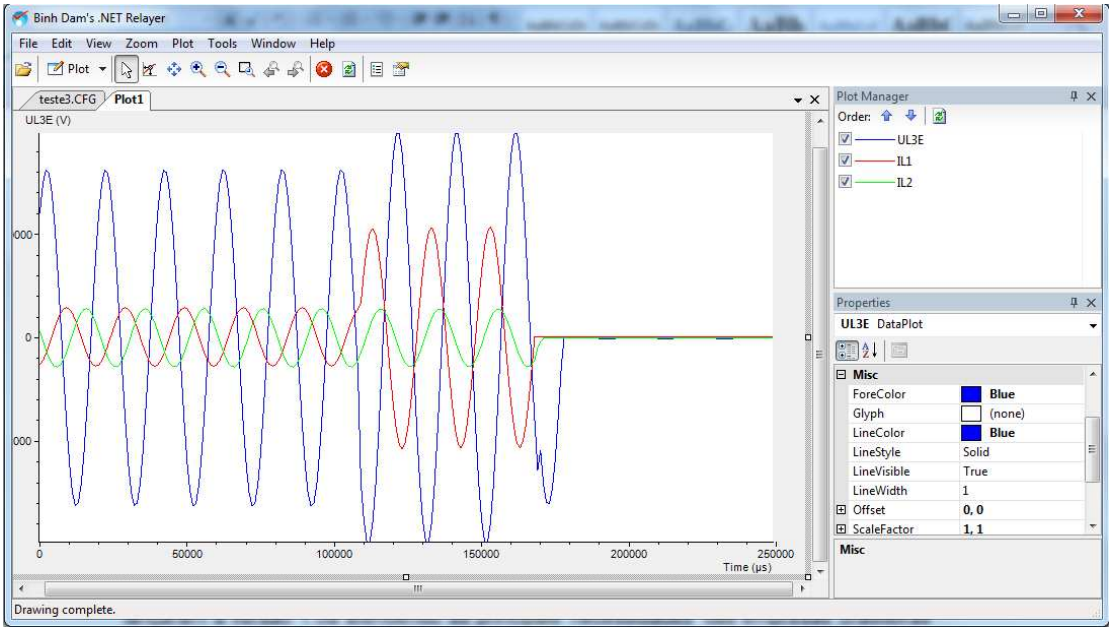

Figura 4 - .NET Relayer [5]

2.4 SINAPE

SINAPE (Sistema Integrado de Apoio à Análise de Perturbações): Este projeto nasceu buscando suprir as mesmas necessidades de mercado, visto que desde a década de 80 havia a necessidade de um *software* específico para a visualização de perturbações e oscilografia na área de proteção. Em 1995 lançaram a versão "1.0a" atendendo as principais necessidades das empresas brasileiras interessadas. O *software*, atualmente na versão 3.0, continua crescendo e se mostra bem completo. Em termos de visualização mostrou-se eficiente, pois permite a comparação de oscilografia das mais variadas formas, ele exibe os valores instantâneos, fasores e eficazes, além de possuir cursores para a comparação entre gráficos. É bastante modular, o que significa que diferentes funcionalidades podem ser incorporadas ao *software* de acordo com o interesse das empresas. Outras funcionalidades que chamam a atenção são a sincronização entre ondas, visualização de canais de diferentes estações, análise harmônica, localização de falta, gráficos de impedância e exportação em formato COMTRADE. Porém o *software* não é livre e, portanto, não pode ser usado para o estudo sem uma licença. Apenas arquivos no formato COMTRADE são importados e exportados [6]. A Figura 5 mostra a interface gráfica do SINAPE.

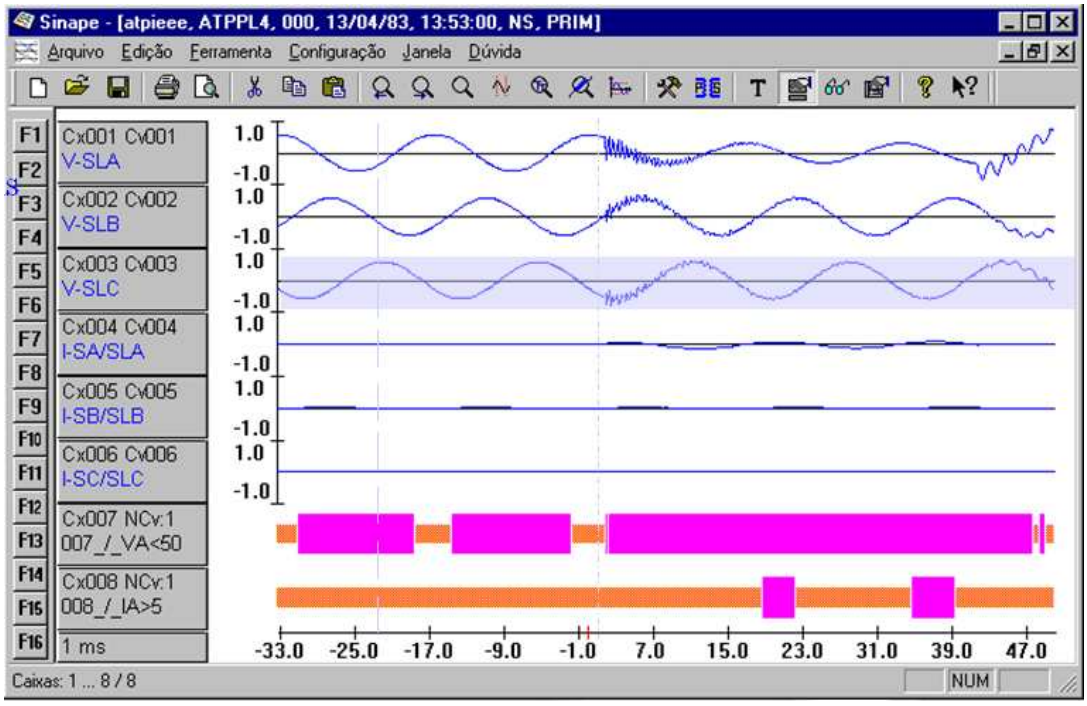

Figura 5 – SINAPE [6]

Embora a maioria das funcionalidades propostas existam nos *softwares* existentes no mercado, o conjunto proposto não é encontrado em um único aplicativo viabilizando a construção de uma ferramenta de código aberto e disponibilizado por uma instituição acadêmica importante como o Laboratório de Sistemas de Energia Elétrica, da Universidade de São Paulo.

O aplicativo deverá se adaptar aos diversos tipos de canais existentes tanto na norma COMTRADE quanto aos novos canais criados dentro de seu domínio e que possuem representações distintas. Também deverá permitir a manipulação de diversos arquivos e formatos conforme o TOP, trabalhar com cursores e gráficos simultâneos conforme o SIGRA, além de possibilitar o uso de diversas janelas em uma mesma área de trabalho (MDI).

3 Ferramentas Computacionais utilizadas no desenvolvimento do aplicativo WAPS

Este capítulo é responsável por apresentar as principais ferramentas computacionais utilizadas no desenvolvimento da ferramenta WAPS. Os principais conceitos de orientação a objetos são introduzidos e em seguida a linguagem Java é definida, já que implementa o paradigma de programação orientada a objetos com suas diversas peculiaridades. São mostrados os benefícios dessa linguagem e a razão pela qual foi adotada como a base para desenvolvimento do WAPS.

3.1 Paradigmas de Programação

Um estilo de programação é um modo de organizar programas com base em um modelo conceitual de programação e uma linguagem apropriada para escrevê-los de forma clara [7]. Existem cinco tipos principais de paradigmas conforme descrito na tabela 1.

Tabela 1 - Paradigmas de Programação

Paradigmas	Exemplificação
Orientada a Procedimentos	Estrutural
Orientada a Objetos	Classes e Objetos
Orientada a Lógica	Objetivos freqüentemente expressados em predicados
Orientada a Regras	if - then - regras
Orientada a Restrições	Relações

Não existe um estilo de programação que seja melhor que todos os outros, mas apenas situações em que determinados paradigmas são mais apropriados. Por exemplo, programação orientada a regras é mais eficiente para projetos envolvendo bases de conhecimento, enquanto programação estrutural é mais apropriada para problemas que envolvem uso intensivo das operações computacionais [7].

3.2 Orientação a Objetos

O termo orientação a objetos é uma quebra de paradigma do desenvolvimento tradicional de *software*, em que estruturas de dados e rotinas são desenvolvidas de forma apenas fracamente acopladas. É um conceito que pressupõe uma organização em termos de coleção de objetos discretos incorporando estrutura e comportamento próprios. Requer um modo de pensamento diferenciado para decomposição do problema e possui uma arquitetura bastante diferenciada dos métodos de projeto estruturados [8].

Métodos de projetos estruturados evoluíram para auxiliar desenvolvedores a construírem sistemas cada vez mais complexos utilizando algoritmos como bloco fundamental. Similarmente, projetos orientados a objetos evoluíram para auxiliar desenvolvedores a explorar o poder das linguagens orientadas a objetos utilizando classes e objetos como blocos fundamentais.

A abordagem de orientação a objetos favorece a aplicação de diversos conceitos considerados fundamentais para o desenvolvimento de bons programas. Mesmo que tais conceitos não sejam exclusivos a esse tipo de organização, o seu desenvolvimento é suportado de uma melhor forma que em outras metodologias. Existem quatro elementos essenciais nesse modelo:

- Abstração
- Encapsulamento
- Modularidade
- Hierarquia

3.2.1 Abstração e Tipo Abstrato de Dados

Pode-se considerar a definição de abstração como o “Ato de separar mentalmente um ou mais elementos de uma totalidade complexa, os quais só mentalmente podem subsistir fora dessa totalidade” [9].

Em termos de desenvolvimento de sistemas, considera-se abstração como o ato de focalizar os aspectos essenciais inerentes a uma entidade e ignorar propriedades específicas,

ou seja, concentrar-se na definição de um objeto antes de decidir como será implementado.

O uso de abstração preserva a liberdade para tomar decisões de desenvolvimento ou de implementação apenas quando há um melhor entendimento do problema a ser resolvido. Toda variável de uma linguagem de programação é uma abstração, pois representa virtualmente determinado objeto que existe no mundo real [8].

Muitas linguagens de programação modernas suportam o conceito de abstração de dados; porém, o uso de abstração juntamente com polimorfismo e herança, como suportado em orientação a objetos, é um mecanismo muito mais poderoso. O uso apropriado de abstração permite que um mesmo modelo conceitual seja utilizado para todas as fases de desenvolvimento de um sistema, desde sua análise até sua documentação.

Correlacionado a esta definição, existe o conceito de tipo abstrato de dados, implementação do conceito de abstração, que consiste em estruturas especificamente construídas para armazenarem determinados tipos de dados. Estas estruturas especificam operadores que permitem a manipulação destes dados, oferecendo recursos para que cada tipo possa ser processado junto com outros [10].

3.2.2 Encapsulamento e Polimorfismo

O conceito de encapsulamento permite que mudanças em programas sejam mais confiáveis através da restrição de informações. A estrutura de um objeto é escondida assim como a implementação de seus métodos.

Desta forma, o encapsulamento separa os aspectos externos de um objeto, os quais são acessíveis a outros objetos, dos detalhes internos de implementação. O uso de encapsulamento permite a modularização do código, evitando que pequenas mudanças tenham repercussões em toda estrutura do código, e melhor reaproveitamento em outras aplicações. Provê barreiras entre diferentes abstrações e como conseqüência separa problemas de forma clara e concisa.

Para a abstração funcionar é necessário que as implementações sejam encapsuladas, ou seja, as classes devem possuir duas partes: uma interface e uma implementação. A

interface é responsável pela comunicação de um objeto com o mundo exterior. Já a implementação representa a abstração assim como os mecanismos para se obter o comportamento esperado [8].

Polimorfismo determina que uma operação sobre um objeto pode se comportar de forma diferente em classes diferentes ou pode ser implementada por mais de um método em uma mesma classe. A linguagem de programação deve ser capaz de selecionar o método correto a partir do nome da operação, classe do objeto e argumentos. Desta forma, novas classes podem ser adicionadas sem necessidade de modificação do código já existente, pois cada classe define apenas os seus métodos e atributos [10].

3.2.3 Modularidade[8]

O ato de particionar um programa em componentes individuais reduz significantemente sua complexidade criando um número definido de interfaces. Na maioria das linguagens as classes e objetos formam a estrutura lógica de um sistema. Organizar essas estruturas em módulos produzindo uma arquitetura física ajuda o gerenciamento de complexidade principalmente em grandes aplicações em que se podem ter centenas de classes.

A modularização consiste em dividir um programa em diferentes módulos que podem ser compilados separadamente, mas que possuem conexões entre si. Servem como *containers* físicos com declarações de classes e objetos do projeto lógico. O objetivo geral da decomposição é a redução do custo do *software* permitindo que os módulos sejam construídos e revisados independentemente.

Decidir sobre o conjunto apropriado de módulos pode ser uma tarefa árdua assim como a construção de abstrações. Quando se tem um problema bem conhecido existem diretivas padronizadas a serem seguidas. Porém, para problemas desconhecidos, é necessário tempo maior de análise.

Em paradigmas tradicionais de programação, a modularização está preocupada com o agrupamento de subprogramas utilizando para isto os critérios de acoplamento e coesão. Em projetos orientados a objetos deve-se definir onde armazenar fisicamente classes e

objetos do projeto lógico.

Como geralmente os módulos são unidades elementares e indivisíveis de um *software*, estes podem ser reutilizados em outros aplicativos. Por isso é importante empacotar classes e objetos de tal forma que sejam reutilizáveis de forma conveniente.

3.2.4 Herança

Técnicas de orientação a objetos promovem compartilhamento em diversos níveis distintos. A Herança de estrutura de dados permite que estruturas comuns sejam compartilhadas entre diversas classes derivadas similares sem redundância. Ainda mais importante que a economia de código é a clareza conceitual do relacionamento e interconexão entre as estruturas de dados, o que reduz o número de casos distintos que devem ser entendidos e analisados [8].

Outra vantagem da utilização do conceito de herança é a possibilidade de reaproveitar códigos em outros projetos. Entretanto, para se alcançar reusabilidade é preciso planejamento e disciplina no uso de termos genéricos, não voltados simplesmente para a aplicação corrente.

Relacionado ao conceito de herança tem-se o conceito de interface. Uma interface modela um comportamento esperado. Ao se modelar um sistema, pode-se pensar apenas nas interfaces de seus objetos, ou seja, em suas funções e relacionamentos. Cria-se então, uma camada extra de abstração. O uso de interfaces sugere a construção de uma planta do sistema de uma forma consistente e clara facilitando o desenvolvimento posterior dos códigos que seguirão os moldes já existentes [10].

3.2.5 O Modelo Orientado a Objetos

O modelo orientado a objetos é fundamentalmente diferente dos modelos tradicionais de programação. Isso não significa que os conceitos e princípios anteriores foram abandonados, ao contrário, na verdade, houve a introdução de novos elementos baseados nas experiências anteriores.

Esse modelo oferece um número significativo de benefícios que outros modelos não

adotam. Também proporciona práticas de desenvolvimento que permitem a estruturação de problemas complexos.

Um modelo de objetos busca capturar a estrutura estática de um sistema mostrando os objetos existentes, seus relacionamentos, atributos e operações que caracterizam cada classe de objetos. É através do uso deste modelo que se enfatiza o desenvolvimento em termos de objetos ao invés de mecanismos tradicionais de desenvolvimento baseado em funcionalidades, permitindo uma representação mais próxima do mundo real [8].

Objetos são abstrações com limites e significados bem definidos para uma aplicação. Existem dois propósitos em sua utilização: promover o entendimento do mundo real e suportar uma base prática para uma implementação computacional. A decomposição de um problema em objetos não é única; depende do julgamento do projetista e da natureza do problema [8].

Para construir um objeto projeta-se uma classe que irá agrupá-los conforme suas similaridades, relacionamentos e semântica. Classes são definidas a partir de propriedades, referentes aos atributos de um objeto, e comportamentos, que estão ligados a operações e métodos suportados por um objeto.

3.2.6 Classes e Objetos

Um objeto possui estado, comportamento e identidade. A estrutura e comportamento de objetos similares são definidos em sua classe comum, ou seja, as classes são como moldes, elas definem quais atributos e funcionalidades os objetos de seu tipo terão. Os objetos são instâncias de suas respectivas classes [8].

O estado de um objeto engloba todas as suas propriedades estáticas e dinâmicas. Já o seu comportamento define como atuará e reagirá a certas ações através de seu estado e passagem de mensagens. A identidade é a propriedade que um objeto possui que o distingue dos demais objetos.

3.3 Java [10]

Java é uma linguagem de programação de alto nível que provê programação orientada o objetos relativamente simples, confiável, portável, interpretada e de alta performance com suporte a ambientes de tempo real. A tecnologia Java também provê uma plataforma que consiste na integração de *hardware* e *software* necessários para rodar programas. A plataforma Java é constituída de dois componentes fundamentais:

a) A Máquina Virtual Java (VJM - *Virtual Java Machine*): é a base da plataforma Java. Foi portada para várias plataformas de *hardware* a fim de se conseguir compatibilidade.
b) A Interface de Programação de Aplicações (API - *Java Application Programming Interface*): é uma grande coleção de componentes de *software* já prontos que provê capacidades extremamente úteis. São agrupadas em bibliotecas, conhecidas como *packages*, ou classes e interfaces.

Como plataforma independente do meio, Java pode ser um pouco mais lenta que códigos nativos. Contudo, vantagens adquiridas a partir da compilação e da tecnologia de máquina virtual traz performance muito perto do esperado em arquiteturas nativas.

Java foi projetada para resolver alguns desafios de desenvolvimento de aplicações em ambientes heterogêneos, principalmente sistemas distribuídos como a Internet. Paralelamente, provê um ambiente seguro com consumo mínimo dos recursos do sistema e implementa alta escalabilidade com a vantagem de portabilidade em qualquer plataforma de *hardware* ou *software* inclusive em dispositivos embarcados.

Os requisitos de concepção são detalhados a seguir.

3.3.1 Simplicidade e Orientação a Objetos

Uma das principais características da linguagem de programação Java é captação rápida por meio dos programadores desde o início do aprendizado sem que haja treinamento intensivo para adquirir habilidades e conhecimentos básicos. O conceito de orientação a objetos foi projetado a partir do zero provendo uma plataforma limpa e

eficiente.

Programadores podem acessar bibliotecas já existentes e exaustivamente testadas que provêem funcionalidades básicas como tipos de dados ou recursos mais avançados como interface de dispositivos de entrada e saída ou sistemas de redes. Essas bibliotecas também podem ser entendidas a fim de se adaptar as requisições dos usuários.

