ELIPSESCADA

HMI/SCADA SOFTWARE

MANUAL DO USUÁRIO

7/1/2015 - Versão 2.29

Índice

Convenções

Estas são convenções utilizadas neste manual:

EXEMPLO	DESCRIÇÃO
SILO6.BMP	Nomes de arquivos e outros termos no nível do sistema operacional são indicados com o tipo de letra Tahoma, em maiúsculas.
Geral	Nomes de campos e opções que devem ser procurados na tela, em menus ou nas abas dos objetos são indicados com tipo de letra Tahoma.
"Agitação"	Caracteres entre aspas devem ser digitados no lugar mencionado, sem a presença das aspas.
`Tela1.Show()`	Partes de programas (*scripts*) são indicadas com o tipo de letra `Courier`. Eles deverão ser digitados nos lugares reservados e depois compilados para a verificação de erros.
Tank01.High	Caracteres em negrito indicam nomes de objetos do Elipse SCADA ou suas propriedades.
<nome do arquivo>	Expressões entre os sinais < > devem ser substituídas pelo nome do objeto em questão.
[Ctrl+Enter]	Expressões entre colchetes indicam nomes de teclas. Quando estiverem acompanhadas de um sinal +, você deve pressionar a segunda tecla enquanto pressiona a primeira.
	Este ícone sozinho significa que o recurso não está disponível para a versão Elipse SCADA CE; acompanhado de nota, significa que há restrições a sua utilização.

Capítulo 1

1. Introdução

Bem-vindo ao Elipse SCADA! A Elipse Software sente-se orgulhosa em apresentar esta poderosa ferramenta para o desenvolvimento de sistemas de supervisão e controle de processos.

O Elipse SCADA alia alto desempenho e grande versatilidade, representados em seus diversos recursos que facilitam e agilizam a tarefa de desenvolvimento de sua aplicação. Totalmente configurável pelo usuário, permite a monitoração de variáveis em tempo real, através de gráficos e objetos que estão relacionados com as variáveis físicas de campo. Também é possível fazer acionamentos e enviar ou receber informações para equipamentos de aquisição de dados. Além disto, através de sua exclusiva linguagem de programação, o Elipse Basic, é possível automatizar diversas tarefas a fim de atender as necessidades específicas de sua empresa.

Agradecemos a sua preferência por nossos produtos e desejamos sucesso com sua nova ferramenta de trabalho!

Equipe Elipse Software

1.1. Pacotes do Elipse SCADA

O Elipse SCADA está disponível em pacotes diferentes, atendendo as demandas de personalização de nossos clientes. A seguir, podemos observar as características de cada pacote:

Elipse View

O Elipse View é indicado para aplicações simples, como por exemplo uma interface com o operador para monitoração e acionamentos. Permite a visualização de variáveis, inclusive com a utilização de animações, programação de setpoints, controle de acesso e funções especiais para *touch-screen*. Este pacote inclui:

- Comunicação com equipamentos via drivers (DLLs) e OPC (Servidor e Cliente);
- Objetos de Tela;
- Visualização de alarmes ativos;
- Comunicação em bloco;

- Scripts;
- Servidor e cliente DDE;
- Servidor de rede Elipse;
- Controle de acesso através de lista de usuários.

O pacote não inclui ferramentas para o registro de dados históricos, alarmes ou relatórios, além de outras funcionalidades que venham a surgir em pacotes mais avançados.

Elipse MMI (Man Machine Interface)

É um software de supervisão completo. Possui banco de dados proprietário, relatórios formatados, históricos, receitas, alarmes e Controle Estatístico de Processos, facilmente implementáveis. Pode, ainda, ser um servidor da dados para outras estações Elipse. Inclui todos os recursos do pacote View, e mais:

Históricos, receitas e relatórios.

- Controle Estatístico de Processos (Módulo CEP);
- Objetos de tela Browser (históricos) e alarmes históricos;
- Registro de alarmes em disco.

O Elipse MMI é indicado para sistemas de qualquer porte, onde não sejam necessárias conexões com bancos de dados externos (ODBC e DAO) ou aplicações de rede, e quando o usuário precisa enxergar outras estações de supervisão.

Elipse Pro

É a mais avançada ferramenta do Elipse SCADA. Permite trocar dados em tempo real com outras estações, transferir/atualizar bancos de dados, realizar comandos e programar setpoints através de rede local ou linha discada. Inclui todos os recursos do pacote MMI, e mais:

- ODBC (Open DataBase Connectivity) e DAO (Data Access Objects);
- Cliente e servidor de rede Elipse (TCP/IP);

O Elipse Pro é a solução ideal para a comunicação com sistemas corporativos, pois suporta ODBC, DAO e diversos protocolos de rede. Além disso, este módulo permite a troca de informações com software dedicado a controle (SoftPLC).

Elipse SCADA CE

Este pacote permite executar aplicações Elipse SCADA em dispositivos baseados no sistema operacional Windows CE, como IHMs, dispositivos sem disco em geral e outros dispositivos móveis. O Elipse SCADA CE não comporta todas as funcionalidades dos pacotes anteriores; quando for este o caso, será indicado no decorrer do manual.

1.2. Módulos de Operação

O Elipse SCADA possui três módulos para sua operação: **Configurador**, **Runtime** e **Master** (inclui os módulos Configurador e Runtime). O módulo ativo é definido a partir do dispositivo de proteção (*hardkey*) acoplado ao computador. Enquanto que os módulos Configurador e Master foram especialmente desenvolvidos para a criação e o desenvolvimento de aplicativos, o módulo Runtime permite apenas a execução destes. Neste módulo, não é possível qualquer alteração no aplicativo por parte do usuário.

Na ausência da hardkey, o software pode ser executado em modo **Demonstração**, que pode ser utilizado para avaliação do software. O modo **Demo** possui quase todos os recursos existentes no módulo **Configurador**, com as seguintes diferenças:

- Não permite salvar aplicações com mais de 20 tags;
- Permite até cinco (5) conexões simultâneas do Elipse Web;
- Permite a execução de uma aplicação e comunicação com equipamentos de aquisição de dados por até duas horas.

Nesse modo, o software pode ser livremente reproduzido e distribuído.

Os módulos Runtime e Master estão também disponíveis em versões **Lite** que possuem as mesmas características, porém são limitadas em número de *tags* (variáveis): **Lite 75** com 75 tags, e **Lite 300** com 300 tags.

Na versão Windows CE, apenas o modo **Runtime** está disponível, em licenças de 75, 300 ou 1500 tags.

1.3. Plug-Ins

Plug-ins são ferramentas adicionais que permitem a expansão dos recursos do Elipse SCADA, acrescentando funcionalidades no software. Eles podem ser adquiridos separadamente e trabalham em conjunto com qualquer versão do software.

Atualmente, estão disponíveis os seguintes plug-ins:

Permite a monitoração de sistemas através de recursos de captura, registro e transmissão digital de imagens em tempo real. Suporta diversos padrões (inclusive MPEG), possibilitando a visualização em janelas com tamanho e qualidade programáveis pelo usuário. Permite a criação de um banco de imagens com busca por período ou evento e transmissão de imagens em tempo real para estações remotas via TCP/IP ou linha discada.

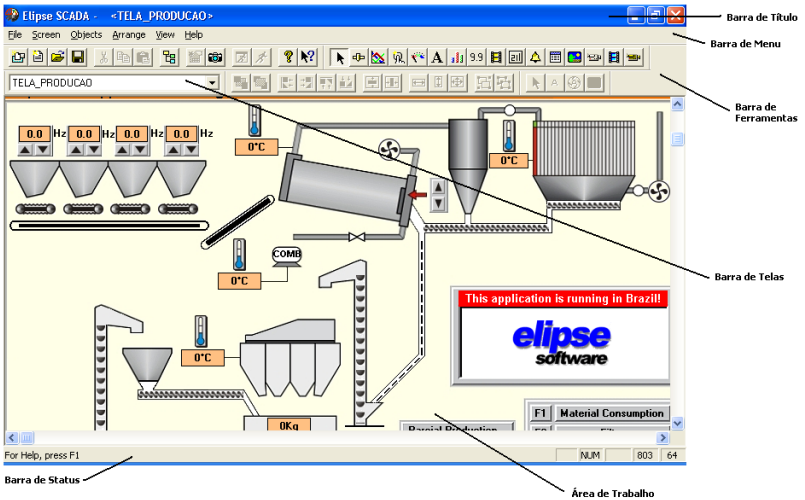
Sistema para supervisão de processos através da Internet. Utilizando qualquer navegador (Internet Explorer, Netscape e outros) é possível conectar-se a uma estação de supervisão remota, recebendo dados em tempo real. Com este recurso é possível visualizar processos de qualquer parte do mundo.

Capítulo 2

2. O Que Há na Tela

Uma maneira fácil de compreender o funcionamento do Elipse SCADA é partir das ferramentas disponíveis e sua apresentação em tela.

A ilustração a seguir mostra a tela principal do Elipse SCADA quando uma aplicação está aberta, no módulo Configurador, identificando seus elementos.



A **Barra de Título** mostra o caminho e o nome de sua aplicação, bem como o título da tela corrente que está sendo mostrada na área de trabalho. A área de trabalho é o espaço onde desenvolvemos a aplicação. A edição de telas e de relatórios é feita nessa área. A **Barra de Telas** mostra o título da tela corrente e permite que você alterne entre uma tela e outra. A **Barra de Menus** permite a escolha das diversas opções para a configuração da aplicação. Os botões da **Barra de Ferramentas** permitem que você execute determinadas tarefas rapidamente sem usar os menus. Assim, com apenas um clique, você pode criar objetos de tela ou chamar o Organizer, por exemplo. A **Barra de Status** mostra várias informações auxiliares quando editando uma aplicação, como por exemplo indicadores da ativação do teclado numérico (NUM), letras maiúsculas (CTRL) e rolagem de tela (SCRL) e coordenadas do ponteiro do mouse. Ela também mostra uma pequena descrição de um determinado objeto, por exemplo um Botão da Barra de Ferramentas ou um item de menu.

As aplicações que rodarão na versão Windows CE precisam ser necessariamente criadas na versão Windows, modo Configurador.

2.1. Opções de Menu

É através das opções de menu que podemos acessar os recursos e funções do software. Descrevemos as opções do Elipse SCADA a seguir.

2.1.1. Menu Arquivo

Figura 1: Menu Arquivo

Opções do Menu Arquivo

COMANDO	AÇÃO
Nova aplicação	Cria uma nova aplicação.
Abrir aplicação	Abre uma aplicação já existente.
Salvar aplicação	Salva a aplicação corrente.
Salvar aplicação como	Salva uma cópia da aplicação corrente em um novo arquivo.
Fechar aplicação	Fecha a aplicação corrente.
Rodar	Executa a aplicação corrente.
Organizer	Chama o Organizer.
Opções...	Permite configurar algumas opções do Elipse SCADA, como criar um arquivo de backup (.BAK) quando salvar a aplicação; configurações do mecanismo de proteção; e o nome do arquivo da biblioteca de língua (o default é INTLBR32.DLL). Nesta opção, também é possível especificar uma aplicação a ser carregada automaticamente quando o Elipse SCADA é ativado remotamente via OPC.
1, 2, 3 e 4	Lista dos quatro arquivos recentemente abertos
Sair	Encerra o Elipse SCADA.

2.1.2. Menu Tela

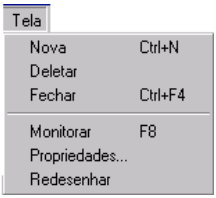


Figura 2: Menu Tela

Opções do Menu Tela

COMANDO	AÇÃO
Nova	Cria uma nova tela (tela em branco).
Deletar	Apaga a tela corrente.
Fechar	Fecha a tela corrente.
Monitorar	Permite monitorar a tela corrente.
Propriedades...	Mostra as propriedades da tela corrente, onde você pode definir uma imagem de fundo e o estilo da janela, entre outras características.

Redesenhar	Redesenha as telas carregadas.

2.1.3. Menu Objetos

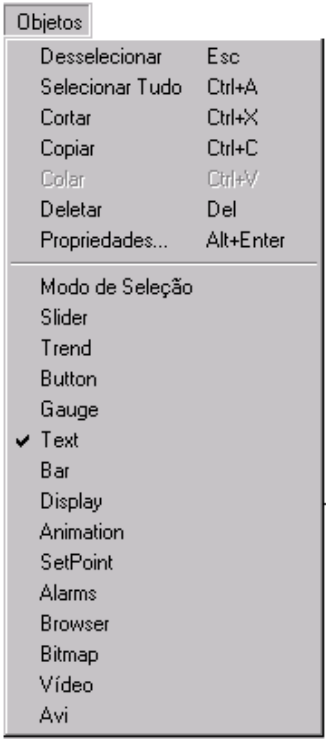


Figura 3: Menu Objetos

Opções do Menu Objetos

COMANDO	AÇÃO
Desselecionar	Desseleciona o objeto corrente.
Selecionar tudo	Seleciona todos os objetos da tela.
Recortar	Recorta o objeto selecionado.
Copiar	Copia o objeto selecionado para a área de transferência (*clipboard*).
Colar	Cola o objeto contido na área de transferência no local indicado.
Deletar	Apaga os objetos selecionados. Para selecionar mais de um objeto, use a tecla [Ctrl].
Propriedades	Mostra as propriedades do objeto selecionado. A mesma função pode ser ativada com um duplo clique sobre o objeto.
Modo de seleção	Liga o modo de seleção, permitindo que o usuário selecione todos os objetos dentro de uma área delimitada pelo mouse.

As demais opções criam objetos de acordo os respectivos nomes. Depois de escolher o Objeto de Tela desejado, deve-se selecionar uma região da tela para colocar o objeto mantendo-se pressionado o botão esquerdo do mouse enquanto ele é movimentado. Um retângulo pontilhado mostra o tamanho e a forma do objeto. Soltando-se o botão do mouse, o objeto será colocado dentro da área especificada.

2.1.4. Menu Arranjar

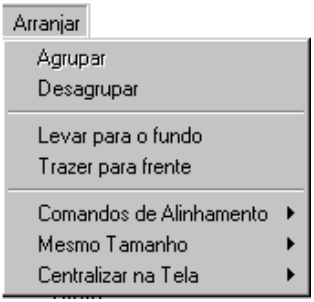


Figura 4: Menu Arranjar

Opções do Menu Arranjar

COMANDO	AÇÃO
Agrupar	Agrupa os objetos selecionados.
Desagrupar	Desagrupa o grupo de objetos selecionados.
Levar para o fundo	Envia o objeto selecionado para o último plano (fundo da tela).
Trazer para frente	Traz o objeto selecionado para o primeiro plano (frente da tela).
Comandos de alinhamento	Alinha objetos selecionados pela esquerda, pela direita, pelo topo ou pela base.
Mesmo tamanho	Faz com que os objetos selecionados tenham o mesmo tamanho.
Centralizar na tela	Centraliza na tela os objetos selecionados.

2.1.5. Menu Visualizar

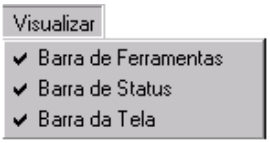


Figura 5: Menu Visualizar

Opções do Menu Visualizar

COMANDO	AÇÃO
Barra de Ferramentas	Mostra ou esconde a Barra de Ferramentas.
Barra de Status	Mostra ou esconde a Barra de Status.
Barra da Tela	Mostra ou esconde a Barra de Telas.

2.1.6. Menu Ajuda

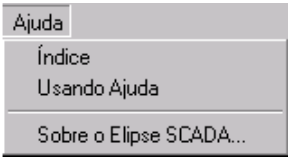


Figura 6: Menu Ajuda

Opções do Menu Ajuda

COMANDO	AÇÃO
Índice	Mostra o índice da ajuda.
Usando Ajuda	Explica como a Ajuda deve ser usada.
Sobre o Elipse SCADA...	Mostra informações a respeito do Elipse SCADA, como a versão, o número do *hardkey* e direitos de cópia.

2.2. Barra de Ferramentas

A **Barra de Ferramentas** oferece um acesso rápido às funções do Elipse SCADA. Essas funções são distribuídas em quatro conjuntos, a saber: **Aplicação**, **Objetos de Tela**, **Arranjar** e **Telas**. Vejamos cada um deles.

2.2.1. Barra de Ferramentas Aplicação

Figura 7: Barra de Ferramentas Aplicação

Opções da Barra de Ferramentas Aplicação

BOTÃO	DESCRIÇÃO
	Cria uma nova aplicação.
	Cria uma nova tela.
	Abre uma aplicação já existente.
	Salva a aplicação corrente.
	Recorta o objeto selecionado copiando para a área de transferência.
	Copia o objeto selecionado para a área de transferência.
	Cola o objeto que está na área de transferência no local indicado na tela.
	Chama o Organizer.
	Mostra as propriedades do objeto selecionado.
	Mostra as propriedades da tela selecionada.
	Executa a aplicação corrente iniciando pelas telas que estão abertas.
	Executa a aplicação corrente.
	Abre a ajuda do sistema.
	Ativa a ajuda sensível ao contexto.

2.2.2. Barra de Ferramentas Objetos

Figura 8: Barra de Ferramentas Objetos

Opções da Barra de Ferramentas Objetos

BOTÃO	DESCRIÇÃO
	Liga o modo de seleção, permitindo que o usuário selecione todos os objetos dentro de uma área delimitada pelo mouse.
	Cria um novo objeto Slider.
	Cria um novo objeto Gráfico de Tendência.
	Cria um novo objeto Botão.
	Cria um novo objeto Gauge (medidor).
	Cria uma nova área de texto (objeto Texto).
	Cria um novo objeto Gráfico de Barra.
	Cria um novo objeto Display.
	Cria uma nova Animação.
	Cria um novo objeto Setpoint.
	Cria um novo objeto Alarme.
	Cria um novo objeto Browser.
	Cria um novo objeto Bitmap.
	Cria um novo objeto Vídeo.
	Cria um objeto AVI.
	Cria um novo objeto Preview.

2.2.3. Barra de Ferramentas Arranjar

A **Barra de Ferramentas Arranjar** possui comandos para edição de Telas agindo sobre os Objetos de Tela que estiverem selecionados; os mesmos comandos estão disponíveis no menu Arranjar. Para selecionar mais de um Objeto de Tela, utilize o botão esquerdo do mouse mantendo a tecla [Ctrl] pressionada; o último objeto selecionado ficará com o foco em vermelho para ser usado como referência. Para desselecionar um objeto use a combinação de teclas: [Ctrl]+[Shift]+BotãoEsq.

Figura 9: Barra de Ferramentas Arranjar

Opções da Barra de Ferramentas Arranjar

BOTÃO	DESCRIÇÃO
	Envia o objeto selecionado para o último plano (fundo da tela).
	Traz o objeto selecionado para o primeiro plano (frente da tela).
	Alinha os objetos selecionados pelo lado esquerdo.
	Alinha os objetos selecionados pelo lado direito.
	Alinha os objetos selecionados pelo topo.
	Alinha os objetos selecionados pela base.
	Centraliza horizontalmente os objetos selecionados em relação à tela.
	Centraliza verticalmente os objetos selecionados em relação à tela.
	Faz com que os objetos selecionados tenham a mesma largura.
	Faz com que os objetos selecionados tenham a mesma altura.
	Faz com que os objetos selecionados tenham o mesmo tamanho.
	Agrupa os objetos selecionados.
	Desagrupa os objetos selecionados.

2.2.4. Barra de Ferramentas Telas

A **Barra de Ferramentas Telas** mostra o nome da tela corrente e permite trocar de tela através de uma lista que mostra o nome de todas as telas existentes na aplicação.

Figura 10: Barra de Ferramentas Telas

2.3. Teclas de Atalho

Outra maneira para acessar rapidamente as funções do Elipse SCADA são as teclas de atalho. Listamos abaixo as teclas disponíveis, agrupadas por função.

Opções Gerais

TECLAS	DESCRIÇÃO
Ctrl + O	Abre a aplicação
Ctrl + Shift + V	Informações "Sobre o Elipse SCADA"
F1	Chama a ajuda
Shift + F1	Chama a ajuda de contexto

Editando uma aplicação

TECLAS	DESCRIÇÃO
Ctrl + S	Salva a aplicação
F10	Roda (executa) a aplicação
Alt + O	Chama o Organizer
Ctrl + N	Nova tela.
F8	Monitorar tela
Ctrl + Alt + Shift + I	Conta o número de itens da aplicação
Ctrl + Shift + F10	Chama a janela de configuração da fonte do Editor de Scripts

Editando Relatórios

TECLAS	DESCRIÇÃO
Ctrl + F4	Fecha o editor de relatórios
Esc	Desseleciona o objeto
Ctrl + A	Seleciona todos objetos
Del	Apaga o objeto

Editando Telas

TECLAS	DESCRIÇÃO
Ctrl + F4	Fechar tela
Esc	Desselecionar objeto
Ctrl + A	Selecionar todos objetos
Del	Apagar objeto
Ctrl + X	Recortar objeto
Ctrl + C	Copiar objeto
Ctrl + V	Colar objeto
Shift + Del	Recortar objeto
Ctrl + Ins	Copiar objeto
Shift + Ins	Colar objeto

2.4. Opções de Linha de Comando

É possível chamar o Elipse SCADA diretamente da linha de comando. O executável ELIPSE32.EXE possui a seguinte sintaxe:

```
ELIPSE32.EXE [-DEMO] [-SETUP] [-EDIT] [<NomeApp>]
```

Onde:

-DEMO	(Opcional) Força o Elipse SCADA a rodar em modo de demonstração, sem verificar os mecanismos de proteção (hardkey). Esta opção reescreve o arquivo .INI configurando a seção [ProtectionType].
-SETUP	(Opcional) Força o Elipse SCADA a rodar o programa de Setup, que permite a você configurar as opções no arquivo de preferências (.INI).
-EDIT	(Opcional) Força o Elipse SCADA a rodar no modo Configurador. Se o nome de uma aplicação for informado na linha de comando, esta aplicação será aberta para configuração.
NomeApp	(Opcional) O nome da aplicação que irá rodar automaticamente ou será aberta para configuração (quando o -EDIT é especificado).


Na versão Windows CE, a única linha de comando disponível é NomeApp.

Capítulo 3

3. Organizer

O desenvolvimento de uma aplicação no Elipse SCADA é baseado na ferramenta **Organizer**. Ele permite uma visão simples e organizada de toda a aplicação, ajudando na edição e configuração de todos os objetos envolvidos no sistema através de uma árvore hierárquica.

A estrutura do Organizer pode ser comparada à árvore de diretórios do Gerenciador de Arquivos do Windows. Desta forma, a estrutura da aplicação começa no canto superior esquerdo com a raiz da aplicação. Todos os objetos da aplicação descem a partir da raiz agrupados de acordo com seu tipo: Tags, Telas, Alarmes, Receitas, Históricos, Relatórios, Drivers, Databases, que constituem os principais elementos de sua aplicação. Selecionando-se qualquer um dos ramos da árvore da aplicação, ele irá se expandir, mostrando seu conteúdo; desta forma, você pode facilmente navegar pela aplicação tendo disponíveis todas as opções de configuração desde a criação de Tags até o redimensionamento de objetos em uma tela específica.

A estrutura básica do Organizer é apresentada a seguir:

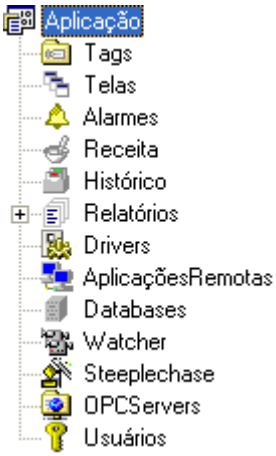


Figura 11: Árvore de classes de objetos no Organizer

Você pode chamar o Organizer somente quando existir uma aplicação aberta selecionando o comando *Organizer* do menu *Arquivo* ou pressionando o botão do *Organizer* na Barra de Ferramentas. A seguinte janela irá aparecer:

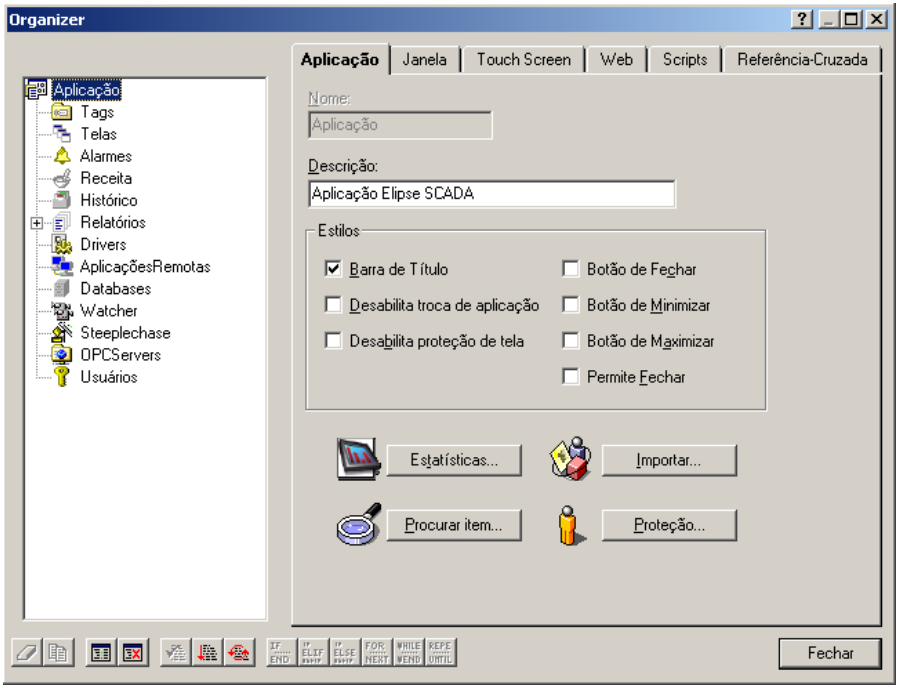


Figura 12: Organizer

A partir do Organizer você pode criar toda a sua aplicação, simplesmente navegando através da árvore da aplicação. Selecionando-se qualquer um de seus ramos, as propriedades do objeto selecionado serão mostradas no lado direito da janela, onde poderão ser editadas. Por exemplo, se você selecionar Tags na árvore do Organizer você poderá criar novos Tags e editar suas propriedades selecionando a página desejada a partir das abas no topo da janela.

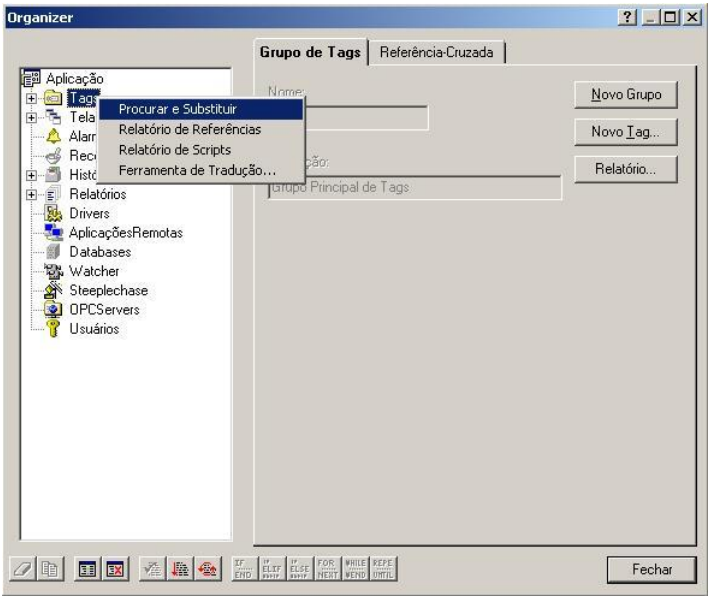


Figura 13: Menu de ações

Ao clicar com o botão direito sobre os objetos do Organizer, as seguintes opções aparecem:

Procurar e Substituir: permite procurar e substituir textos em scripts e propriedades do objeto selecionado e seus filhos. Esta procura é *case sensitive* (diferencia maiúsculas de minúsculas).

Relatório de Referências: produz um relatório com as referências cruzadas do objeto selecionado e seus filhos.

Relatório de Scripts: auxilia o usuário na organização e documentação dos scripts da aplicação.

Ferramenta de Tradução: auxilia na tradução de uma aplicação, mostrando e permitindo editar propriedades com textos.

Os botões na Barra de Ferramentas do Organizer permitem realizar determinadas tarefas rapidamente sem utilizar os menus. Existem 13 botões como pode ser verificado na tabela a seguir:

Ferramentas do Organizer

ICONE	NOME	AÇÃO
	Deletar	Apaga um ou mais itens selecionados no Organizer.
	Duplicar	Duplica o item selecionado na árvore do Organizer.
	AppBrowser	Chama o AppBrowser.
	Referência Cruzada	Chama a Referência Cruzada.
	Compilar	Compila o script que está sendo editado.
	Compilar tudo	Compila todos os scripts que não estão compilados.
	Recompilar tudo	Recompila todos os scripts da aplicação, possibilitando ao usuário acessar cada script com um duplo clique. É gerada uma lista dos scripts compilados, mostrando em vermelho os que estão com erro.
	If	Insere o comando IF...ENDIF no script selecionado, no ponto onde está o cursor.
	Else...If	Insere o comando ELSE...IF no script selecionado, no ponto onde está o cursor.
	Else	Insere o comando ELSE no script selecionado, no ponto onde está o cursor.
	For...Next	Insere o comando FOR...NEXT no script selecionado, no ponto onde está o cursor.
	While...Wend	Insere o comando WHILE...WEND (fim de While) no script selecionado, no ponto onde está o cursor.
	Repeat...Until	Insere o comando REPEAT...UNTIL no script selecionado, no ponto onde está o cursor.

3.1. App Browser

O **AppBrowser** é uma importante ferramenta do Organizer. Ele é composto de uma janela que apresenta a árvore da aplicação com seus objetos. Clicando em qualquer objeto, pode-se visualizar as funções e atributos relacionados a este objeto. Quando estamos escrevendo um script, um botão Copia no Script --> fica disponível nesta janela, permitindo a cópia do atributo ou função em questão para as linhas de programação, facilitando essa tarefa.

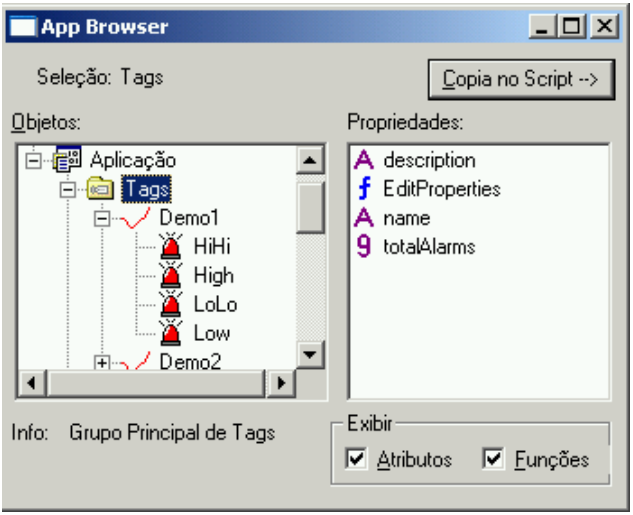


Figura 14: App Browser

3.2. Criando a sua Aplicação

A criação de uma **aplicação** é o ponto de partida para montagem de um sistema utilizando o Elipse SCADA. Em uma aplicação, o usuário reúne todos os objetos necessários para execução das tarefas desejadas. As informações referentes a esta aplicação ficam armazenadas em um arquivo de extensão APP.

Para criar uma nova aplicação, faça isso:

- Escolha no menu Arquivo a opção Nova Aplicação, ou clique no botão correspondente na barra de ferramentas.
- No quadro Salvar Aplicação Nova! escolha um nome e o lugar onde a aplicação será salva.

Além dos arquivos de extensão APP, existem outros gerados e utilizados pelo Elipse SCADA:

Extensões disponíveis

EXTENSÃO	DESCRIÇÃO
.APX	Arquivo com configurações da lista de usuários.
.BAK	Backup da aplicação
.DAT	Arquivo de históricos
.HDR	Cabeçalhos de arquivos de históricos por batelada
.RCP	Arquivo de receitas
.DLL, .SO	Drivers de comunicação
.BMP, .GIF, .JPG	Arquivos de imagens. CE No CE, apenas .BMP está disponível.

NOTA: Usuários de Windows XP deverão ter atenção quanto à ferramenta de restauração do sistema. Esta ferramenta também monitora os arquivos .APP, ou seja, se for feita uma restauração, versões mais antigas dos arquivos serão recuperadas, sobreescrevendo os arquivos mais recentes.

3.2.1. Propriedades Gerais da Aplicação

Quando você seleciona a raiz Aplicação, na árvore do Organizer, suas propriedades são mostradas ao lado direito da árvore. A página de propriedades gerais da Aplicação aparece quando selecionada a aba Aplicação no topo das páginas da Aplicação. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

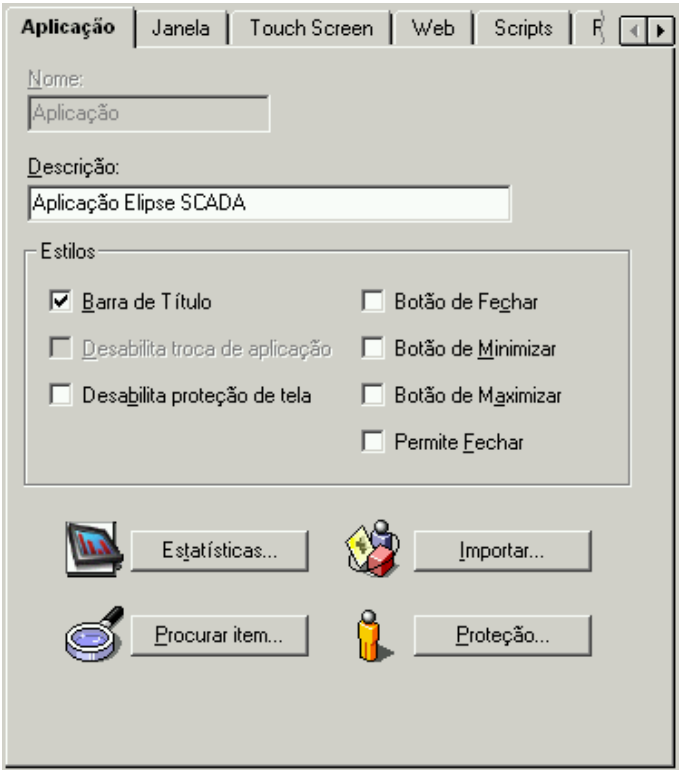


Figura 15: Propriedades da Aplicação

Propriedades da Aba Aplicação

OPÇÃO	DESCRIÇÃO
Descrição	Define o nome da aplicação (que aparecerá na barra de título) caso a opção Barra de Título esteja habilitada.
Barra de título	Habilita a barra de título na janela da aplicação. No CE, a barra de título obedece essa propriedade apenas para telas janeladas. As telas cheia obedecem o padrão do próprio CE, algumas plataformas nunca aparecem, outras sempre aparecem.