A linguagem é muito parecida com C^{++} tornando-se familiar aos usuários, porém sem as dificuldades e complexidades dessa linguagem.

3.3.2 Robustez e Segurança

Java foi projetada para que as criações de *softwares* sejam altamente confiáveis. Provê extensa verificação em tempo de compilação, seguido por um segundo nível de verificação em tempo de execução. Recursos da linguagem guiam os programadores a hábitos confiáveis de programação

O modelo de gerenciamento de memória é extremamente simples. Não existe definição, por meio do programador, de ponteiros de dados ou aritméticos. Um mecanismo de limpeza automática de memória (*garbage collection*) é integrado a própria linguagem.

A tecnologia Java foi projetada para operar em ambientes distribuídos. Desta forma, a segurança é um parâmetro de extrema importância. A linguagem não permite que haja nenhum tipo de invasão externa em aplicativos que estão em tempo de execução. Aplicações escritas em Java estão securas por tentativa de acesso por meio de códigos não autorizados que possam criar vírus em sistemas de arquivos.

3.3.3 Neutralidade de Arquitetura e Portabilidade

A tecnologia Java foi projetada para suportar aplicações desenvolvidas em ambientes heterogêneos. Em tais ambientes, aplicações devem ser capazes de executar em uma variedade de arquiteturas de *hardware* em cima de sistemas operacionais diversos e interoperar com múltiplas interfaces de linguagens de programação. Para se acomodar à grande diversidade dos ambientes, o compilador Java gera arquivos *bytecodes* que é um formato neutro intermediário projetado para suportar código eficientemente em múltiplas

plataformas de *hardware* e *software*. A natureza interpretativa de Java soluciona o problema de versão e distribuição binária, pois o programa em *bytecodes* irá rodar em qualquer plataforma.

A neutralidade de arquitetura é apenas uma parte do verdadeiro sistema de portabilidade. Um estágio mais profundo é tomado definindo-se o tamanho de seus tipos básicos e o comportamento de seus operadores aritméticos. Programas não são diferenciados conforme a máquina que operam e por isto não existe nenhum tipo de incompatibilidade com arquiteturas de *hardwares* e *softwares* existentes.

A neutralidade de arquitetura e portabilidade se traduzem na especificação da Java *Virtual Machine* que é a especificação de uma máquina abstrata que possibilita compiladores Java gerarem os códigos binários necessários para execução de programas. Implementações específicas da Java *Virtual Machine* para *hardwares* e *softwares* específicos provêm a realização concreta da virtualização. A interface portável utilizada é baseada na especificação do padrão comercial POSIX, *Portable Operating System Interface for Unix*.

3.3.4 Alto Desempenho

O desempenho sempre deve ser levado em consideração. A plataforma Java alcança desempenho adotando um esquema em que o interpretador pode rodar em velocidade máxima sem precisar monitorar o meio de execução. O mecanismo de limpeza automática de memória (*garbage collection*) executa em prioridade baixa nos "bastidores" como uma *thread*, ou seja, um processo paralelo, assegurando alta probabilidade de acesso à memória quando esta é requerida. Aplicações que requerem grande parte dos recursos do computador podem ser desenvolvidas de tal forma que seções de consumo intensivo sejam reescritas como código na linguagem nativa da máquina e depois sejam interfaceadas com a plataforma Java.

3.3.5 Interpretação e *Multithreading*

O interpretador Java executa *bytecodes* diretamente em qualquer máquina que o interpretador e o sistema de execução sejam portados. Em uma plataforma interpretada, a fase de link de um programa é simples, incremental e leve.

A tecnologia *multithreading* (suporte a execução de vários threads concorrentes) de Java provê no nível da linguagem os meios para se construir aplicações com mais de um thread de forma simultânea com a adição de primitivas de sincronização sofisticadas. Desta forma, é disponibilizado um alto grau de interatividade para o usuário final.

4 O Padrão COMTRADE

Este capítulo é responsável por explanar a estrutura geral do padrão IEEE COMTRADE e é baseado na norma IEEE incluída nas referências desta monografia [2]. Definem-se os arquivos existentes e suas estruturas básicas bem como os tipos de arquivos de dados suportados (Binário ou ASCII). Por fim, é feita uma rápida revisão sobre como são armazenadas informações em ponto flutuante e as unidades de medida suportadas.

O padrão IEEE COMTRADE (*Common Format for Transient Data Exchange*) define um formato comum para arquivo de dados digitais e meio de troca para intercâmbio de vários tipos de faltas, testes e simulações.

Cada registro COMTRADE possui um conjunto de até quatro arquivos associados. Cada um dos quatro arquivos carrega diferentes classes de informações e são apresentados a seguir:

- *Header;*
- *Configuration;*
- *Data;*
- *Information.*

Todos os arquivos do conjunto devem possuir o mesmo nome, diferindo apenas pela extensão que, neste caso, indica um dos tipos de arquivo listados acima.

Os nomes dos arquivos devem estar no formato xxxxxxxx.yyy. A porção xxxxxxxx (no máximo oito caracteres) é o nome usado para identificar o registro. Já a porção .yyy (exatamente três caracteres), a extensão, é utilizada para identificar o tipo de arquivo representado; é uma sigla associada a cada um dos tipos de arquivo conforme mostrado na tabela 2.

Tabela 2 - Tipos de Arquivos COMTRADE e suas respectivas extensões

Tipo Arquivo	Extensão
Header	.HDR
Configuration	.CFG
Data	.DAT
Information	.INF

Os nomes dos arquivos devem ser compatíveis com a convenção IBM-DOS (MS-DOS Versão 6) de caracteres válidos.

O Padrão COMTRADE também especifica um meio para troca dos arquivos entre sistemas. Na época em que foi proposto, os disquetes de 3’’½, que possuem capacidade de 1,44MB, eram uma opção atrativa para intercambiar as informações. E por isso foi permitida a segmentação do arquivo de dados que pode crescer consideradamente conforme se aumenta o volume de dados de uma amostra.

O arquivo de dados pode ser segmentado em várias sub-partições. Neste caso, os dois últimos caracteres “AT” da extensão “.DAT” são removidos e em seus lugares são inseridos as seqüências dos arquivos gerados. Desta forma, pode-se obter até 100 arquivos referentes a uma mesma amostra com a extensão começando .D00 e indo até o valor D99.

Vale ressaltar que com o avanço tecnológico o uso de disquetes tornou-se obsoleto frente à variedade de alternativas mais atraentes principalmente pela grande capacidade e baixo custo relativo de dispositivos de armazenamento bem como a disseminação de equipamentos que utilizam o tráfego de rede para transmissão e recepção de dados. Desta forma, em muitas aplicações a segmentação do arquivo de dados torna-se desnecessária.

4.1 Header File

O arquivo de cabeçalho é um arquivo de texto no formato ASCII de qualquer tamanho criado pelo gerador dos dados COMTRADE. As informações contidas neste tipo de arquivo podem ser impressas e manipuladas diretamente pelo usuário que pode entender melhor as condições apresentadas.

Não há nenhum padrão a ser seguido; assim, o gerador pode incluir qualquer informação pertinente desejada geralmente em forma narrativa. Considera-se que aplicações computacionais não manipularão as informações contidas neste arquivo.

O IEEE sugere alguns tipos de informações possíveis a serem descritas em um arquivo de cabeçalho:

- Descrição do sistema de energia antes da perturbação;
- Nome da estação;
- Identificação da linha, transformador, reator, capacitor, ou disjuntor que experimenta o regime transiente;
- Comprimento da linha que sofreu a falta;
- Seqüência-positiva, Seqüência-zero de resistência e reatância;
- Acoplamento mútuo entre linhas paralelas;
- Tensão nominal de enrolamento de transformadores, especialmente a potência e a corrente;
- Parâmetros do sistema através dos nós onde os dados são registrados;
- Descrição de como os dados foram obtidos, se foi obtido a partir de uma subestação ou por simulação;
- Descrição do filtro *anti-aliasing* utilizado;
- A seqüência de fase de entrada; e
- O número de discos onde o registro foi gravado.

4.2 Configuration File

O arquivo de configuração é um arquivo de texto no formato ASCII que será lido por *softwares* especializados, e por isso, deve ser escrito em formato específico. Contém as informações necessárias para interpretação do arquivo de dados (.DAT) e por isso não é opcional. Pode ser criado por programas de processamento a partir dos dados de origem do registro de transiente.

As seguintes informações devem estar contidas no arquivo de configuração:

- Nome da estação, identificação do dispositivo de monitoração, ano de revisão do padrão COMTRADE;
- Número e tipos dos canais;
- Nome dos canais, unidades, a fatores de conversão;
- Freqüência da linha;
- Taxa de amostragem;
- Data e tempo do primeiro dado;
- Data e tempo do ponto de acionamento da falta;
- Tipo do arquivo de dados; e
- Fator de multiplicação da estampa de tempo.

O arquivo é dividido em linhas e as linhas em campos. Vírgulas são utilizadas como delimitador de campo, mesmo que o respectivo dado não seja especificado permitindo que o comprimento de cada campo seja variável. A informação em cada linha do arquivo deve ser listada na ordem exata requerida. Qualquer desvio do padrão estipulado invalidará todo o conjunto de arquivos.

Alguns dados são definidos como não críticos e por isso permitem a omissão sem que haja perda na interpretação. Por outro lado, existem informações cruciais definidas como críticas que, caso invalidadas, alteram a compreensão, principalmente do arquivo de dados.

4.3 Data File

O arquivo de dados contém o número da amostra, a estampa de tempo e o valor de cada amostra dos canais de entrada do registro analisado. Os dados devem estar dispostos conforme descrito no arquivo de configuração que também especifica fatores de conversão para converter os valores apresentados em dados concretos, pois os números armazenados representam apenas uma escala dos valores reais.

Além dos dados representativos de entradas analógicas, podem-se representar entradas digitais. Neste padrão, o tipo de entrada é referenciado como entrada de estado associado aos níveis lógicos “1” ou “0”.

O arquivo de dados pode ser escrito tanto em formato ASCII quanto em formato binário e a diferenciação é realizada por um dos campos no arquivo de configuração. No formato ASCII o delimitador de campos é a vírgula enquanto em arquivos binários usa-se o tamanho de cada informação como delimitador.

4.3.1 Arquivo ASCII

O arquivo de dados no formato ASCII é dividido em linhas e colunas. O número de linhas varia de acordo com o tamanho da amostra e afeta diretamente o tamanho total do arquivo. O número de colunas depende da quantidade de canais analisados e também afeta o tamanho total do arquivo. A seguir é mostrada a seqüência de informações necessárias:

- A primeira coluna contém o número da amostra;
- A segunda coluna é a estampa de tempo;
- A terceira coluna representa o conjunto de dados dos canais analógicos; e
- A quarta coluna representa o conjunto de dados dos canais digitais.

Não pode haver quebra de linha entre os dados referentes ao mesmo tempo de amostragem. O arquivo deve terminar com um caractere ASCII de EOF, *End Of File*, referente ao término de arquivo.

4.3.2 Arquivo Binário

O arquivo de dados no formato binário é um sequência contínua de dados binários que possui a mesma estrutura básica utilizada nos arquivos do tipo ASCII.

Não existe delimitador definido com base em caracteres, e não há nenhum tipo de quebra de linha. Os campos são separados conforme tamanho do tipo de dados e pela posição seqüencial no arquivo. Caso algum elemento esteja corrompido ou perdido a seqüência será pedida e o arquivo se tornará inutilizável. O padrão COMTRADE não provê recuperação de dados.

Tem-se a seguinte estrutura:

- Número da amostra e estampa de tempo armazenados na forma binária sem sinal de quatro bytes cada;
- Dados referentes aos canais analógicos armazenados na forma binária em complemento de dois de dois bytes cada. O valor zero é armazenado como 0000h, o valor -1 é armazenado como FFFFh. Desta forma, o maior número valor positivo será 7FFFh e o menor valor negativo será 8001h. O valor 8000 é reservado para marcar dados corrompidos.
- Dados referentes aos canais digitais armazenados em grupos de dois bytes com dezesseis canais digitais cada um. O bit menos significativo da palavra representa o menor canal de entrada digital. Se o número de canais digitais não é divisível por dezesseis, os bits restantes devem ser completados com bits "0".

Os dados são armazenados em formato binário em *big-endian*, ou seja, os bytes mais significativos de uma palavra são armazenados antes dos bytes menos significativos.

4.4 Information File

O arquivo de informação é um arquivo opcional em formato ASCII específico que contém informações extras que a fonte de dados deseja tornar disponível para os usuários. O formato provê informações públicas disponíveis a qualquer usuário bem como informações privadas acessíveis apenas a determinado grupo e não interpretáveis pra propósito geral. Os dois tipos de informações, público e privado, residem em seções separadas do arquivo.

Qualquer *software* que acesse esse tipo de arquivo deverá ser capaz de reconhecer seções públicas de cabeçalho, entradas, ou outros dados definidos no padrão, e tomar as devidas ações necessárias. Existe a imposição que programas que não reconheçam determinados dados não os alterem de forma alguma.

O formato do arquivo de informação é similar ao formato de arquivo Windows™ ".INI". Muitas linguagens de programação incluem funções de leitura e escrita desses tipos de arquivos sendo essa estrutura bastante difundida.

O arquivo é dividido em seções. Cada seção consiste em um cabeçalho seguido por um determinado número de linhas de entrada. Não há limite para o número de seções, porém deve existir pelo menos uma seção por arquivo. Nenhum dado pode residir fora de uma seção. Cada seção é identificada por um uma única linha de cabeçalho.

A estrutura genérica é apresentada a seguir:

Público:

- Cabeçalho de Seção da Informação do Registro (Informações referentes a todo o registro)

  Linhas de Entrada

- Cabeçalho de Seção da Informação do Evento (Informação relacionada a um canal particular ou a uma amostra em um registro)

Linhas de Entrada

- Cabeçalho de Seção da Descrição do Arquivo (Informação equivalente ao arquivo .CFG relacionado a todo o registro)

Linhas de Entrada

- Cabeçalho de Seção do Canal Analógico #1 (Informação equivalente ao arquivo .CFG relacionado ao primeiro canal analógico do registro)

Linhas de Entrada

.

.

.

- Cabeçalho de Seção do Canal Analógico #n (Informação relacionada ao próximo canal analógico do registro, com uma nova seção para cada canal até o número total de canais analógicos)

Linhas de Entrada

- Cabeçalho de Seção do Canal Digital #1 (Informação relacionada ao primeiro canal digital do registro)

Linhas de Entrada

.

.

.

- Cabeçalho de Seção do Canal Digital #n (Informação relacionada ao próximo canal digital do registro, com uma nova seção para cada canal até o número total de canais digitais)

    Linhas de Entrada

Privado:

- Cabeçalho de Informação

    Linhas de Entrada

- Cabeçalho de Informação

    Linhas de Entrada

4.5 Notação de Ponto Flutuante

O padrão COMTRADE define alguns campos como reais e para isto utiliza a representação computacional de ponto flutuante, também conhecido como notação exponencial ou científica, que especifica a mantissa (dígitos significativos) e o expoente (fator multiplicador) de um número real.

Um valor em ponto flutuante com sinal consiste em uma série de dígitos decimais contendo um ponto decimal, seguido por um campo de expoente que contém o caractere "e" ou "E" seguido por um inteiro. Existe a opção de sinal (+ ou -) antes dos dígitos decimais e do expoente. É opcional o uso do ponto decimal e também do expoente.

O expoente é um fator da base 10 e a interpretação correta de números e expoentes negativos requer a inclusão do sinal negativo. Para números e expoentes negativos o sinal é opcional e é assumido o valor positivo caso esteja ausente.

O formato geral é apresentado a seguir:

[+-]dd [.]dddd[E[+-]ddd]

onde:

- Colchetes representam valores opcionais;
- d representa um dígito decimal entre 0 e 9;
- Pelo menos um número deve aparecer no campo
- Se o ponto decimal aparecer, pelo menos um número deve aparecer na esquerda e na direita
- O caractere "e" ou "E" representa exponencial na base 10;
- Se o sinal exponencial aparecer, deve ser seguido por pelo menos um número;
- O valor numérico seguido de "E" deve ser um inteiro.

4.6 Unidades de Medida

As informações contidas nos canais analógicos possuem unidades físicas de medida como corrente, tensão, potência, entre outras. O padrão COMTRADE especifica o uso de unidades de quantidades físicas conforme abreviações ou nomenclatura padronizada de acordo com os padrões IEEE Std 260,1-1993 e IEEE Std 280-1985, caso a nomenclatura exista. Fatores de multiplicação numéricos não devem ser inclusos. Já multiplicadores padrões como k(mil), m(centésimo), M(milhão), etc. são permitidos [11].

5 WAPS: Um *software* aplicado a análise de sinais provenientes do sistema elétrico de potência

Este capítulo é responsável por descrever as diversas funcionalidades projetadas e desenvolvidas para a ferramenta WAPS. É dada uma visão geral sobre sua capacidade e suas funcionalidades assim como a descrição detalhada de cada um de seus diversos módulos.

5.1 Visão Geral

WAPS é um *software* de análise de formas de ondas geradas a partir de sinais processados digitalmente. É um aplicativo voltado aos estudos e análises de perturbações e transitórios de redes de energia elétrica.

O WAPS permite a visualização e manipulação de diversos tipos de sinais bem como a criação de novos canais a partir dos dados importados.

A ferramenta foi desenvolvida na linguagem Java, pois é uma linguagem de alto nível, bastante flexível, amplamente utilizada e que segue o paradigma de programação orientado a objetos. Além disso, fornece a portabilidade necessária para execução do aplicativo em vários sistemas operacionais.

Algumas das características do sistema são descritas a seguir.

- Manipulação de vários arquivos simultaneamente;
- Compatibilidade de arquivos COMTRADE binário e ASCII;
- Suporte a vários tipos de canais (Analógicos, Digitais, etc.);
- Gráficos diferenciados para cada tipo de canal;
- Visualização de sinais pelo valor médio RMS;
- Visualização espectral dos sinais;
- Visualização de formas de onda através de fasores;
- Várias áreas para visualização de gráficos (MDI);

- Vários sinais em um mesmo gráfico;
- Operações algébricas sobre sinais.

5.1.1 Multi documentos

O WAPS trabalha com o conceito de "multi arquivos" isso significa que se pode trabalhar com vários canais de arquivos diferentes ao mesmo tempo bastando-se efetuar diversas operações ABRIR (como será descrito nesta monografia) entre diversos arquivos. A abertura de um arquivo não inviabiliza a visualização de canais previamente selecionados. Caso seja necessário fechar definitivamente um arquivo basta abri-lo e retirar a seleção de seus canais.

5.1.2 Formatos suportados

O padrão IEEE COMTRADE, organizado pela IEEE foi escolhido como padrão para descrição dos dados de entrada devido ao seu grande uso tanto no meio acadêmico quanto na indústria. A ferramenta desenvolvida ainda dá suporte ao incremento de módulos de reconhecimento de outros padrões. Para isso é necessário desenvolver, em separado, código de conversão do padrão requerido para o formato base utilizado pelo WAPS

5.1.3 Tipos de Canais

Existem três tipos de canais no sistema conforme pode ser visualizado na figura 6.

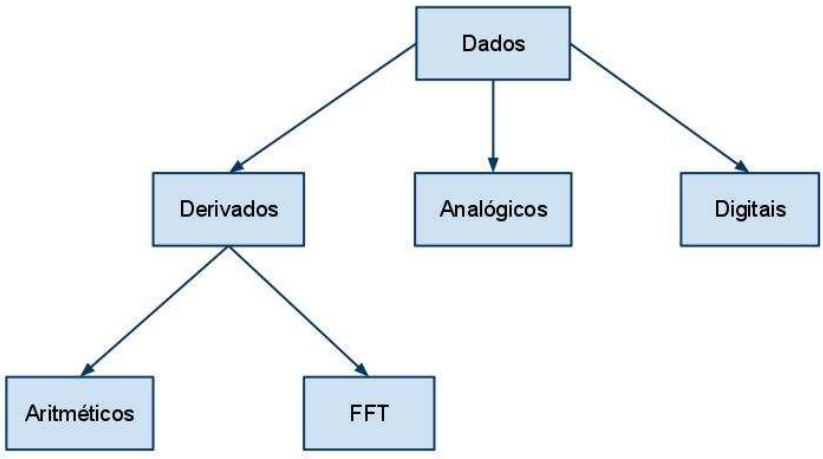

Figura 6 - Tipos de Canais

Canais Analógicos

Os dados de uma amostra representam um sinal contínuo no tempo que foi discretizado para representação computacional.

Canais Digitais

Os dados de uma amostra representam um sinal discreto no tempo e binário contendo apenas os bits "1" e "0".

Canais Derivados

São canais gerados pelo próprio sistema através de operações do usuário. Os canais derivados ainda são divididos em duas subcategorias:

- ARITMETICA: Canais gerados a partir de operações algébricas sobre canais analógicos,
- FFT: Canais gerados a partir de uma transformada discreta de Fourier utilizando o algoritmo FFT sobre canais analógicos

5.2 Princípios Básicos

5.2.1 Instalação

Como o aplicativo WAPS foi desenvolvido na linguagem Java, que foi projetada para suportar aplicações desenvolvidas em ambientes heterogêneos, é possível instalá-lo em diversos sistemas operacionais, como as várias versões de Windows e distribuições Linux que existem no mercado e que dão suporte à máquina virtual Java.

Para os sistemas Windows também foi projetado um instalador nos moldes da extensão .exe com a ajuda da ferramenta Inno Setup, distribuída gratuitamente por Jordan Russell[12] e que proporciona configurações de pastas, verificação da instalação do Java, configuração de menu da barra Iniciar, criação de atalhos e até a desinstalação do *software*.

5.2.2 Splash

Como a inicialização da máquina virtual e as configurações dos parâmetros iniciais, como as bibliotecas de usuários e a construção das interfaces para execução do aplicativo podem levar um tempo significativo aproveita-se, de forma paralela, para apresentar uma tela de apresentação, muitas vezes denominada *splash*, para apresentar o aplicativo

especificando sua versão e dando destaque ao laboratório LSEE onde a ferramenta foi desenvolvida. Uma representação da tela *splash* pode ser vista na figura 7.

Figura 7 - Splash

5.2.3 Tela Inicial

Existe um menu de opções que agrupam determinadas tarefas similares além de existir uma barra de ferramentas com atalhos das funcionalidades mais utilizadas. Algumas dessas funcionalidades são habilitadas apenas em determinados contextos onde podem ser utilizadas.

A área de trabalho é reservada para armazenar as diversas telas de configuração e visualização dos dados. Como o WAPS implementa o conceito de MDI é possível trabalhar com várias janelas ao mesmo tempo dentro da aplicação.

Também existe na parte inferior da janela a barra de status que auxilia o usuário nas diversas tarefas dos sistema dando informações úteis no manuseio da ferramenta.

5.2.4 Ajuda

O WAPS oferece um sistema de ajuda que provê informações relevantes para cada funcionalidade disponível, conforme mostra a figura 8. Para acessar o menu de ajuda basta proceder com uma das seguintes formas:

1. Pressionando a tecla F1.
2. Clicando no botão de ajuda no menu Ferramentas.
3. Clicando no botão de ajuda referente em uma tela que dispõe de orientação individual de ajuda.

Figura 8 - Ajuda

5.2.5 Fluxo Geral

O Fluxo WAPS principal é definido por três módulos interdependentes: Abertura de arquivos, *Math Builder* e Exibição de gráficos. A interação entre os módulos pode ser vista na figura 9.

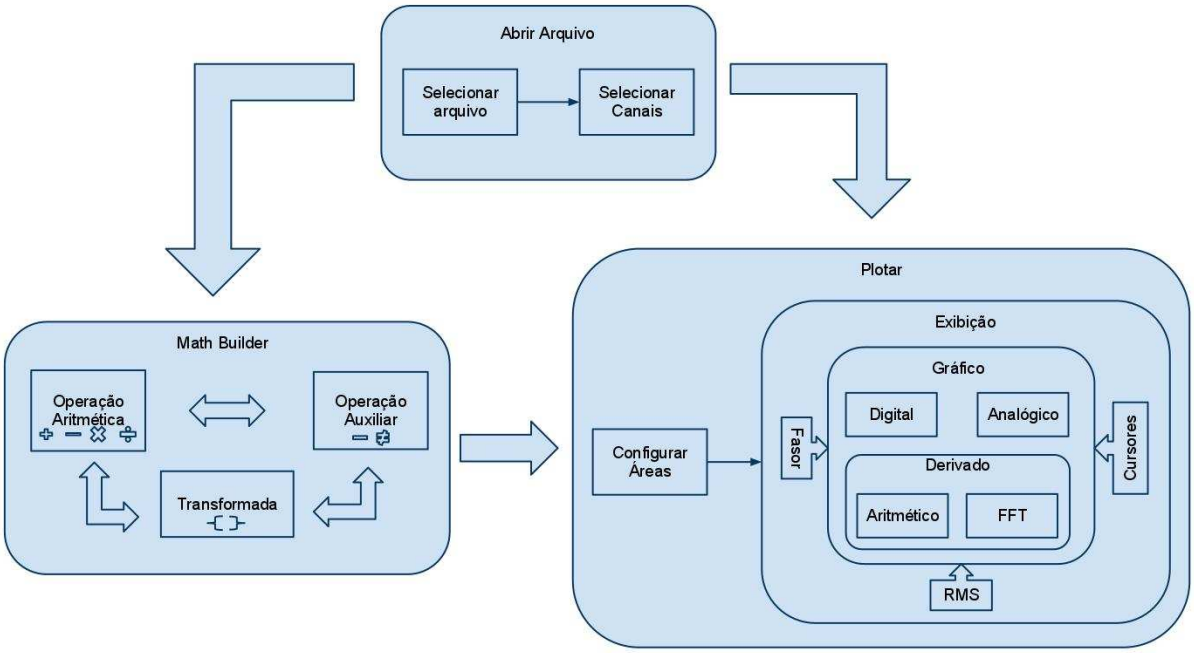

Figura 9 - Fluxo Geral WAPS

5.3 Módulo 1: Abertura de Arquivos

5.3.1 Abrindo documentos

Para abrir determinado arquivo basta acionar o atalho na barra de ferramentas ou escolher o menu “Arquivo” → “Abrir” e uma janela de seleção de arquivos será aberta. É possível percorrer diretórios nas unidades de armazenamento para selecionar um arquivo desejado. Também é possível escrever o caminho completo do arquivo no campo "Nome do Arquivo". Para confirmar a seleção basta clicar no botão "Abrir".

O sistema está configurado para reconhecer arquivos do formato IEEE COMTRADE e por isso apenas os arquivos com extensão .cfg serão visualizados. Pode-se escolher a opção “Arquivos de tipo” e selecionar a opção “Todos os arquivos” para visualizar qualquer arquivo armazenado.

O programa armazena o último diretório de seleção para que em aberturas futuras não seja necessário digitar o caminho completo ou selecionar a seqüência de diretórios necessárias para alcançar arquivos já utilizados ou que estejam na mesma pasta ou em pastas próximas.

A figura 10 mostra um exemplo de execução da tela de seleção de arquivos.

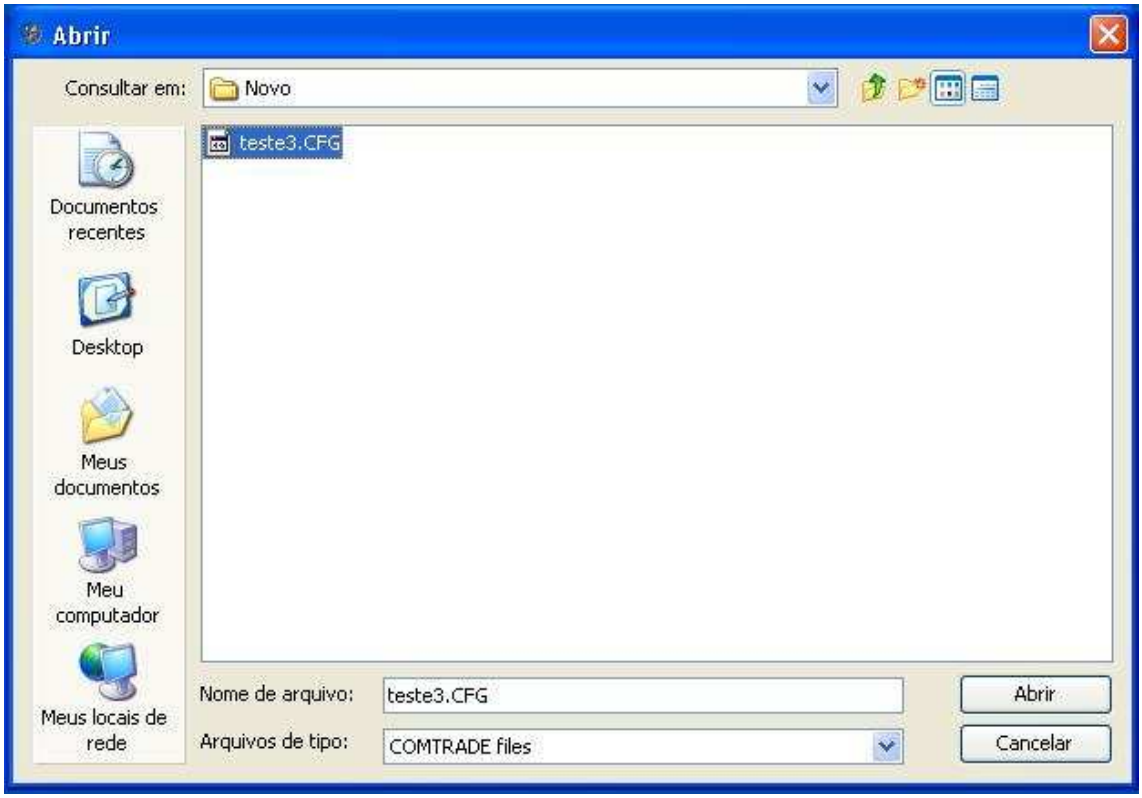

Figura 10 - Seleção de arquivos

5.3.2 Selecionando canais

Após selecionado o arquivo, uma nova tela é exibida para a configuração dos canais. O WAPS permite que apenas alguns canais fiquem visíveis para manipulações. Desta forma, para que os canais se tornem visíveis basta selecioná-los e clicar no botão "OK". A figura 11 mostra um exemplo da tela de seleção de canais.

Os canais são divididos em duas categorias:

- ANALOGICOS;
- DIGITAIS.

Para a categoria de canais analógicos existe um filtro referente à unidade de cada canal. Desta forma, é possível visualizar apenas os canais de determinada unidade:

- Tensão: em Volts (V);
- Corrente: em Ampères (A);
- Potência: em Watts (W);
- Outros: mostra qualquer canal com unidade diferente das acima mencionadas.

O botão "Selecionar Todos" permite a seleção de todos os canais de suas respectivas categorias.

O botão "Cancelar" permite o cancelamento da operação ABRIR. O programa retorna para a janela inicial.

O botão "Nenhum" retira a seleção de todos os canais que estiverem selecionados em suas respectivas categorias.

As seleções de canais efetuadas serão mantidas em memória mesmo após o fechamento da tela ABRIR. Caso algum canal precise ser adicionado ou retirado no escopo de visibilidade basta selecionar novamente a opção ABRIR e manipular os canais apresentados.

Algumas opções como "Exibir Gráficos", "Descartar", "*Math Builder*" só são habilitadas se existe pelo menos um canal visível na aplicação.

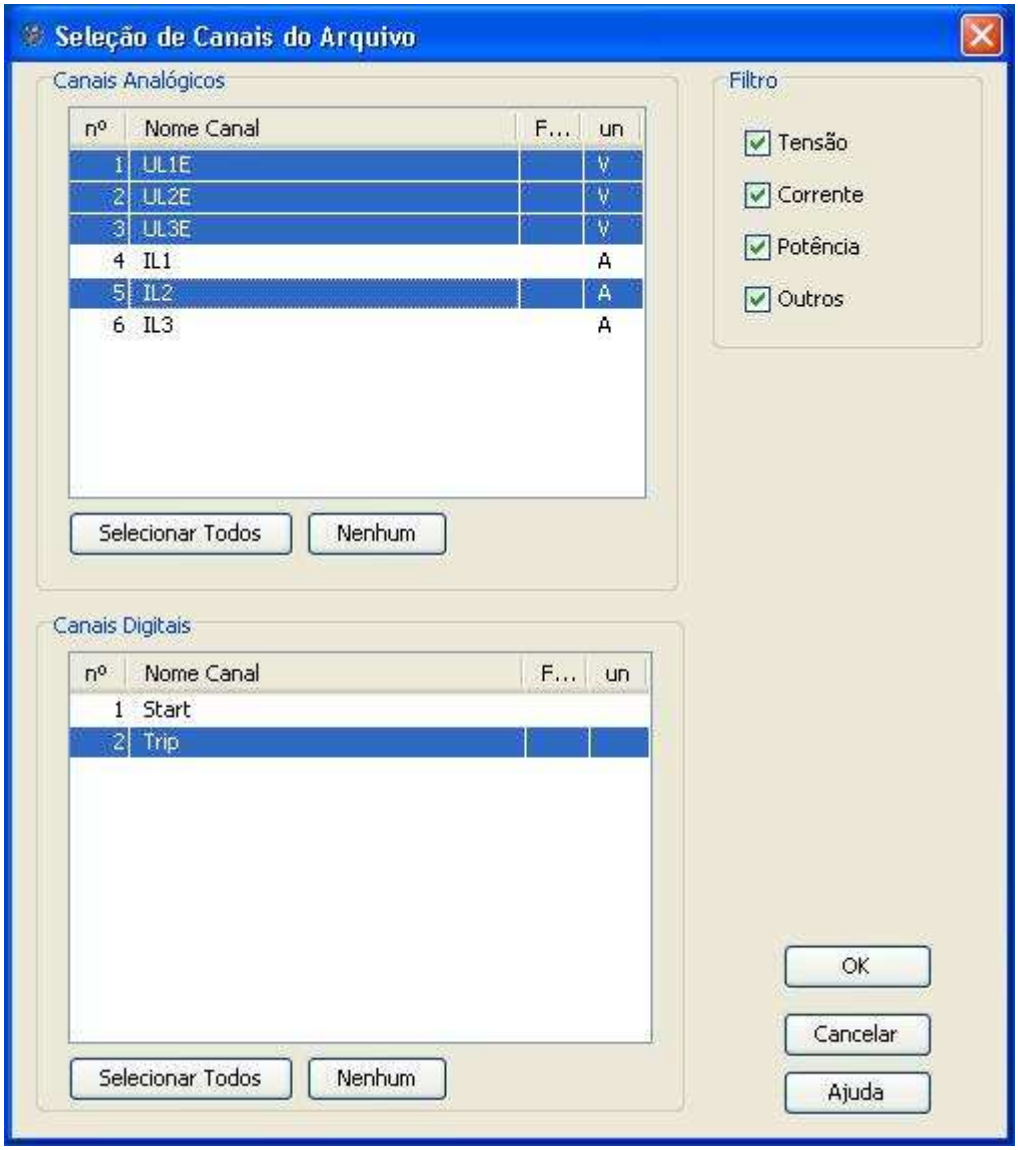

Figura 11 - Seleção de canais

5.3.3 Escopo de visibilidade

O WAPS trabalha com o conceito de visibilidade de canais. Desta forma, canais de diversos arquivos podem ser habilitados ou desabilitados de duas formas:

Seleção de canais do menu ABRIR

O menu ABRIR é a forma mais geral de se configurar a visibilidade de um canal. É possível ver qualquer canal de um arquivo de entrada mesmo que este não esteja visível para o sistema.

O menu ABRIR é a única forma de se habilitar canais não visíveis para o sistema.

Menu Descarte

O menu descarte provê apenas a desabilitação de canais. Para habilitar um canal não visível é necessário reabrir o arquivo de origem e selecioná-lo. É importante ressaltar que o descarte não elimina em definitivo os canais do arquivo de entrada. Eles são apenas retirados do escopo de visibilidade do sistema podendo ser habilitados a qualquer momento.

A tela de descarte, conforme mostrada na figura 12, provê a desabilitação de canais contidos no escopo de visibilidade do sistema. Os canais são divididos conforme as categorias de canais existentes no sistema. Desta forma, para descartar um canal basta selecionar a categoria a qual ele pertence, clicar sobre o canal (ou selecionar um conjunto de canais) e escolher a opção "Descartar".