Propriedades da Aba Aplicação (Quadro Estilo)

OPÇÃO	DESCRIÇÃO
Desabilita troca de aplicação	Desabilita a troca entre programas, ou seja, desabilita o atalho [Alt+Tab] do Windows.
Desabilita proteção de tela	Desabilita qualquer protetor de tela (*screen saver*) enquanto o Elipse SCADA estiver sendo executado.
Botão de Fechar	Habilita o botão de Fechar na barra de título da janela da aplicação.
Botão de Minimizar	Habilita o botão de Minimizar na janela da aplicação.
Botão de Maximizar	Habilita o botão de Maximizar na janela da aplicação.
Permite Fechar	Desligado, faz com que a execução termine apenas quando for chamada a função StopRunning(). Ligado, permite que a aplicação (e o Elipse SCADA) seja terminado via outros meios, como um clique no botão Fechar, desligar do Windows, etc.

Propriedades da Aba Aplicação (Botões)

OPÇÃO	DESCRIÇÃO
Estatísticas...	Abre uma janela que mostra informações estatísticas da aplicação, como: tempo total de edição da aplicação, número de itens na aplicação, número total de tags, número de revisões e versão do Elipse SCADA em que foi gerada a aplicação.
Procura Item...	Abre uma janela que permite buscar um item (objeto, propriedade) em qualquer lugar da aplicação e apresentá-lo para edição.
Importar...	Abre uma janela que permite escolher uma aplicação para a importação. Após a escolha da aplicação origem, uma nova janela é aberta com a árvore das duas aplicações, de modo que o usuário pode arrastar os objetos da aplicação origem para a aplicação destino. OBS: a aplicação-origem não é modificada.
Proteção...	Abre uma janela para proteção da aplicação. Existem duas proteções: para **configuração** (para editar e fazer qualquer

	tipo de modificação) e para **execução**. No caso da utilização de senha para a configuração, o usuário final não poderá alterar a aplicação, a não ser que conheça a senha utilizada. O mesmo vale para a execução, sendo que só pode haver esta senha, se houver uma para a configuração.

3.2.2. Janela de Aplicação

A página Janela permite a configuração da janela principal para execução da aplicação.

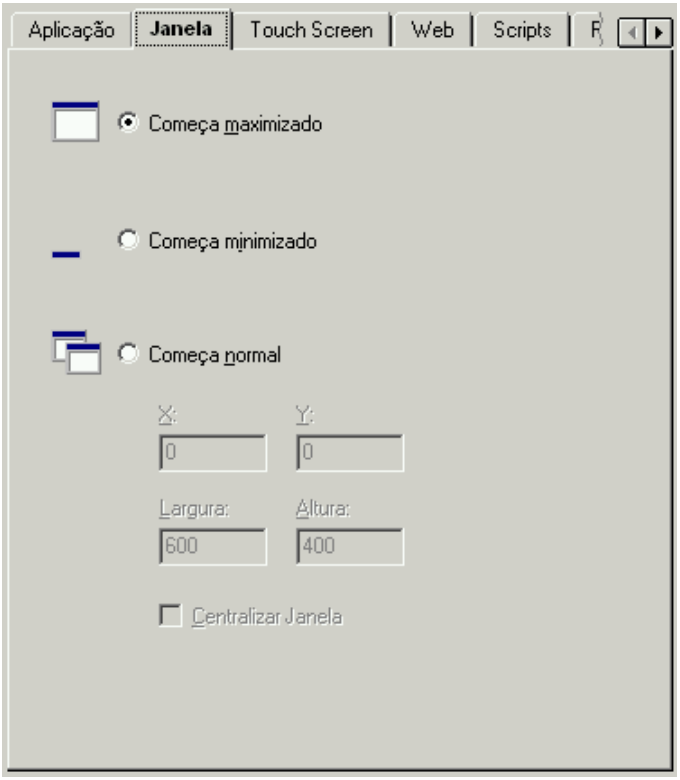


Figura 16: Janela da Aplicação

Propriedades da Aba Aplicação

OPÇÃO	DESCRIÇÃO
Começa Maximizado /Minimizado /Normal	Define a configuração inicial da janela da aplicação.
X, Y, Largura, Altura	Define a posição e tamanho da janela em pixels.
Centralizar Janela	Indica que a janela deverá iniciar em posição central na tela.

3.2.3. Touch Screen

O Elipse SCADA possui suporte especial para a utilização de telas de toque (Touch Screen), permitindo uma interface mais intuitiva para o uso de sua aplicação.

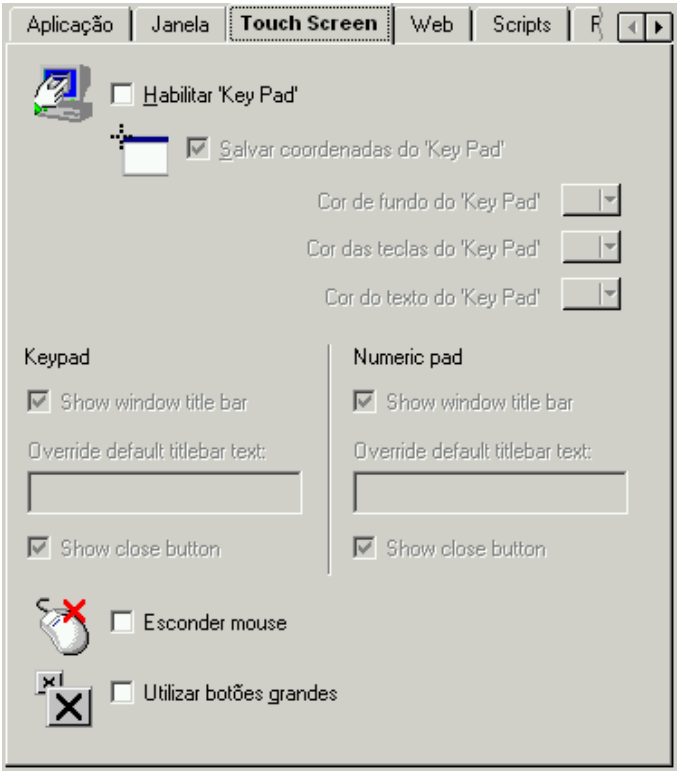


Figura 17: Touch Screen

Propriedades da Aba Touch Screen

OPÇÃO	DESCRIÇÃO
Habilitar "Key Pad"	Habilita o uso do Key Pad em tempo de execução. Atualmente o SCADA CE usa o KeyPad do próprio Win CE então não é possivel escolher cor e dimensões para ele.
Salvar coordenadas do "Key Pad"	Habilita o salvamento das novas coordenadas do Key Pad à medida que sua janela é mudada de posição.
Cor de fundo do "Key Pad"	Define a cor de fundo do Key Pad.
Cor das teclas do "Key Pad"	Define a cor das teclas do Key Pad.
Cor do texto do "Key Pad"	Define a cor do texto do Key Pad.
Esconder mouse	Desabilita o ponteiro do mouse enquanto o Key Pad está sendo mostrado.
Usar botões grandes	Aumenta o tamanho dos botões do Key Pad.

Propriedades do Key Pad e do Numeric Pad

OPÇÃO	DESCRIÇÃO
Show window title bar	Mostra a barra de título da janela do Key Pad e/ou Numeric Pad.
Override default titlebar text	Permite definir um novo título para a barra de título da janela do Key Pad e/ou Numeric Pad.
Show close button	Mostra o botão de fechar na janela do Key Pad e/ou Numeric Pad.

3.2.4. Elipse Web

Através do plug-in Elipse Web, o Elipse SCADA pode gerar documentos para serem visualizados pela Internet, em conjunto com algum Servidor Web comercial, como o **Internet Information Services (IIS)** ou o **Microsoft Personal Web Server (PWS)**.

O Elipse Web pode ser habilitado através do Organizer selecionando o item Aplicação, conforme figura abaixo:

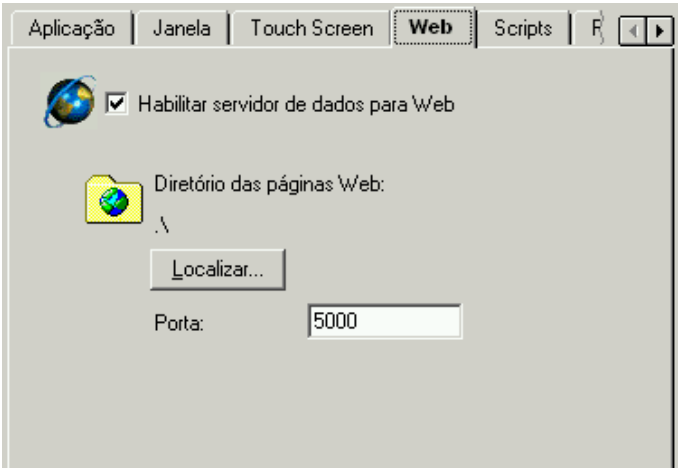


Figura 18: Elipse Web

Propriedades do Elipse Web

OPÇÃO	DESCRIÇÃO
Habilitar servidor de dados para Web	Habilita o Elipse Web.
Diretório das páginas Web	Permite escolher através do botão [Localizar...] o diretório onde serão gerados os documentos a serem visualizados pela Internet. O diretório deverá estar na árvore de documentos do servidor Web.
Porta	Permite definir a porta a qual o servidor Web está associado.

Nota: Para maiores informações, veja o capítulo Elipse Web.

3.3. Scripts

Durante a execução de uma aplicação, diversos procedimentos podem ser disparados através de eventos. Estes procedimentos são descritos por scripts associados a estes eventos. Maiores detalhes sobre scripts serão discutidos em capítulo posterior. Aqui, temos uma lista de eventos/scripts disponíveis em uma aplicação.

Eventos/Scripts de uma Aplicação

EVENTO	DESCRIÇÃO
OnKeyPress	Executa quando uma determinada tecla for pressionada.
OnKeyRelease	Executa quando a tecla é solta.
OnStartRunning	Executa quando a aplicação iniciar a execução.
OnStopRunning	Executa quando a aplicação terminar a execução.
OnUserLogin	Executa quando um usuário inicia a sua seção na aplicação.
OnUserLogout	Executa quando um usuário encerra a sua seção na aplicação.
WhileKeyPressed	Executa enquanto uma determinada tecla estiver sendo pressionada.
WhileRunning	Executa enquanto a aplicação estiver executando. O script irá executar tantas vezes quantas você definir na caixa de texto rodar a cada (aparece somente para scripts whilerunning).

Capítulo 4

4. Tags

A supervisão de um processo com o Elipse SCADA ocorre através da leitura de variáveis de processos no campo. Os valores dessas variáveis são associados a objetos do sistema chamados **Tags**.

Para cada objeto inserido na tela, devemos associar pelo menos um **tag** ou **atributo**. Os **tags** são todas as variáveis (numéricas ou alfanuméricas) envolvidas numa aplicação. Os **atributos** são dados fornecidos pelo Elipse SCADA sobre parâmetros de sistema e componentes da aplicação.

Ao criar tags, o usuário poderá organizá-los livremente em **grupos**, de forma a facilitar a procura e identificação durante o processo de configuração. Para a criação de um grupos, basta selecionar o item Tags no Organizer e clicar em Novo Grupo.

Você pode criar grupos dentro de outros grupos, sem restrições. Para modificar a hierarquia dos grupos e mudá-los de posição (por exemplo, incluir um grupo em outro grupo) basta arrastar o grupo em questão para o lugar desejado.

Você pode criar e editar tags a partir do Organizer, selecionando o ramo Tags na árvore da aplicação e pressionando o botão Novo Tag. Na janela do Organizer você pode dar um duplo clique na opção Tags para ver os tags já definidos para a aplicação, da mesma forma que você faz em uma árvore de diretórios. A medida que a aplicação cresce os tags podem ser agrupados para melhor organizar e editar a aplicação.

4.1. Grupo de Tags

Quando você seleciona a opção Tags na árvore da aplicação no Organizer a seguinte página irá aparecer ao lado direito da árvore. Usando os botões desta página você pode criar um novo grupo de tags ou um novo tag para a sua aplicação. O novo grupo ou tag que for criado irá aparecer automaticamente na árvore da aplicação abaixo da opção Tags.

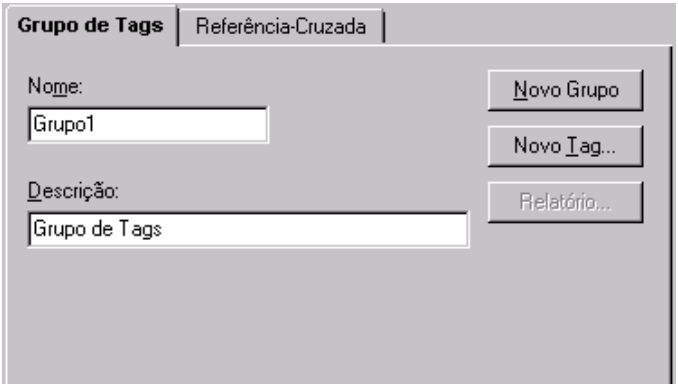


Figura 19: Grupo de Tags

Propriedades da Aba Grupo de Tags

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do grupo de tags selecionado. Um grupo de tags trabalha da mesma forma que um diretório onde vários tags serão definidos.
Descrição	Uma breve descrição sobre o grupo selecionado.
Novo Grupo	Cria um novo grupo de tags a partir do grupo atual.
Novo Tag	Cria um novo tag.
Relatório...	Abre uma janela para configuração de um relatório de todos os tags existentes na aplicação. Podem ser selecionadas apenas as propriedades que se deseja imprimir para cada tipo de tag. O relatório será impresso em um arquivo-texto especificado na caixa Imprime para o arquivo.

4.2. Novo Tag

Quando você pressiona o botão Novo Tag, a janela a seguir irá aparecer. Nesta janela você poderá definir o nome do tag, a quantidade de tags que você deseja criar e o seu tipo. Todos os tags especificados no campo Quantidade serão do mesmo tipo definido no campo Tipo do Tag.

Opções de Tags

OPÇÃO	DESCRIÇÃO
Nome	Nome do Tag. Espaços e caracteres especiais não são permitidos quando os Tags são usados em Scripts.
Quantidade	Define o número de Tags que serão criados com as mesmas características especificadas.
PLC	Tags PLC (CLP) são usados para trocar valores com os equipamentos de aquisição de dados.
DDE	Permite trocar dados com programas que sejam Servidores DDE. DDE (Data Dynamic Exchange) é um protocolo desenvolvido pela Microsoft para comunicação entre aplicações baseadas em Windows.
Demo	Tags Demo são usados para gerar dados randômicos.
Matriz	O Tags Matriz permitem criar matrizes ou vetores de dados.
Expressão	Tags Expressão permitem a entrada de uma expressão numérica ou alfanumérica (permitem concatenação de strings, por exemplo).
Block	Tags Bloco são usados para ler um bloco de valores simultaneamente.
RAM	Tags RAM são usados para armazenar valores na memória.

4.3. Tag Crono

O **Tag Crono** cria um novo cronômetro.

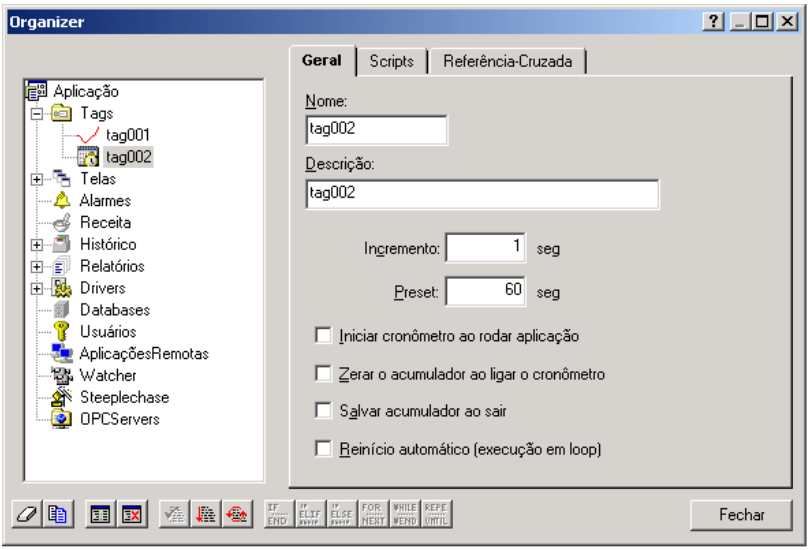


Figura 20: Propriedades do Tag Crono

Propriedades do Tag Crono

OPÇÃO	DESCRIÇÃO
Nome	Nome do Tag. Você pode usar tantos caracteres quanto queira mas espaços e caracteres especiais não são permitidos quando os Tags forem usados nos Scripts.
Descrição	Uma breve descrição sobre o Tag.
Incremento	Determina o incremento do tag.
Preset	Determina o tempo de preset do tag.
Iniciar cronômetro ao rodar aplicação	Habilita a inicialização do cronômetro ao rodar a aplicação.
Zerar o acumulador ao ligar o cronômetro	Habilita zerar o acumulador ao ligar o cronômetro.
Salvar acumulador ao sair	Habilita salvar acumulador ao sair.
Reinício automático (execução em loop)	Habilita o reinício automático em execução de loop.

4.4. Tag PLC

O Tag PLC é usado para trocar informações com os equipamentos de aquisição de dados usando drivers de I/O fornecidos pela Elipse de acordo com o tipo do equipamento. Um arquivo de ajuda é fornecido com cada driver contendo informações importantes a respeito de sua configuração.

Você pode instalar um novo driver pressionando o botão Novo na página de Drivers e selecionando um ou mais drivers da lista. O botão Configurar... abre a janela de propriedades do driver permitindo a configuração dos parâmetros de comunicação [P] de acordo com as informações contidas no arquivo de ajuda. A opção Abortar em erro, encerra a comunicação caso ocorra algum problema, impedindo que uma aplicação fique travada.

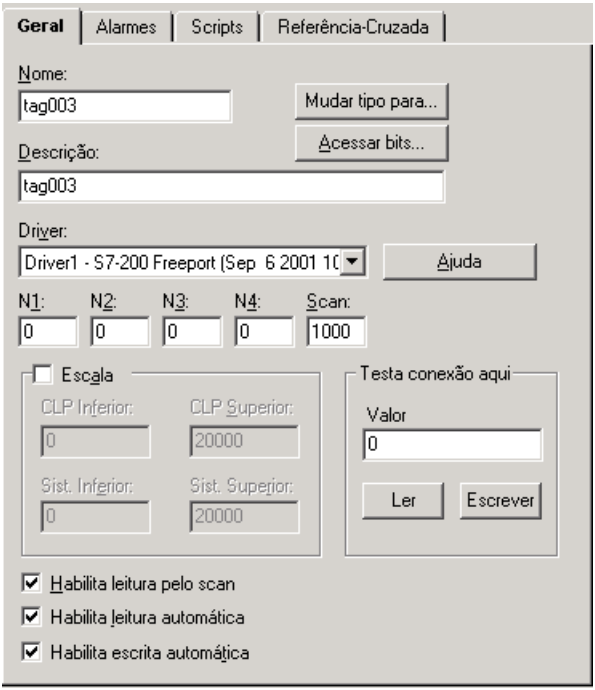


Figura 21: Propriedades do Tag PLC

Propriedades do Tag PLC

OPÇÃO	DESCRIÇÃO
Nome	Nome do Tag. Você pode usar tantos caracteres quanto queira mas espaços e caracteres especiais não são permitidos quando os Tags forem usados nos Scripts.
Mudar tipo para ...	Permite que se mude o tipo do Tag.
Acessar bits...	Permite desmembrar o Tag em bits, criando Tags Bit para cada bit ou conjunto de bits do Tag.
Descrição	Uma breve descrição sobre o Tag.
Driver	Permite a seleção de um driver de comunicação para o Tag corrente. Os drivers devem ser instalados através da janela de Drivers a fim de que estejam disponíveis.
Ajuda	Mostra a Ajuda do Driver selecionado.
Parâmetros "N"	Permite a configuração dos parâmetros de endereçamento "N" para o Tag corrente de acordo com o driver selecionado. Esta configuração está contida na Ajuda do Driver que pode ser acessado pressionando-se o botão "Ajuda". Os valores podem ser expressos em decimais [-32768, 65535], octais [0o, 177777o] ou hexadecimais [0000h, FFFFh].
Scan	Define o intervalo de tempo no qual o valor do tag será lido (em ms)
Escala	Marcando esta opção os valores do Tag serão convertidos para uma nova escala de valores determinada pelo usuário conforme os limites definidos em CLP Inferior e Superior, e Sistema Superior e Inferior.
CLP Inferior	Define o valor mínimo a ser lido do PLC (CLP).
Sistema Inferior	Define o novo valor mínimo para a conversão dos valores lidos.
CLP Superior	Define o valor máximo a ser lido do PLC (CLP).
Sistema Superior	Define o novo valor máximo para a conversão dos valores lidos.
Testar conexão aqui	Testa a comunicação com o PLC, permitindo a leitura e escrita de valores.
Habilita leitura pelo scan	Habilita a leitura pelo scan, ou seja, o valor do tag sempre será atualizado no tempo definido no campo Scan, independente de outras configurações.
Habilita leitura automática	Quando habilitado, o valor do tag só é lido quando necessário. Se a opção Habilita leitura pelo scan estiver habilitada, esta opção é ignorada.
Habilita escrita automática	Habilita escrita automática para o tag PLC (ver tópico abaixo).

Leitura pelo Scan X Leitura Automática

Quando a opção Habilita leitura pelo scan está ligada, o Elipse SCADA atualiza o valor do tag continuamente, na freqüência especificada no campo Scan. Este valor de Scan também é utilizado pela opção Habilita leitura automática para verificar se o valor do tag PLC é antigo, definindo a necessidade ou não de fazer uma releitura. Logo, se a opção Habilita leitura pelo scan estiver ligada, a opção Habilita leitura automática é irrelevante, pois o valor do tag estará sempre atualizado.

A opção Habilita leitura automática pode ser utilizada isoladamente para otimização em casos específicos. Por exemplo, se um tag estiver apenas sendo requisitado por um script, pode-se ligar a opção Habilita leitura automática e desligar a opção Habilita leitura pelo scan. Assim o tag só será lido quando for necessário.

Escrita Automática em Tags PLC

Ao atribuir um valor diretamente a um tag PLC ou elemento de bloco que possua a propriedade **escrita automática** habilitada, o comando é enviado diretamente ao driver de comunicação, que por sua vez o repassa ao equipamento associado. Tal ação não ocorre somente quando o valor atribuído for igual ao conteúdo que já estava no tag. Caso queira forçar uma escrita mesmo assim, deve ser executada a função Write() do tag, em algum script (ver capítulo Scripts).

4.5. Tag DDE

O **Tag DDE** é usado para troca de dados entre o Elipse SCADA e outras aplicações (Excel, Access, Word, etc.) usando DDE (*Dynamic Data Exchange*). Para tanto você precisa especificar a aplicação servidora, o tópico e o item, conforme você pode ver na janela abaixo.

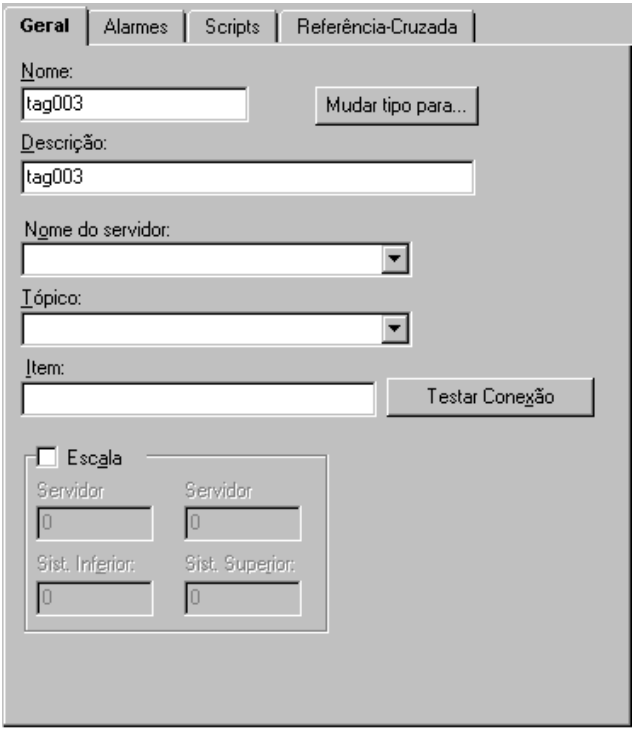


Figura 22: Propriedades do Tag DDE

Propriedades do Tag DDE

OPÇÃO	DESCRIÇÃO
Nome	Nome do Tag. Você pode usar tantos caracteres quanto queira mas espaços e caracteres especiais não são permitidos quando os Tags forem usados nos Scripts.
Mudar tipo para	Permite que se mude o tipo do Tag.
Descrição	Uma breve descrição sobre o Tag.
Nome do Servidor	Define o nome do servidor DDE que pode ser uma aplicação Windows (ex: Excel, Word, etc) ou um driver DDE fornecido pelo fabricante do seu equipamento. A lista de programas disponíveis para servidores DDE aparece quando a seta ao lado desta caixa é pressionada.
Tópico	Define o nome do Tópico do Servidor DDE, dependendo do tipo do Servidor. Pode ser um documento (ex: uma tabela do Excel). A lista de Tópicos disponíveis aparece quando a seta ao lado desta caixa é pressionada.
Item	Define o nome do Item do Servidor DDE, dependendo do tipo do Servidor. Pode ser um item de um documento (ex: uma célula em uma tabela do Excel).
Testar Conexão	Permite que você teste a configuração DDE. Uma mensagem pode indicar um erro de conexão ou o valor recebido pelo item configurado.
Escala	Marcando esta opção os valores do Tag serão convertidos para uma nova escala de valores determinada pelo usuário conforme os limites definidos em Server Low, System Low, Server High e System High.
Servidor Inferior	Define o valor mínimo a ser lido do Servidor.
Sistema Inferior	Define o novo valor mínimo para a conversão dos valores lidos.
Servidor Superior	Define o valor máximo a ser lido do Servidor.
Sistema Superior	Define o novo valor máximo para a conversão dos valores lidos.

4.6. Tag Demo

O **Tag Demo** é usado para a simulação de valores. Ele permite a você gerar curvas definidas ou valores randômicos conforme o tipo de curva selecionada nos seis botões da página Geral do tag Demo (veja figura abaixo).

Tags Demo podem ajudá-lo a testar sua aplicação ou podem ser usados, por exemplo, em um objeto de tela animação para mostrar os quadros da animação de acordo com a variação do tag.

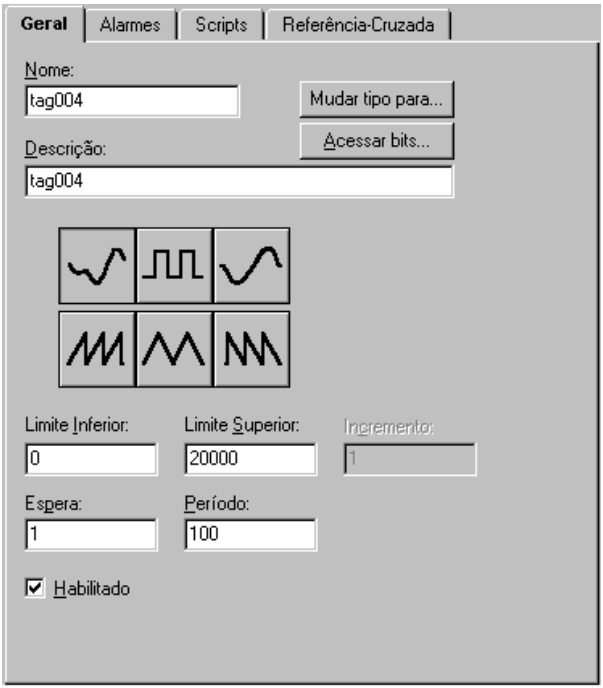


Figura 23: Propriedades do Tag Demo

Propriedades do Tag Demo

OPÇÃO	DESCRIÇÃO
Nome	Nome do Tag. Espaços e caracteres especiais não são permitidos quando os Tags forem usados em Scripts.
Mudar tipo para	Permite que se mude o tipo do Tag.
Acessar bits...	Permite desmembrar o Tag em bits, criando Tags Bit para cada bit.
Descrição	Uma breve descrição sobre o Tag.
Tipo	Define o tipo de curva a ser usada pelo Tag Demo corrente.
Limite inferior	Define um valor mínimo para o Tag Demo.
Limite superior	Define um valor máximo para o Tag Demo.
Incremento	Define o incremento para o Tag Demo para uma curva dente de serra.
Espera	Define o número de períodos entre cada geração de valor para o Tag Demo. Por exemplo, se for 1 um valor é gerado a cada período, se for 2, gera um valor a cada dois períodos, e assim por diante. É usado junto com o atributo *period* para controlar o intervalo de tempo para a variação dos dados.
Período	Define o número de milisegundos entre a geração de cada novo valor para o Tag Demo. É usado junto com o atributo *delay* para controlar o intervalo de tempo para a variação dos dados.
Habilitado	Define a condição inicial do Tag Demo: Habilitada ou Desabilitada. Os valores do Tag Demo são gerados somente quando esta opção estiver marcada, caso contrário, o valor do Tag permanece o mesmo.

4.7. Tag Expressão

O **Tag Expressão** permite que você atribua uma expressão numérica ou alfanumérica a um tag. Você pode criar equações envolvendo outros tags e strings. As mesmas funções, operadores e constantes usadas nos Scripts podem ser usadas nos Tags Expressão.

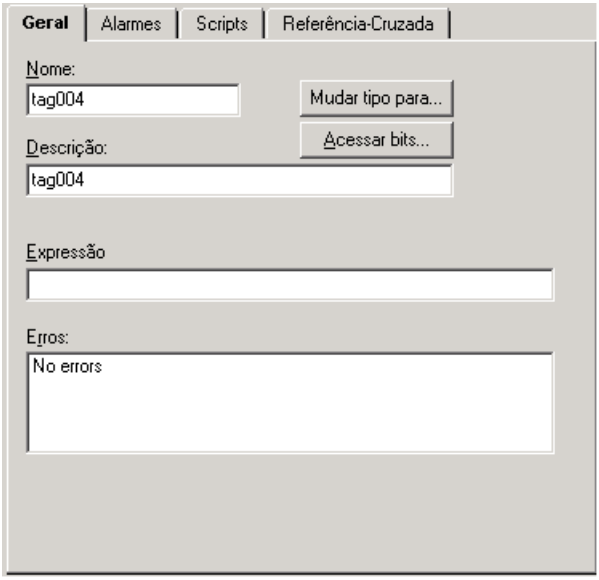


Figura 24: Propriedades do Tag Expressão

Propriedades do Tag Expressão

OPÇÃO	DESCRIÇÃO
Nome	Nome do tag. Espaços e caracteres especiais não são permitidos quando os tags forem usados em scripts.
Mudar tipo para	Permite que se mude o tipo do tag.
Acessar bits...	Permite desmembrar o tag em bits, criando tags Bit para cada bit.
Descrição	Uma breve descrição sobre o tag.
Expressão	Permite a entrada de qualquer expressão válida para o tag.
Erros	Lista erros de sintaxe encontrados na expressão. Os erros são mostrados durante a edição da expressão e para que se tenha uma expressão válida a mensagem "No errors" (Sem erros) deve aparecer neste campo.

4.8. Tag Bloco

Tags Blocos permitem a comunicação em bloco com os equipamentos de aquisição de dados usando drivers de I/O fornecidos pela Elipse Software de acordo com o tipo do equipamento. Um arquivo de ajuda é fornecido com cada driver contendo informações importantes a respeito de sua configuração.

Você pode instalar um novo driver pressionando o botão Novo na página de Drivers e selecionando um ou mais drivers da lista. O botão Configurar... abre a janela de propriedades do driver permitindo a configuração dos parâmetros de comunicação "P" de acordo com as informações contidas no arquivo de ajuda. A opção Abortar em erro encerra a comunicação caso ocorra algum problema, impedindo que uma aplicação fique "travada".

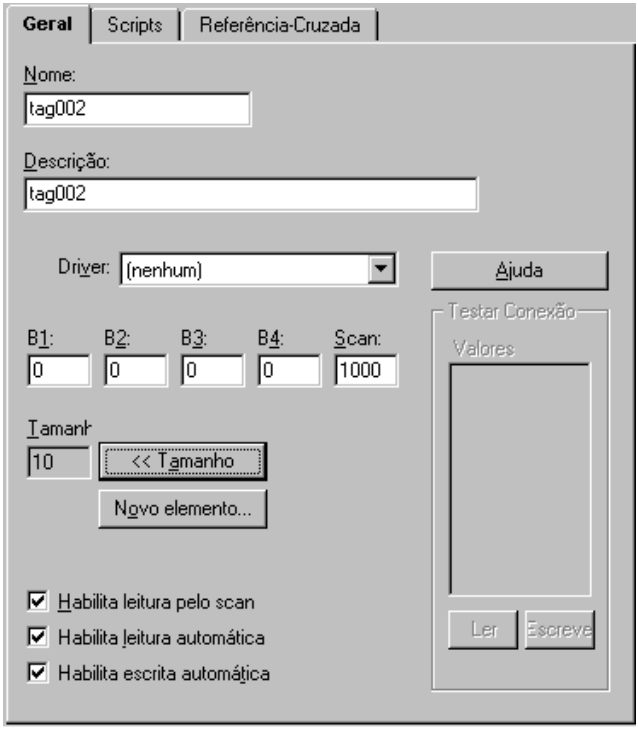


Figura 25: Propriedades do Tag Bloco

Propriedades do Tag Bloco

OPÇÃO	DESCRIÇÃO
Nome	Nome do Tag. Você pode usar tantos caracteres quanto queira mas espaços e caracteres especiais não são permitidos quando os Tags forem usados nos Scripts.
Mudar tipo para	Permite que se mude o tipo do Tag.
Descrição	Uma breve descrição sobre o Tag.
Driver	Permite a seleção de um driver de comunicação para o Tag corrente. Os drivers devem ser instalados através da janela de Drivers a fim de que estejam disponíveis.
Ajuda	Mostra a Ajuda do Driver selecionado.
Parâmetros "B"	Permite a configuração dos parâmetros de endereçamento "B" para o Tag corrente de acordo com o driver selecionado. Esta configuração está contida na Ajuda do Driver que pode ser acessado pressionando-se o botão "Ajuda". Os valores podem ser expressos em decimais [-32768, 65535], octais [0o, 177777o] ou hexadecimais [0000h, FFFFh].
Scan	Define de quanto em quanto tempo os valores do Tag serão atualizados (ms).
Mudar tamanho	Muda o tamanho do bloco a ser monitorado pelo Elipse SCADA.
Adicionar Elemento	Permite que você adicione um novo elemento ao Tag selecionado.
Habilitar leitura pelo scan	Habilita leitura em bloco.
Habilitar leitura automática	Habilita leitura automática para o bloco.
Habilitar escrita automática	Habilita escrita automática para o bloco.
Tamanho	Configura o tamanho do tag bloco.
Novo elemento...	Novo elemento no tag bloco.

4.8.1. Elemento de Bloco

Cada elemento do tag Bloco possui suas propriedades que podem ser acessadas selecionando-se o elemento desejado na árvore da aplicação no Organizer. A página a seguir será, então, mostrada no lado direito da janela.