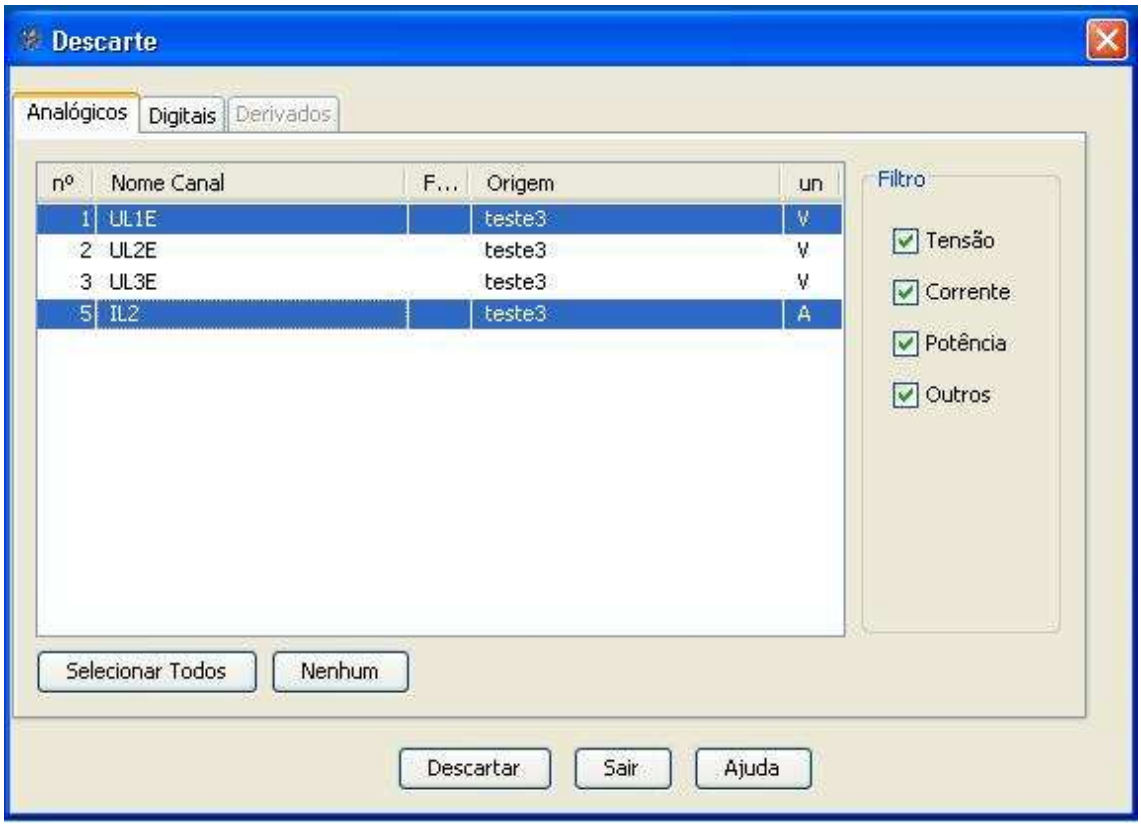

Figura 12 - Descarte de canais

5.4 Módulo 2: *Math Builder*

A ferramenta *Math Builder* provê manipulações de canais através da criação de novos objetos gerados a partir das operações matemáticas disponibilizadas. Tais objetos herdam as características do canal ao qual foi derivado podendo ser utilizado em novas operações ou na visualização de novos gráficos.

O *Math Builder* trabalha com operações apenas sobre canais analógicos e derivados. Não há operações sobre canais digitais. Na criação de um novo canal o nome e a fase definidos devem ser únicos. Além disso, determinadas operações agem apenas sobre determinados tipos de canais. Caso o segundo argumento também seja um canal deve-se levar em consideração a compatibilidade com o primeiro argumento.

5.4.1 Argumentos

Operações Aritméticas requerem a definição de dois argumentos, enquanto operações Auxiliares e de Transformadas exigem apenas um parâmetro.

Primeiro Argumento

O primeiro argumento diz respeito ao canal sobre o qual será aplicada a operação. Deve-se levar em conta que existem algumas operações restritas a determinada categoria de canais.

A operação escolhida será aplicada sobre um canal sem, contudo, modificá-lo. Para isso, um novo canal será gerado e, portanto, torna-se necessário a definição de um novo canal com os dados gerados a partir do resultado obtido. Por padrão, o sistema já insere as informações do novo canal tomando como base as informações do canal de origem, porém é possível modificar as seguintes informações:

- índice: número do canal
- nome: nome de identificação do canal
- fase: fase do canal
- unidade: unidade de medida do canal

Segundo Argumento

O segundo argumento diz respeito à segunda variável da operação. Caso seja escolhido a opção "Constante", deve-se inserir um valor real (são possíveis valores negativos e/ou em ponto flutuante) que corresponderá ao fator da operação. Caso seja escolhido a opção "Curva", deve-se selecionar o canal que corresponderá ao segundo parâmetro da operação.

5.4.2 Operações Aritméticas

Operações Aritméticas são as operações mais básicas e não necessitam do menu "Parâmetros". Necessitam sempre de dois operandos e só podem ser realizadas sobre canais analógicos ou derivados do tipo ARITMÉTICA.

Soma

O segundo argumento pode ser uma constante que será somada a cada valor da amostra. Caso o segundo argumento seja uma curva as duas amostras serão somadas conforme seus respectivos tempos.

Subtração

O segundo argumento pode ser uma constante que subtrairá o valor de cada amostra. Caso o segundo argumento seja uma curva cada valor de sua amostra subtrairá os valores correspondentes da primeira amostra.

Multiplicação

O segundo argumento pode ser uma constante que multiplicará o valor de cada amostra. Caso o segundo argumento seja uma curva cada valor de sua amostra multiplicará os valores correspondentes da primeira amostra.

Divisão

O segundo argumento pode ser uma constante que dividirá o valor de cada amostra. Caso o segundo argumento seja uma curva cada valor de sua amostra dividirá os valores correspondentes da primeira amostra.

Caso uma constante ou os dados de uma amostra sejam muito próximas (ou iguais) ao valor zero, então o resultado correspondente não poderá ser visualizado.

A figura 13 mostra um exemplo de operação aritmética de soma entre dois canais analógicos. São selecionados os canais UL1E e UL2E. O sistema é responsável por preencher as informações do novo canal automaticamente com base no primeiro canal selecionado. O usuário pode ainda modificar as alterações conforme desejar.

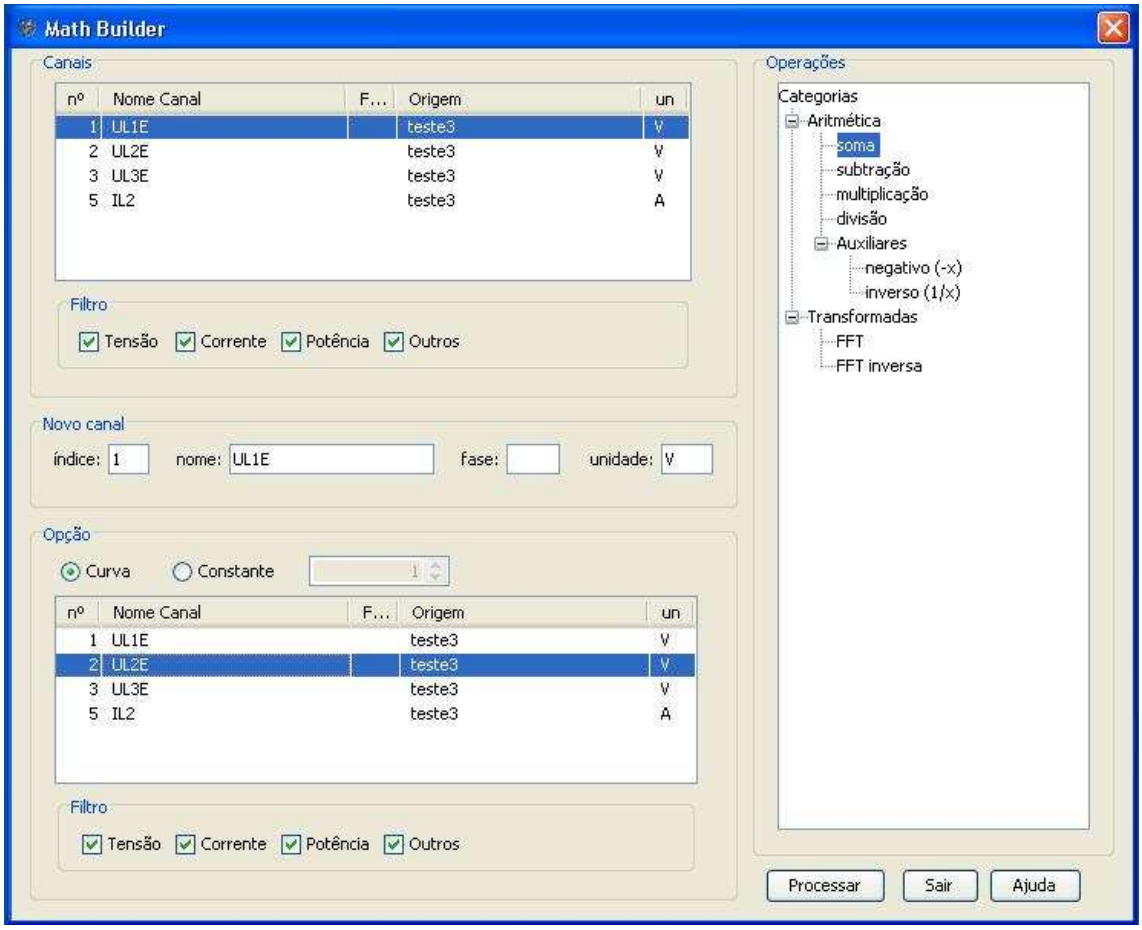

Figura 13 - Math Builder: Operação Aritmética

5.4.3 Operações Auxiliares

Operações Auxiliares não necessitam do menu "Parâmetros" bem como não há a necessidade de especificar um segundo argumento. Caso o valor de uma amostra seja muito próxima (ou igual) ao valor zero então o resultado correspondente não poderá ser visualizado.

Negativo (-x)

Os valores da amostra do canal selecionado são multiplicados pelo valor -1 obtendo-se desta forma um novo canal com simetria em relação ao eixo da abscissa.

Inverso (1/x)

Os valores da amostra do canal selecionado são invertidos, ou seja, o valor resultante será igual a 1 dividido pelo valor da amostra.

5.4.4 Operações de Transformadas

Operações de transformadas utilizam um novo domínio de referência diferente do tempo. Este domínio, denominado Domínio da Freqüência, representa seus sinais em componentes espectrais e por isso existem algumas restrições de operações conforme o tipo de canal. Um novo menu é disponibilizado (menu "Parâmetros") para a configuração de alguns parâmetros necessários para a transformação de um sinal no domínio do tempo para o domínio da freqüência.

FFT

A operação FFT (*Fast Fourier Transform*) é um algoritmo otimizado que transforma sinais analógicos discretos no tempo em sinais espectrais no domínio da freqüência. Só pode ser aplicada sobre canais analógicos ou derivados do tipo ARITMÉTICA. Além disso, não é necessário especificar a unidade de medida, pois o sinal gerado será um sinal espectral no domínio da freqüência e as respectivas amplitudes terão valores adimensionais representando o valor inicial DC seguido das múltiplas harmônicas do sinal.

A operação FFT disponibiliza um menu parâmetro para a configuração de algumas propriedades:

- Freq. Fundamental (Hz): Valor da freqüência fundamental (em Hertz) que é a base para o cálculo das harmônicas múltiplas. O valor inicial é o valor da freqüência do sistema fornecido no arquivo de configuração.
- Tempo inicial (ms): Tempo (em milissegundos) do começo da transformação. Dados obtidos antes desse tempo serão desconsiderados. O valor inicial é 0.
- Número de ciclos: Número de ciclos da amostra. O valor inicial é 1. Caso seja fornecido um valor maior que o número máximo de ciclos da amostra então será adotado o valor máximo para o cálculo.

FFT inversa

A operação FFT inversa age sobre um canal do tipo FFT transformando sinais espectrais no domínio da freqüência para sinais discretos no domínio do tempo. Desta forma, só pode ser aplicada sobre canais do tipo FFT. Um exemplo dessa transformação pode ser vista na figura 14.

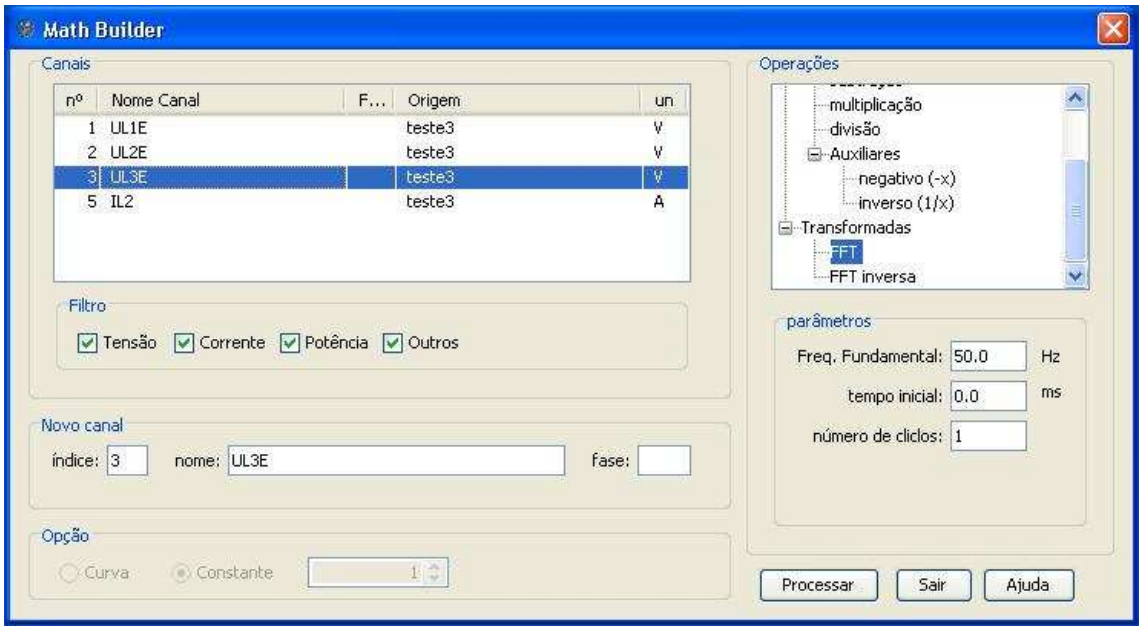

Figura 14 - *Math Builder*: Operação FFT

5.5 Módulo 3: Exibir Gráficos

5.5.1 Configurar exibição de gráficos

Antes de exibir os dados importados é necessário realizar algumas configurações. Isso é necessário, pois o sistema deve tratar diversas peculiaridades como a aglutinação de canais oriundos de arquivos diferentes. Além disso, existem canais incompatíveis que não podem ser exibidos em uma mesma área devido à natureza de sua representação.

Áreas de Exibição

Por padrão, existem 4 áreas, numeradas de 1 a 4, disponíveis para a exibição do sistema. Desta forma, pode-se trabalhar separadamente em diferentes áreas e não há a necessidade de se utilizar todas nem respeitar uma ordem pré-definida de uso.

Cada área pode receber vários canais de diferentes arquivos. Quando isso acontece os dados dos diferentes canais são mesclados em uma mesma área de exibição, ou seja, uma mesma área conterá gráficos de diferentes canais.

Existem restrições de exibição conjunta de tipos de canais. Ou seja, o sistema não permite a exibição de gráficos, em uma mesma área, de canais cujos tipos são de naturezas distintas.

Canais

Cada palheta referente a cada tipo de canal só é habilitada caso exista pelo menos um canal correspondente visível para o sistema.

Os canais devem ser selecionados e transferidos para uma área desejada. Pode-se selecionar vários canais simultaneamente através do mouse (botão esquerdo) e pelo teclado (tecla Ctrl).

Existem 3 formas de transferência de canais:

1. Arrastar e soltar

Após selecionados os canais de interesse basta pressionar o botão esquerdo do mouse sobre um dos itens da seleção e mantê-lo pressionado enquanto arrasta-se o mouse até uma área soltando em seguida o botão pressionado.

2. Botões

O botão "Mover Selecionados" permite que os canais previamente selecionados sejam movidos automaticamente para a área que estiver selecionada. O botão "Mover Todos" move todos os canais da palheta em destaque para a área selecionada independente de qualquer seleção previamente realizada.

3. Imagem de transferência

Os canais podem ser movidos individualmente através da imagem ao final de cada linha que representa a transferência para a área que estiver em destaque.

A figura 15 mostra um exemplo de tela de configuração de gráficos.

Restrições de Exibição

A tela de configuração de Gráficos permite que se mesclem vários tipos de canais para se compor uma área de exibição que provê a visualização de diversos gráficos simultaneamente.

Porém alguns tipos de canais não podem dividir o mesmo espaço de visualização devido a natureza diversa dos canais suportados pelo sistema. Desta forma, algumas restrições são impostas a fim de manter a consistência e evitar estados inválidos na criação dos gráficos.

A seguir são mostrados os tipos de áreas existentes e os tipos de canais suportados por eles.

1. Área de canais Digitais

   Uma área de exibição de canais digitais suporta apenas canais do tipo Digital.

2. Área de canais Analógicos

   Uma área de exibição de canais analógicos suporta apenas canais do tipo Analógico bem como canais derivados do tipo Aritmético.

3. Área de canais FFT

   Uma área de exibição de canais FFT suporta apenas canais do tipo FFT.

Escolhida a configuração para cada área de exibição basta clicar no botão "OK" para que uma nova janela interna seja criada com os gráficos correspondentes aos canais selecionados.

Operação inversa

Caso seja necessário modificar os itens transferidos para as áreas de exibição devolvendo-os para suas respectivas categorias de canais o sistema disponibiliza as operações inversas, porém deve-se respeitar a compatibilidade entre canais.