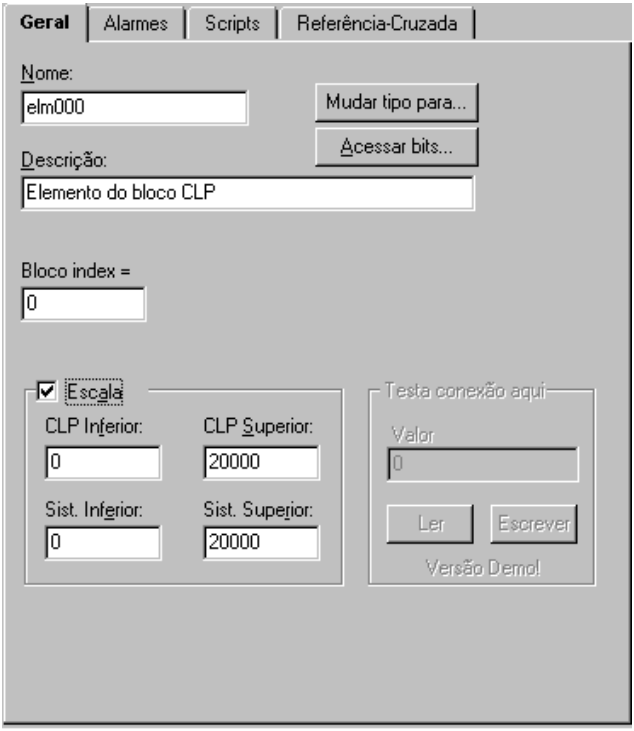


Figura 26: Propriedades do Elemento de Bloco

Propriedades do Elemento de Bloco

OPÇÃO	DESCRIÇÃO
Nome	Nome do Tag. Você pode usar tantos caracteres quanto queira mas espaços e caracteres especiais não são permitidos quando os Tags forem usados nos Scripts.
Descrição	Uma breve descrição sobre o Tag.
Mudar tipo para...	Permite que se mude o tipo do Tag.
Acessar bits...	Permite desmembrar o Tag em bits, criando Tags Bit para cada bit ou conjunto de bits do Tag.
Bloco index	Permite mudar a ordem do elemento no bloco digitando o índice desejado.
Escala	Marcando esta opção os valores do Tag serão convertidos para uma nova escala de valores determinada pelo usuário conforme os limites definidos em CLP Inferior, Sistema Inferior, CLP Superior e Sistema Superior.
CLP Inferior	Define o valor mínimo a ser lido do PLC (CLP).
Sistema Inferior	Define o novo valor mínimo para a conversão dos valores lidos.
CLP Superior	Define o valor máximo a ser lido do PLC (CLP).
Sistema Superior	Define o novo valor máximo para a conversão dos valores lidos.
Testar comunicação aqui	Testa a comunicação com o PLC lendo e escrevendo alguns valores.

4.9. Tag RAM

Tags RAM são usados internamente para armazenar valores em memória. Este tipo de tag é volátil, portanto mantém seus valores somente enquanto a aplicação está executando.

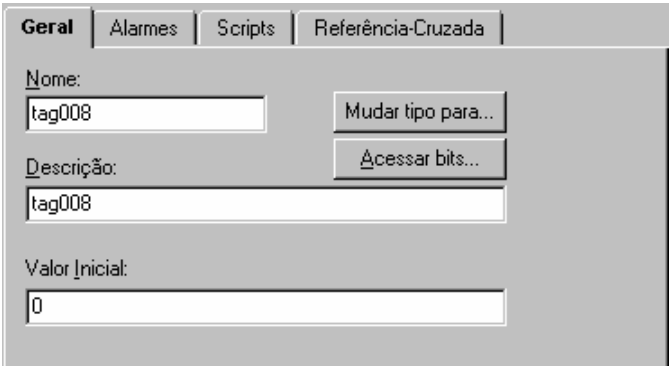


Figura 27: Propriedades do tag RAM

Propriedades do Tag RAM

OPÇÃO	DESCRIÇÃO
Nome	Nome do tag. Você pode usar tantos caracteres quanto queira mas espaços e caracteres especiais não são permitidos quando os Tags forem usados nos Scripts.
Descrição	Uma breve descrição sobre o tag.
Mudar tipo para...	Permite que se mude o tipo do tag.
Acessar bits...	Permite desmembrar o tag em bits, criando tags Bit para cada bit ou conjunto de bits do tag.
Valor inicial	Define um valor inicial para o tag. Este valor pode ser um número ou um string entre aspas duplas (ex.: "abc").

4.10. Tag Matriz

O **Tag Matriz** permite criar matrizes ou vetores de dados que podem ser usados em cálculos, armazenamentos etc. É possível associar cada célula de uma matriz a um tag ou propriedade. As operações sobre matrizes sempre tem linha e coluna começando com o índice 1.

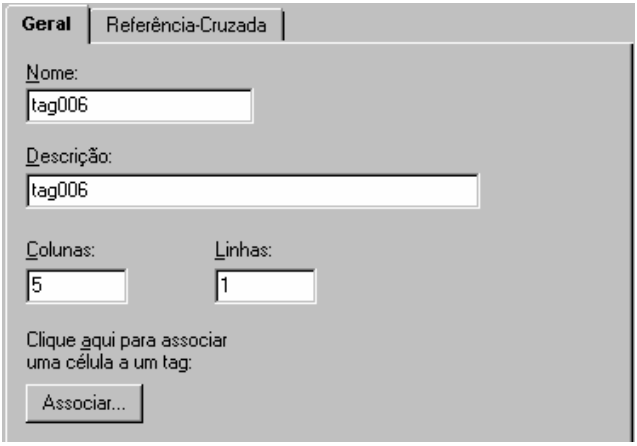


Figura 28: Propriedades do Tag Matriz

Propriedades do Tag Matriz

OPÇÃO	DESCRIÇÃO
Nome	Nome do tag. Você pode usar tantos caracteres quanto queira mas espaços e caracteres especiais não são permitidos quando os Tags forem usados nos Scripts.
Descrição	Uma breve descrição sobre o tag.
Colunas	Define o número de colunas da matriz
Linhas	Define o número de linhas da matriz.
Associar	Mapeia todas ou somente algumas células da matriz para um tag.

Associar Células a Tags

Você pode associar uma célula da matriz ou vetor para um tag pressionando o botão Associar... na página Geral do tag Matriz e especificando qual ou quais as células que deseja associar.

Cada célula associada aparece abaixo do tag Matriz na árvore da aplicação no Organizer. Ao selecionar uma célula específica suas propriedades são mostradas ao lado direito da árvore. Cada célula associada possui 4 páginas de propriedades: Geral, Alarmes, Scripts e Tags. As 3 primeiras páginas são as mesmas de qualquer tag e a página de tags permite associar um tag ou propriedade à célula da matriz da mesma forma em que tags e propriedades são associados à objetos de tela.

4.11. Tag Bit

O **Tag Bit** somente pode ser criado a partir de um outro tag e permite acessar individualmente cada bit do mesmo. Os tags que permitem o desdobramento em bits são: PLC, Demo, Expressão, Elemento de Bloco, Ram ou Remoto.

Você pode criar um tag Bit a partir da página Geral de qualquer um dos tags citados acima. Clicando no botão Acessar bits... a seguinte janela irá aparecer, onde você poderá selecionar os bits que deseja mapear. A seleção dos bits é feita usando-se o mouse e as teclas [Shift] ou [Ctrl], da mesma forma em que se selecionam itens em uma list box do Windows.

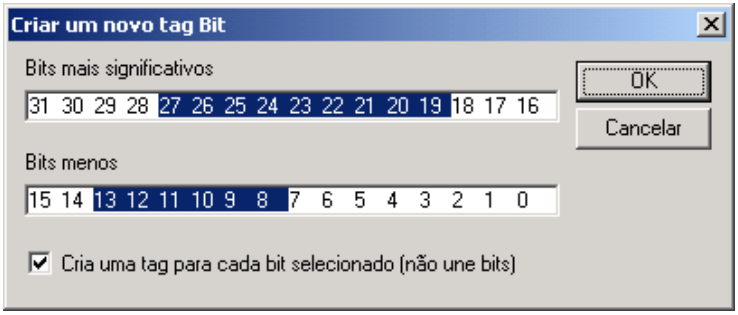


Figura 29: Tag Bit

O tag Bit pode ser tanto um único bit quanto um conjunto de bits, desde que sejam contínuos. Isto quer dizer que você pode mapear para um único tag Bit, por exemplo, os bits 0, 1, e 2 mas não os bits 10, 11 e 24. A *check box* existente nesta janela permite especificar se devem ser criados um tag para cada bit selecionado ou se os bits contínuos que estejam selecionados devem ser agrupados em um único tag.

No exemplo acima serão criados cinco tags Bit da seguinte forma:

TAGS	bitField	bitField2	bitField3	bitField4	bitField5
BITS	0, 1, 2	5	8	12	20, 21, 22

Os tags Bit criados aparecem abaixo do respectivo tag na árvore da aplicação no Organizer. Ao selecionar um tag Bit específico suas propriedades são mostradas ao lado direito da árvore.

A página de Propriedades Gerais do tag Bit aparece quando selecionada a aba Geral no topo das páginas do tag Bit. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

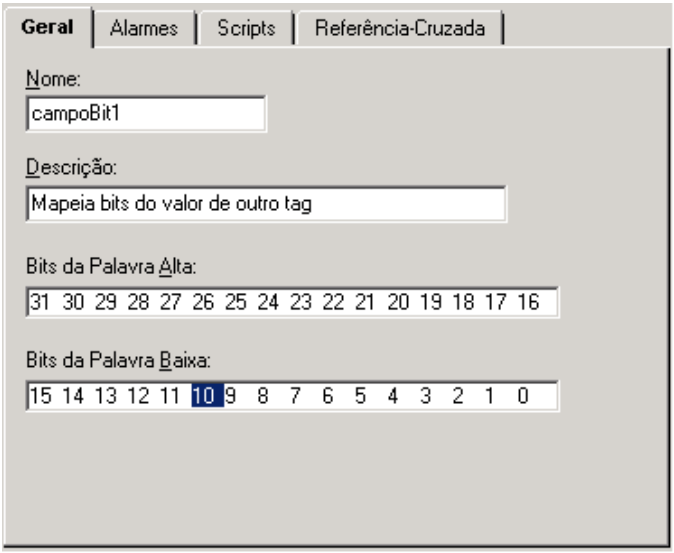


Figura 30: Propriedades gerais do Tag Bit

Propriedades do Tag Bit

OPÇÃO	DESCRIÇÃO
Nome	Nome do tag. Espaços e caracteres especiais não são permitidos quando os tags forem usados nos scripts.
Descrição	Uma breve descrição sobre o tag.
Bit de palavra alta e bit de palavra baixa	Define quais os bits ou bit que fazem parte daquele tag.

4.12. Página de Alarmes

Cada tag que você define possui uma página de Alarmes como a mostrada abaixo, onde podem ser configurados 4 intervalos de valores e prioridades para Alarmes. Alarmes são usados para sinalizar algum problema e então tomar as ações apropriadas usando scripts. Para visualizar os Alarmes configurados para um tag você precisa criar um objeto de tela Alarme e atribuir o tag a ele. Este objeto pode mostrar, também, alarmes já ocorridos que estejam registrados em um arquivo de históricos ou alarmes ativos no sistema. Para imprimir os alarmes ocorridos no

sistema você pode definir um relatório através do Organizer e executar a função especial Print() em um script.

A página de Alarmes dos Tags aparece quando selecionada a aba Alarmes no topo das páginas do tag. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

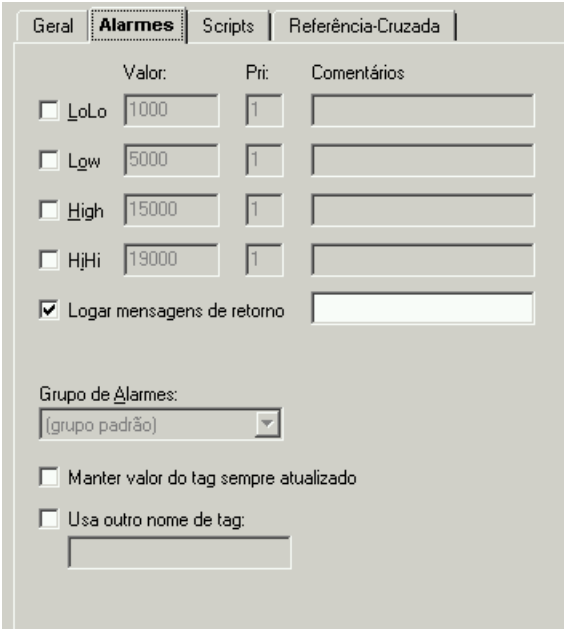


Figura 31: Página de Alarmes

Propriedades dos Alarmes (associados a Tags)

OPÇÃO	DESCRIÇÃO
LoLo	Alarme Baixo Crítico. Define um intervalo de valores (menor igual) onde o Tag é considerado em um estado de Alarme Baixo Crítico. É usado quando o valor do tag está abaixo de um mínimo, ou seja, extremamente baixo.
Low	Alarme Baixo. Define um intervalo de valores (menor igual) onde o tag é considerado em estado de alarme baixo. É usado quando o valor do tag está abaixo do normal.
High	Alarme Alto. Define um intervalo de valores (maior igual) onde o tag é considerado em estado de Alarme Alto. É usado quando o valor do tag está mais alto do que o normal.
HiHi	Alarme Alto Crítico. Define um intervalo de valores (maior igual) onde o tag é considerado em estado de Alarme Alto Crítico. É usado quando o valor do tag é está acima de um máximo, ou seja, extremamente alto.
Valor	Define os limites para cada situação possível de alarme (lolo, low, hi, hihi).
Pri	Define a prioridade para cada situação de alarme. Números pequenos indicam alta prioridade (a prioridade deve ser um número entre 0 e 999). Para um melhor controle os alarmes de maior prioridade irão aparecer em primeiro plano na janela de alarmes (objeto de tela Alarme).
Comentários	Um comentário ou mensagem pode ser definido para cada alarme.
Logar mensagens de retorno	Habilita o registro (log) das mensagens de retorno de alarme.
Grupo de Alarmes	Define o grupo de Alarmes do tag corrente. O grupo de Alarmes deve ser definido na opção Alarmes do Organizer.
Manter o valor do tag sempre atualizado	Define que o sistema irá supervisionar o tag constantemente, mesmo que ele não esteja sendo utilizado em nenhum lugar da aplicação, a fim de não perder nenhum alarme deste tag.
Usar outro nome de tag	Permite definir um nome alternativo para tag no registro (log) de alarmes.

4.13. Scripts de Tags

É possível executar scripts associados a mudanças nos valores dos tags e a estados de alarmes dos tags.

Scripts Disponíveis

Você pode associar scripts a tags executando-os em uma das situações a seguir:	
EVENTOS	**DESCRIÇÃO**
OnAck	Executa o script quando o alarme for sinalizado como reconhecido pelo usuário.
OnAlarmHigh	Executa o script quando o valor definido para alarme Alto (High) for alcançado.
OnAlarmHiHi	Executa o script quando o valor definido para o alarme HiHi for alcançado.
OnAlarmLow	Executa o script quando o valor definido para o alarme baixo (Low) for alcançado.
OnAlarmLoLo	Executa o script quando o valor definido para o alarme LoLo for alcançado.
OnAlarmReturn	Executa o script assim que uma situação de alarme tiver terminado.
OnValueChanged	Executa o script quando mudar o valor do tag
OnRead	Somente para tags PLC e Bloco. Executa o script quando o tag é lido do driver.

Capítulo 5

5. Telas

Uma **Tela** pode ser definida como uma janela para monitoramento de um processo. Cada aplicação pode ter um número ilimitado de telas.

Você pode criar uma nova Tela pressionando o botão Nova Tela na Barra de Ferramentas ou usando o comando Novo no menu Tela. Nesta nova tela você pode definir **Objetos de Tela**, um desenho de fundo e outras características específicas.

Para uma melhor visualização do sistema que você está monitorando alguns bitmaps de fundo podem ser definidos para as telas. Um bitmap do Windows é um arquivo gráfico com extensão BMP que pode ser criado em diversas aplicações Windows específicas para desenho, como por exemplo o Paintbrush.

Você pode criar seus desenhos (bitmaps) em qualquer tamanho e cores que desejar, Elipse SCADA irá importá-los automaticamente sem a necessidade de qualquer processo de conversão.

Para visualizar ou editar as propriedades da tela corrente dê um duplo clique em um espaço vazio da tela ou use o comando Propriedades do menu Tela.

Quando a opção Telas é selecionada na árvore do Organizer, a janela a seguir aparece, contendo uma lista de todas as telas da sua aplicação. Você pode criar uma nova tela usando o botão Novo à direita da página ou remover uma tela existente selecionando-a na lista e pressionando o botão Deletar.

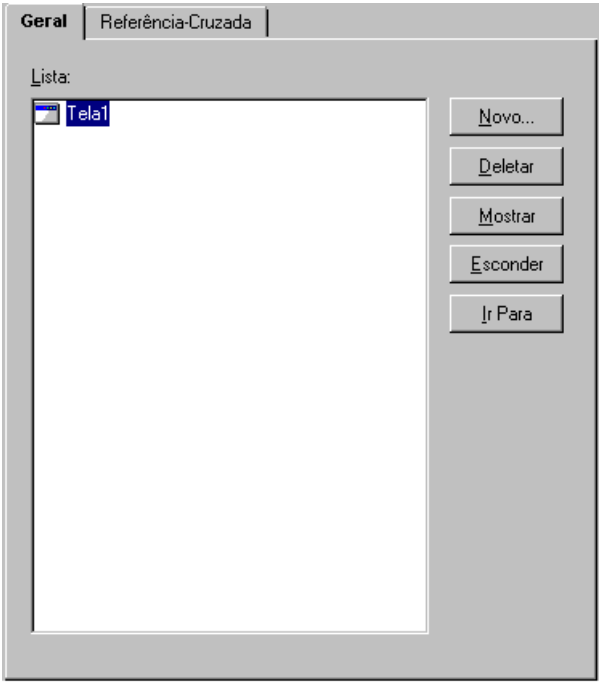


Figura 32: Janela de criação de Telas

Propriedades da Janela de Criação das Telas

OPÇÃO	DESCRIÇÃO
Lista	Mostra uma lista de todas as Telas existentes na sua aplicação. As mesmas Telas aparecem na árvore do Organizer logo abaixo da opção Telas e quando selecionadas suas respectivas propriedades são mostradas.
Nova	Cria uma nova Tela.
Deletar	Apaga a Tela selecionada da Lista.
Ir para	Fecha o Organizer e mostra a Tela corrente.
Mostrar	Mostra a Tela corrente sem selecioná-la.
Fechar	Esconde a Tela corrente mantendo-a na aplicação.

5.1. Propriedades Gerais da Tela

Cada tela que você cria para a aplicação aparece abaixo da opção Telas na árvore da aplicação no Organizer. Ao selecionar uma tela específica suas propriedades são mostradas ao lado direito da árvore. A página de propriedades gerais de telas aparece quando selecionada a aba Geral no topo das páginas da tela. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

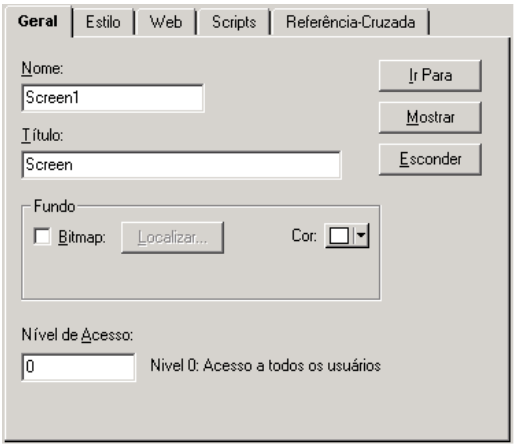


Figura 33: Propriedades da Tela

Propriedades da Tela

OPÇÃO	DESCRIÇÃO
Nome	Define um nome para a tela corrente. Usando este nome você pode abrir a tela de qualquer parte da aplicação usando botões ou teclas de função, bem como associá-la a scripts.
Titulo	Define um título para a tela, usado também como sua descrição.
Cor	Define a cor de fundo para a tela corrente. Este parâmetro é usado quando não existe um bitmap selecionado ou quando o bitmap não preenche toda a tela.
Bitmap	Habilita ou desabilita o uso de um bitmap como fundo para a tela corrente.
Localizar...	Permite navegar na estrutura de diretórios a fim de encontrar os arquivos BMP que serão usados como fundo para a tela. O caminho e nome do bitmap aparecem abaixo.
Nível de acesso	Define o nível de acesso para a tela.

5.2. Propriedades de Estilo de Telas

A página propriedades de estilo de telas aparece quando selecionada a aba Estilo no topo das propriedades da tela. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

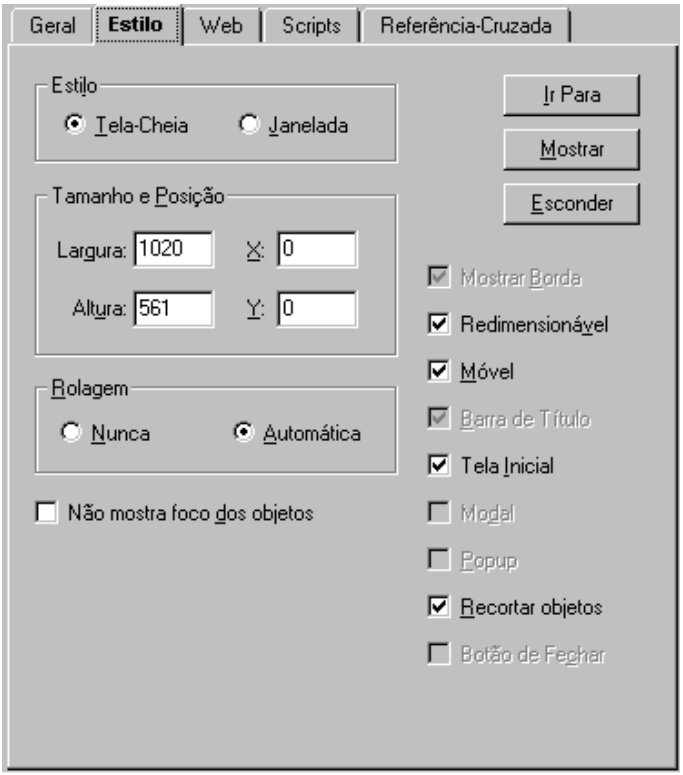


Figura 34: Propriedades de Estilo da Tela

Propriedades de Estilo da Tela

OPÇÃO	DESCRIÇÃO
Tela Cheia	Determina que a Tela ocupe toda a janela da aplicação.
Janelada	Determina que a Tela apareça dentro de uma janela sem ocupar toda a Tela da aplicação conforme especificado nas opções *Tamanho* e a *Posição*.
Largura	Define a largura da Tela em *pixels*.
Altura	Define a altura da Tela em *pixels*.
X	Determina a coordenada X para o canto superior esquerdo da Tela em *pixels*. Por exemplo, se você esta usando uma resolução no Windows de 640x480 (VGA) a sua coordenada X pode variar de 0 (zero) a 640 pixels para uma janela sem scroll bar.
Y	Determina a coordenada Y para o canto superior esquerdo da Tela em *pixels*. Por exemplo, se você esta usando uma resolução no Windows de 640x480 (VGA) a sua coordenada Y pode variar de 0 (zero) a 480 pixels para uma janela sem scroll bar.
Rolagem	**Nunca**: determina que a janela não tenha barras de rolagem, mesmo quando necessário. **Automático**: barras de rolagem aparecerão automaticamente quando se fizer necessário.
Não mostra foco dos objetos	Não mostra o foco dos objetos da tela em tempo de execução.
Mostrar Borda CE	Mostra uma borda de 1 pixel de largura em volta da tela. Somente para telas janeladas e de tamanho fixo.
Redimensionável CE	Permite o redimensionamento da janela em tempo de execução.
Móvel	Permite que a janela seja movida em tempo de execução.
Barra de Título	Mostra ou esconde a Barra de Título.
Visível	Torna a Tela visível em tempo de execução.
Modal	Somente válido para Telas janeladas. Define a janela como sendo modal, isto é, deve ser fechada para que o foco possa passar para outras janelas abertas.
Popup	Somente válido para Telas janeladas. Define a janela como sendo popup, isto significa que ela é fechada automaticamente quando perde o foco.
Recortar objeto	Define o uso do clipping para o redesenho dos objetos de tela relativo ao fundo, otimizando o redesenho. Esta opção deve ser habilitada somente se os objetos de tela não estão sobrepostos e não serão movidos, caso contrário poderá não surtir o efeito desejado.
Botão de Fechar	Habilita o botão fechar na janela (botão do canto superior direito). Este botão só pode ser visto em tempo

	de execução.

5.3. Scripts de Tela

Scripts de telas geralmente estão associados a uma ação executada na tela. Isto significa, por exemplo, que eles podem ser executados ao abrir ou fechar a tela, ou mesmo quando uma tecla for pressionada enquanto a tela é monitorada. Os scripts disponíveis para as telas são descritos na tabela abaixo.

Scripts Disponíveis

Você pode associar scripts a tags executando-os em uma das situações a seguir:	
EVENTOS	**DESCRIÇÃO**
OnHide	Executado após uma tela ter sido fechada.
OnKeyPress	Executado quando a tecla especificada no botão Nova tecla for pressionada.
OnKeyRelease	Executado quando a tecla especificada no botão Nova tecla for solta.
OnKillFocus CE	Executado quando a tela perde o foco de teclado ou mouse
OnPreHide	Executado antes da tela ser fechada. Chame a função Show() neste script para manter a tela aberta.
OnPreShow	Executado antes da tela ser mostrada. Chame a função Hide() neste script para cancelar a abertura da tela.
OnSetFocus CE	Executado quando a tela ganha o foco de teclado ou mouse.
OnShow	Executado quando a tela é mostrada.
WhileKeyPressed	Executado enquanto a tecla especificada no botão Nova tecla estiver sendo pressionada.
WhileRunning	Executado enquanto a tela estiver ativa.

Capítulo 6

6. Objetos de Tela

Quando o programador está desenvolvendo a sua aplicação, muitas vezes ele necessita mostrar algum resultado, fazer um alerta ou receber informações e acionamentos do usuário. Essa interação com o usuário é feita através do que chamamos **interface**. Para que o programador possa construir a interface de sua aplicação, o Elipse SCADA oferece uma série de recursos chamados "Objetos de Tela".

Objetos de Tela são elementos gráficos e representações de objetos reais do processo que ajudam o usuário a interagir e acompanhar a aplicação que está sendo executada no Elipse SCADA.

Neste capítulo, veremos os objetos de tela que estão disponíveis no Elipse SCADA, seu significado, utilidade e funcionamento. Para uma melhor compreensão, organizamos os objetos de tela em duas categorias: objetos de visualização e objetos de interação.

6.1. Edição dos Objetos de Tela

Os objetos de tela podem ser criados a partir da barra de ferramentas Objetos já descrita em seção anterior ou através do menu Objetos. Uma vez selecionado o objeto que se deseja criar mantenha o botão esquerdo do mouse pressionado na área da tela enquanto movimenta o mouse (um retângulo pontilhado mostra o tamanho e a forma do objeto). Ao soltar o botão o objeto será posicionado dentro da área especificada.

A edição dos objetos na tela, como alinhamento, tamanho, posição e agrupamento é feita através da barra de ferramentas **Arranjar** já descrita ou através do menu. O último objeto selecionado fica com o foco em vermelho para ser usado como referência. Para deselecionar um objeto use a combinação de teclas: [Shift]+[Ctrl]+[BotãoEsq].

6.2. Propriedades dos Objetos de Tela

Em geral, os objetos de tela possuem diversas propriedades comuns que estão agrupadas nas páginas descritas a seguir.

6.2.1. Página Tamanho e Pos

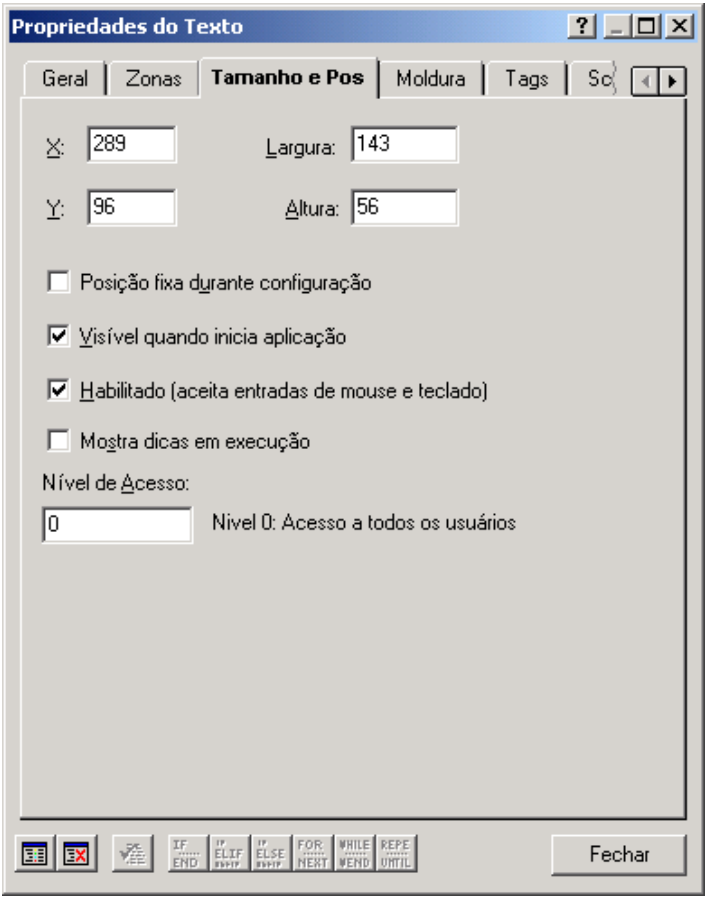


Figura 35: Página Tamanho e Pos

Propriedades da Página de Tamanho e Posição

OPÇÃO	DESCRIÇÃO
X	Define a coordenada *X* para o canto superior esquerdo do objeto, em pixels. É usado juntamente com o atributo *Y* para definir a posição do objeto a partir da origem da tela (0,0)
Y	Define a coordenada *Y* para o canto superior esquerdo do objeto, em pixels. É usado juntamente com o atributo *X* para definir a posição do objeto a partir da origem da tela (0,0)
Largura	Determina a largura do objeto em pixels. É usado juntamente com o atributo altura para definir o tamanho do objeto
Altura	Determina a altura do objeto em pixels. É usado juntamente com o atributo largura para definir o tamanho do objeto
Posição fixa durante a configuração	Determina que o objeto não possa ser movido durante a configuração (bloqueia a mudança de posição do objeto).
Visível quando iniciar aplicação	Define que o objeto será visível no momento em que a aplicação iniciar.
Habilitado (aceita entrada de mouse e teclado)	Habilita o acesso do teclado e mouse ao objeto (válido somente para objetos que permitem entradas via mouse ou teclado. Exemplo: slider).
Mostrar dica em execução	Habilita o objeto a mostrar uma dica (*tip*) quando o mouse está sobre ele.
Nível de Acesso	Permite associar um nível de acesso ao objeto (0 para acesso livre).

6.2.2. Página Moldura

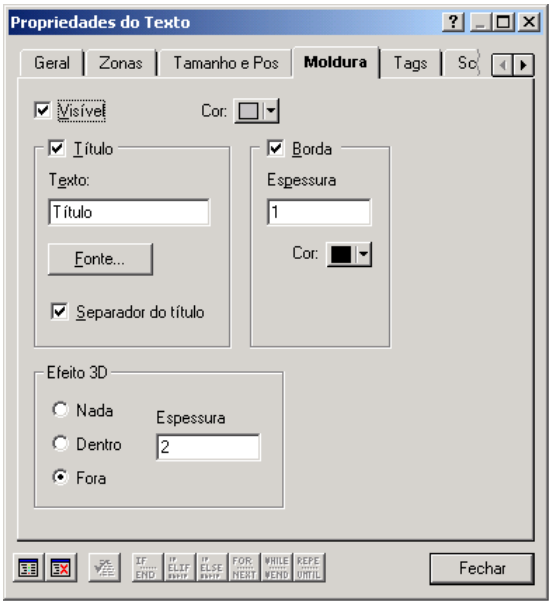


Figura 36: Página de Moldura

Propriedades da Página de Moldura

OPÇÃO	DESCRIÇÃO
Visível	Habilita ou desabilita uma moldura em volta do objeto.
Cor	Define a cor da moldura do objeto.
Título	Habilita ou desabilita um título na moldura do objeto.
Texto	Define o texto do título.
Fonte	Define fonte, cor e tamanho da fonte do Título.
Separador de título	Habilita ou desabilita uma linha separadora entre o texto e o objeto.
Borda	Habilita ou desabilita uma borda na moldura.
Espessura	Define a espessura do moldura em pixels.
Cor	Define a cor da borda do moldura.
Efeito 3D	Seleciona um efeito 3D dentro ou fora do moldura do objeto.
Espessura	Define a espessura em pixels para o efeito 3D.

6.2.3. Página de Tags

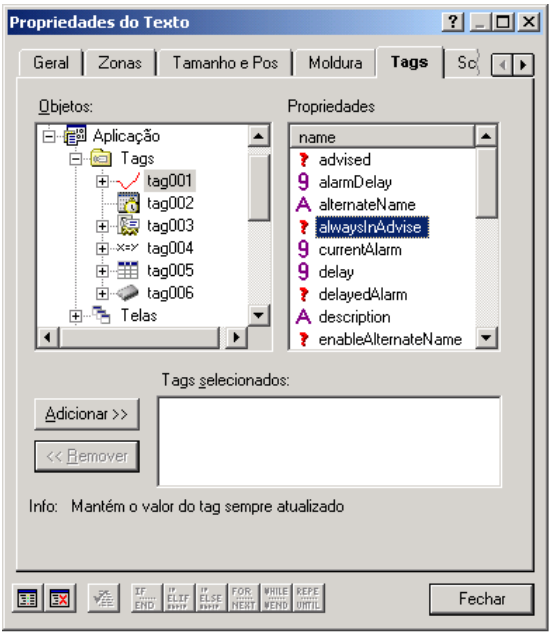


Figura 37: Página de Tags

Propriedades da Página de Tags

OPÇÃO	DESCRIÇÃO
Objetos	Mostra a árvore da aplicação. Conforme o objeto selecionado, suas propriedades aparecerão na janela de *Propriedades*.
Propriedades	Permite a seleção de qualquer propriedade do objeto selecionado na árvore.
Tags selecionados	Lista os Tags que estão associados ao objeto.
Adicionar	Adiciona os Tags marcados à lista de Tags selecionados.
Remover	Remove os Tags selecionados da lista.

6.3. Scripts de Objetos de Tela

Existem alguns scripts disponíveis para todos os objetos de tela, estes scripts são descritos na tabela abaixo. Scripts específicos de um objeto são descritos mais adiante na respectiva seção do objeto. Maiores detalhes a respeito do uso de scripts veja no capítulo específico.

Figura 38: Página de Scripts

Scripts Disponíveis

Você pode associar scripts a tags executando-os em uma das situações a seguir:	
EVENTOS	**DESCRIÇÃO**
OnKillFocus	Executado quando o objeto perde o foco do teclado ou mouse.
OnLButtonDblClk	Executado quando o botão esquerdo do mouse é pressionado duas vezes sobre o objeto.
OnLButtonDown	Executado quando o botão esquerdo do mouse é pressionado sobre o objeto.
OnLButtonUp	Executado quando o botão esquerdo do mouse é solto sobre o objeto.
OnMouseMove	Executado quando o mouse é movido sobre o objeto.
OnRButtonDblClk	Executado quando o botão direito do mouse é pressionado duas vezes sobre o objeto.
OnRButtonDown	Executado quando o botão direito do mouse é pressionado sobre o objeto.
OnRButtonUp	Executado quando o botão direito do mouse é solto sobre o objeto.
OnSetFocus	Executado quando o objeto recebe o foco do teclado ou mouse.