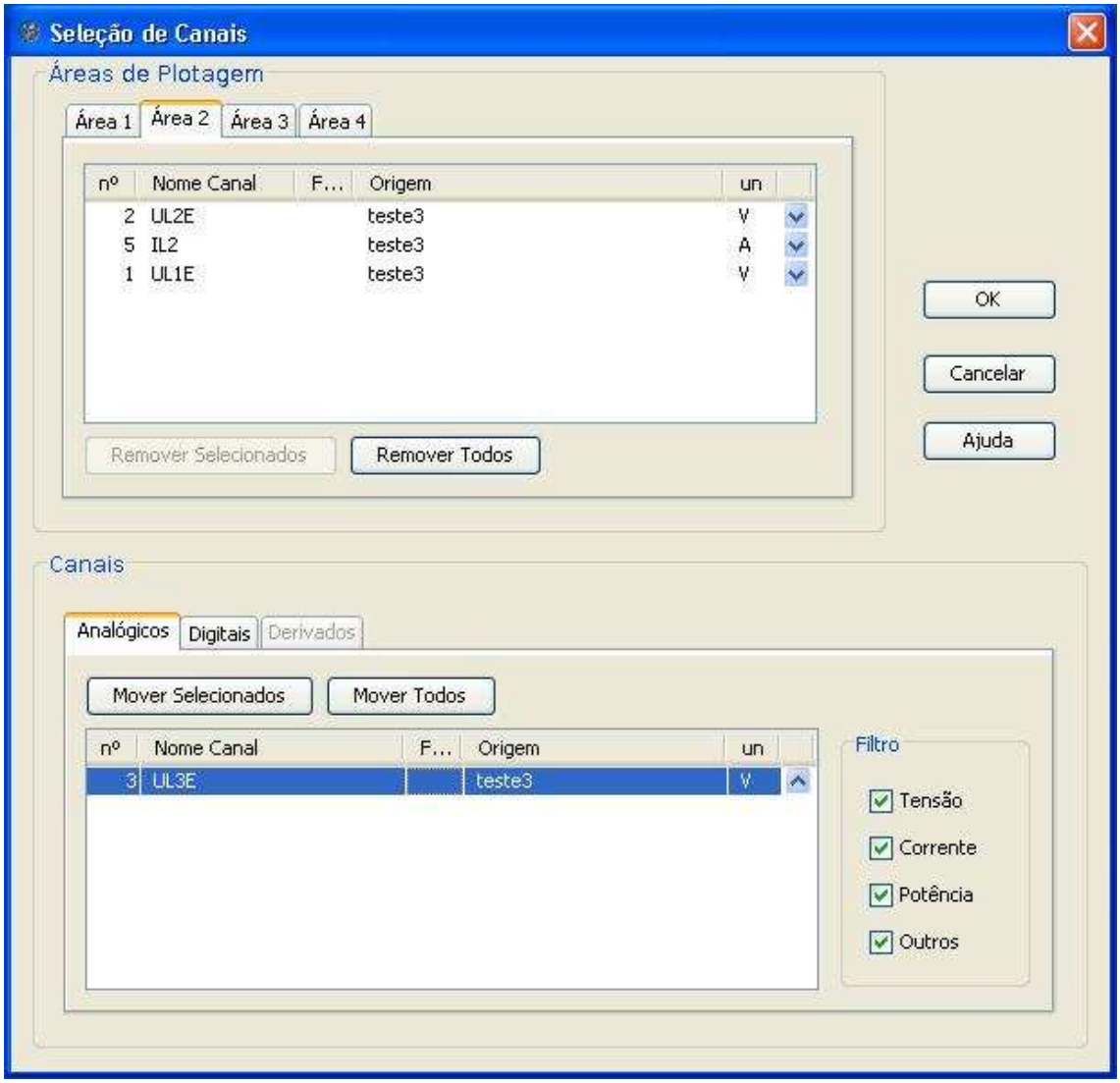

Figura 15 - Configuração de gráficos

5.5.2 Exibição de Gráficos

Os gráficos são apresentados em até quatro painéis sendo que cada painel representa uma das áreas de exibição do sistema. O número de painéis varia de acordo com o número de áreas preenchidas pelo usuário quando o mesmo configurou a exibição dos gráficos. Para cada canal colocado em uma área de exibição haverá uma série correspondente exibida no gráfico, ou seja, tem-se uma série representando cada canal. Além dos gráficos, existe, na janela interna, um painel de opções.

Tipos de Gráficos

Cada tipo de canal possui seu respectivo gráfico. A seguir são apresentadas suas características:

1. Gráficos de Canais Analógicos e Derivados Aritméticos

O gráfico correspondente a esses canais apresenta no eixo das abscissas o tempo em milissegundos, que varia de zero a um determinado tempo equivalente ao maior tempo entre todas as séries. O eixo das ordenadas representa a unidade da série variante no tempo. Cada unidade possuirá um eixo no gráfico, e esses eixos são posicionados alternadamente entre a direita e a esquerda do gráfico.

Os canais possuem séries apresentando os valores RMS e fasoriais. Além disso, cursores verticais possibilitam a análise dos valores em cada instante de tempo.

2. Gráficos de Canais Digitais

Os canais digitais são representados por gráficos de categoria na horizontal. O eixo das abscissas representa o tempo em milissegundos. O eixo das ordenadas apresenta o nome do canal.

As barras desse tipo de gráfico possuem apenas duas cores, azul e vermelho. Cada cor representa um dos níveis lógicos do sinal digital. O azul corresponde ao nível lógico "0" e o vermelho ao nível lógico "1". Esse tipo de gráfico também possui cursores.

3. Gráficos de Canais Derivados FFT

Os canais derivados FFT são representados por gráficos de categoria na vertical. O eixo das abscissas apresenta as harmônicos do sinal enquanto o eixo das ordenadas apresenta a magnitude. Caso dois canais tenham freqüências fundamentais diferentes, um novo gráfico será criado para diferenciá-los. O novo gráfico compartilha o eixo das ordenadas.

A figura 16 exemplifica a exibição de gráficos aglutinados (analógico e derivado aritméticos), digitais e FFT .

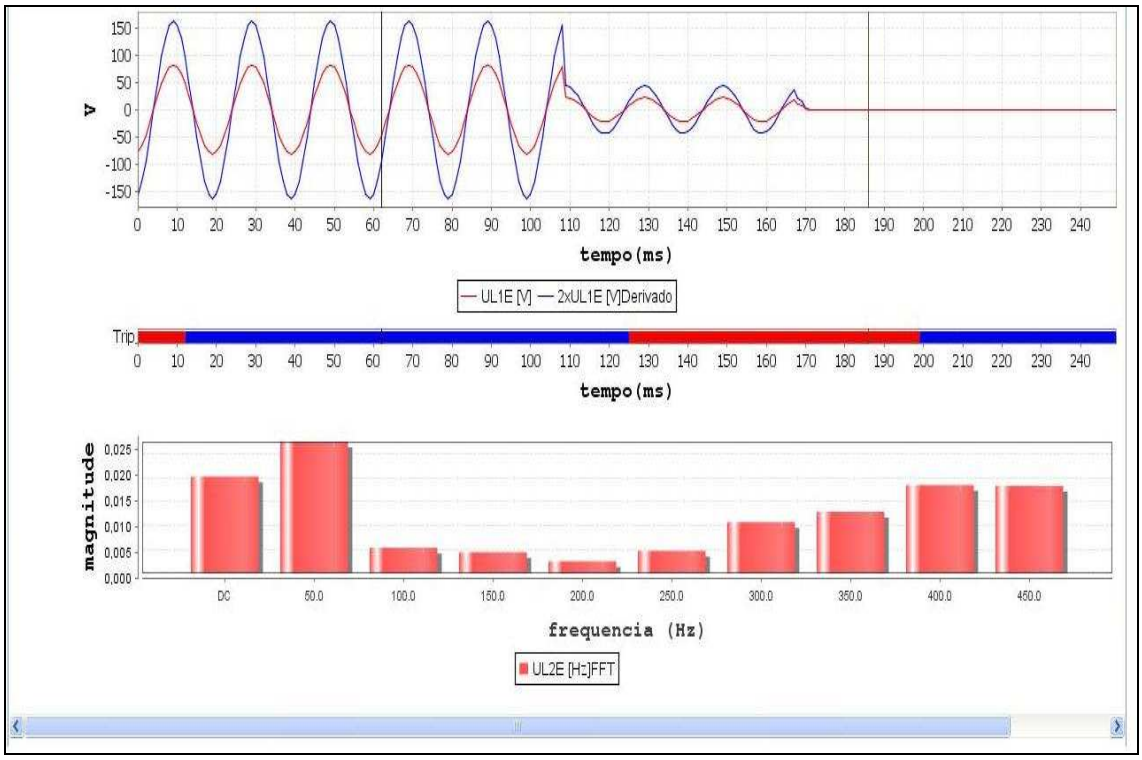

Figura 16 - Graficos genéricos

Valor RMS

Cada canal analógico ou derivado aritmético terá um gráfico com o valor RMS correspondente. O valor RMS representa o valor quadrático eficaz da magnitude de uma quantidade variável.

Os sinais dos valores RMS são colocados no mesmo gráfico de seus respectivos canais analógicos ou derivados aritméticos, com as mesmas unidades podendo também ser analisadas pelos cursores, conforme mostrado na figura 17.

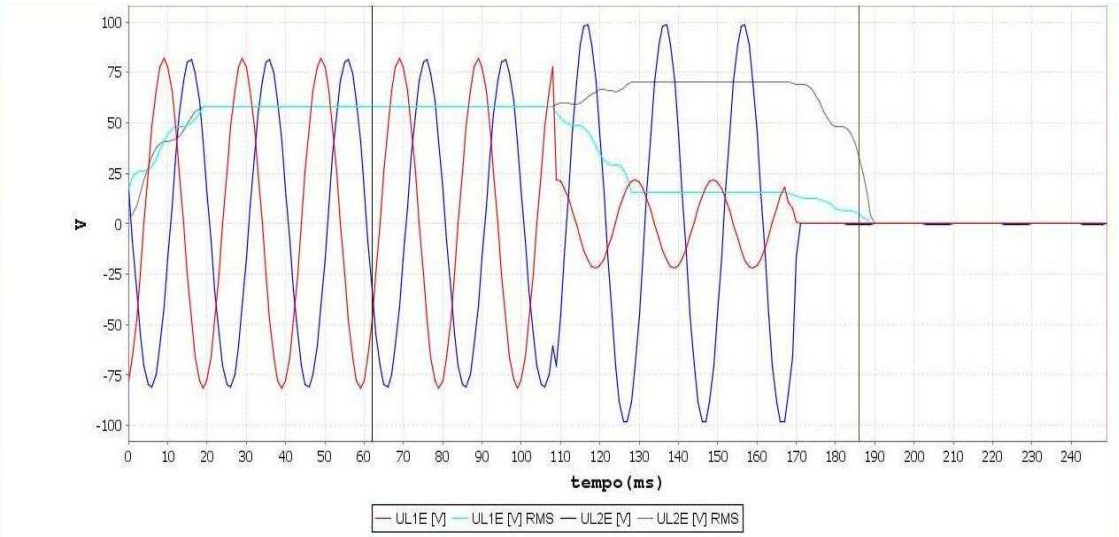

Figura 17 – Gráfico RMS

Gráficos Fasoriais

Cada canal analógico ou derivado aritmético terá uma série representando o ângulo e a amplitude máxima em um certo tempo. Essas séries serão mostradas em uma interface da própria janela interna de exibição de gráficos, sendo que esta possui o painel de opções, dois painéis e dois controles deslizantes. Os controles deslizantes determinam o tempo que será representado o ângulo e a fase das séries nos gráficos.

Encontra-se nos painéis um gráfico do tipo vetorial, esse gráfico apresenta um vetor correspondente a cada série que é a projeção bidimensional do sinal no tempo. Na ponta desse vetor é mostrado sua amplitude e fase. Conforme o controle deslizante se movimenta o gráfico é modificado. A figura 18 mostra um exemplo desse tipo de gráfico.

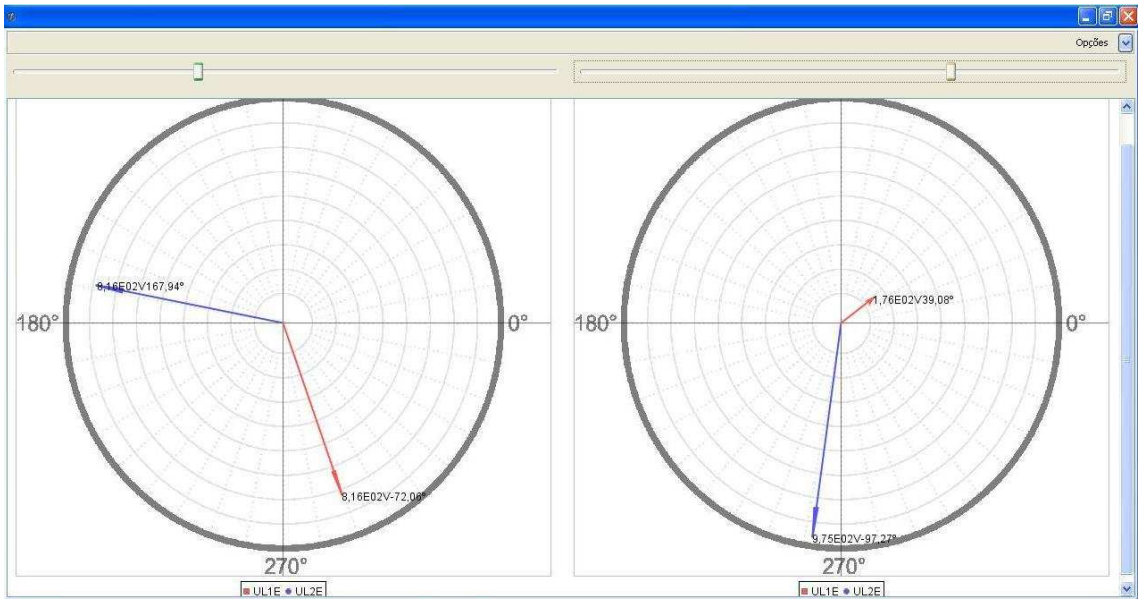

Figura 18 - Gráfico fasor

Cursores

As linhas verticais que passam pelos gráficos da janela interna de exibição possibilitam a visualização dos valores e das operações entre gráficos. Esses valores são exibidos em uma tabela de operações no painel de opções e também na tabela de tempo, conforme pode ser visualizado na figura 19.

Para movimentar os cursores, é preciso habilitá-los através dos botões "Ativar Cursor 1" e "Ativar Cursor 2" localizados na tabela de tempo. Quando os cursores não estão habilitados, a única maneira de movimentá-los é através dos controles deslizantes da interface de exibição de gráficos fasoriais.

Depois de habilitado, pode-se alterar a posição do cursor de 2 maneiras:

1. Duplo Clique

Após a realização de um duplo clique na área de exibição de gráficos, o cursor habilitado passa a ocupar a posição horizontal do mouse.

2. Clique sobre linha vertical

Ao posicionar o mouse próximo ou em cima da linha vertical habilitada, nota-se que o ícone do mouse é modificado para cursor de redimensionamento. Realizando um clique, a

linha vertical passa a acompanhar a posição horizontal do mouse. Após outro clique o cursor a movimentação é desabilitada.

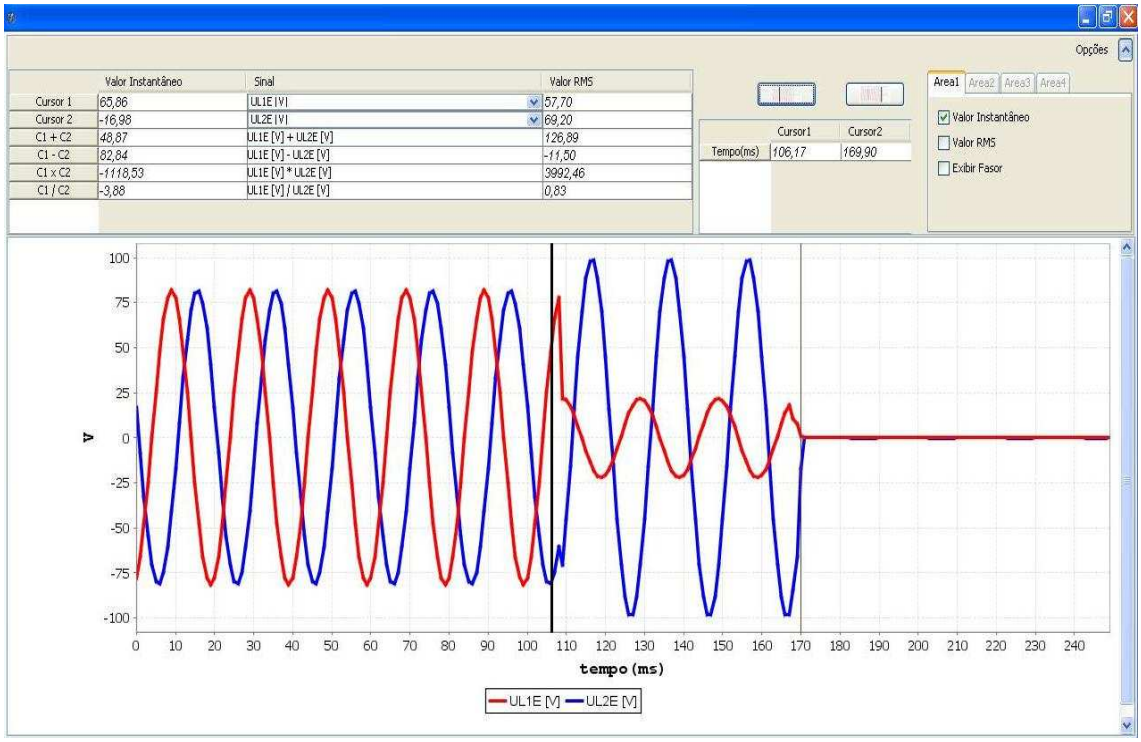

Figura 19 - Gráfico com cursores

Painel de Opções

O painel de opções da janela interna de exibição de gráficos está dividido em três partes: Tabela de Operações, Tabela de Tempo e Painel Tabulado. A figura 20 mostra um painel de opções e suas respectivas partes.

1. Tabela de Operações

Essa tabela apresenta seis linhas e três colunas. As linhas da coluna 1 apresentam o valor dos cursores 1 e 2 juntamente com as quatro operações aritméticas básicas entre eles. A duas primeiras linhas da segunda coluna apresentam Caixas de Combinação com os nomes de todas as séries analógicas e derivadas aritméticas dos gráficos exibidos. Enquanto nenhum canal é escolhido, as operações não são realizadas e, portanto, a tabela mantém os seus valores em zero. Portanto, caso não haja canais analógicos ou aritméticos, a tabela não atualiza seus valores. Quando um canal é selecionado, o traço no gráfico fica mais largo para facilitar sua visualização.

Por fim, na última coluna da tabela encontram-se os valores dos cursores 1 e 2 referentes ao sinal RMS da série selecionada nas caixas de combinação. Analogamente a primeira coluna, abaixo das linhas onde encontra-se os valores dos cursores 1 e 2, observa-se as quatro operações aritméticas básicas entre os cursores.

2. Tabela de Tempo

A tabela apresenta duas células sendo elas responsáveis por mostrar os tempos em milissegundos sobre os quais os cursores passam. Além da tabela de tempo encontra-se logo acima os botões "Ativar Cursor 1" e "Ativar Cursor 2".

3. Painel Tabulado

O painel tabulado da barra de opções apresenta 4 abas, sendo que cada uma é responsável por um dos gráficos analógicos ou derivados aritméticos exibidos. Quando houver gráficos digitais ou derivados FFT, a aba é desabilitada. O número de abas é igual ao número de áreas configuradas na tela de configuração de gráficos.