6.4. Referência Cruzada

A página de Referência-Cruzada *(Cross-Reference)* lista todos os links para o objeto corrente e suas propriedades, ou seja, quais outros objetos se referem a este e quais propriedades ou valores estão sendo usados. Se você deseja ir ao item selecionado basta dar um duplo-clique sobre. A caixa Mostrar itens filhos permite visualizar na lista os itens "filhos" associados aos itens "pai" listados.

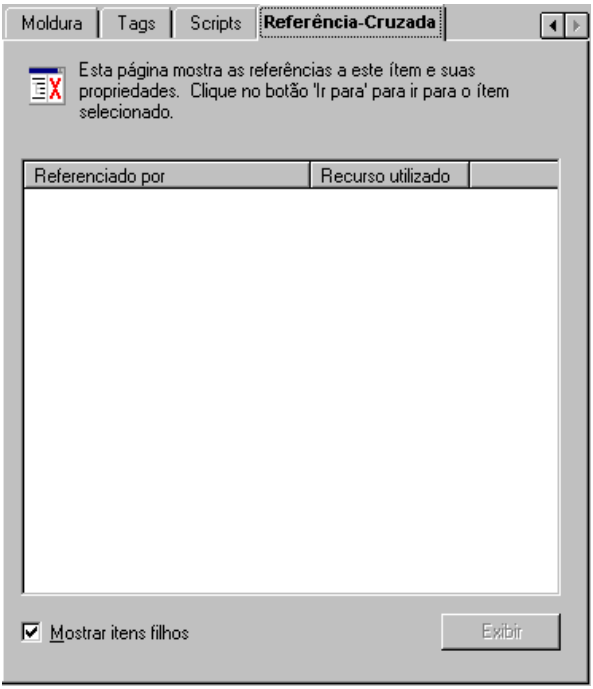


Figura 39: Página de Referência Cruzada

6.5. Objetos de Visualização

6.5.1. Texto

O objeto **Texto** permite atribuir mensagens a intervalos de valores dos tags, denominados **Zonas**. Podem ser definidas diversas zonas cada uma delas contendo sua própria mensagem.

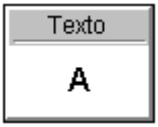


Figura 40: Texto

As Propriedades do Texto permitem definir cores e mensagens para cada zona. Também é possível definir uma zona padrão para os valores que não pertencem a nenhum intervalo específico. Você pode editar as propriedades de um texto dando um duplo-clique sobre o mesmo.

Propriedades Gerais do Texto

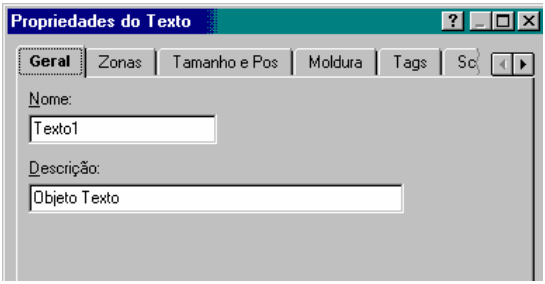


Figura 41: Propriedades Gerais do Texto

Propriedades Gerais do Texto

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do texto que será usado na árvore do Organizer e nos scripts.
Descrição	Uma breve descrição sobre o texto.

Zonas de Texto

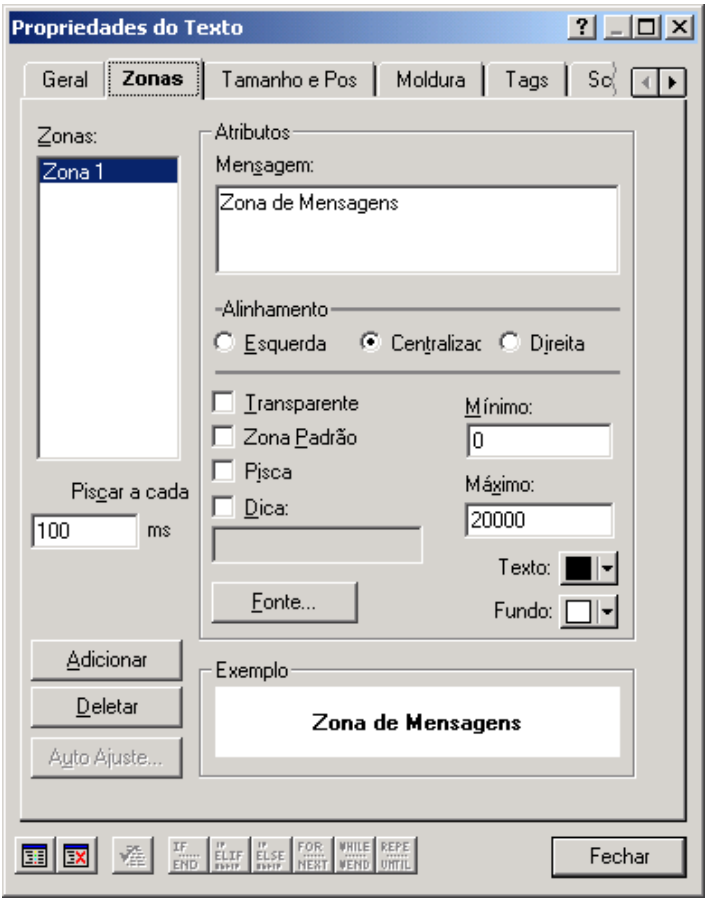


Figura 42: Propriedades da Zona de Texto

Propriedades da Zona de Texto

OPÇÃO	DESCRIÇÃO
Zonas	Lista todas as Zonas definidas e permite sua edição.
Adicionar	Adiciona uma nova Zona na lista.
Deletar	Remove da lista a Zona selecionada.
Auto Ajuste	Ajusta o intervalo das Zonas automaticamente.
Messagem	Texto associado a zona selecionada. Esta mensagem será mostrada quando o valor do Tag associado ao objeto Texto estiver dentro da zona.
Alinhamento	Define o alinhamento dos caracteres para o objeto Texto.
Transparente	Define que o fundo do objeto será transparente.
Zona padrão	Define a Zona selecionada como zona padrão (*default*), ou seja, uma zona que contém todos os valores que não estão definidos em outras zonas.
Pisca	Habilita o pisca-pisca para a zona. Uma vez ativa e definida como "Pisca" a zona alterna com a zona default conforme o tempo em Piscar a cada.
Piscar a cada	Define o tempo para o pisca-pisca entre as zonas Blinking e a default.
Dica	Permite configurar uma dica (*tip*) para cada zona. Se a zona não possui uma dica, o sistema utiliza a descrição do objeto.
Minimo	Define um valor mínimo para a Zona selecionada.
Maximo	Define um valor máximo para a Zona selecionada.
Fundo	Define a cor de fundo do Texto.
Fonte	Define fonte, cor e tamanho para o texto da mensagem.

6.5.2. Display

O objeto **Display** é usado para mostrar os valores dos tags em tempo real.

As propriedades do display permitem definir o tamanho, cor, fonte, alinhamento, efeitos 3D e outras características. Você pode editar as propriedades do display dando um duplo clique sobre o mesmo.

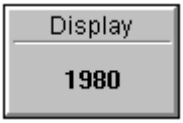


Figura 43: Display

Propriedades Gerais do Display

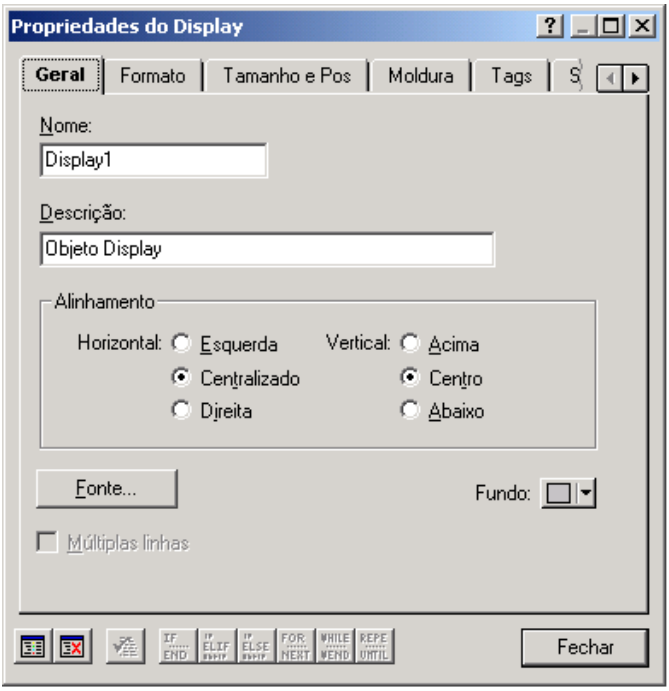


Figura 44: Propriedades Gerais do Display

Propriedades Gerais do Display

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do display que será usado na árvore do Organizer e nos scripts.
Descrição	Uma breve descrição sobre o display.
Alinhamento	Define o alinhamento dos caracteres para o display.
Fonte	Define a fonte, cor e tamanho do texto do display.
Fundo	Define a cor de fundo do display.
Múltiplas linhas	Define o display como tendo múltiplas linhas. Somente disponível se o formato do setpoint (página de formato) é string.

Formato do Display

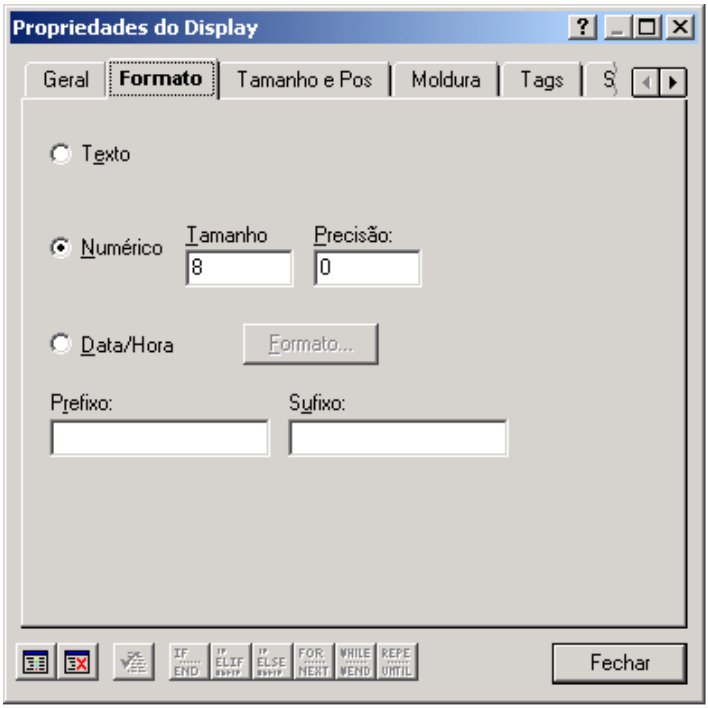


Figura 45: Propriedades de Formato do Display

Propriedades do Formato do Display

OPÇÃO	DESCRIÇÃO
Texto	Mostra o valor do Tag em formato string.
Numérico	Mostra o valor do Tag em formato numérico.
Tamanho	Define o número de dígitos a serem mostrados incluindo o ponto decimal.
Precisão	Define quantos dígitos do tamanho serão decimais.
Prefixo	Adiciona um prefixo tipo string ao valor mostrado.
Sufixo	Adiciona um sufixo tipo string ao valor mostrado.

6.5.3. Browser

O objeto **Browser** permite visualizar seus arquivos de bancos de dados. Estes arquivos devem estar no formato Elipse SCADA, ou seja, devem ter sido criados pelo Elipse. Para localizar estes arquivos você deve procurar pelas extensões .DAT (Históricos ou Alarmes) ou .HDR (Batelada). Você pode editar as propriedades do browser dando um duplo-clique sobre o mesmo.

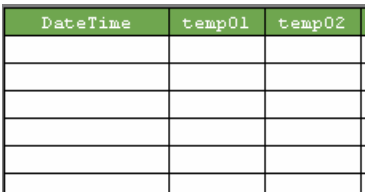


Figura 46: Browser

Propriedades Gerais do Browser

Figura 47: Propriedades Gerais do Browser

Propriedades Gerais do Browser

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do Browser que será usado na árvore do Organizer e nos Scripts.
Descrição	Uma breve descrição sobre o Browser.
Fontes	Define a fonte, cor e tamanho do texto a ser mostrado no Browser.
Ordem	Define a ordem (ascendente ou descendente) dos registros do Browser.
Título de fundo	Define a cor de fundo para o título do Browser.
Texto doTítulo	Define a cor do texto para o título do Browser.
Fundo das linhas	Define a cor de fundo das linhas de texto do Browser.
Texto das linhas	Define a cor do texto das linhas do Browser.
Fundo selecionado	Define a cor de fundo para as linhas selecionadas do Browser.
Texto selecionado	Define a cor do texto para as linhas selecionadas do Browser.
Grades	Define a cor da grade do Browser.
Fundo	Define a cor da área externa do Browser.

Configurações do Browser

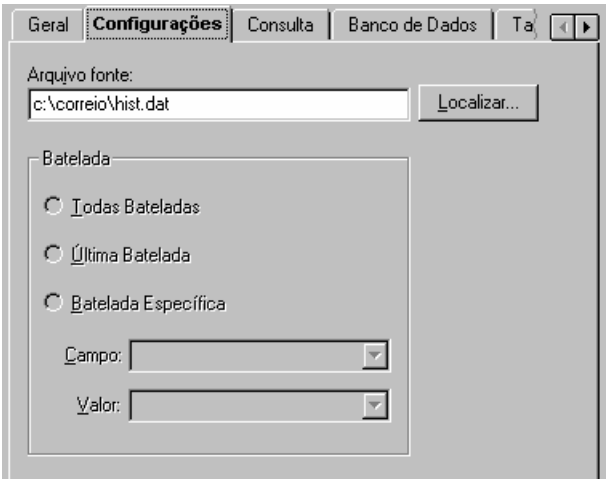
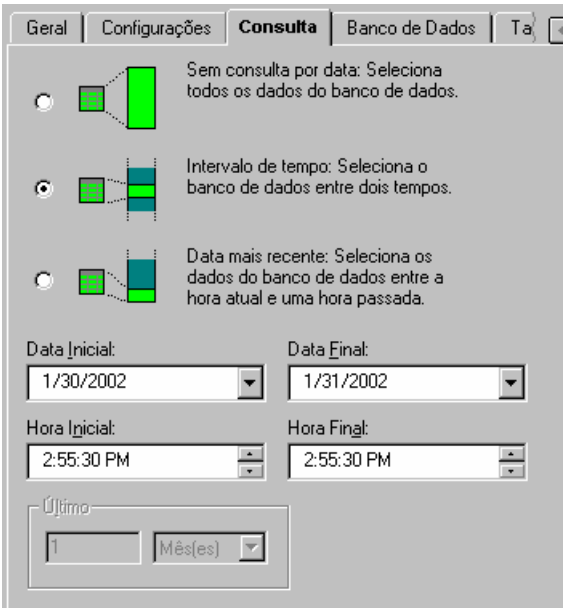


Figura 48: Configurações do Browser

Propriedades das Configurações do Browser

OPÇÃO	DESCRIÇÃO
Arquivo Fonte	Define o nome do arquivo a ser mostrado no objeto Browser. Este arquivo deve ter extensão DAT ou HDR.
Browser	Permite localizar o arquivo fonte a ser usado pelo Browser.
Todas as Bateladas	Seleciona todas as bateladas para serem impressas. Esta opção está disponível somente se o arquivo fonte (*Source Filename*) é um arquivo de Histórico com processo de batelada habilitado.
Últimas bateladas	Seleciona a última batelada para ser impressa. Esta opção está disponível somente se o arquivo fonte (*Source Filename*) é um arquivo de Histórico com processo de batelada habilitado.
Batelada Específica	Seleciona uma batelada específica para ser impressa, conforme o especificado nos campos *Campo* (*Field*) e *Critério* (*Criteria*). Esta opção está disponível somente se o arquivo fonte (*Source Filename*) é um arquivo de Histórico com processo de batelada habilitado.
Campos	Lista os campos disponíveis para seleção de uma batelada específica. Somente campos tipo string serão listados.
Valor	Define o valor a ser buscado quando selecionada uma batelada específica.

Consulta do Browser



Propriedades da Consulta do Browser

OPÇÃO	DESCRIÇÃO
Sem consulta por data	Não será usado filtro, ou seja, seleciona todos os dados.
Intervalo de tempo	Seleciona os dados dentro de um intervalo de tempo especificado.
Dados mais recentes	Seleciona apenas os dados mais novos.
Data Inicial	Determina a data inicial do intervalo de tempo.
Data Final	Determina a data final do intervalo de tempo.
Hora Inicial	Determina o horário inicial do intervalo de tempo.
Hora Final	Determina o horário final do intervalo de tempo.
Último	Define o número de unidades para selecionar os dados mais recentes.
Unidade	Define a unidade usada para selecionar os dados mais recentes.

Banco de Dados do Browser

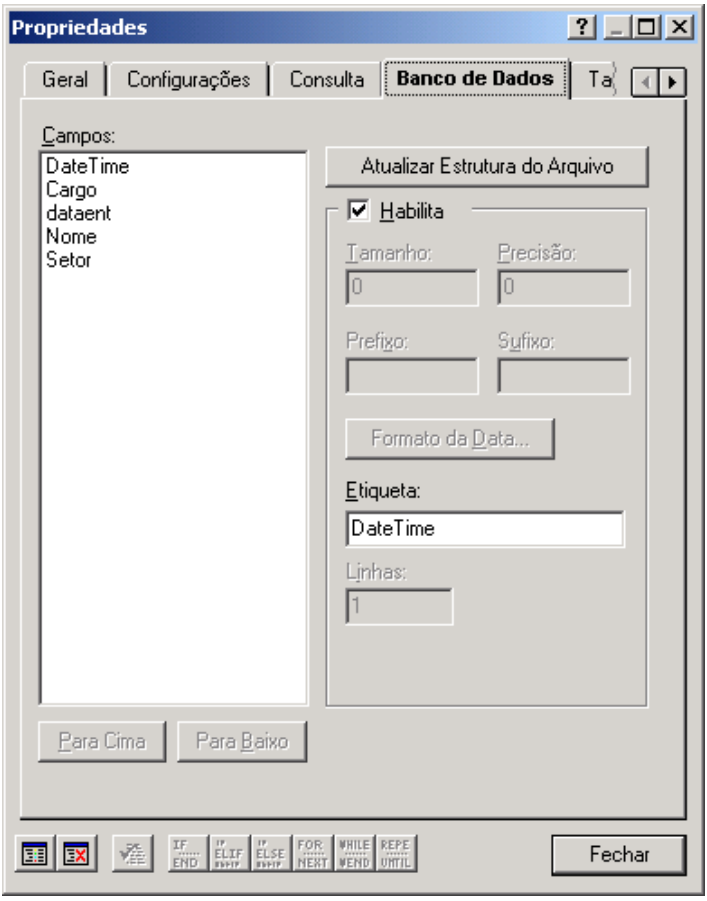


Figura 49: Propriedades do Banco de Dados do Browser

Propriedades do Banco de Dados do Browser

OPÇÃO	DESCRIÇÃO
Campos	Lista os campos disponíveis.
Para Cima	Move o campo selecionado uma posição para cima.
Para Baixo	Move o campo selecionado uma posição para baixo.
Habilita	Permite que o campo selecionado seja impresso.
Atualiza estrutura do arquivo	Recarrega a lista de campos atual do arquivo nos campos do browser. Quando é feita alguma modificação na estrutura do arquivo usado pelo browser este botão deve ser pressionado para atualização dos campos.
Tamanho	Define o número de dígitos a serem mostrados incluindo o ponto decimal.
Precisão	Define quantos dígitos do tamanho serão decimais.
Prefixo	Adiciona um prefixo tipo string ao valor mostrado.
Suffixo	Adiciona um sufixo tipo string ao valor mostrado.
Formato de data	Define o formato da marcação de tempo (timestamp) do browser.
Etiqueta	Permite trocar o nome do campo selecionado.
Linhas	Define o número de linhas que o campo selecionado irá ter no browser.

Scripts do Browser

Os eventos disponíveis para Scripts exclusivos do Browser são descritos na tabela abaixo. Maiores detalhes a respeito do uso de Scripts veja no capítulo específico.

Scripts Disponíveis

Você pode associar scripts a tags executando-os em uma das situações a seguir:	
EVENTOS	**DESCRIÇÃO**
OnDrawRow	Executado logo antes de cada linha do Browser ser desenhada. Permite que a cor de fundo e do texto da linha sejam modificadas através das funções SetTempRowColor() e SetTempRowTextColor(), de acordo com o valor dos campos da linha (este valor pode ser pego com a função GetField()).

6.5.4. Bitmap

O objeto **Bitmap** permite inserir imagens, figuras ou desenhos nas suas telas de sua aplicação, desde que estejam nos formatos BMP, GIF ou JPEG. Este objeto pode ser redimensionado para ter o mesmo tamanho da figura e possui suporte a transparência, bastando habilitar e selecionar uma cor. Você pode editar as propriedades do Bitmap dando um duplo clique sobre o mesmo.

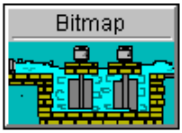
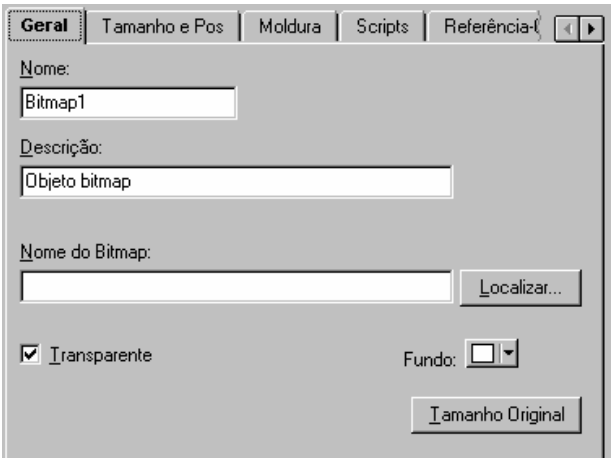


Figura 50: Bitmap

Nota: No Elipse SCADA CE, apenas o formato .BMP está disponível.

Propriedades Gerais do Bitmap



Propriedades Gerais do Bitmap

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do Bitmap que será usado na árvore do Organizer e nos Scripts.
Descrição	Uma breve descrição sobre o Bitmap.
Nome do Bitmap	Define o nome do arquivo correspondente.
Localizar...	Permite indicar a localização do arquivo a ser usado.
Transparente	Determina que a cor epecificada em Fundo seja transparente.
Fundo	Define a cor de fundo do Bitmap.
Tamanho Original	Ajusta o objeto Bitmap de forma que ele tenha o mesmo tamanho da figura original.

6.5.5. Animação

O objeto **Animação** é um meio fácil de criar uma animação usando imagens (bitmaps) associados a quadros (*frames*) definidos pelo usuário que são mostrados em sequência. Um Tag deve ser associado à animação de forma que o valor do Tag determine qual o quadro da animação que será mostrado. Os valores do Tag são associados a Zonas, que correspondem a um determinado quadro.

É possível usar imagens de quaisquer tamanho e cores, desde que sejam arquivos nos formatos suportados: GIF, JPEG e Bitmap (.BMP). Você pode editar as propriedades da Animação dando um duplo clique sobre o objeto.

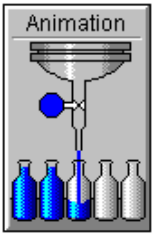


Figura 51: Animação

Nota: No Elipse SCADA CE, apenas o formato .BMP está disponível.

Propriedades Gerais da Animação

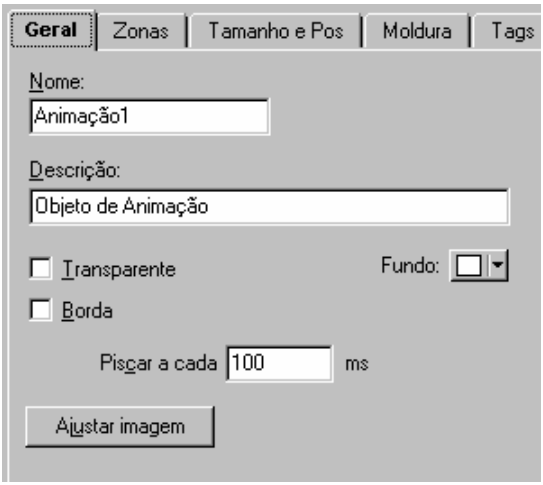


Figura 52: Propriedades Gerais da Animação

Propriedades Gerais da Animação

OPÇÃO	DESCRIÇÃO
Nome	Define o nome da animação.
Descrição	Uma breve descrição sobre o objeto.
Transparente	Define que a cor especificada na opção Fundo será transparente, permitindo que o fundo da tela apareça.
Borda	Habilita uma borda (linha preta) ao redor dos quadros.
Fundo	Define uma cor de fundo para a animação.
Piscar a cada	Tempo em milisegundos para o pisca-pisca entre as zonas Pisca e Padrão.
Ajustar imagem	Ajusta o objeto animação de forma que ele tenha o mesmo tamanho do primeiro quadro da animação.

Zonas da Animação

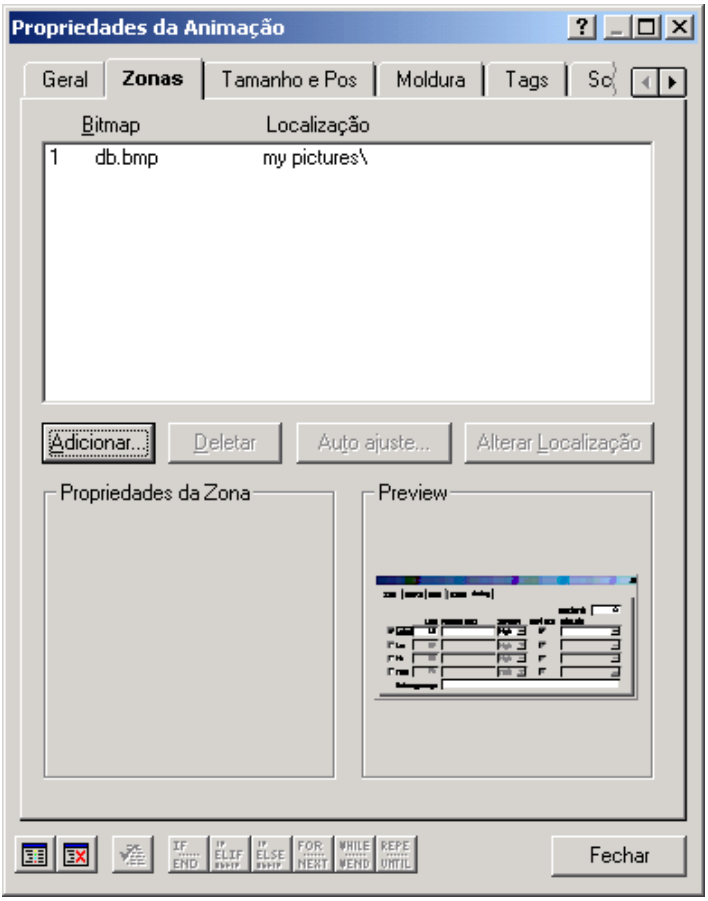


Figura 53: Propriedades da Zona de Animação

Propriedades da Zona de Animação

OPÇÃO	DESCRIÇÃO
Bitmap	Lista os bitmaps dos quadros da animação. Selecione um dos bitmaps para ver e editar suas propriedades.
Localização	Mostra o caminho dos bitmaps.
Adicionar	Usado para adicionar um bitmap ou um grupo de bitmaps à lista.
Deletar	Remove um bitmap ou um grupo de bitmaps da lista.
Auto ajuste...	Ajusta automaticamente as zonas dividindo o total do intervalo igualmente entre as zonas definidas.
Alterar Localização	Usado para trocar a localização (*path*) do bitmap.
Zona Padrão	Define o quadro selecionado na lista como padrão; desta forma, ele será mostrado quando o valor do tag estiver fora das outras zonas definidas.
Minimo	Define um valor mínimo para a zona selecionada.
Maximo	Define um valor máximo para a zona selecionada.
Pisca	Indica que essa zona será utilizada no pisca-pisca.
Dica	Permite configurar uma dica (*tip*) para cada zona. Se a zona não possui uma dica habilitada então é utilizada a descrição do objeto.
Preview	Mostra uma prévia do quadro selecionado.

6.5.6. AVI

O objeto **AVI** é utilizado para visualização de arquivos de extensão AVI no sistema. Através deste objeto, é possível localizar arquivo AVI no sistema e mostrá-lo na tela no momento da execução do projeto. Este objeto aceita somente arquivos AVI.

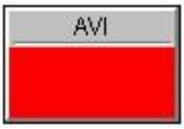


Figura 54: AVI

Propriedades Gerais do AVI

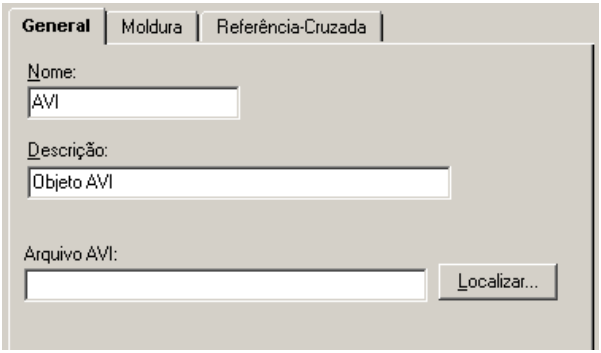


Figura 55: Propriedades Gerais do AVI

Propriedades Gerais do AVI

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do objeto no sistema.
Descrição	Uma breve descrição sobre o video.
Arquivo AVI	Mostra o caminho do arquivo AVI no sistema.
Localizar...	Permite navegar nos diretórios para localizar o arquivo AVI.

6.5.7. Video

O objeto **Video** é utilizado para visualização de uma imagem de video. Através desta opção, é possível visualizar arquivos do sistema ou criados a partir da opção **Watcher.**

Propriedades Gerais do Video

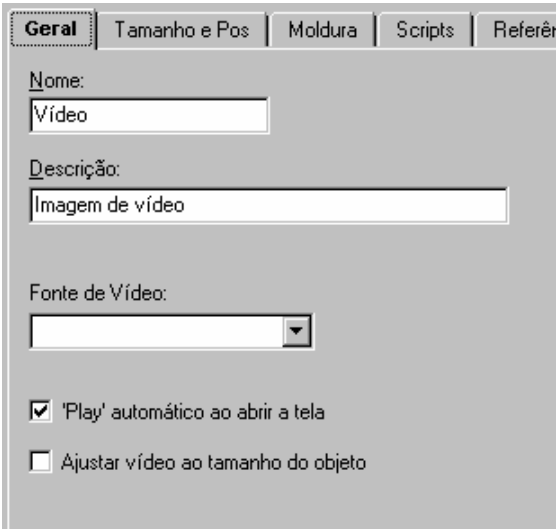


Figura 56: Propriedades Gerais do Video

Propriedades Gerais do Video

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do objeto no sistema.
Descrição	Uma breve descrição sobre o video.
Fonte do Video	Determina a fonte que o video irá utilizar para a visualização do objeto.
Play automático ao abrir a tela	Habilita a visualização do video no momento da execução do objeto.
Ajustar video ao tamanho do objeto	Habilita o ajuste do video conforme o tamanho do objeto.

6.5.8. Preview

O objeto **Preview** é utilizado para visualização de arquivos de video que foram gerados através da placa XPressPlayer. Através deste objeto, é possível configurá-lo na placa XPressPlayer para conectar-se a câmera determinada ou indicar o arquivo XPressPlayer. A visualização do objeto será mostrada na execução do projeto.

Figura 57: Preview

Propriedades Gerais do Preview

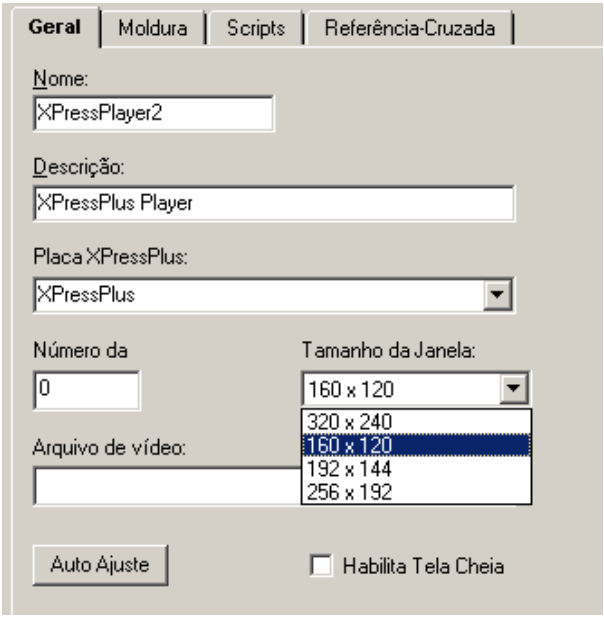


Figura 58: Propriedades Gerais do Preview

Propriedades Gerais do Preview

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do objeto no sistema.
Descrição	Uma breve descrição sobre o video.
Placa XPressPlus	Indica o objeto do XPressPlus do Watcher a ser associado ao objeto Preview.
Número da câmera	Indica o número da câmera que será conectada ao objeto.
Tamanho da Janela	Indica o tamanho da janela para a visualização do objeto.
Arquivo de video	Mostra o caminho do arquivo de video no sistema.
Auto Ajuste	Ajusta automaticamente a janela de visualização conforme as especificações da opção Tamanho da Janela.
Habilitar tela cheia	Habilita a visualização em "tela cheia".

6.5.9. Tendência

Este objeto é usado para visualizar um gráfico de tendência com até 16 tags. O gráfico é constantemente atualizado a medida que o processo evolui e os valores dos tags mudam.

Usando tendências, você pode fazer gráficos como Valor x Tempo e Valor x Valor. Devido a estas características pode-se efetuar a análise dos dados a medida que o processo evolui.

O objeto Tendência possui vários aspectos que podem ser modificados através de várias páginas de propriedades, que serão vistas a seguir.

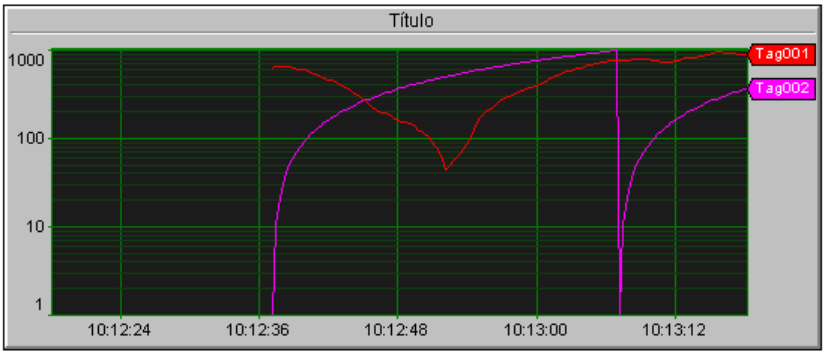


Figura 59: Tendência

Propriedades Gerais da Tendência

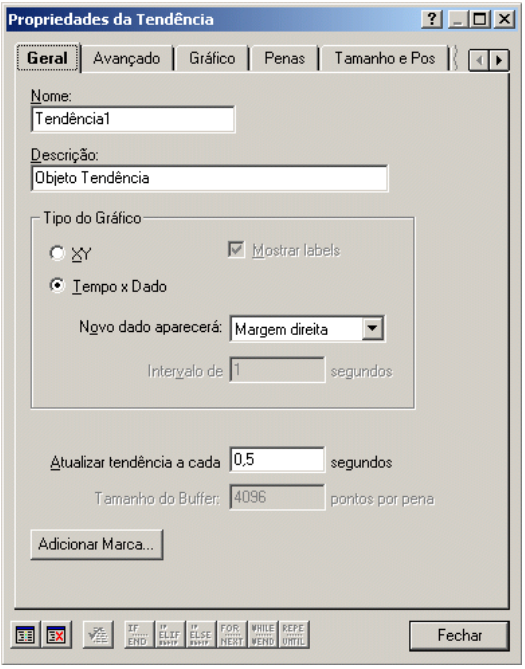


Figura 60: Propriedades Gerais da Tendência

Propriedades Gerais da Tendência

OPÇÃO	DESCRIÇÃO
Nome	Define o nome para identificação do objeto Tendência.
Descrição	Uma breve descrição sobre a Tendência.
XY	Habilita um gráfico em tempo real em função de duas variáveis, uma em cada eixo de coordenadas. As variáveis podem ser agrupadas aos pares, de forma que um objeto possa mostrar várias funções XY.
Tempo x Dado	Define um gráfico de dados em função do tempo.
Mostrar labels	Mostra uma legenda com o significado para cada pena da Tendência.
Novo dado aparecerá	Define de onde virão os novos dados da Tendência.
Intervalo de	Define o intervalo total de tempo para amostragem do gráfico (em segundos). Esta opção é desabilitada se a Tendência é histórica.
Atualizar tendência a cada	Define o intervalo de tempo entre cada atualização do gráfico. Esta opção é desabilitada se a Tendência é histórica.
Tamanho do Buffer	Define o número de pontos (de 1 a 1.000.000) que a tendência irá armazenar para cada pena (cada ponto ocupa 20 bytes). Esta opção é desabilitada se a Tendência é histórica.
Adicionar Marca	Adiciona uma linha horizontal, uma linha vertical ou um ponto para referência no gráfico da Tendência.

Propriedades Avançadas da Tendência

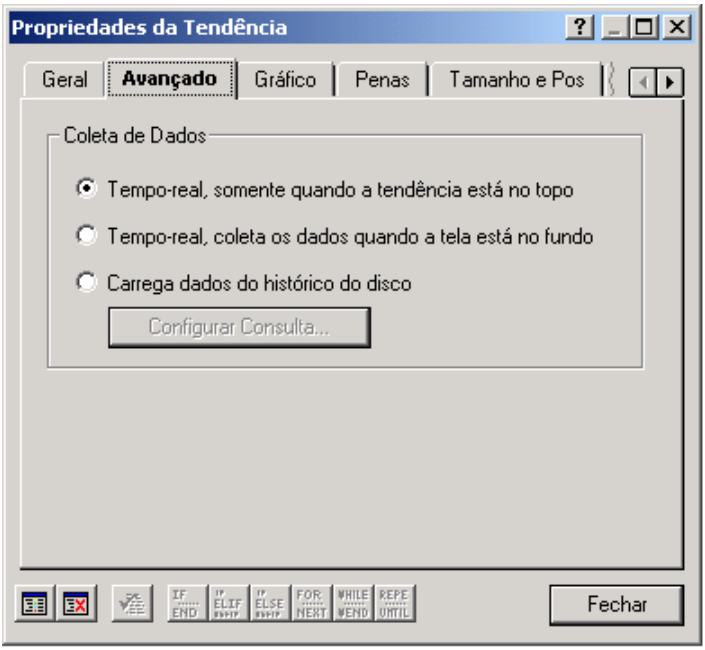


Figura 61: Propriedades Avançadas da Tendência

Propriedades Avançadas da Tendência

OPÇÃO	DESCRIÇÃO
Tempo real, somente quando a tendência está no topo	Atualiza os valores da Tendência em tempo real somente se a Tendência está visível.
Tempo real, coleta de dados quando a tela está no fundo	Atualiza os valores da tendência em tempo real mesmo quando a Tendência não está visível.
Carregar dados do histórico em disco	Atualiza a Tendência carregando dados de um Histórico selecionado.
Configurar consulta...	Abre uma janela que permite configurar a Consulta da Tendência Histórica.

Propriedades do Gráfico de Tendência

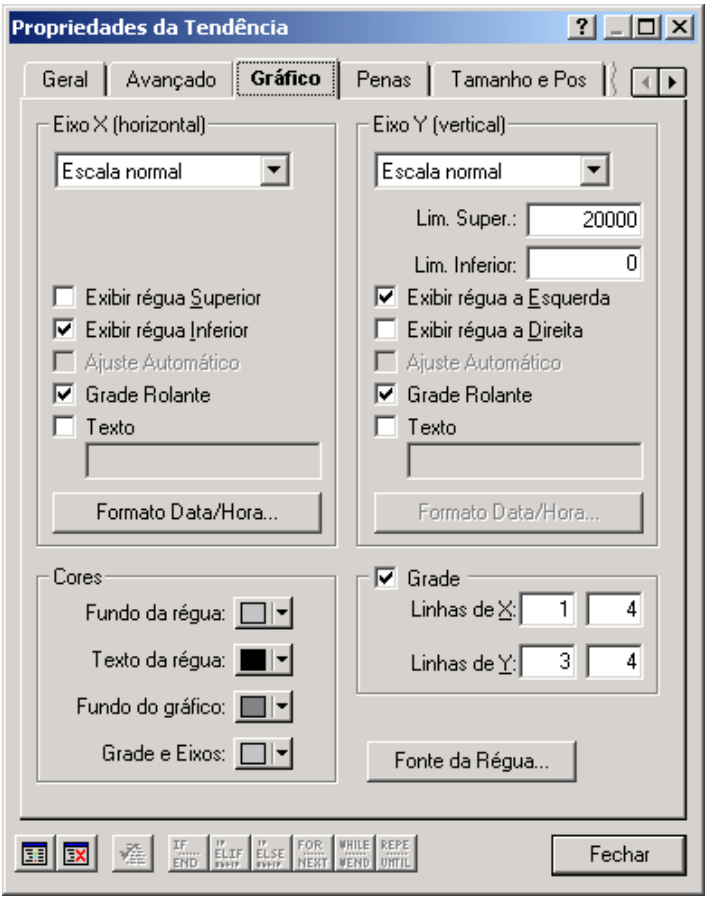


Figura 62: Propriedades da Tendência

Propriedades para os Eixos X e Y

OPÇÃO	DESCRIÇÃO
(Escala)	A primeira caixa de seleção indica o tipo de escala a ser usado no eixo, a saber: escala normal, escala logarítmica ou fator de potência.
Limite	Define o limite esquerdo do eixo X no gráfico. Esta opção está disponível somente se o gráfico é do tipo XY.
Limite direito	Define o limite direito do eixo X no gráfico. Esta opção está disponível somente se o gráfico é do tipo XY.
Exibir régua superior	Exibe uma régua no topo do gráfico, com os valores do eixo X.
Exibir régua inferior	Exibe uma régua na base no gráfico, com os valores do eixo X.
Limite superior	Define o limite superior do eixo Y no gráfico.
Limite inferior	Define o limite inferior do eixo Y no gráfico.
Exibir régua a esquerda	Exibe uma régua à esquerda do gráfico, com os valores do eixo Y.
Exibir régua a direita	Exibe uma régua à direita do gráfico, onde os valores do eixo Y.
Grade Rolante	Faz com que a grade do gráfico acompanhe o movimento gerado pela entrada dos dados na Tendência.
Texto	Permite colocar uma legenda para os eixos X e Y.
Formato Data/Hora	Premite definir o formato de data e hora, para Tendências do tipo Valor x Tempo.

Propriedades de Cores do Gráfico

OPÇÃO	DESCRIÇÃO
Fundo da régua	Define a cor de fundo para as réguas do gráfico.
Texto da régua	Define a cor para o texto que será mostrado nas réguas.
Fundo do gráfico	Define a cor de fundo para o gráfico.
Grades e eixos	Define a cor da grade e dos eixos do gráfico.

Propriedades da Grade do Gráfico

OPÇÃO	DESCRIÇÃO
Grade	Habilita a grade no gráfico (linhas de orientação).
Linhas de X	Define o número de linhas horizontais da grade.
Linhas de Y	Define o número de linhas verticais da grade.
Fonte da régua...	Define fonte, tamanho e cor para os caracteres do gráfico.

Cores das Penas da Tendência

Figura 63: Propriedades das Cores das Penas da Tendência

Propriedades das Cores das Penas da Tendência

OPÇÃO	DESCRIÇÃO
	Insere uma pena associada a um tag.
	Insere uma pena associada a um campo do histórico (a opção Carrega Dados do Histórico do Disco deve estar habilitada na aba Avançado).
	Insere uma marca indicativa no gráfico.
	Apaga uma pena ou uma marca selecionada na lista.
	Define que o desenho da pena será do tipo Linha. Os estilos de desenho de pena suportados pelo CE são apenas o sólido e o tracejado. Caso um estilo não suportado for solicitado pela aplicação, será utilizado o estilo sólido.
	Define que o desenho da pena será do tipo Área.
	Define que o desenho da pena será do tipo Barra.
	Muda a ordem da pena selecionada.

Label	Essa coluna lista as penas selecionadas para o gráfico. Permite mudar o texto da legenda e ativar/desativar cada pena.
Y Axis	Seleciona o tag a ser visualizado no eixo Y.
X Axis	Seleciona o tag a ser visualizado no eixo X.
	Mostra o quadro Cores das Penas.

6.5.10. Gráfico de Barras

O **Gráfico de Barras** é utilizado quando se deseja visualizar os dados na forma de volume. Editando as propriedades do gráfico podem ser definidas uma série de características como escala gráfica, orientação das barras (vertical/horizontal), réguas e cores.

Cada tag associado ao Gráfico de Barras é representado por uma barra com uma cor específica, podendo ser mostradas até um máximo de 16 barras (ou seja, 16 tags). O objeto possui ainda um segundo modo, o **Bar Gauge**. Nesse modo, os valores dos tags são indicados por uma agulha que percorre uma escala disposta sobre cada barra.

Você pode editar as propriedades do Gráfico de Barras dando um duplo clique sobre o mesmo. Nas propriedades do Gráfico de Barras temos várias páginas (abas ou *tabs*) para a sua configuração.

Figura 64: Gráfico de barras

Propriedades Gerais do Gráfico de Barras

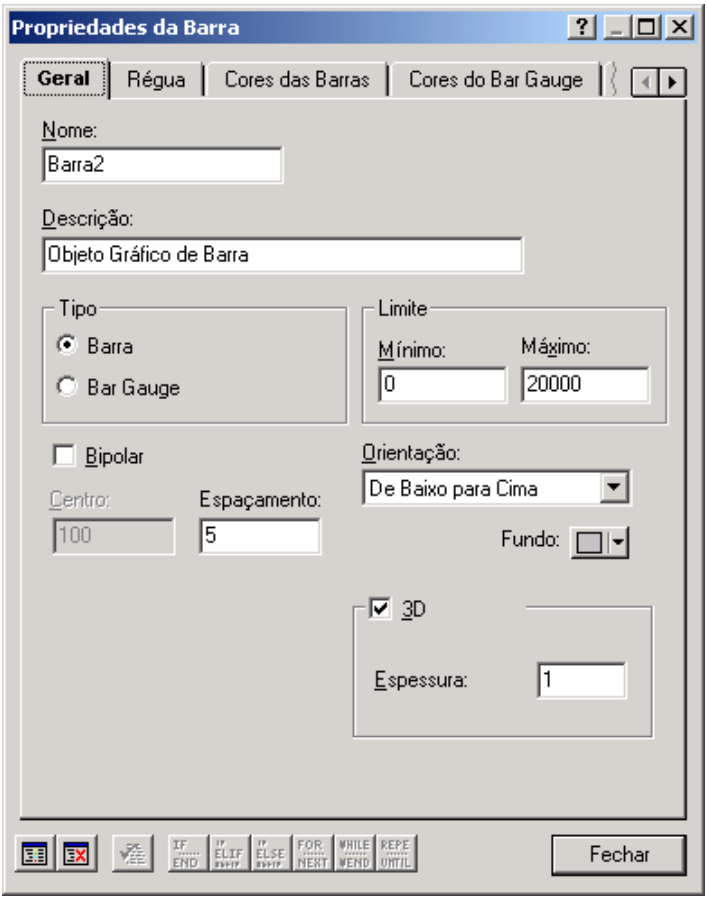


Figura 65: Propriedades Gerais do Gráfico de Barras

Propriedades Gerais do Gráfico de Barras

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do objeto, que será usado na árvore do Organizer e nos Scripts.
Descrição	Uma breve descrição sobre o gráfico.
Tipo	Define qual o tipo de gráfico de barras: Barra (simples) ou Bar Gauge (medidor em barra).
Limite	Define os valores mínimo e máximo para o Gráfico de Barras.
Orientação	Define a orientação do Gráfico. As opções disponíveis são de baixo para cima, de cima para baixo, da esquerda para a direita, da direita para a esquerda.
Fundo	Define uma cor de fundo para o Gráfico de Barras.
Bipolar	Determina que as barras do gráfico tenham dois lados divididos por um centro. As barras crescem do centro em direção aos limites do gráfico.
Centro	Define o centro para um Gráfico de Barras bipolares.
Espaçamento	Define o espaço entre as barras, em pixels.
3D	Habilita ou desabilita um efeito tridimensional nas barras do gráfico. Você também pode definir a largura do efeito 3D.

Propriedades da Régua

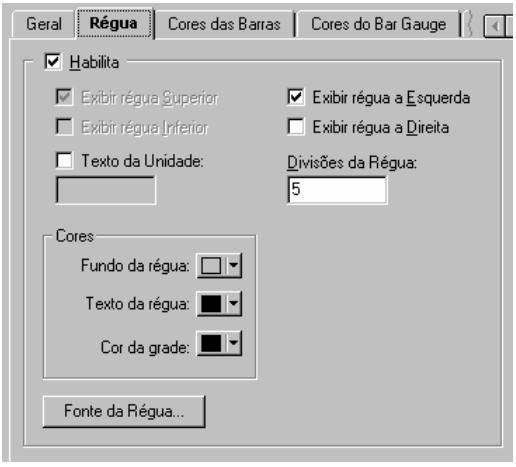


Figura 66: Propriedades da Régua

Propriedades da Régua

OPÇÃO	DESCRIÇÃO
Habilita	Permite que uma régua seja mostrada no Gráfico de Barras.
Exibir régua superior	Exibe uma régua superior no Gráfico de Barras, onde são mostrados os valores dos Tags.
Exibir régua inferior	Exibe uma régua inferior no Gráfico de Barras, onde são mostrados os valores dos Tags.
Exibir régua a esquerda	Exibe uma régua a esquerda do Gráfico de Barras, onde são mostrados os valores dos Tags.
Exibir régua a direita	Exibe uma régua a direita do Gráfico de Barras, onde são mostrados os valores dos Tags.
Texto da unidade	Unidade a ser mostrada na régua do Gráfico de Barras.
Divisões da régua	Define o número de divisões da régua.
Cores	Define as cores do fundo da régua, do texto e da grade (divisões).
Fonte da régua...	Define fonte, tamanho e cor para os caracteres na régua do Gráfico de Barras.

Cores da Barra

Esta aba permite habilitar legendas que ficarão ao lado do desenho das barras no gráfico. Isso é feito marcando o quadro Mostrar labels. Cada número indica a barra do gráfico, sua respectiva legenda e cor.

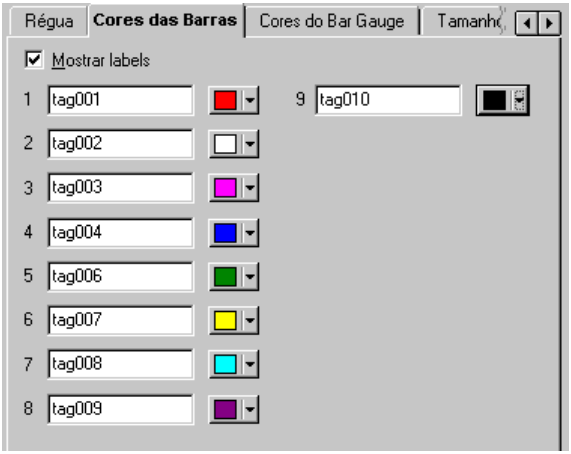


Figura 67: Propriedades das Barras

Cores do Bar Gauge

Esta página permite ajustar as propriedades do Bar Gauge. Ela só está disponível quando a opção Bar Gauge é especificada na página de propriedades gerais do gráfico.

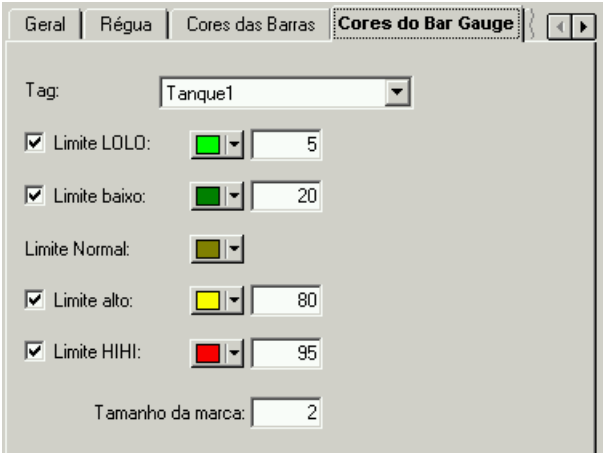


Figura 68: Definição das cores do Bar Gauge

Propriedades do Bar Gauge

OPÇÃO	DESCRIÇÃO
Tag	Permite escolher qual a barra que será editada.
Limite LOLO	Habilita a indicação da faixa de valor baixo crítico na barra. Especifica a cor da faixa e o valor limite.
Limite baixo	Habilita a indicação da faixa de valor baixo na barra. Especifica a cor da faixa e o valor limite.
Limite Normal	Especifica a cor da faixa de valores considerados normais para o tag.
Limite alto	Habilita a indicação da faixa de valor alto na barra. Especifica a cor da faixa e o valor limite.
Limite HIHI	Habilita a indicação da faixa de valor alto crítico na barra. Especifica a cor da faixa e o valor limite.
Tamanho da marca	Define o tamanho da agulha que fará a indicação do valor do tag.

6.5.11. Gauge

O objeto **Gauge** é bastante útil para mostrar variáveis com resultados analógicos, funcionando como um medidor contínuo. Os valores de escala e as propriedades do gauge podem ser definidos pelo usuário, além de sua posição na tela que pode variar em 0, 90, 180 ou 270 graus de rotação. Você pode editar as propriedades do Gauge dando um duplo clique sobre o mesmo.

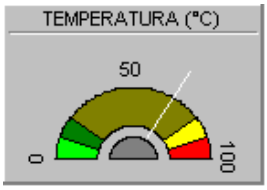


Figura 69: Gauge

Propriedades Gerais do Gauge

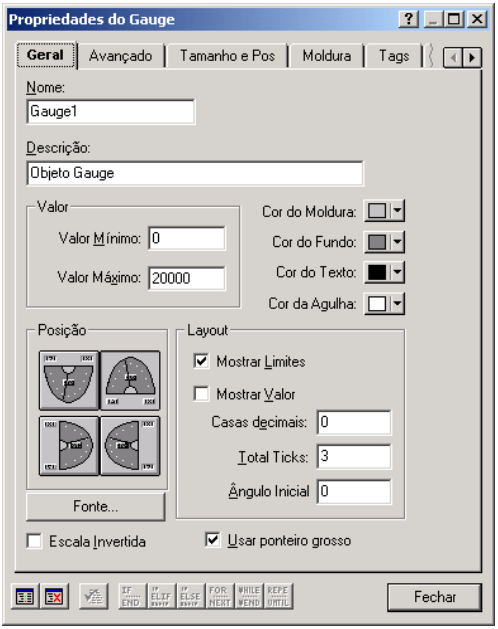


Figura 70: Propriedades do Gauge

Propriedades Gerais do Gauge

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do Gauge.
Descrição	Uma breve descrição sobre o Gauge.
Valor mínimo	Define o valor mínimo para o Gauge.
Valor máximo	Define o valor máximo para o Gauge.
Cor da moldura	Define a cor de fundo do Gauge.
Cor do fundo	Define a cor do Gauge.
Cor do texto	Define uma cor para os números de escala do Gauge.
Cor da agulha	Define a cor do ponteiro do Gauge.
Posição	Define a posição do Gauge (0°, 90°, 180° e 270°).
Mostrar limites	Mostra os números-limite da escala do Gauge.
Mostrar valor	Mostra o valor nominal indicado no Gauge.
Casas decimais	Define o número de casas decimais para o valor nominal do Gauge.
Total ticks	Define o número de divisões na escala gráfica usada no Gauge.
Ângulo inicial	Define o ângulo inicial para a agulha do Gauge.
Fontes...	Define fonte, cor e tamanho para o texto do Gauge.
Usar ponteiro grosso	Define uma agulha mais grossa para o Gauge.

Propriedades Avançadas do Gauge

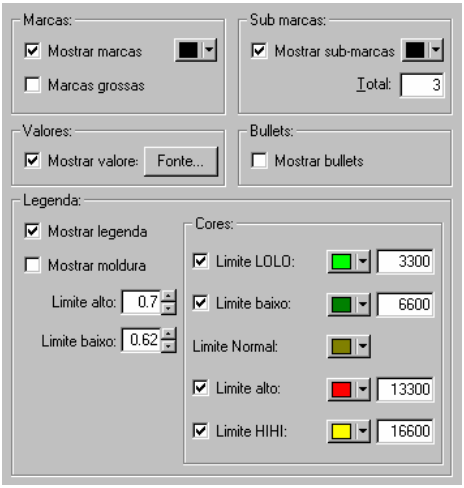


Figura 71: Propriedades Avançadas do Gauge

Propriedades Avançadas do Gauge

OPÇÃO	DESCRIÇÃO
Mostrar marcas	Exibe as marcas principais no objeto Gauge.
Marcas grossas	Determina que as marcas principais sejam grossas.
Mostrar sub-marcas	Exibe sub marcas no objeto Gauge.
Total	Define o número total de sub-marcas a serem exibidas entre as marcas principais.
Mostrar valores	Exibe os valores numéricos no objeto Gauge.
Mostrar bullets	Mostra as marcas em forma de bullets.
Fonte	Permite definir a fonte dos valores do Gauge.
Mostrar legenda	Exibe uma barra ao longo do Gauge onde podem ser configuradas diferentes cores dependendo da faixa de valores.
Mostrar moldura	Mostra um frame ao longo do percurso da agulha.
Limite alto	Define a altura da borda superior da legenda.
Limite baixo	Define a altura da borda inferior da legenda.
Cores	Define os valores e cores para os limites: LowLow, Low, Normal, High e HighHigh a serem mostrados na legenda.

6.6. Objetos de Interação

6.6.1. Slider

Este objeto é usado para ler ou escrever valores em um Tag selecionado. Você pode atribuir valores ao Tag selecionado deslizando o potenciômetro (botão deslizante) ou usando as setas de direção nas extremidades do Slider. O valor vai variar conforme uma escala definida no objeto. Você pode editar as propriedades do Slider dando um duplo clique sobre o mesmo.

Figura 72: Slider

Propriedades Gerais do Slider

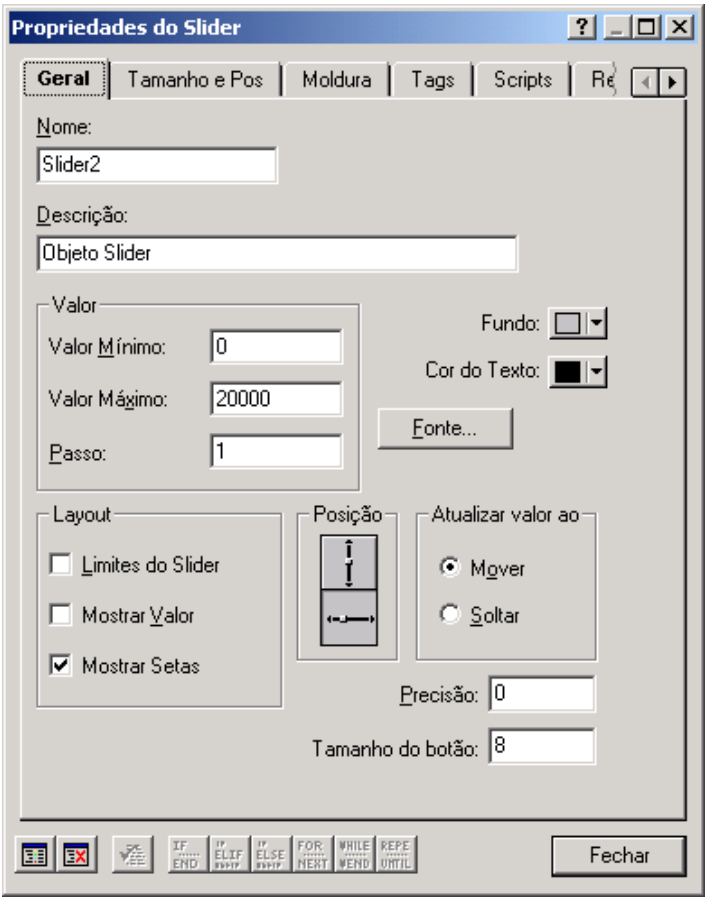


Figura 73: Propriedades Gerais do Slider

Propriedades Gerais do Slider

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do Slider.
Descrição	Uma breve descrição sobre o Slider.
Valor Mínimo	Define o valor mínimo para o Slider.
Valor Máximo	Define o valor máximo para o Slider.
Passo	Define o passo de variação do valor do Slider (Tag associado) quando as setas de direção são pressionadas.
Fundo	Define a cor de fundo do Slider.
Cor do texto	Define a cor dos números que serão mostrados no Slider.
Fontes...	Permite configurar as fontes utilizadas no slider.
Limite do Slider	Habilita ou desabilita que os limites (máximos e mínimos) do Slider sejam mostrados no objeto.
Mostrar valor	Habilita ou dasabilita que os valores do Slider sejam mostrados.
Mostrar setas	Habilita ou dasabilita que as setas do Slider sejam mostradas.
Posição vertical	Posiciona o Slider verticalmente.
Posição horizontal	Posiciona o Slider horizontalmente.
Mover	Permite a atualização do valor do Slider mesmo quando o botão está sendo movido.
Soltar	Atualiza o valor do Slider somente quando o botão deslizante é solto.
Precisão	Define o número de dígitos decimais a serem mostrados no Slider.
Tamanho do botão	Modifica o tamanho do botão deslizante do Slider.

6.6.2. Botão

Este objeto é utilizado para acionamentos ou execuções de tarefas especificadas pelo usuário através do mouse ou teclado e seu funcionamento é igual aos dos botões standard do Windows. O Elipse SCADA possui uma série de modelos de botão entre interruptores, chaves e outros, para facilitar a implementação de várias funções através de botões. O pressionamento de botões também gera eventos que podem ser tratados por Scripts. Os botões são bastante flexíveis e podem ser amplamente configurados através das páginas de propriedades.

Figura 74: Botão

Propriedades Gerais do Botão

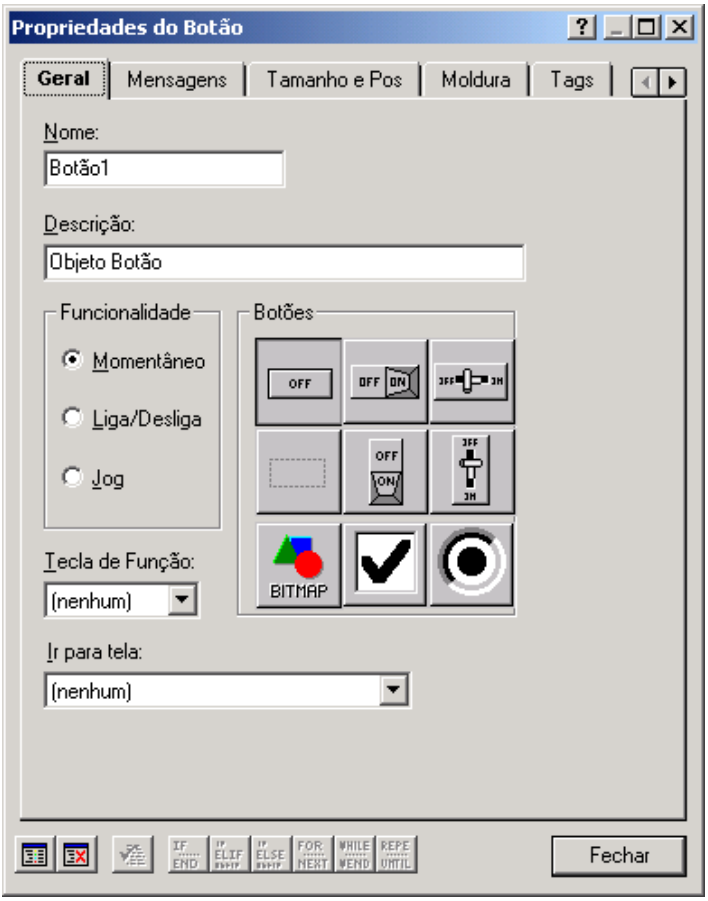


Figura 75: Propriedades do Botão

Propriedades do Botão

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do Botão que será usado na árvore do Organizer e nos Scripts.
Descrição	Uma breve descrição sobre o Botão.
Momentâneo	Este tipo de botão funciona como um botão de OK de uma caixa de diálogo e é usado para atribuir um valor ao Tag associado ou trocar entre telas.
Liga/Desliga	Este tipo de botão funciona como um chave. Ele possui dois estados (ligado e desligado) que podem ser definidos na página Mensagens.
Jog	Este botão alterna entre dois valores: um quando está pressionado e outro quando é solto. Ele funciona como o botão de reset do computador.
Botões	Conjunto de modelos que definem a aparência do botão. Estão disponíveis chaves, interruptores e bitmaps, além de um quadro transparente que permite transformar em um botão qualquer objeto sob ele.
Teclas de função	Associa uma tecla de função ao botão.
Ir para a tela	Define uma tela para ser chamada quando o botão é pressionado.

Mensagem do Botão

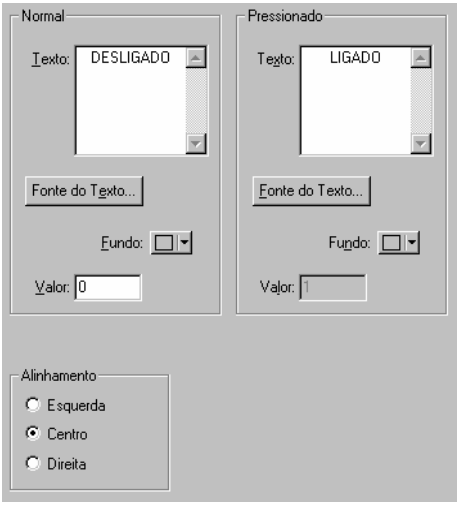


Figura 76: Propriedades das Mensagens do Botão

Propriedades da Mensagem do Botão – Estado Normal

OPÇÃO	DESCRIÇÃO
Texto	Define um texto a ser mostrado no botão quando ele está despressionado.
Fonte do texto	Define a cor, tamanho e tipo da fonte do texto quando o botão está despressionado.
Fundo	Define uma cor de fundo para o botão quando ele está despressionado.
Valor	Define o valor a ser atribuído ao tag associado, quando o botão está despressionado.

Propriedades da Mensagem do Botão – Estado Pressionado

OPÇÃO	DESCRIÇÃO
Texto	Define um texto a ser mostrado no botão quando ele está pressionado.
Fonte do texto	Define a cor, tamanho e tipo da fonte do texto quando o botão está pressionado.
Fundo	Define uma cor de fundo para o botão quando ele está pressionado.
Valor	Define o valor a ser atribuído ao Tag associado, quando o botão está pressionado.
Alinhamento	Alinha o texto do botão à esquerda, ao centro ou à direita.

Scripts de Botões

Os scripts disponíveis exclusivamente para botões são descritos na tabela abaixo. Maiores detalhes a respeito do uso de scripts veja no capítulo específico.

Scripts Disponíveis

Você pode associar scripts a botões executando-os em uma das situações a seguir:	
EVENTOS	**DESCRIÇÃO**
OnPress	Executa o script quando o botão é pressionado.
OnRelease	Executa o script quando o botão é solto.
WhilePressed	Executa o script enquanto o botão está sendo pressionado, conforme o ciclo (em ms) definido.

6.6.3. Setpoint

O objeto **Setpoint** trabalha como uma caixa de edição do Windows, assim basta digitar um valor e pressionar [Enter] para atribuir este valor ao tag associado. As propriedades do setpoint permitem que você defina o tipo do valor de entrada, a fonte, tamanho e cor dos caracteres que serão mostrados no objeto. Você pode editar as propriedades do Setpoint dando um duplo clique sobre o mesmo.

Figura 77: Setpoint

Propriedades Gerais do Setpoint

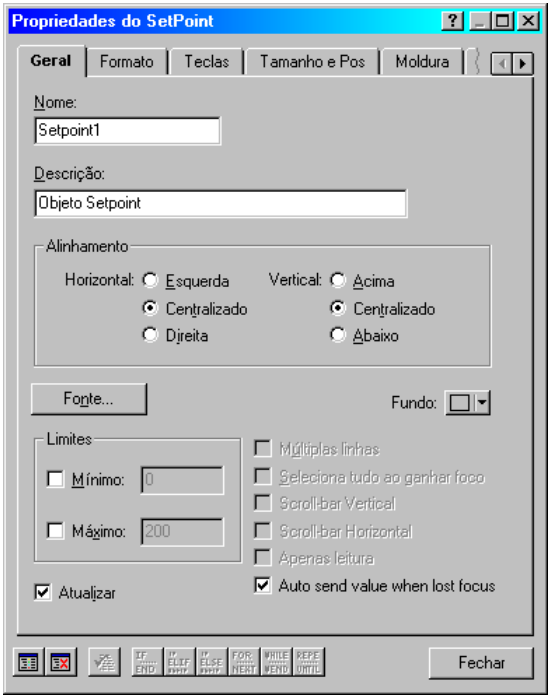


Figura 78: Propriedades Gerais do Setpoint

Propriedades Gerais do Setpoint

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do Setpoint que será usado na árvore do Organizer e nos Scripts.
Descrição	Uma breve descrição sobre o Setpoint.
Alinhamento	Define o alinhamento dos caracteres do Setpoint. No CE, o alinhamento vertical é sempre centralizado.
Fundo	Define uma cor de fundo para o Setpoint.
Limites	Define os limites para os dados de entrada do Setpoint.
Fontes	Define a fonte, cor e tamanho do texto do Setpoint.
Atualizar	Atualiza o valor do Setpoint sempre que o valor do Tag mudar.
Linhas múltiplas	Define o Setpoint como tendo múltiplas linhas. Somente disponível se o formato do Setpoint (página de Formato) é texto.
Seleciona tudo ao ganhar foco	Seleciona todos os caracteres do Setpoint quando ele recebe o foco.