Cada aba possui três caixas de seleção. Quando a primeira é marcada, os gráficos da série passam a ser mostrados. Quando essa caixa de seleção é desmarcada, verifica-se se a segunda caixa de seleção(valor RMS) está marcada. Caso positivo, os gráficos passam a exibir somente as séries de valor RMS. Se desmarcada a janela interna de exibição de gráficos exibe a interface de gráficos fasoriais referente aos canais do gráfico correspondente a aba que possui a caixa de seleção.

É possível visualizar o valor RMS concomitantemente ao valor instantâneo. O gráfico fasorial exige representação própria e por isso é mostrado independentemente das outras formas.

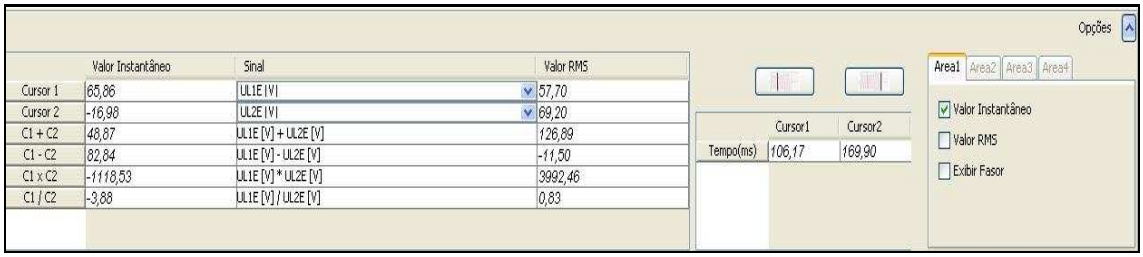

Figura 20 - Painel de opções

Propriedades

1. Zoom

Para melhor visualizar os gráficos na janela interna de exibição de gráficos, pode-se aplicar um zoom nos mesmos, independente do tipo de gráfico gerado. Também é possível realizar a operação inversa e retornar ao tamanho original da imagem.

2. Copiar

Para copiar a imagem do gráfico basta clicar com o botão direito do mouse sobre a imagem e escolher a opção do menu *pop-up* “Copiar”.

3. Imprimir

Para imprimir a imagem do gráfico basta clicar com o botão direito do mouse sobre a imagem e escolher a opção do menu *pop-up* "Imprimir...".

4. Salvar

Para salvar a imagem do gráfico basta clicar com o botão direito do mouse sobre a imagem do gráfico e escolher a opção do menu *pop-up* “Salvar Como...”.

Na caixa “Nome do arquivo”, escolha o diretório e o nome do arquivo e clique em Salvar. Deve-se levar em consideração que o único formato reconhecido atualmente é o PNG (*Portable Network Graphics*).

6 Testes Realizados com a ferramenta WAPS

Este capítulo é responsável por analisar a confiabilidade da ferramenta WAPS. Para isso alguns testes foram realizados a partir de um projeto de testes bem definido e que engloba a maioria das funcionalidades da ferramenta como: a utilização do conceito de multi arquivos, uso de canais analógicos e digitais, geração de arquivos derivados aritméticos e de transformada, exibição de valores instantâneos e RMS, verificação fasorial, gráficos digitais e de transformadas, etc.

Para exemplificar e analisar a confiabilidade da ferramenta, o projeto de testes será dividido em duas fases para englobar todas as funcionalidades da ferramenta.

6.1 Projeto de testes Fase 1

O projeto da fase 1 pode ser visualizado na figura 21. Dois arquivos serão selecionados. O primeiro arquivo possui tanto arquivos analógicos quanto digitais, mas serão selecionados apenas 4 canais analógicos e 2 canais digitais de seu repositório. O segundo arquivo possui apenas canais analógicos e serão selecionados 3 destes canais. Em seguida, serão realizadas duas operações aritméticas: soma entre dois canais analógicos e "negativação" da amostra. Para verificação dos resultados obtidos serão mostrados os gráficos representativos de cada canal, além disso as representações fasoriais e RMS serão visualizadas.

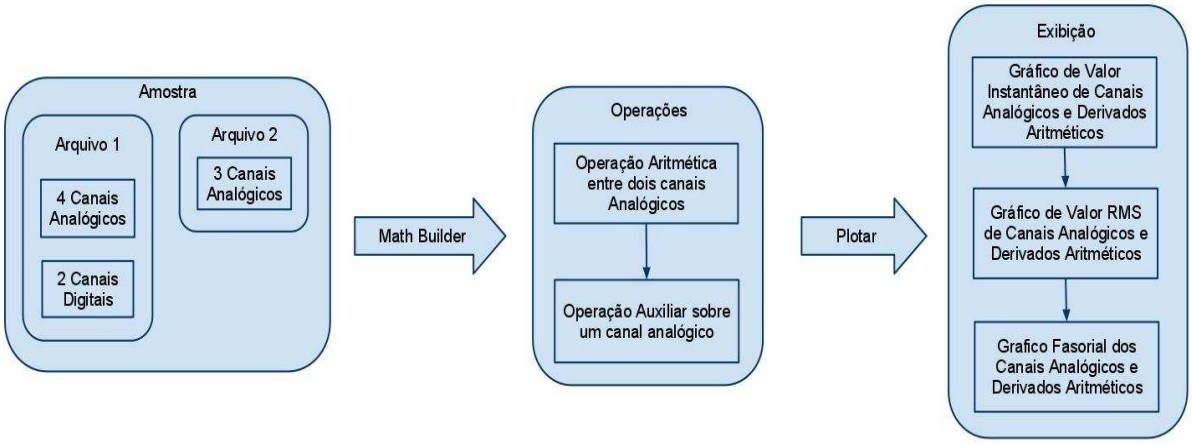

Figura 21 - Planejamento do teste fase 1

6.1.1 Abertura de Arquivo e Seleção de canais

Dois arquivos genéricos serão importados pela ferramenta. O primeiro possui seis canais analógicos sendo três destes canais de tensão e outros três canais de corrente. Existem ainda neste primeiro arquivo dois canais digitais que serão selecionados para importação assim como dois canais de tensão e dois canais de corrente conforme mostrado na figura 22.

O segundo arquivo possui seis canais sendo três de tensão e três de corrente. Dois canais de tensão e outro de corrente serão selecionados, conforme mostra a figura 23.

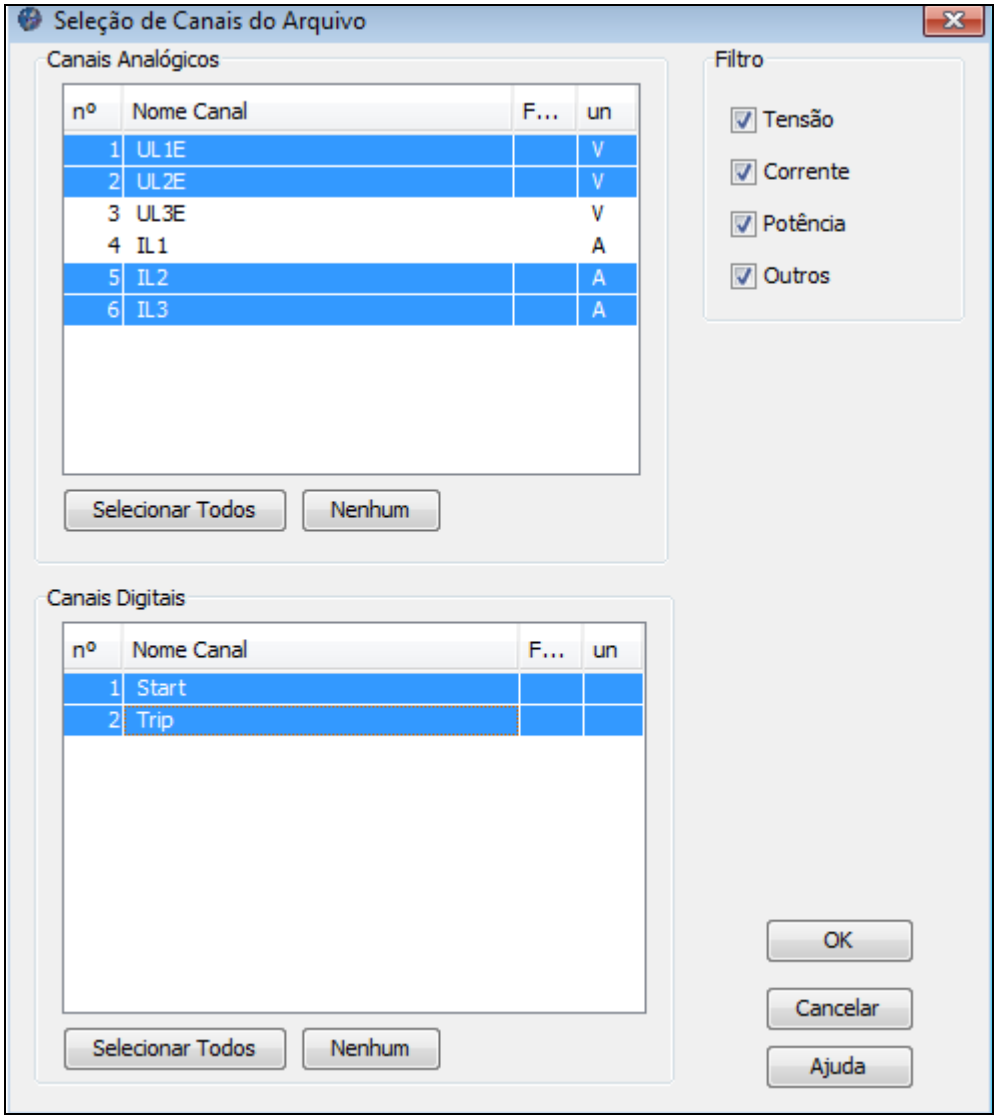

Figura 22 - Arquivo 1 Fase1: Seleção de Canais

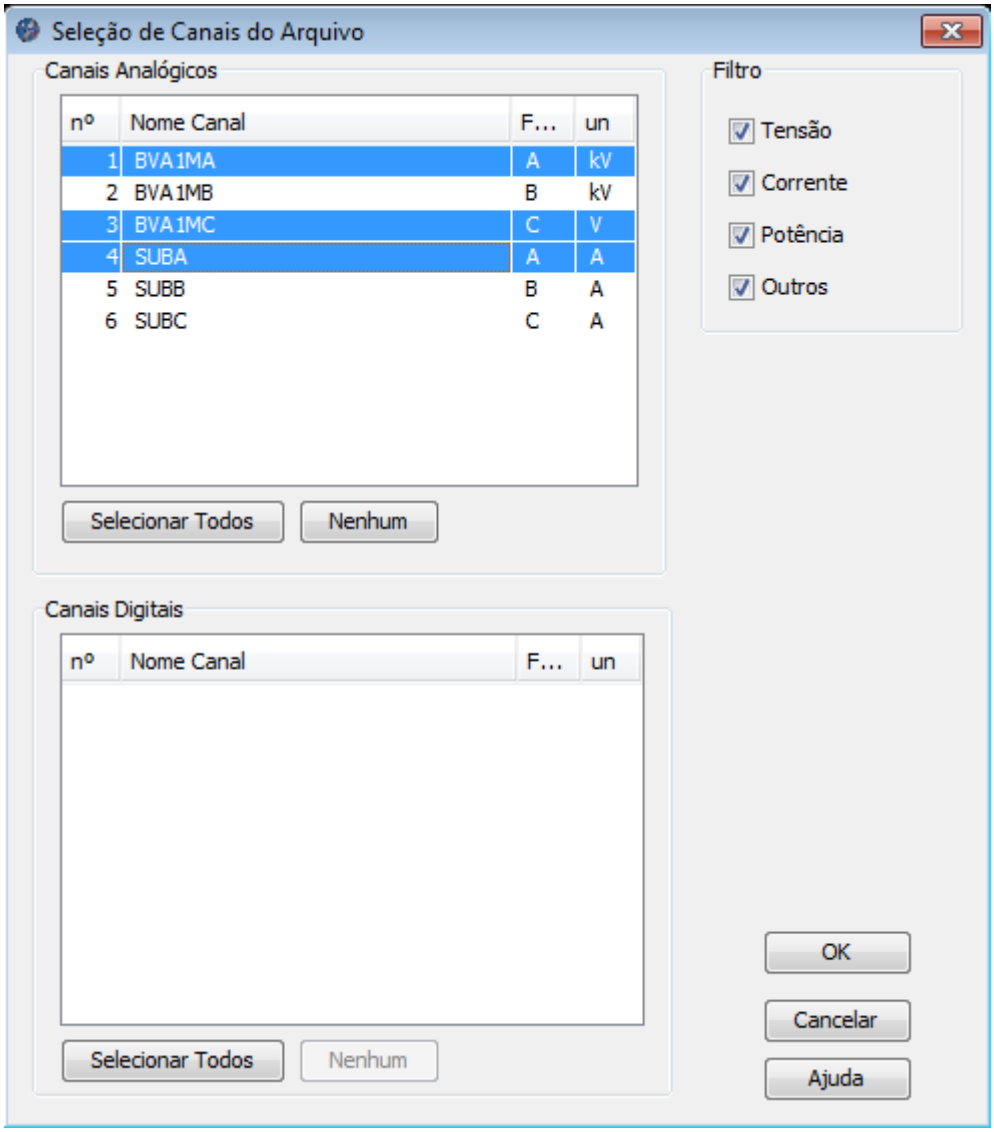

Figura 23 - Arquivo 2 Fase1: Seleção de canais

6.1.2 Manipulação de Canais com *Math Builder*

A primeira manipulação diz respeito à operação aritmética de soma entre dois canais analógicos. Foram escolhidos os canais UL1E e UL2E para análise. A figura 24 exemplifica tal transformação.

A segunda manipulação será realizada sobre o canal SUBA do arquivo 2. Uma operação de negativo (-x) é efetuada conforme figura 25.

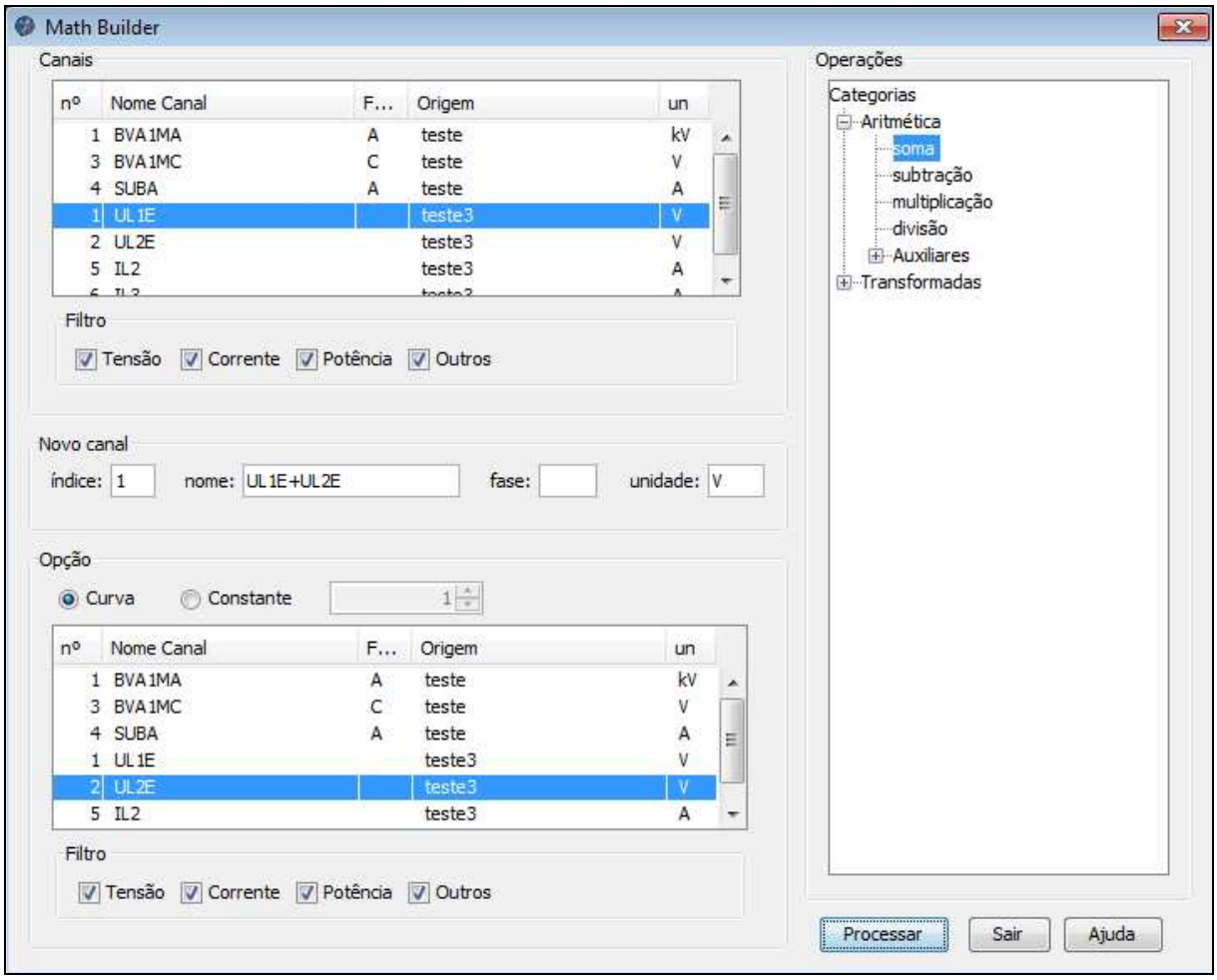

Figura 24- Math Builder: Soma

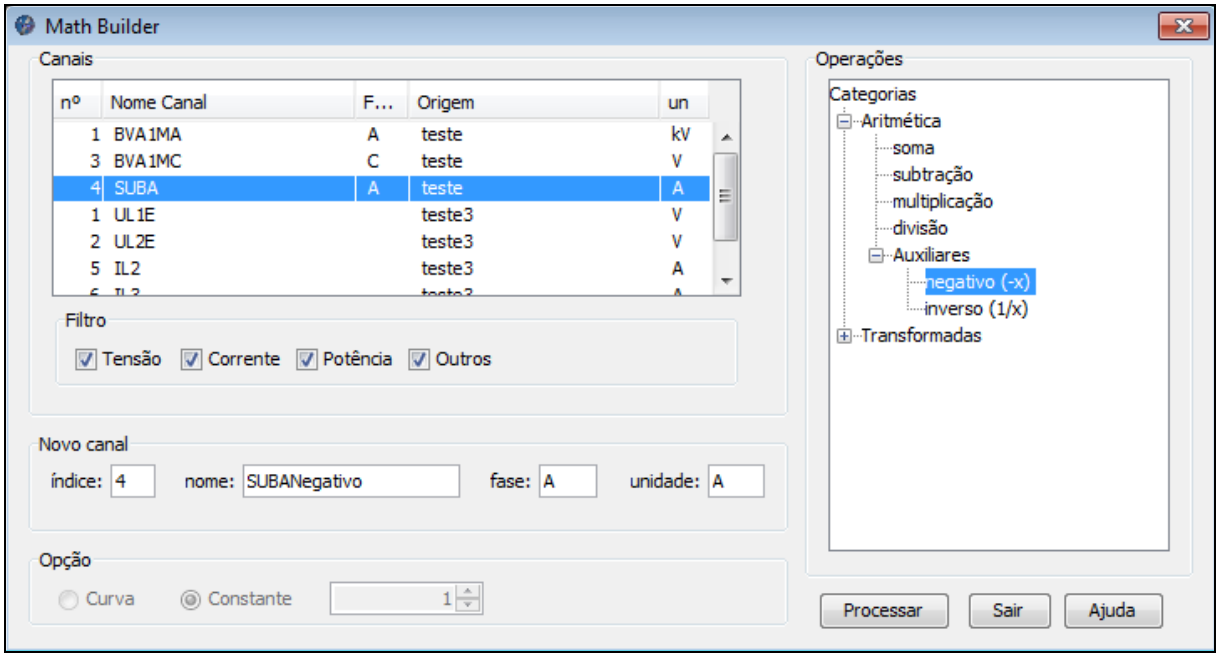

Figura 25 - Math Builder: Negativo

6.1.3 Resultados obtidos

A figura 26 mostra os valores instantâneos dos canais UL1E(vermelho), UL2E(azul) e do canal criado UL1E+UL2E(verde). Pode-se observar que o canal UL1E+UL2E realmente representa a soma dos outros dois canais que estão defasados. Ele segue os canais ficando entre ambos. Perto do tempo de 110ms, após o começo da amostragem, é possível ver a distorção do sinal UL1E ocorrida devida a alguma falta. Como seu valor diminui em relação ao canal UL2E o canal soma possui valores muito próximos a esse último canal.

Os canais Start e Trip são canais digitais e por isso possuem representação em formato de barras horizontais. Os valores "0" e "1" são diferenciados pela cor de cada segmento. Os valores "0" são representados pela cor azul enquanto os valores "1" são representados pela cor vermelha. Canais digitais geralmente representam eventos lógicos que ocorrem na linha de transmissão.

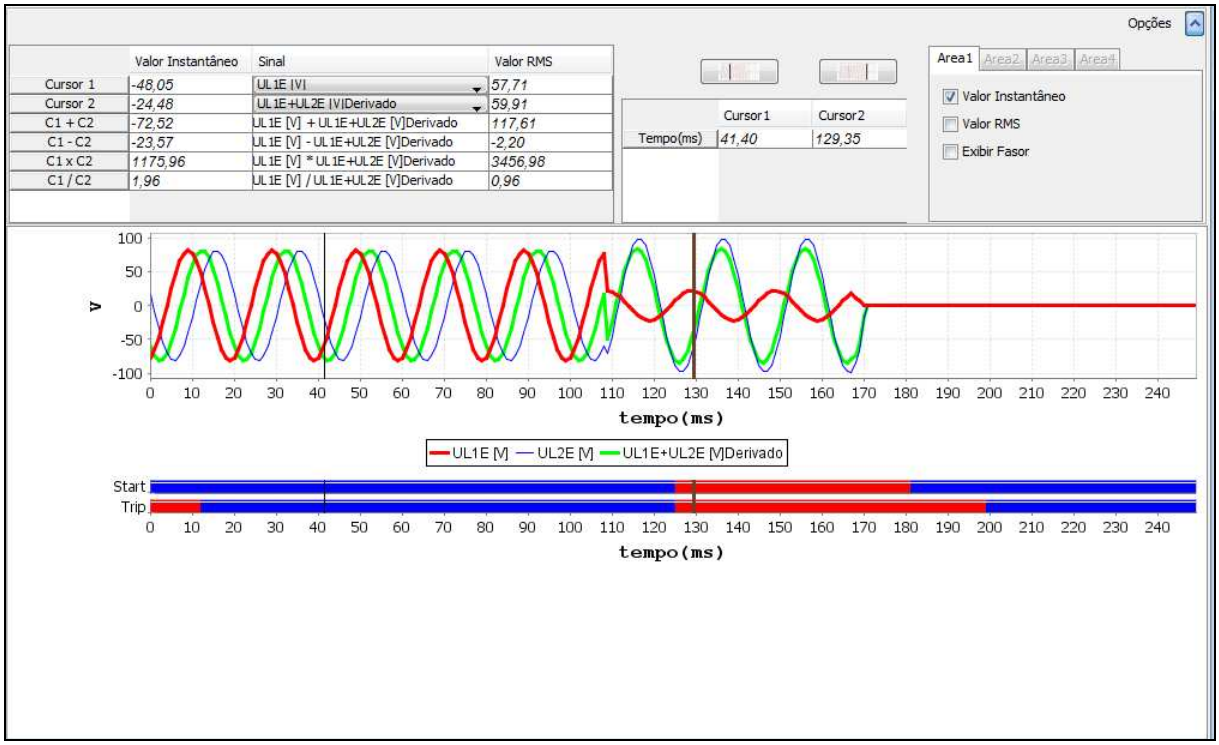

Figura 26 - Gráfico: valores instantâneos

A figura 27 é a representação dos canais em valores RMS. Pode-se perceber que o valor eficaz de todos os canais antes de ocorrer a falta se mantém constante e com mesmo valor. Isso ocorre porque os canais UL1E e UL2E estão defasados e como a soma entre ambos possui mesma amplitude o valor eficaz se mantém constante. Após a falta o sinal

UL1E cai em relação ao valor do canal UL2E então o canal soma possui valor muito próximo do canal UL2E.

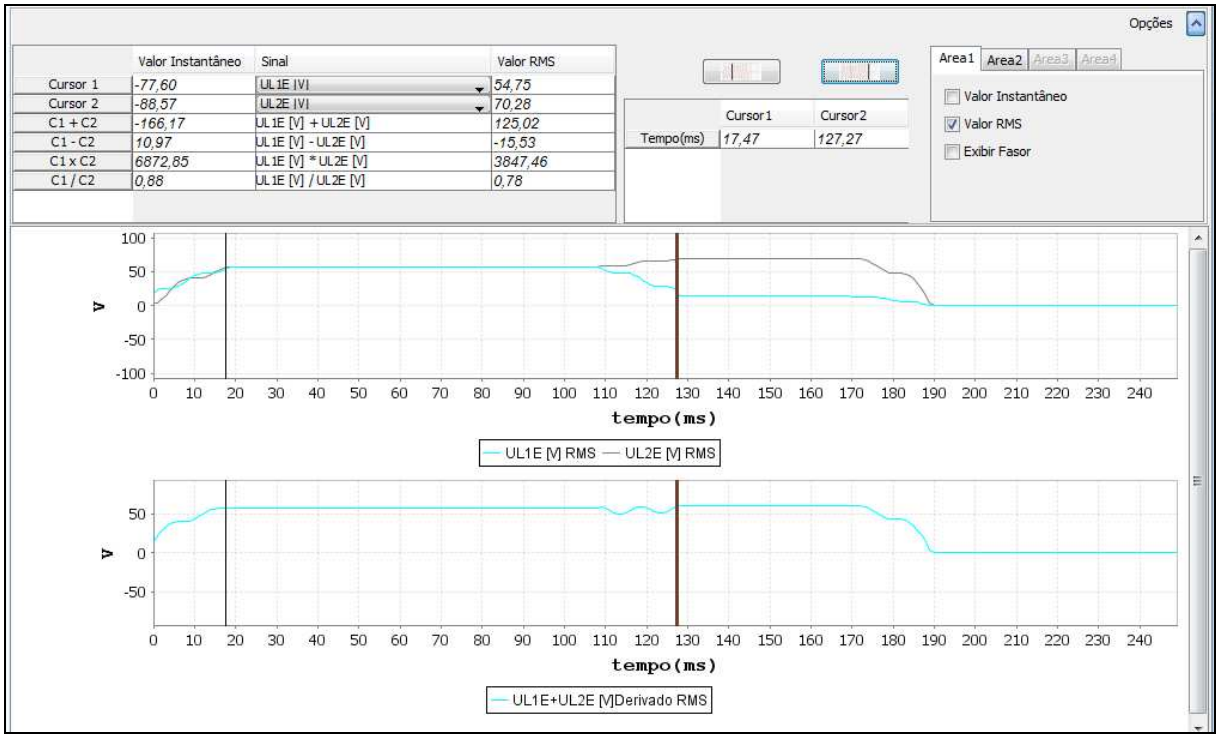

Figura 27 - Gráfico: Valores RMS

A figura 28 mostra a representação fasorial dos canais UL1E e UL2E. É possível ver como ocorre a defasagem entre ambos os canais conforme se desloca a barra dos cursores no tempo.

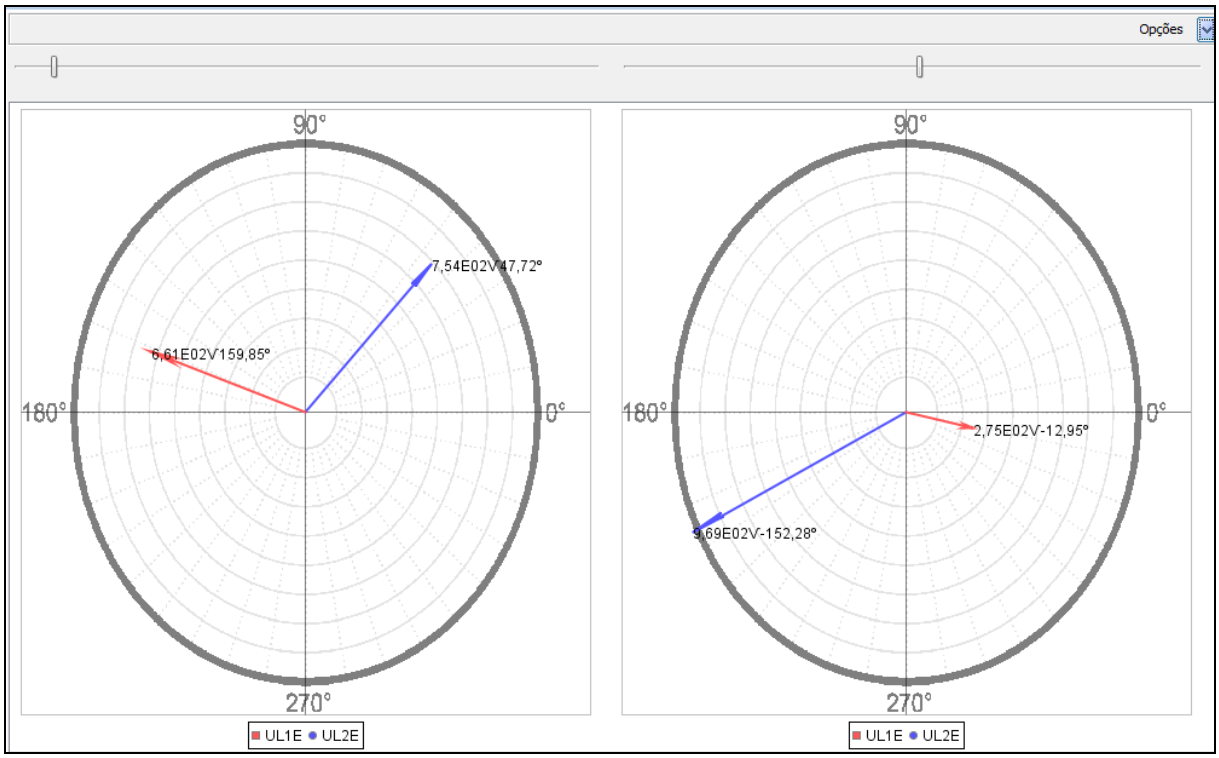

Figura 28 – Gráfico: Representação Fasorial

6.2 Projeto de testes Fase 2

O projeto da fase 2 pode ser visualizado na figura 29. Nessa fase apenas um arquivo será selecionado. Tal arquivo possui somente canais analógicos e desses canais será selecionado para a realização de duas operações. A primeira corresponde à transformação rápida de Fourier, FFT. A segunda é justamente a operação inversa para obtenção do sinal original. Para verificação dos resultados obtidos serão mostrados os gráficos representativos de cada canal.

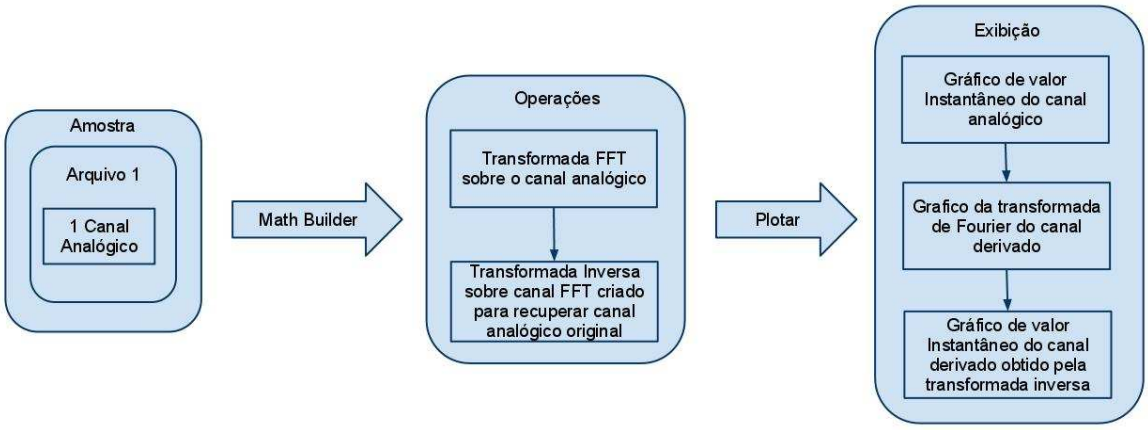

Figura 29 - Planejamento do teste fase 2

6.2.1 Abertura de Arquivo e Seleção de canais

Apenas um arquivo será aberto desta vez. O arquivo 2 da fase 1 possui apenas canais analógicos sendo 3 de tensão e 2 de corrente. Será selecionado o canal BVA1MA, conforme mostrado na figura 30.

6.2.2 Manipulação de Canais com *Math Builder*

A primeira manipulação diz respeito à transformada de Fourier a partir do algoritmo FFT sobre o único canal selecionado. Desta forma pode-se ver posteriormente o espectro de freqüências do sinal, a figura 31 mostra a realização da operação.

A segunda manipulação será realizada sobre o canal BVA1MAFFT. Esse é o mesmo canal criado no passo anterior, ou seja, será realizada a operação de FFT inversa para obtenção do sinal original. A operação pode ser vista na figura 32.