Formato do Setpoint

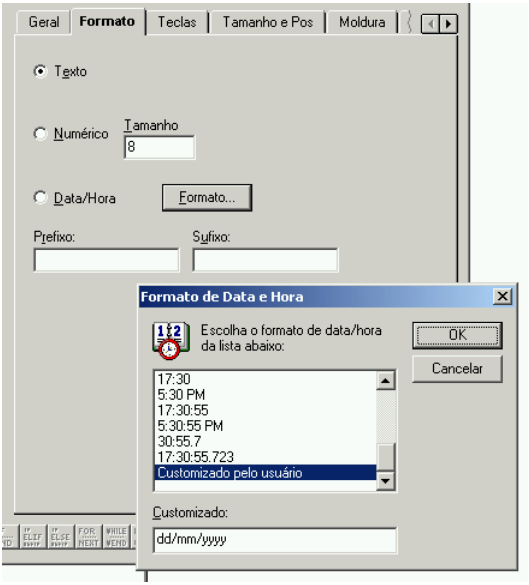


Figura 79: Propriedades do Formato do Setpoint

Propriedades do Formato do Setpoint

OPÇÃO	DESCRIÇÃO
Texto	Mostra o valor do Setpoint em formato string.
Numerico	Mostra o valor do Setpoint em formato numérico.
Tamanho	Define o número de dígitos a serem mostrados incluindo o ponto decimal.
Data/Hora	Mostra o valor do Setpoint em formato data e hora.
Formato...	Permite definir um string que descreve o formato data e hora.
Prefixo	Adiciona um prefixo tipo string ao valor mostrado.
Sufixo	Adiciona um sufixo tipo string ao valor mostrado.

Teclas do Setpoint

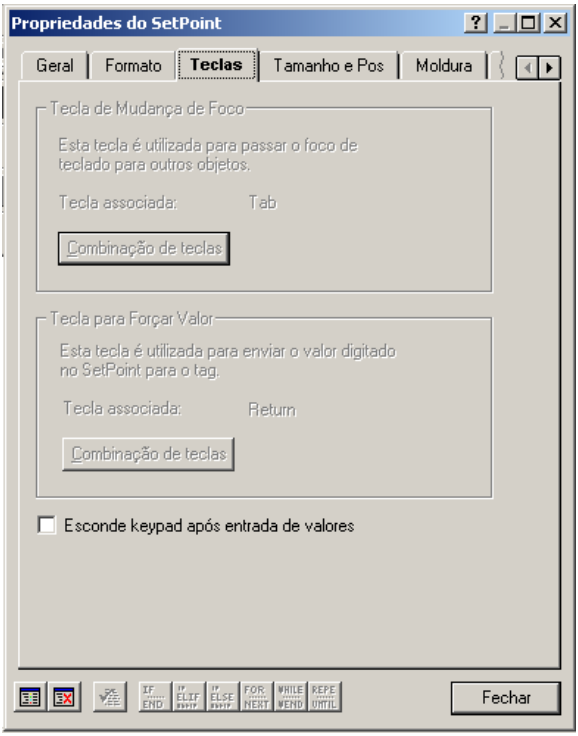


Figura 80: Propriedades das Teclas do Setpoint

Propriedades das Teclas do Setpoint

OPÇÃO	DESCRIÇÃO
Tecla de mudança de foco	Permite a seleção de uma combinação de teclas ou uma tecla para alternar o foco entre as entradas de dados dos Objetos de Tela. Substitui a tecla [Tab] quando o Setpoint é multilinhas.
Teclas para forçar valor	Permite a seleção de uma combinação de teclas ou uma tecla para enviar os dados digitados do Setpoint para o Tag selecionado. Substitui a tecla [Enter] quando o Setpoint é multilinhas.
Esconde keypad após entrada de valores	Indica que o keypad irá desaparecer depois da entrada de valores pelo usuário. O keypad nunca fica escondido no CE.

6.6.4. Alarmes

O objeto **Alarme** permite a verificação dos alarmes ativos ou dos alarmes registrados ("logados") no arquivo de alarmes. Alarmes podem ser disparados quando os valores dos tags associados são verificados em quatro situações e prioridades diferentes: LoLo (baixo crítico), Low (baixo), High (alto), HiHi (alto crítico). Fora dessas faixas, os valores dos tags são considerados normais.

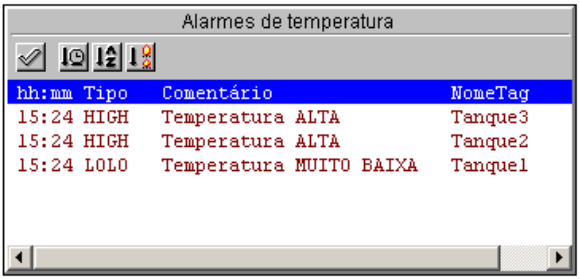


Figura 81: Janela do objeto Alarme

Na janela de alarmes, encontramos uma barra de ferramentas que permite algumas funções, a saber:


Reconhece os alarmes selecionados com o mouse


Lista os alarmes pela data


Lista os alarmes pelo nome, usando a data como chave secundária

Lista os alarmes por prioridade (iniciando pela prioridade 1), usando a data como chave secundária

Propriedades Gerais dos Alarmes

As propriedades dos alarmes permitem definir o formato da data, hora e mensagens, bem como as cores para cada situação de alarme. Você pode editar as propriedades dando um duplo clique sobre o objeto **Alarmes**.

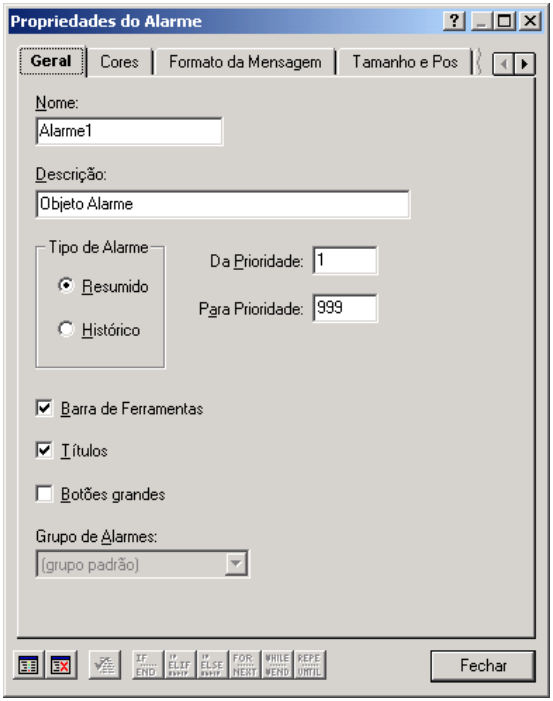


Figura 82: Propriedades Gerais dos Alarmes

Propriedades Gerais dos Alarmes

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do objeto Alarmes.
Descrição	Uma breve descrição sobre o objeto Alarmes.

OPÇÃO	DESCRIÇÃO
Tipo de alarme	Resumido: mostra os alarmes ativos, ou seja, Tags que estão em situação de alarme no momento. Histórico: mostra os alarmes que estão registrados no arquivo de alarmes. Obs: a opção Habilita Reg. no objeto Alarmes no Organizar deve estar habilitada.
Da Prioridade	Define a prioridade mais alta de alarmes a serem mostrados no objeto.
Para Prioridade	Define a prioridade mais baixa de alarmes a serem mostrados no objeto.
Barra de Ferramentas	Mostra ou esconde a barra de ferramentas do objeto Alarme.
Titulos	Habilita uma barra de título no objeto Alarme, mostrando uma linha de cabeçalho com selecionados na página Formato da Mensagem.
Botões grandes	Habilita botões grandes na barra de ferramentas do objeto de Alarmes.
Grupo de alarmes	Seleciona o grupo de alarmes que será mostrado no objeto de Alarmes corrente.

Cores dos Alarmes

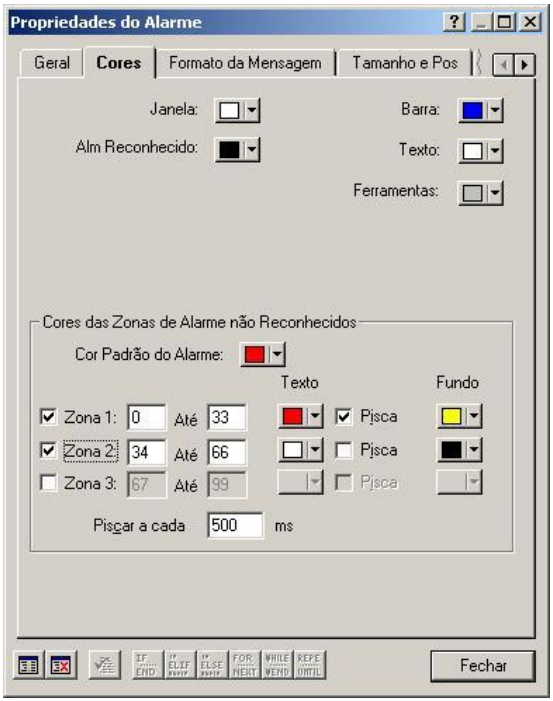


Figura 83: Propriedades das Cores do Alarme

Propriedades das Cores do Alarme

OPÇÃO	DESCRIÇÃO
Janela	Cor de fundo da janela de alarmes.
Alm Reconhecido	Cor para alarmes reconhecidos.
Barra	Cor da barra de títulos das colunas.
Texto	Cor dos títulos das colunas.
Ferramentas	Cor da barra de ferramentas.
Cor padrão do alarme	Cor de fundo para os alarmes não-reconhecidos.
Zona 1	Permite definir o intervalo de prioridades para a Zona 1, cores do texto e fundo do alarme, e se a linha deve piscar ou não.
Zona 2	Permite definir o intervalo de prioridades para a Zona 2, cores do texto e fundo do alarme, e se a linha deve piscar ou não.
Zona 3	Permite definir o intervalo de prioridades para a Zona 3, cores do texto e fundo do alarme, e se a linha deve piscar ou não.
Piscar a cada ... ms	Permite especificar a freqüência da alternância de cores dos alarmes.

Formato das Mensagens dos Alarmes

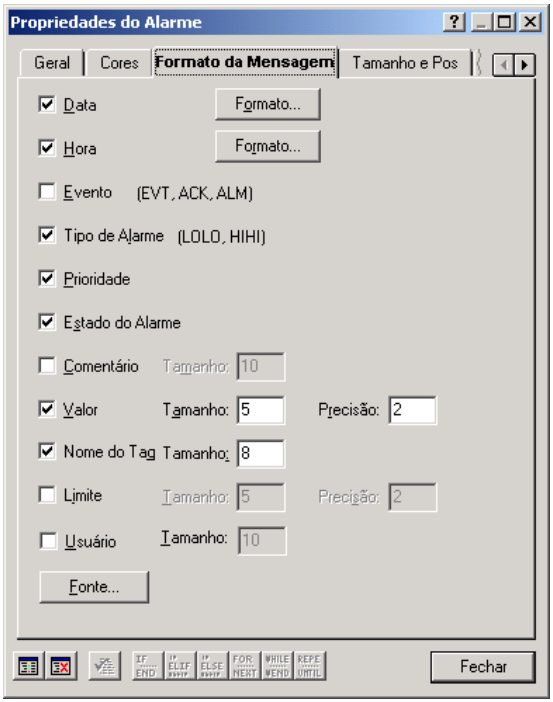


Figura 84: Propriedades das Mensagens dos Alarmes

Propriedades das Mensagens dos Alarmes

OPÇÃO	DESCRIÇÃO
Data	Habilita a visualização da data no objeto Alarme, conforme o formato especificado na opção Formato.
Data/Formato	Define o formato da data a ser mostrada no objeto.
Hora	Habilita a visualização da hora no objeto Alarme, conforme o formato especificado na opção Formato.
Hora/Formato	Define o formato da hora a ser mostrada no objeto.
AM/PM	Habilita o formato de hora americano.
Mostrar milisegundos	Mostra os milisegundos na coluna de hora do objeto Alarme.
Evento	Habilita a visualização da coluna de eventos no objeto.
Tipo do alarme	Habilita a visualização da coluna do tipo do alarme (lolo, low, high, hihi) no objeto Alarme.
Prioridade	Habilita a visualização da coluna de prioridades de alarmes (lolo, low, high, hihi) no objeto Alarme.
Estado do Alarme	Habilita a visualização da coluna de status (UnAck, Ack) dos alarmes no objeto Alarme.
Comentários	Habilita a visualização dos comentários definidos para os Tags no objeto Alarme.
Comentários/Tamanho	Define o tamanho dos comentários em números de caracteres.
Valor	Habilita a visualização da coluna de valores dos tags no objeto Alarme. Os valores serão mostrados variando para alarmes tipo resumidos ou fixos para alarmes tipo histórico.
Valor/Tamanho	Define o tamanho dos valores dos tags em número de caracterers.
Valor/Precisão	Define quantos dígitos do tamanho serão decimais.
Nome do tag	Habilita a visualização da coluna de valores dos Tags no objeto Alarme.
Nome do Tag/Tamanho	Define o tamanho do nome dos tags em números de caracteres.
Limite	Habilita a visualização dos limites definidos para os Alarmes conforme o seu tipo (Low, LoLo, High, HiHi)
Limite/Tamanho	Define o tamanho dos valores limites de cada alarme em número de caracteres.
Limite/Precisão	Define quantos dígitos do tamanho serão decimais.
Usuário	Habilita a visualização do nome do usuário responsável.
Usuário/Tamanho	Define o tamanho do nome do usuário em números de caracteres.

Capítulo

7. Alarmes

A opção Alarmes da árvore da aplicação no Organizer permite que sejam definidas algumas características para o gerenciamento dos Alarmes do sistema. Você pode especificar um arquivo para gravar todos os alarmes que irão ocorrer no seu sistema, bem como configurar sons e mensagens de alerta.

As propriedades do Gerenciador de Alarmes afetam o comportamento de todos os Alarmes do sistema especificados na página de Alarmes dos Tags.

7.1. Propriedades Gerais dos Alarmes

A página de Propriedades Gerais dos Alarmes aparece quando selecionada a aba *Alarmes* no topo das páginas dos Alarmes. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

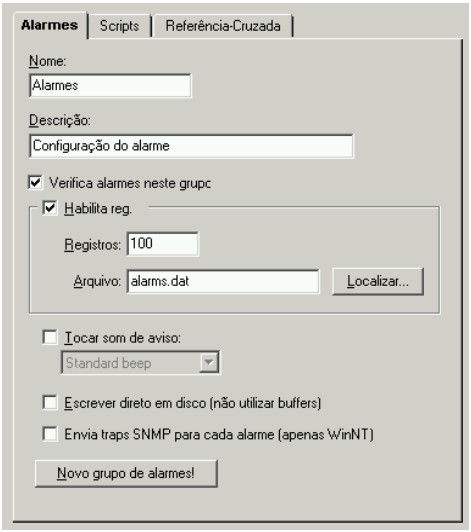


Figura 85: Propriedades Gerais dos Alarmes

Propriedades Gerais dos Alarmes

OPÇÃO	DESCRIÇÃO
Verifica alarmes neste grupo	Permite habilitar ou desabilitar a verificação de alarmes do grupo.
Habilitar reg.	Habilita o registro (log) dos alarmes do grupo atual.
Registros	Define o tamanho do arquivo de Alarmes em número de registros. Cada alarme é um registro que ocupa 42 bytes. O tamanho do cabeçalho é 848 bytes determinando o tamanho inicial do arquivo. A atualização deste arquivo se dá de forma cíclica.
Arquivo	Define o nome do arquivo de Alarmes (extensão DAT)
Localizar	Permite localizar ou definir um diretório para o arquivo de Alarmes.
Tocar som de aviso	Habilita um som de alerta para os Alarmes. O som pode ser escolhido a partir da lista embaixo.
Escrever direto em disco	Força a escrita direta dos dados no disco, sem utilizar buffers. Isso diminui a performance, mas aumenta muito a segurança de dados no caso de falhas.
Envia traps SNMP para cada alarme	(Disponível somente nos sistemas Windows NT.) Faz com que o sistema gere um evento (*trap*) SNMP a cada nova mensagem de alarme.
Novo grupo de alarmes!	Cria um novo grupo de Alarmes. O usuário pode escolher a que grupo de alarmes um Tag ou um objeto de tela Alarmes está associado.

7.2. Scripts dos Alarmes

Os scripts do Gerenciador de Alarmes geralmente estão associados a uma situação de alarme. Isto significa, por exemplo, que eles podem ser executados quando um Alarme ocorrer.

Os scripts disponíveis para os alarmes são descritos na tabela abaixo. Maiores detalhes a respeito veja no scripts.

Scripts Disponíveis

Você pode associar scripts a alarmes executando-os em uma das situações a seguir:

EVENTOS	DESCRIÇÃO
OnAlarm	Executa o script na ocorrência e retorno de um alarme.

Capítulo 8

8. Receitas

Uma **Receita** é um conjunto de valores pré-definidos que podem ser carregados para um grupo de tags a fim de configurar um processo específico. Esta lista de tags também chamamos de **modelo de receita**.

Por exemplo, seja uma máquina que fabrica diferentes tipos de parafusos. As variáveis envolvidas no processo são sempre as mesmas, mas seus valores provavelmente irão mudar dependendo do tipo de parafuso que se quer produzir. Supondo que você tem diferentes configurações de máquina para cada tipo de parafuso, estes valores poderiam ser gravados em uma receita e serem posteriormente carregados em *tags* de controle, facilitando a tarefa do operador e evitando erros.

Dessa maneira, podemos criar um modelo de receita "Parafuso" com diversas receitas "Fenda Philips", "Fenda Torx", "Fenda Simples" e assim por diante.

Para que sejam recuperados quando necessário, os modelos e os dados de uma receita são armazenados em disco, em um "arquivo de receitas" com a extensão .RCP.

Você pode definir uma receita no Organizer durante a configuração da aplicação ou em tempo de execução usando **Funções Especiais** através de scripts.

8.1. Propriedades Gerais da Receita

Cada receita que você cria para a aplicação aparece abaixo da opção Receitas na árvore do Organizer. Ao selecionar uma receita específica, suas propriedades são mostradas ao lado direito da árvore. A seguir, podemos ver as propriedades das receitas.

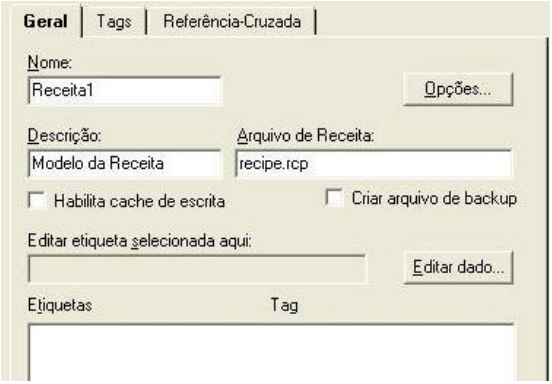


Figura 86: Propriedades Gerais da Receita

Propriedades Gerais da Receita

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do modelo de receita.
Descrição	Uma breve descrição sobre o modelo de receita.
Arquivo Receita	Define o nome do arquivo para o modelo de receita. O nome do arquivo pode ter até 8 caracteres e não deve conter extensão (o Elipse SCADA irá sempre usar a extensão RCP). Você pode especificar também o caminho do arquivo, que poderá ser uma localização absoluta ("C:\ELIPSE\RECIPES\RCP1") ou relativa ("RECIPES\RCP1"). Localizações relativas são recomendadas se você deseja copiar sua aplicação para outro computador.
Habilita cache de escrita	Permite que seja utilizado o cache do Windows para agilizar o processo de escrita do arquivo de receitas.
Criar arquivo de backup	Habilita a criação automática de uma cópia de backup do arquivo .RCP, que contém as informações da receita. Em caso de problemas com o arquivo principal, é possível recuperar as informações anteriores a partir deste backup.
Editar receita selecionada aqui	Permite a edição da etiqueta que identifica o tag selecionado na receita.
Editar Dado...	Abre a janela Editar Receita onde se pode acrescentar e modificar os valores das diversas receitas.
Etiquetas	Mostra os campos associados aos tags do modelo de receita.
Tag	Mostra os tags selecionados para o modelo de receita corrente.

8.2. Editando Receitas

Para acrescentar, editar ou apagar receitas já criadas, utilizamos a janela Editar Receita que é chamada através do botão Editar dado... na página de propriedades de receitas. Todas as receitas criadas utilizando o modelo escolhido são listadas no campo Receitas, onde podem ser selecionados para edição. Selecionando qualquer receita da lista você poderá editar a sua descrição e os valores de cada tag associado.

Veremos os campos dessa janela a seguir:

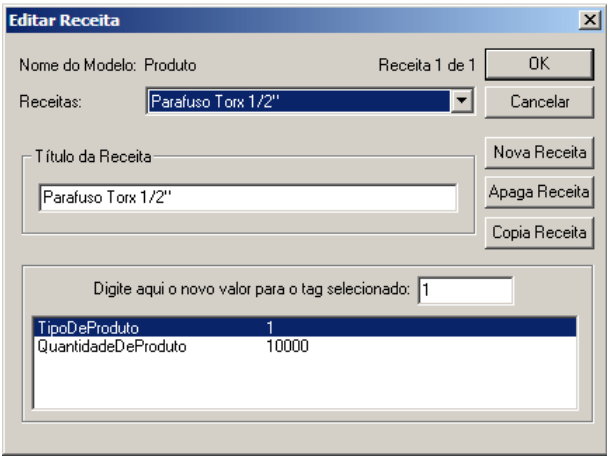


Figura 87: Editando Receitas

Campos da janela Editar Receitas

OPÇÃO	DESCRIÇÃO
Receitas	Permite a seleção de uma receita no modelo corrente.
Título da Receita	Nome ou descrição da receita.
Nova Receita	Cria uma nova receita. Cada modelo de receita pode ter várias receitas (conjunto de valores) relacionadas.
Apaga Receita	Apaga a receita selecionada.
Copia Receita	Permite copiar uma receita já definida, a fim de tornar mais fácil a criação de receitas com muitas variáveis.
Digite aqui o novo valor para o tag selecionado	Permite a edição do valor do tag para a receita corrente, que são listados no quadro inferior. Os tags são identificados por suas etiquetas associadas. Use as setas de direção do teclado ou o mouse para selecionar o tag a ser editado.

Capítulo 9

9. Históricos

Os **Históricos** permitem o armazenamento da variação dos dados de um processo ao longo do tempo, possibilitando análises futuras. Estes Históricos podem ser gerados de duas maneiras diferentes: de forma **Contínua** ou em **Bateladas** (em *batch*).

Na forma contínua, o Elipse SCADA armazena os dados continuamente durante a execução da aplicação. Na forma de bateladas, o histórico é feito por lotes. Nessa modalidade, é necessário enviar um comando via script para iniciar e terminar a geração do histórico.

9.1. Propriedades Gerais dos Históricos

A página de propriedades gerais dos históricos aparecem no lado direito do Organizer, quando clicamos no item Histórico na árvore da aplicação. Esta página é a que segue.

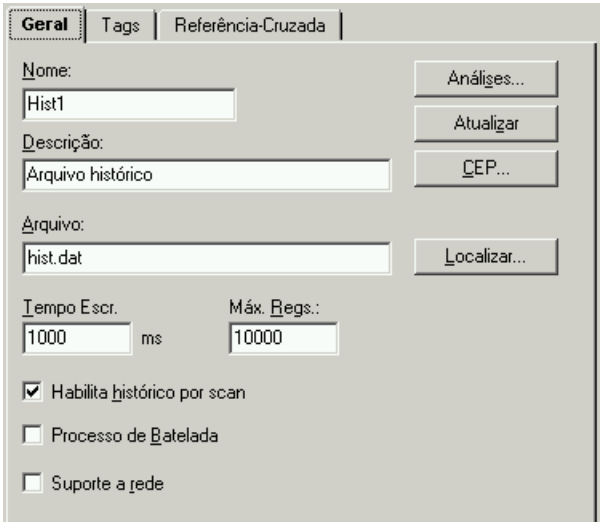


Figura 88: Propriedades Gerais dos Históricos

Propriedades Gerais dos Históricos

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do objeto histórico.
Descrição	Uma breve descrição do objeto.
Análises...	Permite a visualização dos dados do histórico em forma gráfica. Você pode chamar a Análise Histórica em tempo de execução, através da função Analysis() do Histórico.
Atualizar	Recalcula a análise histórica e o CEP para o histórico.
CEP...	Chama a tela do Controle Estatístico de Processos.
Arquivo	Define o nome do arquivo do histórico.
Localizar...	Permite localizar ou definir um diretório para o arquivo de histórico.
Tempo Escr.	Define a freqüência com que os dados serão escritos no arquivo.
Máx. Regs.	Número máximo de registros a serem armazenados no arquivo. O arquivo de histórico é rotativo, ou seja, quando os dados excederem o tamanho do arquivo, os primeiros registros serão substituídos.
Habilita histórico por scan	Habilita a escrita no arquivo de histórico a partir do início da execução da aplicação segundo a taxa de varredura definida. Deixe esta opção desmarcada se você deseja controlar manualmente (usando scripts) a geração dos dados do histórico.
Processo de Batelada	Define o tipo do histórico como sendo por batelada (por lotes). Quando esta opção está marcada, um arquivo de cabeçalho (extensão HDR) é criado com o mesmo nome que o arquivo de histórico. O arquivo de cabeçalho guarda informações sobre cada batelada.
Suporte a rede	Habilita o suporte a rede para o histórico, isto é, permite que o histórico seja acessado (somente para leitura) por outras aplicações Elipse na rede, através de um browser ou relatório do tipo Análise Histórica.

9.2. Análise Histórica

Quando um Histórico é criado, o Elipse SCADA automaticamente associa a ele um objeto **Análise Histórica** (*HAnalysis*). Este objeto possui diversas características que podem ser ajustadas em tempo de execução.

Ao ser chamada a Análise Histórica, uma janela Histórico é aberta em sua aplicação, mostrando a página Análises com o gráfico para análise dos dados. Esta janela possui mais seis páginas de configuração (disponíveis de acordo com a programação

da aplicação). São elas: Gráfico, Penas, Cores das Penas, Configurações, Consultas e Impressão. Cada uma destas páginas aparece quando selecionada a sua guia correspondente do topo da janela. Vejamos, a seguir, uma descrição da cada uma.

Análise

A página de **Análise** mostra o gráfico da Análise Histórica e permite a configuração das variáveis a serem mostradas no mesmo. Possui recursos de zoom, rolagem e impressão, que podem ser acessados por meio da barra de ferramentas que se localiza na parte inferior da janela.

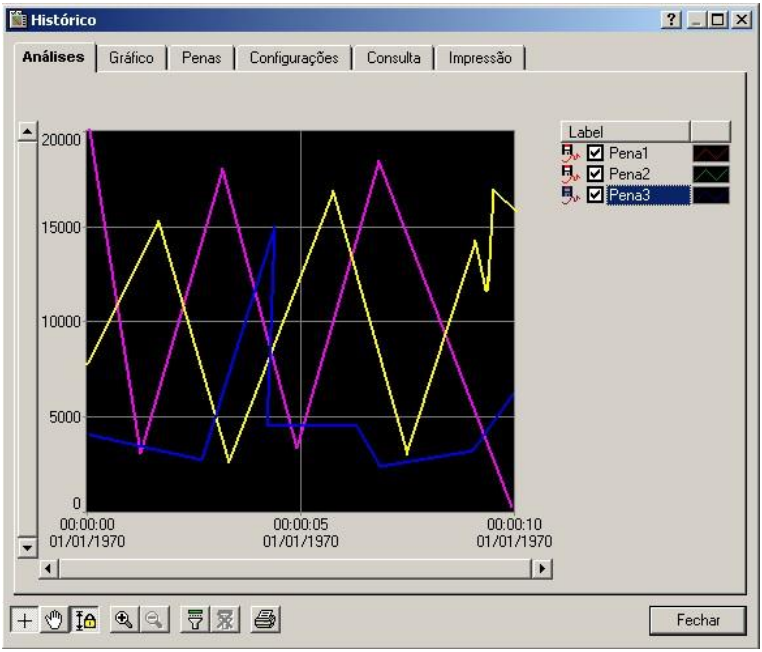


Figura 89: Página de análise

À direita, podemos ver as penas do gráfico, cada uma delas associada a um tag, o qual se quer observar a evolução histórica. Estas penas podem ser ativadas ou desativadas e pode-se mudar o tipo de cor e estilo de linha, de acordo com a vontade do usuário.
Na parte superior, o Elipse SCADA mostra algumas informações quando o mouse está posicionado dentro da área do gráfico: coordenadas do mouse e valor do ponto selecionado no gráfico demarcado pelas linhas (horizontal e vertical) tracejadas. Para selecionar um ponto do gráfico basta clicar sobre o mesmo.

Gráfico

Esta página permite a configuração da aparência do Gráfico da Análise Histórica, de acordo com o que segue:

Figura 90: Página de configuração do gráfico

Opções para o eixo X (horizontal)

OPÇÃO	DESCRIÇÃO
Limite esquerdo	Limite esquerdo do eixo X. Disponível se o gráfico é do tipo XY.
Limite direito	Limite direito do eixo X. Disponível se o gráfico é do tipo XY.
Exibir régua superior	Exibe uma régua para o eixo X na parte superior do gráfico.
Exibir régua inferior	Exibe uma régua para o eixo X na parte inferior do gráfico.

Opções para o eixo Y (vertical)

OPÇÃO	DESCRIÇÃO
Limite superior	Limite superior do eixo Y.
Limite inferior	Limite inferior do eixo Y.
Exibir régua à esquerda	Exibe uma régua a esquerda do gráfico, com os valores do eixo Y.
Exibir régua à direita	Exibe uma régua a direita do gráfico, com os valores do eixo Y.

Opções comuns

OPÇÃO	DESCRIÇÃO
Ajuste automático	Calcula automaticamente os limites para os eixos X e/ou Y.
Grade Rolante	Faz com que a grade de fundo se movimente à medida que os valores do gráfico são atualizados.
Texto	Permite acrescentar um título para as réguas.
Formato Data/Hora	Ajusta o formato de data e hora quando configurados para o eixo.
Cores/Fundo da régua	Define a cor de fundo para as réguas.
Cores/Texto da régua	Define a cor para o texto que será mostrado nas réguas.
Cores/Fundo do gráfico	Define a cor de fundo para o gráfico.
Cores/Grades e eixos	Define a cor da grade e dos eixos do gráfico.
Grade/Linhas de X	Define o número de linhas horizontais da grade.
Grade/Linhas de Y	Define o número de linhas verticais da grade.
Fonte da régua	Define fonte, tamanho e cor para os caracteres do gráfico.

Penas

Esta página permite determinar os tags que serão mostrados no gráfico, associando a cada um, uma "pena" que irá descrever a evolução da variável no processo.

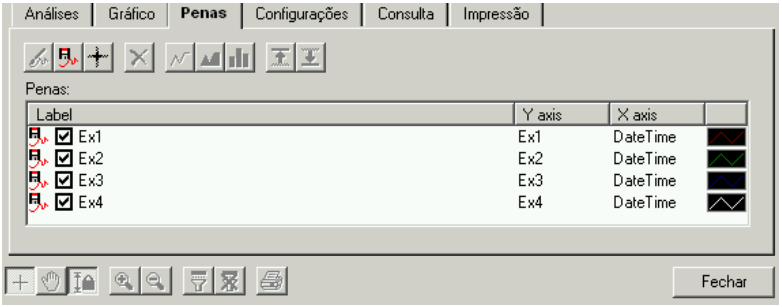


Figura 91: Guia de configuração das penas

No quadro central, são listadas as penas que serão usadas no gráfico. Pode-se ajustar quais estarão ativas. Clicando-se no retângulo preto mais à direita, é possível ajustar as características da representação gráfica da pena, tais como espessura da linha e cor. Os outros campos são descritos abaixo:

Opções de configuração das penas

OPÇÃO	DESCRIÇÃO
Label	Nome ou descrição da pena. Por padrão, o nome do tag é sugerido, mas pode ser mudado.
Y axis	Variável a ser usada na coordenada Y.
X axis	Variável a ser usada na coordenada X.

Através dos botões da barra de ferramentas, posicionada na parte superior da janela, é possível:

 acrescentar mais uma Pena

 subir a pena de posição

 acrescentar uma Marca

 descer a pena de posição

 apagar uma Pena ou Marca

 ajustar a representação em linha para o *tag* selecionado

 ajustar a representação em preenchimento para o *tag* selecionado

 ajustar a representação em barra para o *tag* selecionado

Configurações

Esta página permite especificar o arquivo de dados a ser usado na análise. Se a opção Processo de Batelada estiver marcada (ver propriedades gerais do Histórico), o grupo Batelada nesta página estará disponível para a escolha da batelada.

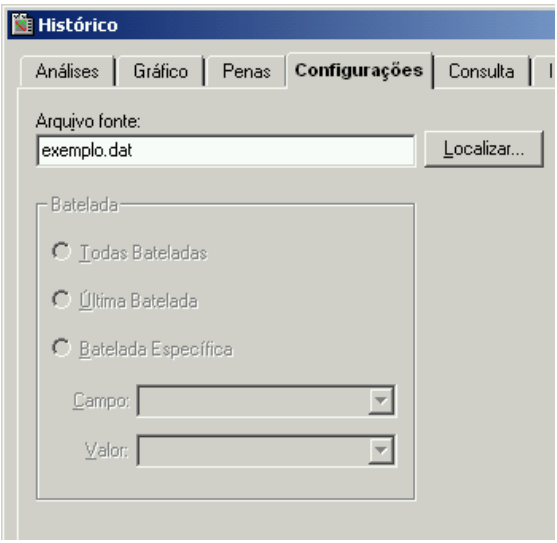


Figura 92: Guia de configuração do arquivo de histórico

Campos de configuração do arquivo de histórico

OPÇÃO	DESCRIÇÃO
Nome do arquivo	Define o nome do arquivo fonte para Análise Histórica e CEP (.DAT).
Todas Bateladas	Seleciona a impressão de todas as bateladas do arquivo.
Última batelada	Seleciona para ser impressa a última batelada registrada.
Batelada Específica	Seleciona uma batelada específica para ser impressa, conforme o especificado nos campos Campo e Valor.

Consulta

Esta página permite criar uma "consulta" (*query*) para o arquivo de histórico. A partir dessa consulta é possível definir um intervalo de tempo para limitar os dados com os quais se quer trabalhar.