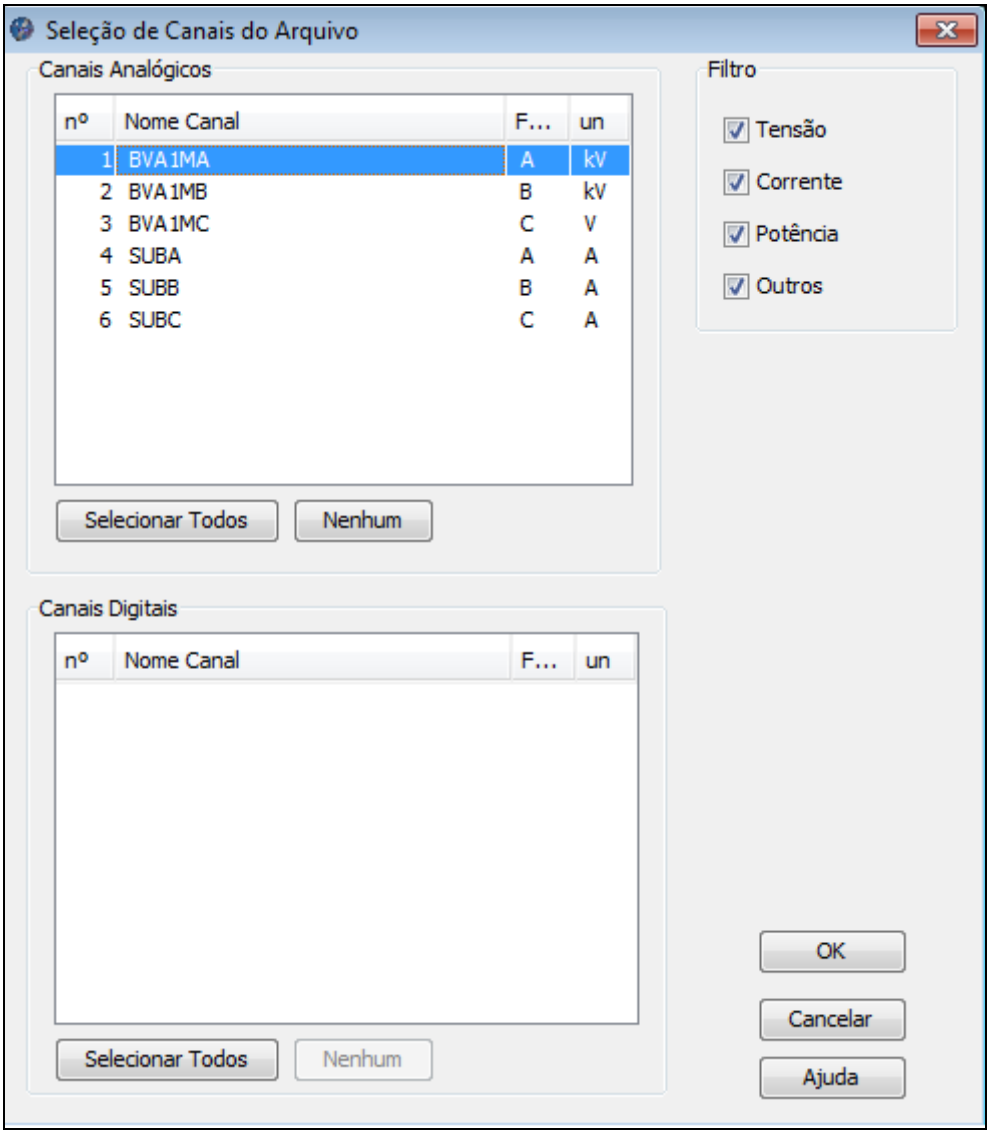

Figura 30 - Arquivo 2: Seleção de Canal

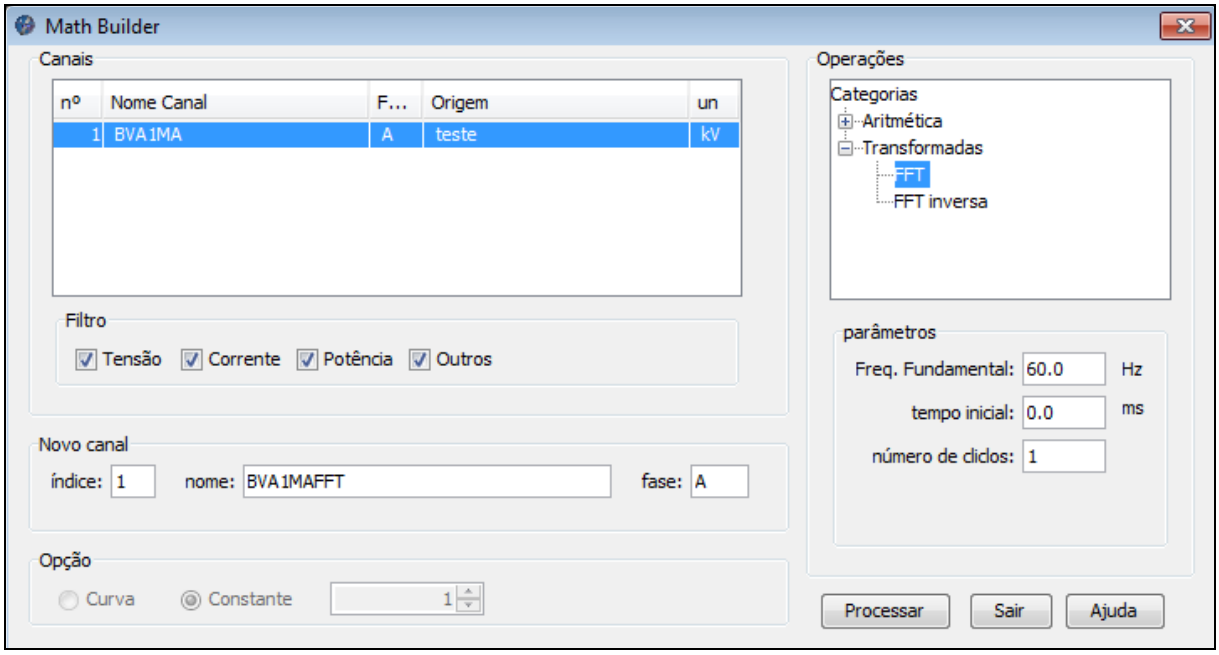

Figura 31 – Math Builder: Operação FFT

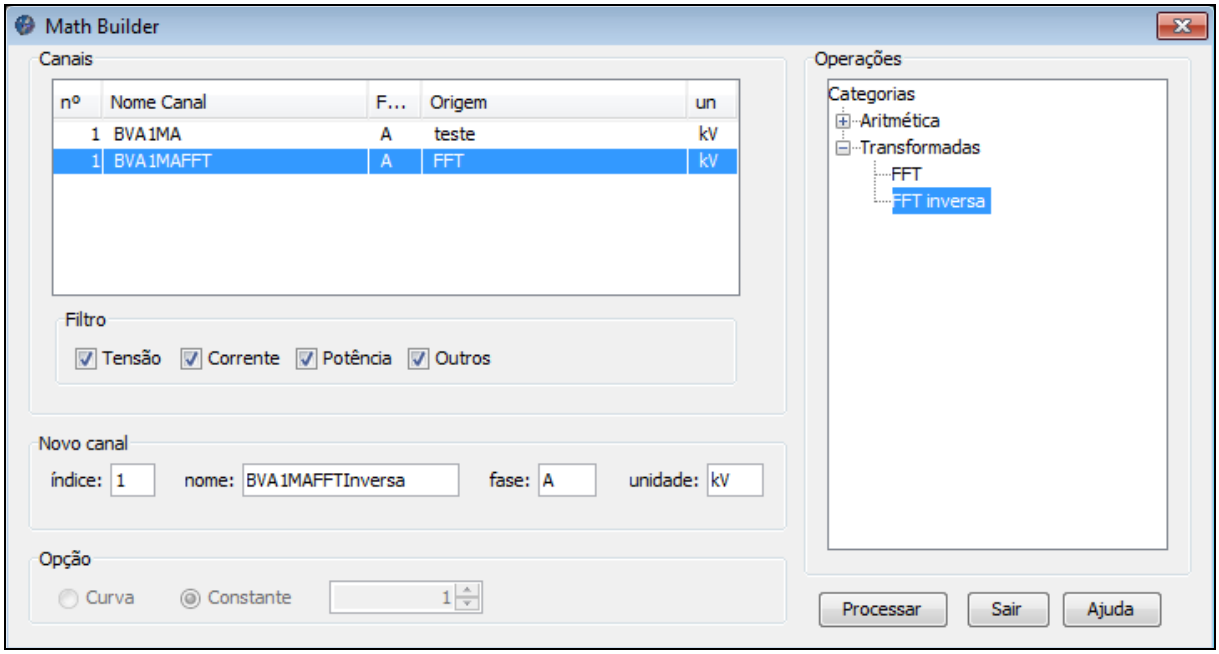

Figura 32- Math Builder: Operação FFT Inversa

6.2.3 Resultados obtidos

A figura 33 aglutina os gráficos dos três canais analisados. O primeiro gráfico diz respeito ao próprio sinal analógico BVA1MA. É uma representação no domínio do tempo. O segundo gráfico diz respeito a transformada FFT sobre esse canal. É uma representação no domínio da freqüência e, portanto, apresenta o valor DC do sinal bem como suas harmônicas nas freqüências múltiplas da freqüência fundamental. O terceiro gráfico é a transformada inversa que recupera o sinal original.

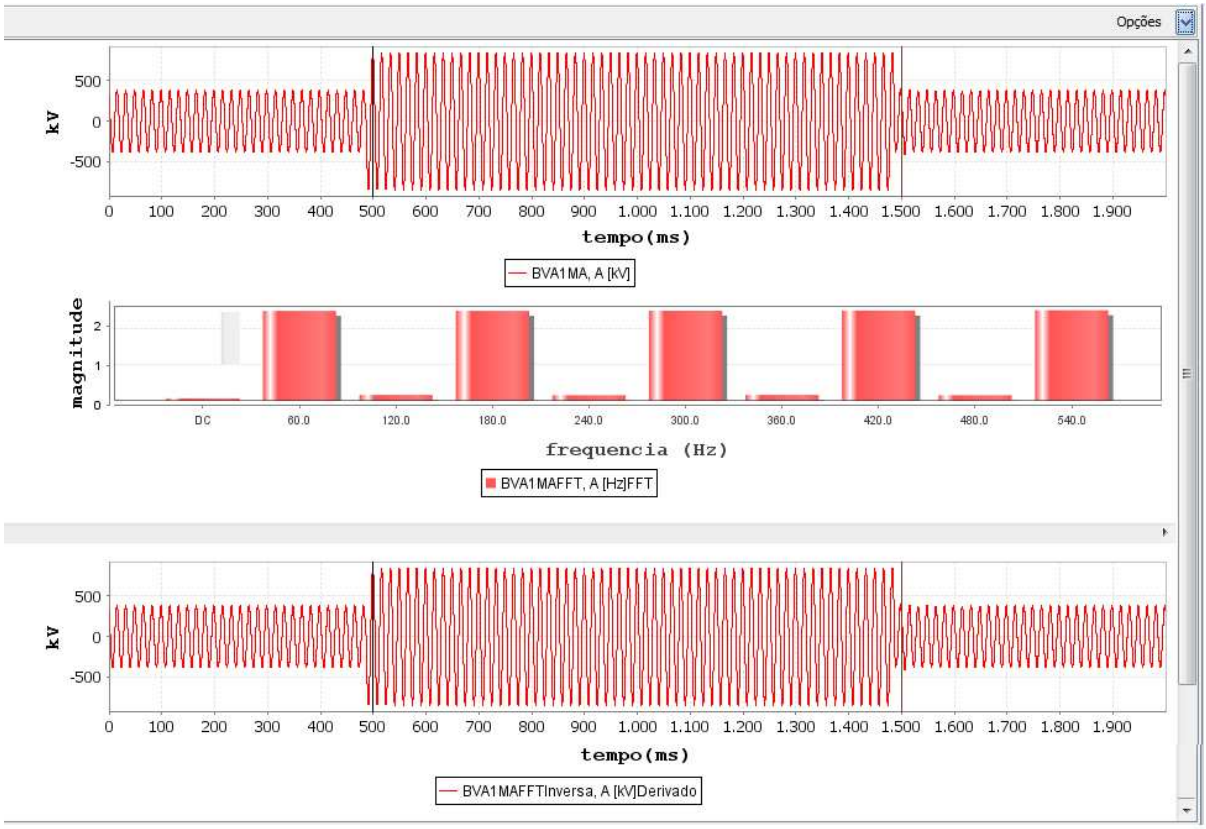

Figura 33 - Gráfico: valor instântâneo e RMS

Apenas como exemplificação fica a figura 34 como um exemplo genérico e não significativo de uma representação espectral quando sinais possuem freqüências fundamentais distintas. É criado um gráfico para cada sinal já que eles não podem dividir a multiplicidade das freqüências harmônicas.

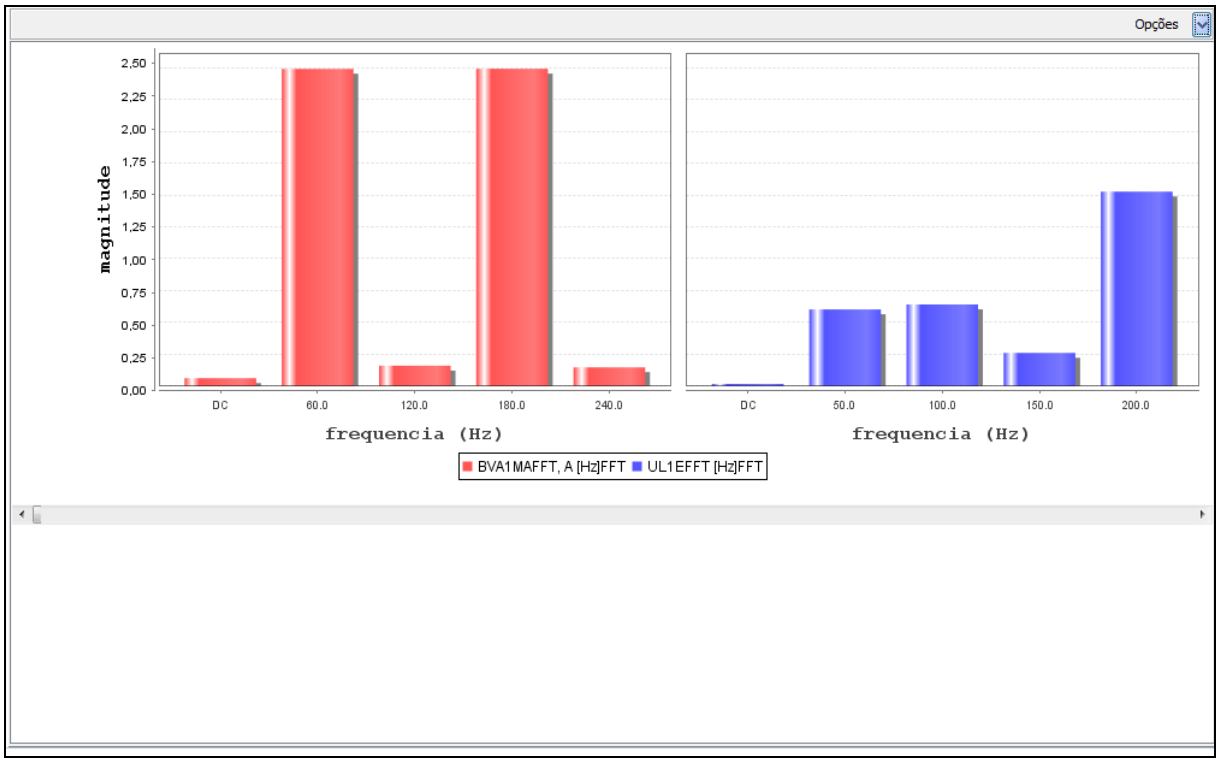

Figura 34 – Exemplo de representação espectral de sinais com freqüências fundamentais distintas

Pode-se perceber que a visualização dos dados é diferenciada para cada tipo de canal e, para a maioria dos canais, existe mais de um tipo de representação ajudando o usuário da aplicação a obter o maior número de informações de forma rápida e fácil. Para aumentar a flexibilidade, a tela de configuração permite ao usuário definir como serão exibidas as diversas informações podendo-se agrupá-las da forma que achar necessário.

Canais são diferenciados por cor permitindo a distinção imediata de suas formas. As diversas legendas são separadas por gráfico e representam seus respectivos canais indicando suas unidades de medidas que são expostas nos eixos coordenados.

Os resultados obtidos revelam a concordância com o valor esperado para as operações propostas indicando a viabilidade da ferramenta WAPS. O teste revela a capacidade da ferramenta em um ambiente multi arquivos com canais distintos. A manipulação dos canais com a criação de novos canais derivativos ocorreu de forma simples já que a interface foi desenvolvida para ser utilizada de forma amigável e intuitiva.

7 Conclusões

Este trabalho aborda a construção de uma ferramenta de código livre, WAPS (*Wave Analyser for Power System*), voltada ao estudo e análise de perturbações e transitórios de redes de energia elétrica. Essa ferramenta foi desenvolvida no Laboratório de Sistemas de Energia Elétrica – LSEE pertencente à Escola de Engenharia de São Carlos – EESC localizada na Universidade de São Paulo – USP e reconhece arquivos do padrão IEEE COMTRADE.

Como a ferramenta foi construída sobre a premissa de *software* livre, permite à comunidade científica voltada à área de proteção de sistemas elétricos de potência utilizá-la de forma gratuita e adaptá-la as suas necessidades.

O uso dos conceitos de orientação a objetos permitiu construir um *software* modular e bem organizado ajudando possíveis desenvolvedores que queiram modificá-lo. A utilização da linguagem Java permitiu a portabilidade essencial para disseminar a ferramenta em diversos ambientes.

A ferramenta possui uma interface simples e amigável que permite ao usuário rápido aprendizado além de fornecer um sistema de ajuda integrado em todas as telas do sistema. Mesmo a barra de status localizada na parte inferior do *software* exibe informações úteis que permite fácil navegação entre os módulos.

Para se retirar informações de tipos de canais distintos concluiu-se que era necessário uma representação distinta para cada tipo de forma que facilitasse a manipulação do usuário da aplicação. A API utilizada para geração de gráficos embora, de certa forma genérica, não possui suporte gratuito para o seu entendimento. Desta forma, despendeu-se muito tempo para personalização das funcionalidades, porém o resultado obtido foi recompensador.

A proposição de modularidade da ferramenta gerou diversas dificuldades. O padrão inicial proposto, COMTRADE, embora poderoso é muito simples e pouco flexível. Outros padrões existentes no mercado como o padrão PQDIF, *Power Quality Data Interchange Format*, é muito mais robusto, porém não se adéqua ao padrão existente do WAPS e por isso

é necessário revisão da estrutura principal do programa. As modificações necessárias devem se restringir localmente devido ao conceito de modularidade explanado no capítulo 3.

As tarefas e projetos realizados contribuíram para a compreensão de novos conceitos principalmente relacionados à prática de programação orientada a objetos, projeto de sistemas, conceitos sobre sistemas elétricos de potência, e o desenvolvimento do raciocínio lógico, aumentado-se de forma geral a capacidade para a solução de situações problemas.

Para trabalhos futuros na ferramenta WAPS destaca-se a revisão da estrutura principal do programa para prover maior flexibilidade de acoplamento dos módulos de leitura de padrões distintos.

O padrão PQDIF poderá ser adicionado como ferramenta reconhecida pelo programa devido a sua grande importância na área de proteção, principalmente no que diz respeito à qualidade de energia.

Novas operações devem ser agregadas ao módulo *Math Builder* para que atenda as necessidades do mercado e seja mais robusto. Algumas operações sugeridas podem envolver o cálculo de deslocamento no tempo das amostras, cálculo de logaritmo e exponenciação em base escolhida pelo usuário. Seria ideal se esse módulo parecesse mais com uma calculadora, conforme é visto em outras aplicações.

Os gráficos também devem ser mais personalizáveis permitindo alterações de cor, espessamento de linha, etc. Seria importante que uma vez exibidos os gráficos fosse possível alterá-los arrastando-os nas diversas áreas existentes.

8 Referências Bibliográficas

[1] “IEEE Standard common format for transient data exchange (COMTRADE) for power systems”, Editado por Power System Relaying Committee of the IEEE Power Engineering Society, 1-55937-156-0, IEEE Press, 1991.

[2] “IEEE Standard common format for transient data exchange (COMTRADE) for power systems”, Editado por “Power System Relaying Committee of the IEEE Power Engineering Society”, 0-7381-1667-X, IEEE Press, 1999.

[3] ELECTROTEK CONCEPTS, INC. TOP, The Output Processor®: Software for Visualizing Monitoring Data and Simulation Results. Disponível em: <http://www.pqsoft.com/top/>. Acesso em: junho. 2011.

[4] SIEMENS, POWER TRANSMISSION AND DISTRIBUTION – OPERACIONAL SECTOR – SECUNDARY TECHNOLOGY. SIGRA 4 Demo. Download disponível em: <http://siemens.siprotec.de/download_neu/index_e.htm>. Acesso em: junho. 2011.

[5] BINH DAM, Q..NET RelayerTM Software Quick Start Guide: IEEE COMTRADE Viewer and Software for Protective Relay Simulation and Experiments. 2009.

[6] CEPEL. SINAPE: Sistema Integrado de Apoio à Análise de Perturbações. Informações disponíveis em: < http://www.sinape.cepel.br>. Acesso em: junho. 2011.

[7] Bobrow, D. and Stefik, M. February 1986. Perspectives on Artificial IntelligenceProgramming. Science vol. 231.

[8] Booch Grady, OBJECT-ORIENTED, Analysis and Design, 2ª Edição. Addison-Wesley. 1993.

[9] FERREIRA, Aurélio B. de Hollanda. Dicionário Aurélio da Língua Portuguesa. 5. ed. Rio de Janeiro: Nova Fronteira, 2010.

[10] Deitel, Harvey M. e Deitel, Paul J., JAVA, How to Program, 4ª edição. 2002.

[11] IEEE Std 280-1985 (R1996), IEEE Standard Letter Symbols for Quantities Used in Electrical Science and Electrical Engineering (DoD). 1996

[12] Russell Jordan, Inno Setup. Download disponível em: <http://www.jrsoftware.org/isinfo.php>. Acesso em: 4 de julho. 2011.