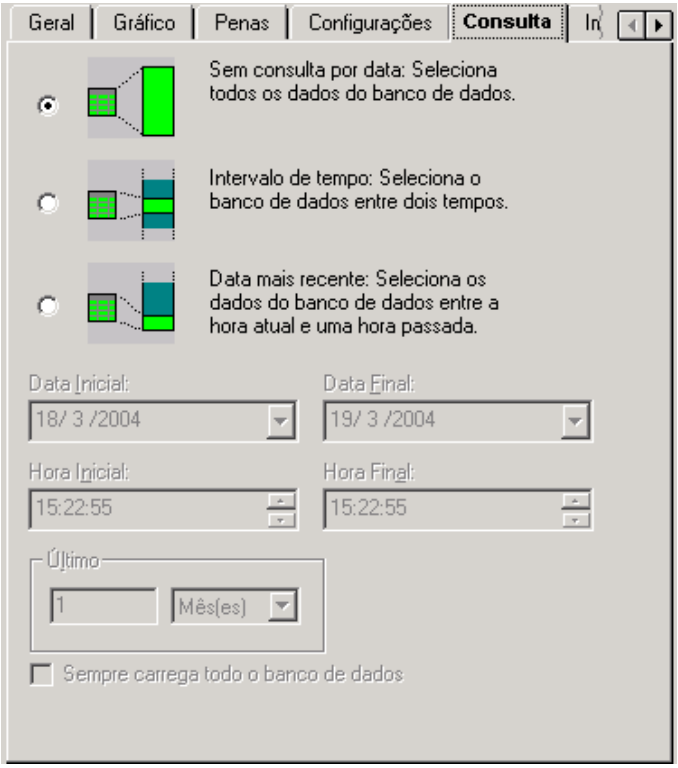


Figura 93: Guia de configuração da consulta no histórico

Campos de configuração da consulta no histórico

OPÇÃO	DESCRIÇÃO
Sem consulta por data	Não será usado filtro, ou seja, seleciona todos os dados.
Intervalo de Tempo	Seleciona os dados dentro de um intervalo de tempo especificado.
Dados mais recentes	Seleciona os dados a partir de um tempo passado até agora.
Data inicial	Determina a data inicial do intervalo de tempo.
Hora inicial	Determina a hora inicial do intervalo de tempo.
Data final	Determina a data final do intervalo de tempo.
Hora final	Determina a hora final do intervalo de tempo.
Último (valor)	Define o número de unidades de tempo dos dados mais recentes.
Último (unidade)	Define a unidade de tempo dos dados mais recentes.
Sempre carrega todo o banco de dados	Carrega todos os dados do histórico. Os limites de visualização são restringidos conforme o configurado pelo usuário.

Impressão

Esta página permite a configuração de opções para a impressão da análise histórica. O desenvolvedor da aplicação poderá gerar arquivos de configuração de impressora (arquivos .PTR) que são carregados pelo usuário em tempo de execução.

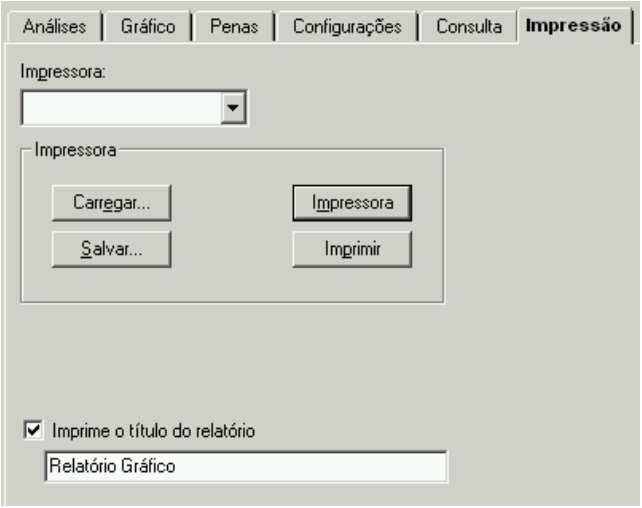


Figura 94: Configurações de impressão

Configurações de impressões

OPÇÃO	DESCRIÇÃO
Caixa Impressora	Permite a escolha da impressora onde se quer imprimir o relatório.
Botão Carregar	Permite carregar arquivos de configuração.
Botão Salvar	Permite salvar uma configuração específica.
Botão Impressora	Chama a configuração da impressora corrente.
Imprimir	Comanda a impressão da Análise Histórica.
Imprimir o título do relatório	Permite especificar um título para o relatório (no campo embaixo).

9.3. Controle Estatístico de Processos

O **Controle Estatístico de Processos** (ou CEP) permite uma análise estatística de um processo monitorado. É uma poderosa ferramenta para que você possa verificar se o processo está executando de acordo com as suas necessidades e controlar suas variáveis para obter um melhor resultado. Assim como a Análise Histórica, o CEP também está associado a um objeto Histórico e é possível habilitar a sua configuração em tempo de execução.

Quando o CEP é chamado, uma Janela de Propriedade é aberta. Através dela, é possível configurar as opções do CEP, que possui 7 páginas: Configurar, Cores das Penas , Gráfico de Média, Gráfico de Dispersão, Gráfico de Histograma, Configurações e Consulta.

As notações a seguir serão usadas na explicação das fórmulas do CEP:

- k : número de amostras;
- $\overline{x}_i$: média aritmética dos valores da amostra *i*, onde $i = 1, 2, 3, ..., k$
- n : tamanho de cada amostra;
- x_{ij} : valor *j* da amostra *i*, onde $j = 1, 2, 3, ..., n$

Configurar

Esta é a página principal do CEP e permite a configuração de suas propriedades gerais. Nesta tela também encontramos botões para chamar as telas auxiliares do CEP, que permitem visualizar gráficos e resultados da análise estatística.

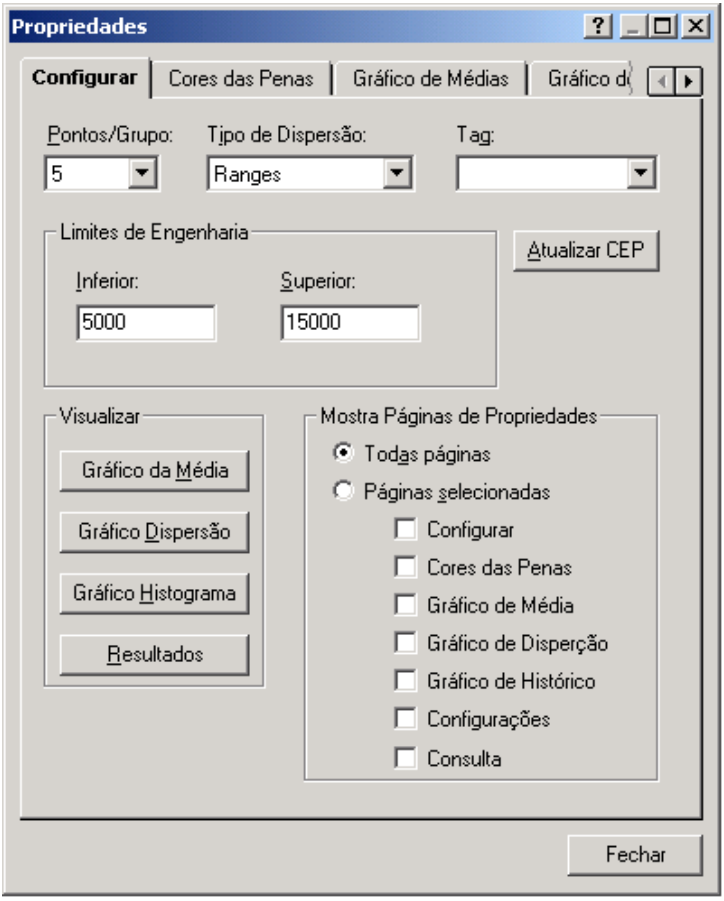


Figura 95: Propriedades do CEP

Configuração das propriedades do CEP

OPÇÃO	DESCRIÇÃO
Pontos/Grupo	Define o número de itens para cada amostra
Tipo de dispersão	Define o método a ser usado: Sigma ou Range
Tag	Define qual o *Tag* do Histórico que será analisado
Limites de Engenharia/Inferior	Define o limite inferior de Engenharia ou Especificação do intervalo de tolerância para os dados das amostras do CEP
Limites de Engenharia/Superior	Define o limite superior de Engenharia ou Especificação do intervalo de tolerância para os dados das amostras do CEP

Após a alteração de algumas destas características, é necessário apertar o botão Atualizar CEP para o sistema refazer os cálculos.

Cores das Penas

Esta página permite selecionar as cores para cada pena (*tag*) do Histórico e as demais linhas de controle, que serão mostradas nos gráficos do CEP, a saber: média, limite superior/inferior de controle, limite superior/inferior de engenharia, curva normal e cor da barra.

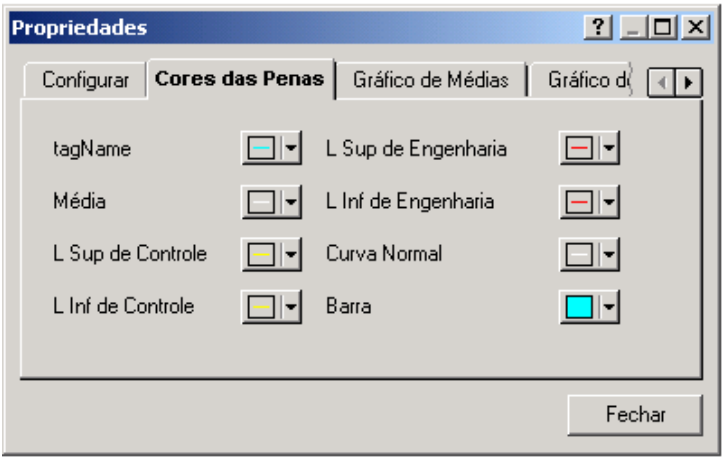


Figura 96: Configurações das Cores das Penas

Gráficos de Médias

Esta página permite configurar a aparência do gráfico de médias do CEP. Seus controles são descritos a seguir.

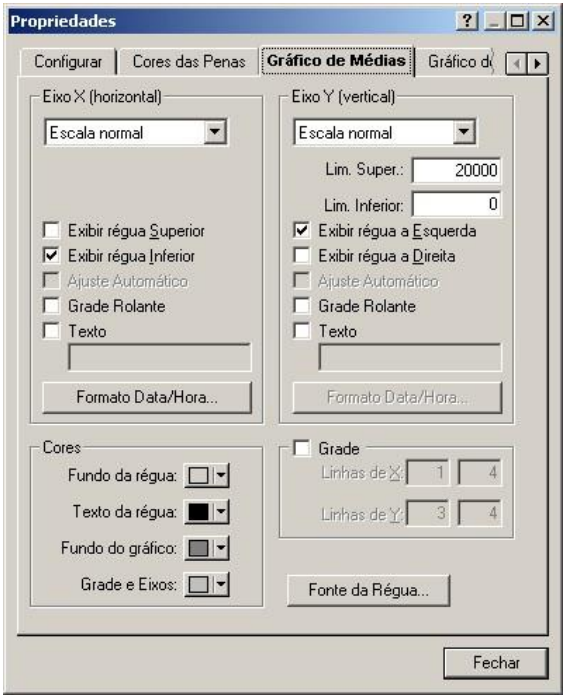


Figura 97: Configurações do Gráfico de Média

Opções para o eixo X (horizontal)

OPÇÃO	DESCRIÇÃO
Limite esquerdo	Limite esquerdo do eixo X. Disponível se o gráfico é do tipo XY.
Limite direito	Limite direito do eixo X. Disponível se o gráfico é do tipo XY.
Exibir régua superior	Exibe uma régua para o eixo X na parte superior do gráfico.
Exibir régua inferior	Exibe uma régua para o eixo X na parte inferior do gráfico.

Opções para o eixo Y (vertical)

OPÇÃO	DESCRIÇÃO
Limite superior	Limite superior do eixo Y.
Limite inferior	Limite inferior do eixo Y.
Exibir régua à esquerda	Exibe uma régua a esquerda do gráfico, com os valores do eixo Y.
Exibir régua à direita	Exibe uma régua a direita do gráfico, com os valores do eixo Y.

Opções comuns

OPÇÃO	DESCRIÇÃO
Ajuste automático	Calcula automaticamente os limites para os eixos X e/ou Y.
Grade Rolante	Faz com que a grade de fundo se movimente à medida que os valores do gráfico são atualizados.
Texto	Permite acrescentar um título para as réguas.
Formato Data/Hora	Ajusta o formato de data e hora quando configurados para o eixo.
Cores/Fundo da régua	Define a cor de fundo para as réguas.
Cores/Texto da régua	Define a cor para o texto que será mostrado nas réguas.
Cores/Fundo do gráfico	Define a cor de fundo para o gráfico.
Cores/Grades e eixos	Define a cor da grade e dos eixos do gráfico.
Grade/Linhas de X	Define o número de linhas horizontais da grade.
Grade/Linhas de Y	Define o número de linhas verticais da grade.
Fonte da régua	Define fonte, tamanho e cor para os caracteres do gráfico.

Métodos de Cálculo

O gráfico da média é calculado da maneira seguinte. Para cada amostra *i* (1, 2, ..., k), é calculada a média $\overline{X}_i$:

$$\overline{X}_i = \frac{1}{n}\left(x_{11} + x_{12} + x_{13} + ... + x_{1n} \right)$$

Uma linha central representada por $\overline{\overline{X}}$, é a média das médias de cada amostra. Você pode obter o valor do desvio padrão das médias $\sigma_{\overline{X}}$ usando um de dois métodos:

a) Método Sigma

Para cada amostra i (i = 1, 2, ..., k) calcula-se:

$$S_i = \left[\left(\sum_{j=1}^{n} x_j^2 - n.\bar{x}^2 \right) \Big/ n \right]^{1/2}$$

A seguir, calcula-se o desvio médio:

$$\bar{S} = \frac{1}{k}\left(S_1 + S_2 + ... + S_k\right)$$

Utiliza-se, para a estimativa do desvio padrão, um fator de correção, anotado por c_2. Esta constante varia conforme *n* e, como você pode ver na tabela do CEP, quando n é maior que 25, $c_2 = 1$, o que significa que não há mais correção a fazer.

O desvio padrão será, então:

$$\sigma_{\bar{x}} = \frac{\bar{S}}{C_2}$$

Recomenda-se utilizar este método para amostras grandes, com *n* maior que 10.

b) Método Range

Para cada amostra i (i = 1, 2, ..., k), calcula-se:

$$R_i = \left(X_{\max} - X_{\min} \right)$$

A seguir calcula-se a amplitude média:

$$\overline{R} = \frac{1}{k}\left(R_1 + R_2 + ... + R_k \right)$$

Utiliza-se, para a estimativa do desvio padrão, um fator de correção, anotado por d_2. Esta constante varia conforme *n* e, como você pode ver na tabela do CEP, quanto maior o valor de *n* maior a correção a ser aplicada. O desvio padrão será, então:

$$\sigma_{\bar{x}} = \frac{\overline{R}}{D_2}$$

Recomenda-se utilizar este método para amostras pequenas, com *n* menor ou igual a 10 (*n*<=10).

O gráfico terá, então, os seguintes pontos no eixo *X*.

XDB: Linha central = $\overline{\overline{X}}$

LSC: Limite Superior de Controle

$$LSC = \overline{X} + a_2 \cdot \overline{R} ;$$

LIC: Limite Inferior de Controle

$$LIC = \overline{X} - a_2 \cdot \overline{R} ;$$

LSE: Limite Superior de Engenharia;

LIE: Limite Inferior de Engenharia;

Gráfico de Dispersão

Esta página permite configurar a aparência do gráfico de dispersão do CEP. Ela tem os mesmos controles e funcionamento do gráfico das Médias.

Métodos de Cálculo

a) Método Sigma

A linha central do gráfico é representada por $\overline{S}$ e calculada como segue:

$$\overline{S} = \frac{1}{k}\left(S_1 + S_2 + ... + S_k\right)$$

Utiliza-se, para o cálculo dos limites de controle, duas constantes c_3 e c_4 (ver tabela SPC). Desta forma calcula-se os limites:

$$\text{LSC} = \overline{S}\left(c_4\right)$$

$$\text{LIC} = \overline{S}\left(c_3\right)$$

Este gráfico é recomendado para amostras grandes com n maior que 10 (n>10);

b) Método Range

Da mesma forma que no gráfico anterior, a linha central do gráfico é representada por $\overline{R}$ e calculada como segue:

$$\overline{R} = \frac{1}{k}\left(R_1 + R_2 + ... + R_k\right)$$

Utiliza-se, para o cálculo dos limites de controle, duas constantes d_3 e d_4 (ver tabela CEP). Desta forma calcula-se os limites:

$$\text{LSC} = \overline{R}.d_4$$

$$\text{LIC} = \overline{R}.d_3$$

Os valores para cálculo das linhas centras e limites de controle são definidos de acordo com a tabela que segue:

Tabela CEP (valores para cálculo das linhas centrais e limites de controles)

N	a_2	d_2	c_2	d_3	d_4	c_3	c_4
1	1,880	1,128	0,5642	0,000	3,267	0,000	3,267
2	1,023	1,693	0,7236	0,000	2,575	0,000	2,568
3	0,729	2,059	0,7979	0,000	2,282	0,000	2,266
4	0,577	2,326	0,8407	0,000	2,115	0,000	2,089
5	0,483	2,534	0,8686	0,000	2,004	0,030	1,970
6	0,419	2,704	0,8882	0,076	1,924	0,118	1,882
7	0,373	2,847	0,9027	0,136	1,864	0,185	1,815
8	0,337	2,970	0,9139	0,184	1,816	0,239	1,761
9	0,308	3,078	0,9227	0,223	1,777	0,284	1,716
10	0,285	3,173	0,9300	0,256	1,744	0,321	1,679
11	0,266	3,258	0,9359	0,284	1,716	0,354	1,646
12	0,249	3,336	0,9410	0,308	1,692	0,382	1,618
13	0,235	3,407	0,9453	0,329	1,671	0,406	1,594
14	0,223	3,472	0,9490	0,348	1,652	0,428	1,572
15	0,212	3,532	0,9523	0,364	1,636	0,448	1,552
16	0,203	3,588	0,9551	0,379	1,621	0,466	1,534
17	0,194	3,640	0,9576	0,392	1,608	0,482	1,518
18	0,187	3,689	0,9599	0,404	1,596	0,497	1,503
19	0,180	3,735	0,9619	0,414	1,586	0,510	1,490
20	0,173	3,778	0,9638	0,425	1,575	0,523	1,477
21	0,167	3,819	0,9655	0,434	1,566	0,534	1,466
22	0,162	3,858	0,9670	0,433	1,557	0,545	1,455
23	0,157	3,895	0,9684	0,452	1,548	0,555	1,445
24	0,153	3,931	0,9696	0,459	1,541	0,565	1,435

Gráfico de Histograma

O Gráfico de Histograma mostra um volume grande de dados de forma clara, permitindo uma melhor visualização da tendência central, da dispersão ao longo da escala de medição e da freqüência de valores.

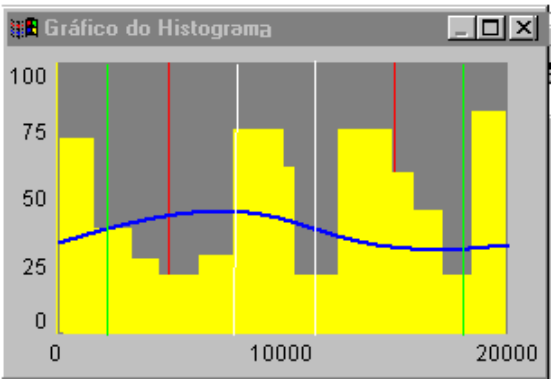


Figura 98: Gráfico de Histograma

Na página Gráfico de Histograma podemos configurar a aparência deste tipo de gráfico gerado a partir do CEP. Esta página tem os mesmos controles e funcionamento do gráfico das médias. O Elipse SCADA gera o Gráfico de Histograma baseado nos seguintes elementos:

Número de Barras

Total de itens	Número de barras
20-50	6
51-100	7
101-200	8
201-500	9
501-1000	10
Mais de 1000	11-20

Intervalo de Barra (I)

A largura de cada barra é dada por um intervalo, determinado diminuindo-se o menor item do maior e dividindo-se o resultado pelo número de barras. Elipse SCADA define que item pertence a que barra e calcula a freqüência total f para cada barra do gráfico. A distribuição normal também aparece no gráfico do Histograma e é determinada pela seguinte fórmula:

$$y = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)2}{2\sigma^2}}$$

onde: μ = média e σ = desvio padrão

A linha central da média, e os limites de controle (LSC e LIC) e engenharia (LSE e LIE) também aparecem no gráfico.

Índices de Capabilidade

Antes de descrever cada índice de capabilidade, as seguintes fórmulas devem ser definidas:

Ponto médio = (LSE + LIE) / 2

Tolerância = LSE - LIE

CP - Capabilidade Inerente do Processo

$$\text{CP} = \frac{Tolerancia}{6\sigma}$$

Se as amostras não estiverem no ponto médio elas tendem a zero.

VALOR DO CP	DESCRIÇÃO
Maior que 1.33	Processo é capaz.
Entre 1.0 e 1.33	Processo é capaz, mas precisa ser monitorado com CP próximo de 1.0.
Menor que 1.0	Processo não é capaz.

CR - Taxa de Capabilidade

É o inverso do CP.

$$\text{CR} = \frac{6\sigma}{Tolerancia}$$

valores de CR menores que 0.75 indicam capabilidade.

K - média do processo versus média especificada

K é a comparação entre a média e o ponto médio, mostrando o quanto os dados estão centralizados de acordo com a especificação.

$$K = \frac{(Media - PtoMedio)}{\left(\frac{Tolerancia}{2}\right)}$$

CPK - Capabilidade em relação a média especificada

CPK é a capabilidade do processo baseada no pior caso de dados. CPK é o menor valor de:

$$\frac{(LSE - Media)}{3\sigma} \quad \text{ou} \quad \frac{(Media - LIE)}{3\sigma}$$

Um valor negativo do CPK indica que a média está fora dos limites de engenharia. Um CPK igual a zero indica que a média é igual a um dos limites de engenharia. Um CPK entre 0 e 1.0 significa que parte dos 6 limites sigma caem fora dos limites de engenharia. Um CPK igual a 1 indica que um final dos 6 limites sigma cai em um limite de engenharia. Um CPK maior que um significa que os 6 limites sigma caem completamente dentro dos limites de engenharia.

Configurações

Esta página é a mesma existente na consulta do Histórico e permite a configuração do arquivo de histórico que será usado pelo CEP.

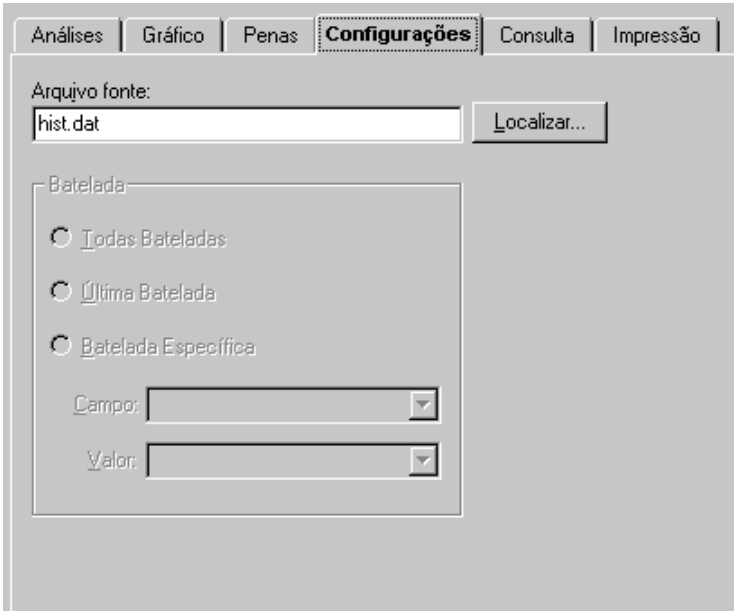


Figura 99: Configurações

Consulta

Esta página é a mesma existente na consulta do Histórico e permite definir um intervalo de tempo para o arquivo de Histórico.

Figura 100: Configurações das Consultas

Capítulo 10

10. Relatórios

Relatórios permitem imprimir os dados de arquivos de Histórico ou de Alarmes ou ainda valores de tags em tempo real.

Atualmente, os relatórios têm a seguinte restrição: não permitem carregar/salvar configuração de impressora (a janela de configuração da impressora é mostrada sempre).

Você pode definir um Relatório no Organizer durante a configuração da aplicação ou em tempo de execução usando **Funções Especiais** através de Scripts.

Existem quatro tipos de Relatórios disponíveis:

- **Texto:** imprime os dados de arquivos de históricos ou de alarmes em formato texto;
- **Gráfico**: imprime os dados de arquivos de históricos ou de alarmes de forma gráfica;
- **Formatado**: usado para imprimir dados em tempo real, como por exemplo o valor de um Tag em determinado momento;
- **Análise Histórica** : é um relatório em tela que possui dentro dele um relatório gráfico.

Para criar ou editar um Relatório você precisa selecionar a opção *Relatórios* na árvore da aplicação no Organizer, a página abaixo será mostrada contendo uma lista de todos os Relatórios existentes na aplicação.

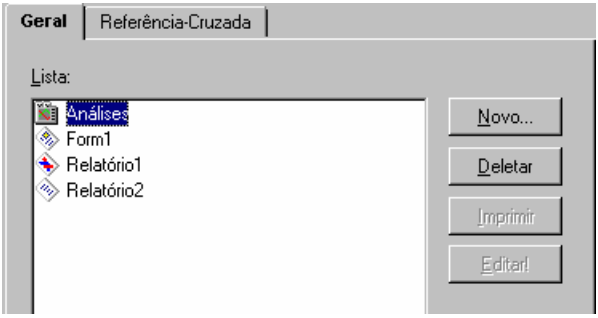


Figura 101: Tela de criação de relatórios

Você pode criar um novo Relatório usando o botão *Novo* à direita da página ou remover um existente selecionando-o na lista e pressionando o botão *deletar*. A janela para escolha do tipo do Novo Relatório é mostrada a seguir:

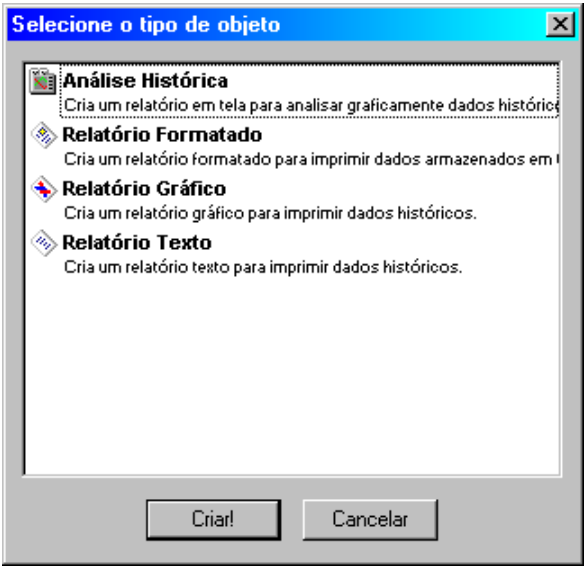


Figura 102: Tela dos tipos de relatórios

Cada Relatório que você cria para a aplicação aparece abaixo da opção *Relatórios* (*Reports*) na árvore da aplicação no Organizer. Ao selecionar um Relatório específico suas propriedades são mostradas ao lado direito da árvore.

10.1. Propriedades Gerais

Para relatórios do tipo: Texto, Gráfico e Formatado

A página de propriedades Gerais dos Relatórios Texto, Gráfico e Formatado aparece quando selecionada a aba *Geral* no topo das páginas do Relatório. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

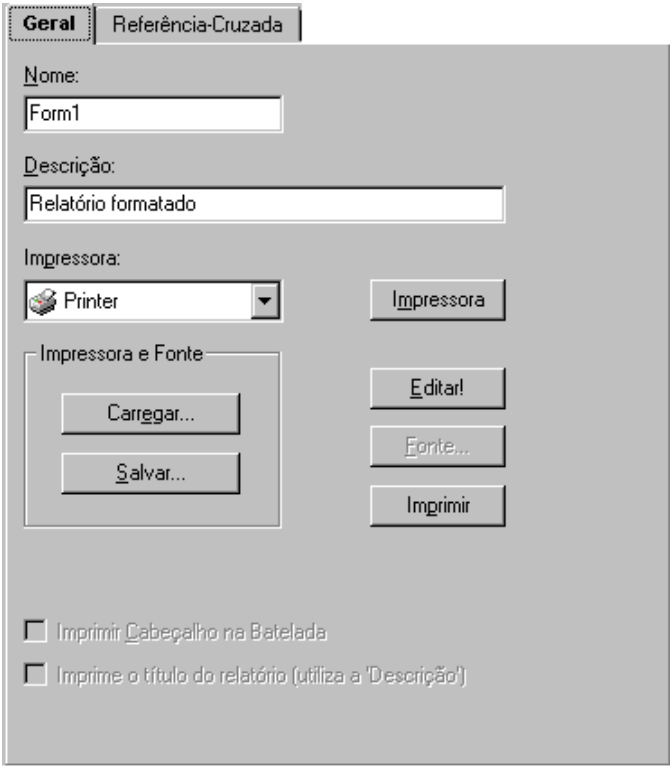


Figura 103: Propriedades gerais dos relatórios

Propriedades Gerais dos Relatórios

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do Relatório que será usado na árvore do Organizer e nos Scripts.
Descrição	Uma breve descrição sobre o Relatório.
Imprimir cabeçalho na batelada	Habilita a impressão de uma página contendo dados do cabeçalho da Batelada.
Imprime o título do relatório	Somente disponível para Relatórios TEXTO e GRÁFICO. Imprime como título o texto informado no campo Descrição.
Impressora	Permite escolher a definição de impressora para o relatório. O botão mostra a caixa de diálogo de Configuração de Impressora.
Editar!	Permite editar o desenho de um relatório FORMATADO.
Fonte...	Define a fonte (tipo, cor e tamanho) a ser usada no Relatório.
Imprimir	Imprime o Relatório.
Impressora e Fonte	Os botões Carregar e Salvar permitem carregar e salvar configurações de impressora previamente definidas e salvas no sistema.

10.2. Configurações

Para relatórios do tipo: Texto e Gráfico

Permite a especificação do arquivo a ser impresso: Histórico (extensão DAT), Alarmes (extensão DAT) ou Batelada (extensão HDR). Se a opção Processo de Batelada estiver marcada (ver Propriedades Gerais do Histórico) o quadro Batelada nesta página estará disponível para a escolha da Batelada.

A página de Configurações dos Relatórios aparece quando selecionada a aba Configurações no topo das páginas do Relatório Texto ou Gráfico. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

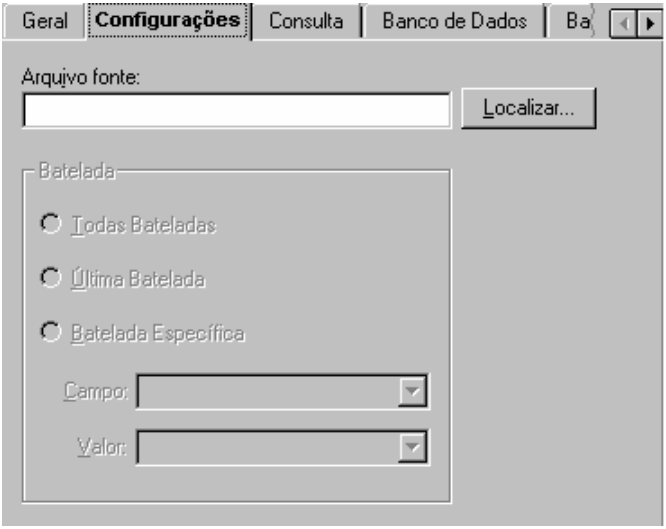


Figura 104: Configurações do Relatório

Configurações dos Relatórios

OPÇÃO	DESCRIÇÃO
Arquivo fonte	Define o nome do arquivo fonte para o Relatório (.DAT ou .HDR).
Localizar...	Permite localizar o arquivo fonte a ser usado pelo Relatório.
Todas as bateladas	Seleciona todas as bateladas para serem impressas. Esta opção está disponível somente para arquivos de Histórico por batelada.
Ultima batelada	Seleciona a última batelada para ser impressa. Esta opção está disponível somente para arquivos de Histórico por batelada.
Batelada específica	Seleciona uma batelada específica para ser impressa, conforme o especificado nos campos Campo e Valor.
Campo	Lista os campos disponíveis para seleção de uma batelada específica. Somente campos tipo string serão listados.
Valor	Define o valor a ser buscado quando numa batelada específica.

10.3. Consulta

Para relatórios do tipo: Texto e Gráfico

Permite definir um intervalo de tempo para selecionar os dados do arquivo a ser impresso. A página de Consulta do Relatório aparece quando selecionada a aba *Consulta* no topo das páginas dos Relatórios Texto ou Gráfico. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

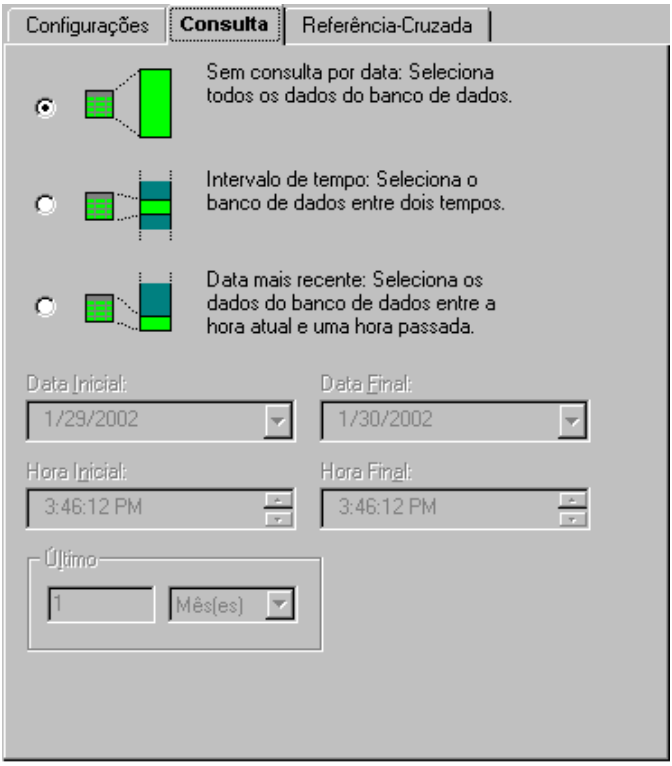


Figura 105: Propriedades das Consultas

Propriedades das Consultas

OPÇÃO	DESCRIÇÃO
Sem consulta por data	Não será usado filtro, ou seja, seleciona todos os dados.
Intervalo de tempo	Seleciona os dados dentro de um intervalo de tempo especificado.
Dados mais recentes	Seleciona apenas os dados mais novos.
Data inicial	Determina a data inicial do intervalo de tempo.
Hora inicial	Determina o horário inicial do intervalo de tempo.
Data final	Determina a data final do intervalo de tempo.
Hora final	Determina o horário final do intervalo de tempo.
Último	Número de unidades usadas para selecionar os dados mais recentes.
Unidade	Define a unidade usada para selecionar os dados mais recentes.

10.4. Banco de Dados

Relatório Texto

A página Banco de Dados do Relatório aparece quando selecionada a aba Banco de Dados no topo das páginas do Relatório Texto.

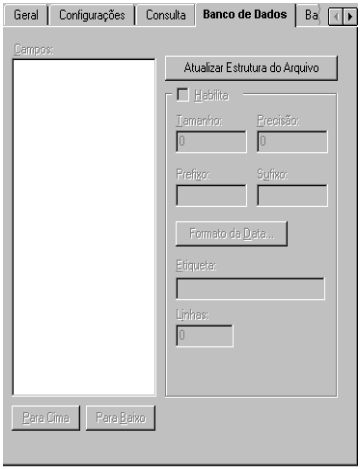


Figura 106: Propriedades de Bancos de Dados em Relatórios

Propriedades do Banco de Dados

OPÇÃO	DESCRIÇÃO
Campos	Lista os campos disponíveis.
Para Cima	Move o campo selecionado uma posição para cima.
Para Baixo	Move o campo selecionado uma posição para baixo.
Atualizar Estrutura do Arquivo	Recarrega a lista de campos atual do arquivo nos campos do Relatório. Quando é feita alguma modificação na estrutura do arquivo usado pelo Relatório este botão deve ser pressionado para atualização dos campos.
Habilitado	Permite que o campo selecionado seja impresso.
Tamanho	Define o número de dígitos a serem mostrados incluindo o ponto decimal.
Precisão	Define quantos dígitos do tamanho serão decimais.
Prefixo	Adiciona um prefixo tipo string ao valor mostrado.
Sufixo	Adiciona um sufixo tipo string ao valor mostrado.
Formato da Data	Define o formato do campo DataTime.
Etiqueta	Define um nome alternativo aos campos que serão impressos no relatório.
Linhas	Define o número de linhas que o campo selecionado irá ter no Relatório.

10.5. Batelada

Relatório Texto

A página de Batelada do Relatório Texto aparece quando selecionada a aba Batelada no topo das páginas do Relatório Texto. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

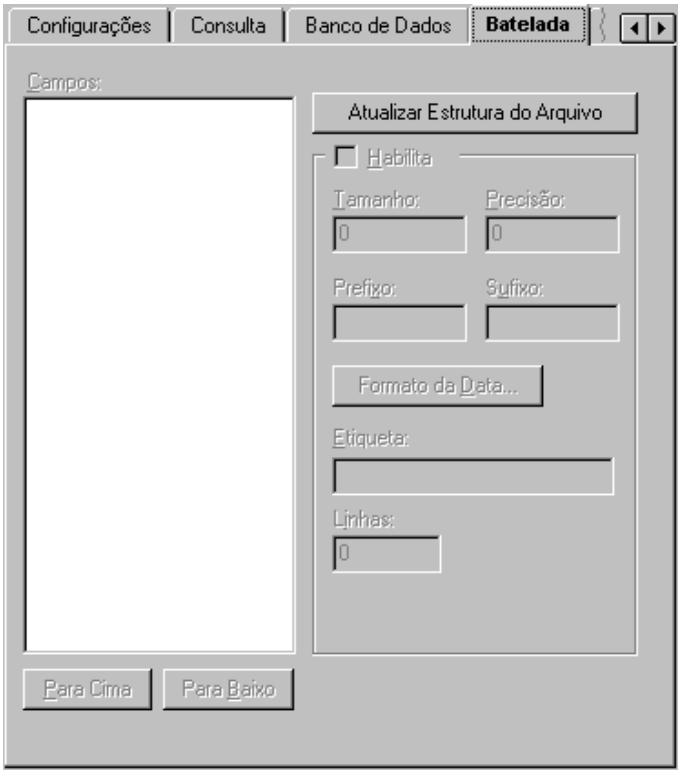


Figura 107: Propriedades de Bateladas em Relatórios

Propriedades de Bateladas

OPÇÃO	DESCRIÇÃO
Campos	Lista os campos disponíveis.
Para cima	Move o campo selecionado uma posição para cima.
Para baixo	Move o campo selecionado uma posição para baixo.
Habilitado	Permite que o campo selecionado seja impresso.
Atualizar estrutura de arquivo	Recarrega a lista de campos atual do arquivo nos campos do Relatório. Quando é feita alguma modificação na estrutura do arquivo usado pelo Relatório este botão deve ser pressionado para atualização dos campos.
Tamanho	Define o número de dígitos a serem mostrados incluindo o ponto decimal.
Precisão	Define quantos dígitos do tamanho serão decimais.
Prefixo	Adiciona um prefixo tipo string ao valor mostrado.
Sufixo	Adiciona um sufixo tipo string ao valor mostrado.
Formato da Data	Define o formato do campo DataTime.
Etiqueta	Define um nome alternativo aos campos que serão impressos no relatório.
Linhas	Define o número de linhas que o campo selecionado irá ter no Relatório.

10.6. Gráfico

Relatório Gráfico

A página Gráfico do Relatório Gráfico aparece quando selecionada a aba Gráfico no topo das páginas do Relatório Gráfico. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue:

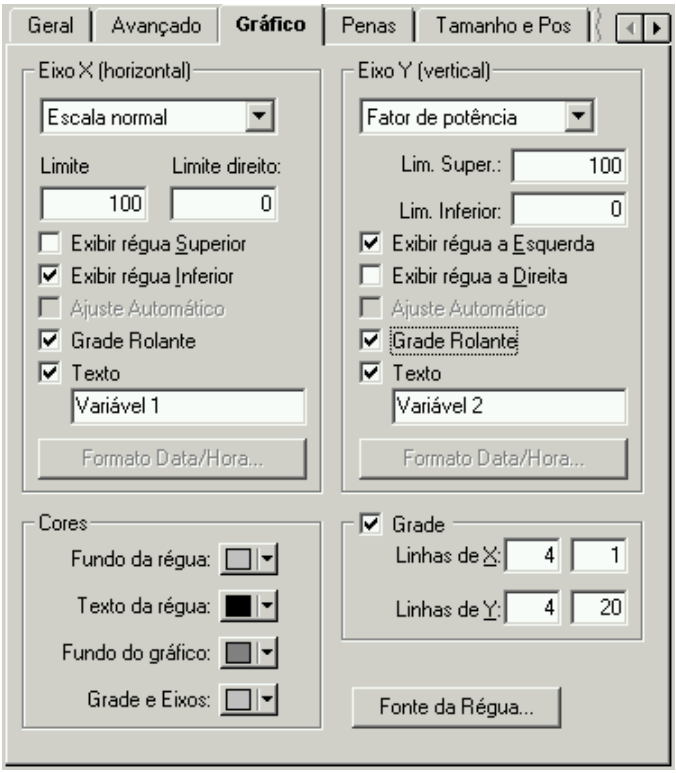


Figura 108: Propriedades do Gráfico

Propriedades do Gráfico (Eixo X e Eixo Y)

OPÇÃO	DESCRIÇÃO
Escala	Define se a escala do gráfico será linear (escala normal) ou logarítmica (fator de potência).
Limite esquerdo	Define o limite esquerdo do eixo X do gráfico. Esta opção está disponível somente se o gráfico é do tipo XY.
Limite direito	Define o limite direito do eixo X do gráfico. Esta opção está disponível somente se o gráfico é do tipo XY.
Exibir régua superior	Exibe uma régua superior no gráfico, onde são mostrados os valores do eixo X.
Exibir régua inferior	Exibe uma régua inferior no gráfico, onde são mostrados os valores do eixo X.
Limite superior	Define o limite superior do eixo Y do gráfico.
Limite inferior	Define o limite inferior do eixo Y do gráfico.
Exibir régua a esquerda	Exibe uma régua a esquerda do gráfico, onde são mostrados os valores do eixo Y.
Exibir régua a direita	Exibe uma régua a direita do gráfico, onde são mostrados os valores do eixo Y.
Ajuste automático	Calcula automaticamente os limites dos eixos X ou Y do gráfico.
Grade rolante	Define que a grade de orientação do gráfico irá rolar na direção em que o gráfico está sendo desenhado acompanhando os valores, ao invés de ficar fixa no objeto.
Texto	Legenda que irá aparecer ao lado da escala do eixo X e eixo Y.
Formato Data/Hora	Define o formato da marcação de tempo que irá aparecer no eixo X nos gráficos Tempo X Dado.

Propriedades do Gráfico (Cores e Grade)

OPÇÃO	DESCRIÇÃO
Fundo da régua	Define a cor de fundo para as réguas do gráfico.
Texto da régua	Define a cor para o texto que será mostrado nas réguas.
Fundo do gráfico	Define a cor de fundo para o gráfico.
Grades e eixos	Define a cor da grade e dos eixos do gráfico.
Grade	Mostra uma grade (linhas de orientação vertical e horizontal) no gráfico.
Linhas de X	Define o número de linhas horizontais da grade.
Linhas de Y	Define o número de linhas verticais da grade.
Mostrar milisegundos	Mostra os milisegundos no eixo do gráfico que representa o tempo.
Fonte da régua...	Define fonte, tamanho e cor para os caracteres do gráfico.

10.7. Penas

Relatório Gráfico

A página Penas do Relatório Gráfico aparece quando selecionada a aba Penas no topo das páginas do Relatório Gráfico. Esta página é mostrada abaixo e seus respectivos controles e campos são descritos na tabela que segue.

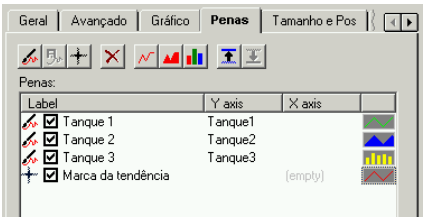


Figura 109: Propriedades de Penas

Propriedades das Penas

OPÇÃO	DESCRIÇÃO
	Insere uma pena associada a um tag.
	Insere uma pena associada a um campo do histórico (a opção Carrega Dados do Histórico do Disco deve estar habilitada na aba Avançado).
	Insere uma marca indicativa no gráfico.
	Apaga uma pena ou uma marca selecionada na lista.
	Define que o desenho da pena será do tipo Linha.
	Define que o desenho da pena será do tipo Área.
	Define que o desenho da pena será do tipo Barra.
	Muda a ordem da pena selecionada.
Label	Essa coluna lista as penas selecionadas para o gráfico. Permite mudar o texto da legenda e ativar/desativar cada pena.
Y Axis	Seleciona o tag a ser visualizado no eixo Y.
X Axis	Seleciona o tag a ser visualizado no eixo X.
	Mostra o quadro Cores das Penas (ver seção a seguir).

Cores das Penas

O quadro **Cores das Penas** aparece quando clica-se no ícone  ao lado das penas do gráfico. Nesse quadro, podemos definir a cor, o formato e o tipo do gráfico que será desenhado para a pena em questão: linha, área ou barra. O botão Outros... permite a escolha de outras cores que não as listadas no quadro.

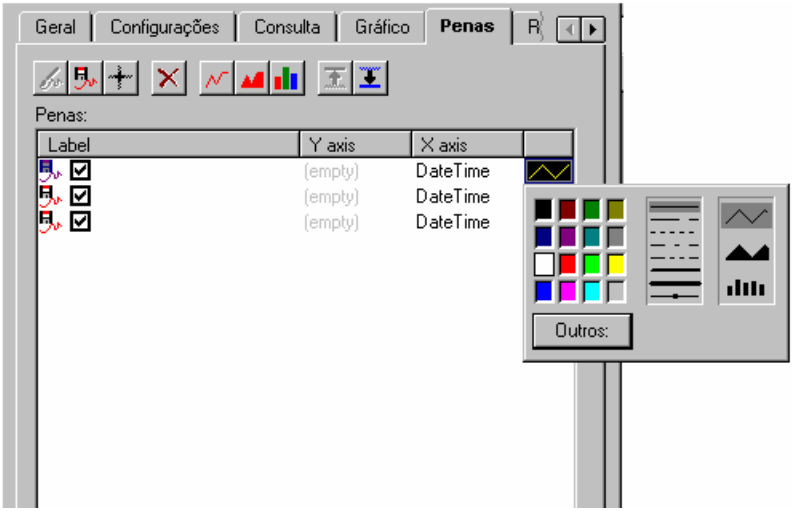


Figura 110: Configurações das Cores das Penas

10.8. Scripts

Relatório Gráfico

Os scripts permitem fazer pré-configurações nos relatórios, antes de sua impressão. Por exemplo, é possível configurar a escala de um gráfico a ser impresso. Maiores detalhes a respeito do uso de scripts veja no capítulo específico.

SCRIPTS DISPONÍVEIS

EVENTO	DESCRIÇÃO
OnBeforePrint	Disparado na execução do método Print() do relatório, logo depois de executar a consulta, antes do relatório ser impresso de fato.

10.9. Relatório Formatado

O **Relatório Formatado** é usado quando se deseja imprimir dados em tempo real como por exemplo o valor de um determinado tag em dado momento. Este relatório pode ser editado usando-se objetos disponíveis em modo de edição.

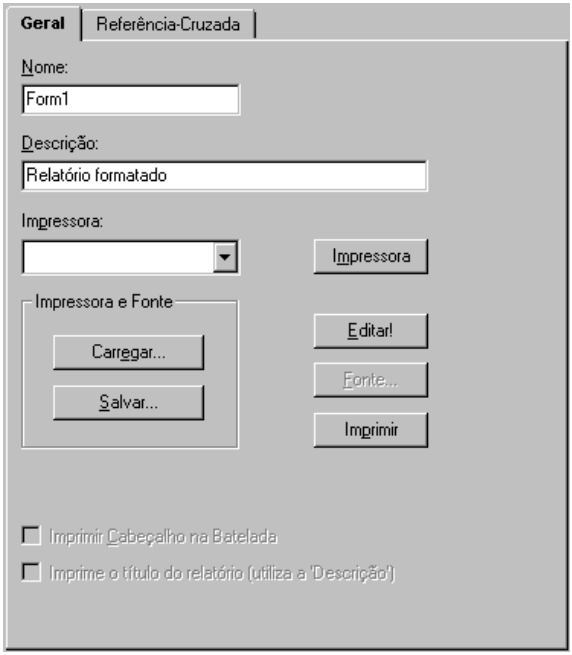


Figura 111: Relatório Formatado

10.10. Relatório Análise Histórica

O **Relatório Análise Histórica** é um relatório em tela que cria um objeto Análise Histórica, podendo também ser impresso. A configuração e as páginas de propriedade deste Relatório são feitas em tempo de execução, de acordo com o que já foi descrito nessa seção.

É possível configurar a janela da Análise Histórica e ativar ou desativar as páginas de configuração nas propriedades gerais do objeto Análise Histórica no Organizer.

Capítulo

11. Drivers

O Elipse SCADA permite a comunicação com equipamentos de aquisição de dados e com outros computadores executando o Elipse SCADA, através de drivers de E/S ou drivers de rede fornecidos pela Elipse Software. Um documento é fornecido com cada driver contendo informações importantes a respeito de sua configuração.

Verifique se o driver desejado já está disponível para o SCADA CE. Os arquivos .dll para SCADA Win e SCADA CE são diferentes.

Para instalar ou configurar um driver você precisa selecionar a página de drivers em um tag PLC ou Bloco associado ou acessar o objeto Drivers no Organizer. Nesse caso, a página abaixo será mostrada contendo uma lista de todos os drivers instalados na aplicação. Você pode instalar um novo driver pressionando o botão Novo ou remover um existente selecionando-o na lista e pressionando o botão Deletar.

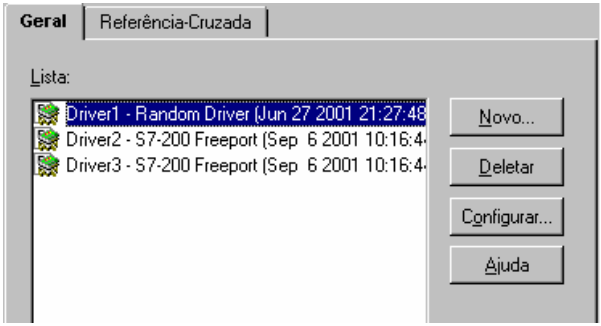


Figura 112: Drivers do Sistema

11.1. Configurando Drivers

Cada novo driver que você adiciona na aplicação, aparece abaixo da opção Drivers na árvore da aplicação no Organizer. Ao selecionar um driver específico, suas propriedades são mostradas ao lado direito da árvore.

A página de Configuração do Driver aparece quando selecionada a respectiva aba no topo das páginas de configuração do driver ou quando você pressiona o botão Configurar... à direita da lista de Drivers.

11.1.1. Drivers PLC

Para os **Drivers PLC,** você pode configurar os parâmetros de comunicação "P" e outras propriedades do driver de acordo com as informações contidas no arquivo de ajuda que acompanha o driver. A opção Abortar em erro encerra a comunicação caso ocorra algum problema, impedindo que a aplicação fique "travada".

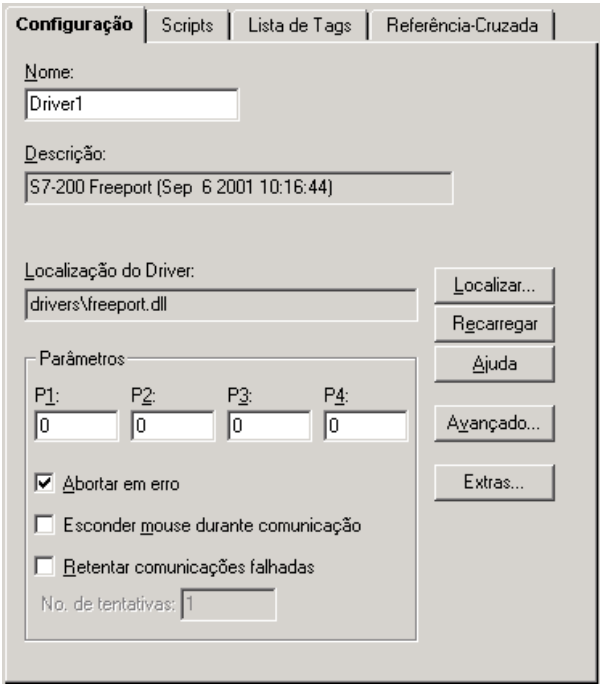


Figura 113: Configuração do driver

Propriedades de Configuração de Drivers

OPÇÃO	DESCRIÇÃO
Nome	Nome do objeto.
Descrição	Mostra informações do driver, como fabricante, versão e data.
Localização do Driver	Mostra o caminho do arquivo do driver.
Parâmetros	Permite a configuração dos parâmetros de comunicação "P" para o driver selecionado, conforme especificado na ajuda do driver.
Abortar em erro	Aborta a comunicação se algum problema ocorrer.
Esconder mouse durante a comunicação	Esconde o cursor do mouse durante a comunicação.
Retentar comunicações falhadas	Define que o sistema irá tentar reestabelecer uma comunicação perdida com o driver.
No. de tentativas	Define o número de tentativas de reestabelecer a comunicação com o driver.
Localizar	Permite navegar pelos diretórios para localizar o arquivo do driver.
Recarregar	Recarrega o driver, reestabelecendo a conexão.
Ajuda	Mostra a ajuda do driver selecionado.
Avançado	Abre uma janela para configuração das opções avançadas do driver.
Extras	Abre uma janela paea configuração dos parâmetros extras do driver.

Propriedades de Configuração de Drivers (Avançado)

OPÇÃO	DESCRIÇÃO
Manter comportamento 16-bits	Se ligado desabilita a multitarefa, fazendo com que o driver 32-bits rode como na versão 16-bits. Essa opção é necessária para alguns drivers mais antigos.
Escrever em background	Controla como as escritas serão efetuadas no driver. Algumas escritas geradas pelo Elipse SCADA podem ser efetuadas em background (segundo plano), aumentando a performance. Se esta opção estiver desabilitada todas as escritas serão em foreground (primeiro plano), forçando a aplicação a esperar a comunicação com o driver para continuar sua execução.
Prioridade de escrita	Permite escolher se a prioridade das escritas será maior (*High*) ou igual (*Low*) à prioridade das leituras.
Iniciar o driver ao rodar a aplicação	Permite desabilitar o início automático do driver. Por exemplo: em um driver *dial-up*, muitas vezes é interessante esperar um comando para iniciar a comunicação.
Manter o driver em memória	Muitos drivers têm um tempo de carregamento e início muito lento. Essa opção mantém o driver em memória, minimizando o tempo de início e agilizando a comunicação.

O botão Extras... chama configurações especiais dos drivers. Essas configurações variam de driver para driver. Consulte o arquivo de ajuda que acompanha o driver para obter maiores informações. A caixa de diálogo pode mudar de acordo com o driver que está inserido no sistema, mas em geral, é mostrado o que está abaixo:

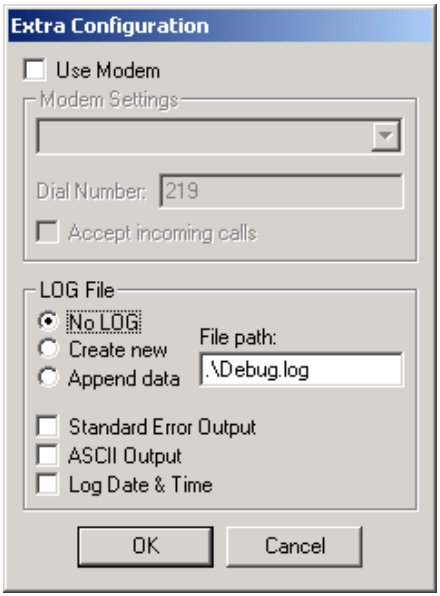


Figura 114: Configurações extras

As opções disponíveis são as seguintes:

Opções das Configurações Extras

OPÇÃO	DESCRIÇÃO
Use Modem	Habilita a utilização do modem.
Modem Settings	Determina o modem a ser utilizado.
Dial number	Determina o número a ser discado pelo modem.
Accept incoming calls	Configura o modem, após o driver ser conectado ao mesmo, para atender chamadas vindas de outro modem, após um número configurável de toques.
No Log	Não utiliza o arquivo de log.
Create new	Cria um novo arquivo de log e insere no local indicado no campo file path. Se já existir arquivo de log no path especificado, o arquivo de log será sobre-escrito.
Append data	Cria um novo arquivo de log e insere no local indicado no campo file path. Se já existir arquivo de log no caminho especificado, as informações serão acrescentadas ao final do mesmo.
Standard Error Output	Utilizada para depuração. Esta opção deve ser mantida desabilitada.
ASCII Output	Converte os bytes de comunicação em caracteres texto, conforme a tabela ASCII.
Log Date & Time	Mostra o timestamp, isto é, a data e o horário de cada mensagem no arquivo.

11.1.2. Drivers de Rede

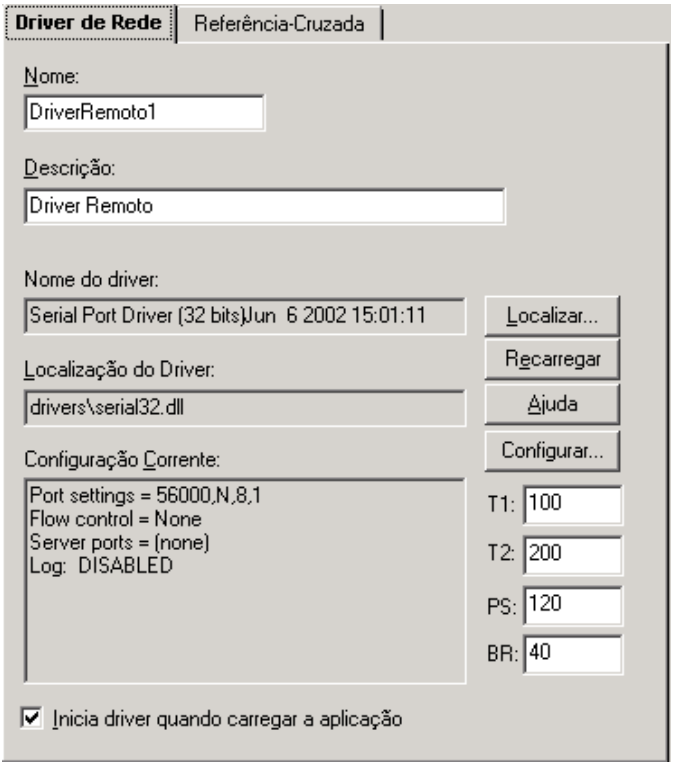


Figura 115: Configurações Comuns para Drivers de Rede

Configurações Comuns para Drivers de Rede

OPÇÃO	DESCRIÇÃO
Nome	Nome do objeto.
Descrição	Breve descrição do objeto.
Nome do driver	Nome do driver, definido pelo fabricante.
Localização do driver	Permite navegar nos diretórios para indicar o arquivo do driver.
Configuração corrente	Mostra a configuração corrente do driver de rede.
Localizar...	Permite navegar pelos diretórios para localizar o arquivo do driver.
Recarregar	Recarrega o driver, reestabelecendo a conexão.
Ajuda	Mostra a ajuda do driver selecionado.
Configurar...	Abre a janela de configuração do driver. Esta janela varia conforme o driver selecionado. Consulte a documentação do driver para obter mais informações
Inicia driver quando carregar a aplicação	Ativa o driver no início da execução da aplicação.
T1, T2, PS, BR	Estes parâmetros governam o funcionamento do protocolo de troca de mensagens entre diversas instâncias do Elipse SCADA. O protocolo é *full-duplex* e permite que cada estação envie até 16 mensagens sem confirmação (janela de mensagens). O tamanho da mensagem em bytes é definido no campo PS (Packet Size). O pacote de dados é dividido entre a troca de dados de tags e arquivos remotos, e o envio de dados para transmissão de vídeo em tempo real para o Elipse Watcher. O campo BR (Band Reserve) determina o percentual destes pacotes que está reservado para a troca de dados. O restante pode ser utilizado pelo Watcher. Se não houverem dados de vídeo, o pacote inteiro é utilizado para dados de tags, e vice-versa. O campo T1 determina o tempo de envio do *keepalive* (em milisegundos), isto é, uma mensagem que informa que o Elipse ainda está ativo. Esta mensagem só é enviada se a estação não estiver enviando dados no momento. O campo T2 indica o timeout (tempo-limite), em milisegundos, para o reenvio de mensagens não-reconhecidas. Se uma mensagem não foi reconhecida como recebida, ela é reenviada após este tempo expirar. Se o envio de uma mensagem falhar dez vezes, a conexão é abortada.

11.2. Scripts de Drivers

Scripts de drivers normalmente estão associados ao status da comunicação. Isto significa, na prática, que um script pode ser executado quando um erro de comunicação ocorrer. Maiores detalhes a respeito do uso de scripts veja no capítulo específico.

Scripts Disponíveis

Você pode associar scripts a drivers executando-os em uma das situações a seguir:

OnCommError — Executado cada vez que um erro de leitura ou escrita ocorrer no driver.

11.3. Lista de Tags Associados

A página Lista de Tags mostra os tags PLC e Bloco que estão associados ao driver em questão, permitindo a edição dos parâmetros de endereçamento e varredura diretamente na lista, num processo semelhante a uma planilha eletrônica.

Figura 116: Cabeçalho da Lista de tags associados

Opções disponíveis na Lista de tags associados

OPÇÃO	DESCRIÇÃO
Nome	Nome do Tag
N1/B1 – N4/B4	Parâmetros de endereçamento do tag .
Scan	Tempo de atualização (leitura) do valor do tag.
Size	Mostra o tamanho de um tag Bloco (não disponível para tags PLC).
Exportar Tags...	Permite criar um arquivo no formato CSV com a definição dos parâmetros dos tags listados. Este arquivo pode ser importado no Elipse E3 ou qualquer outro software compatível (como por exemplo, o Microsoft Excel).

Capítulo

12. Database

A opção Databases permite criar e manipular bancos de dados dentro do Elipse SCADA. Este objeto utiliza o padrão ODBC do Windows que efetua a manipulação do banco de dados enviando e recebendo dados. Open Database Connectivity (ODBC) é uma interface criada pela Microsoft que oferece uma interface universal para acesso a diferentes bancos de dados incluindo Oracle, Access, MySQL, Interbase entre outros.

Você pode definir um Banco de Dados no Organizer durante a configuração da aplicação ou em tempo de execução através de Scripts. Para utilizar um banco de dados no Elipse SCADA você precisa selecionar a opção Databases na árvore da aplicação no Organizer. A página abaixo será mostrada contendo uma lista de todos os Databases existentes na aplicação.

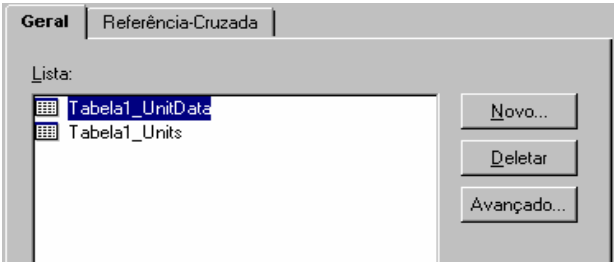


Figura 117: Database

Propriedades do Database

OPÇÃO	DESCRIÇÃO
Lista	Lista os databases existentes na aplicação.
Novo	Chama o Assistente de Nova Conexão que permite acrescentar uma conexão ODBC para um banco de dados.
Deletar	Remove da lista o database selecionado.
Avançado	Permite a consulta e edição do string de conexão ODBC.

Você pode conectar o Elipse SCADA a um banco de dados usando o Assistente de Nova Conexão ao pressionar o botão Novo à direita da página. Uma janela será mostrada perguntando se você deseja criar uma conexão com uma tabela já existente ou criar uma nova tabela. A seguir uma lista contendo as conexões (Data Sources) existentes será mostrada, se desejar uma nova clique no botão Novo e escolha o driver ODBC que você deseja usar: MS Access, MS Fox Pro, MS Excel, CA-Clipper, dBase, Oracle e outros. Selecionado o driver, clique OK e configure as opções do mesmo, entre elas: Nome da Conexão (Data Source Name) e o arquivo ou diretório que contém os dados. Se você estiver criando uma nova tabela uma janela será apresentada para que você informe o Nome da Tabela, seus respectivos campos e o tipo de dados de cada um.

O Elipse SCADA irá mostrar na árvore do Organizer a nova tabela e seus respectivos campos, que poderão ser modificados usando as funções especiais do ODBC nos Scripts. Você pode remover um banco de dados existente selecionando-o na lista e pressionando o botão Deletar. Para uma melhor compreensão da conectividade do ODBC veja a figura a seguir:

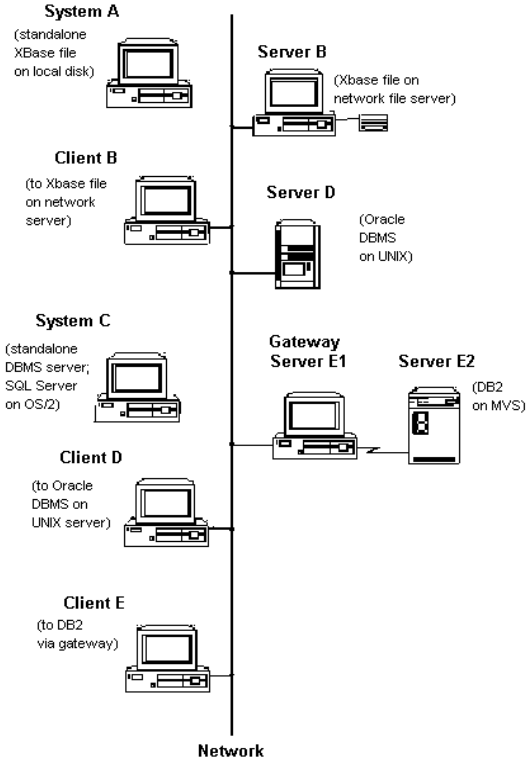


Figura 118: Exemplo de conectividade ODBC

Exemplo usando Database

O exemplo a seguir mostra uma conexão do Elipse SCADA com um banco de dados Access.

- No Access crie um banco de dados contendo uma ou mais tabelas. Ex: arquivo SUPPORT.MDB com a tabela Main Table.
- Feche o banco de dados (SUPPORT.MDB).
- No Elipse SCADA selecione a opção Databases na árvore do Organizer. Pressione o botão Novo e o Assistente de Nova Conexão será mostrado para que você escolha fazer uma conexão com uma tabela já existente ou criar uma nova tabela. Selecione a conexão com uma tabela já existente.

Figura 119: Assistente de Nova Conexão de Banco de Dados

- Selecione a aba Machine Data Source e uma lista com as conexões (Data Source Name) disponíveis será mostrada. Se você desejar alguma não disponível na lista pressione o botão Novo e escolha o Driver ODBC que deseja usar. Neste exemplo você deve escolher o banco de dados Microsoft Access e clicar OK.

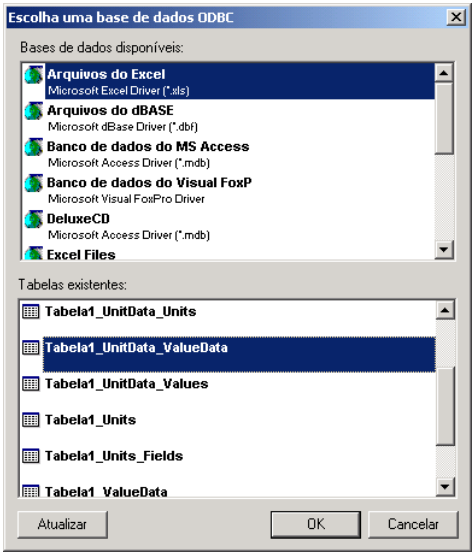


Figura 120: Bases de dados ODBC

- A janela Criar nova Database será mostrada para que voce indique o tipo da fonte de dados. Selecione Sistema Database e clique Next.
- Na janela seguinte escolha o Driver ODBC que deseja usar. Neste exemplo você deve escolher o banco de dados Microsoft Access e clicar Next;
- Aparecerá a janela de Configuração do Driver onde você deve informar o nome da conexão (Data Source Name) e pressionando o botão Select escolher o arquivo que você criou (SUPPORT.MDB);
- A próxima janela permite a seleção das tabelas do banco de dados que serão utilizadas. Selecione a tabela Main Table;
- Agora os títulos das tabelas selecionadas aparecem abaixo da opção Databases na árvore da aplicação e seus respectivos campos aparecem abaixo do título.
- Feito isso, os campos do Banco de Dados podem ser associados aos Objetos de Tela, bem como a Tags (variáveis do sistema). A manipulação dos registros do Banco de Dados é feita através de Funções Especiais que podem ser encontradas usando-se o App Browser quando em uma página de Scripts (ver Capítulo de Scripts - Funções Especiais - Funções de Bancos de Dados).

Capítulo

13

13. Usuários

Elipse SCADA permite a você controlar o acesso a uma aplicação através de uma lista de nomes, podendo atribuir uma senha a cada usuário e configurar níveis de segurança no seu sistema.

O usuário "Administrador" possui acesso ilimitado ao sistema. Os demais usuários possuem um nível de segurança associado que permite a eles acessarem apenas as características atribuídas ao seu nível de acesso.

Você pode criar uma lista de usuários selecionando a opção Usuários na árvore da aplicação no Organizer, a página abaixo será mostrada contendo uma lista de todos os usuários cadastrados na aplicação.

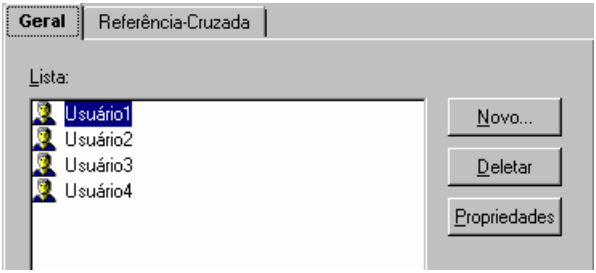


Figura 121: Lista de usuários

Opções disponíveis na Lista de usuários

OPÇÃO	DESCRIÇÃO
Lista	Lista de todos os usuários cadastrados.
Novo	Adiciona um novo usuário na lista de usuários.
Deletar	Remove o usuário selecionado.
Propriedades	Mostra as propriedades do usuário selecionado.

Você pode adicionar um novo usuário usando o botão Novo à direita da página ou remover um existente selecionando-o na lista e pressionando o botão Deletar.

Ao adicionar um novo usuário no sistema ele irá aparecer na árvore do Organizer logo abaixo de Usuários. Selecione o novo usuário na árvore e a janela a seguir irá aparecer, onde deverão ser informados o login e a senha do usuário.

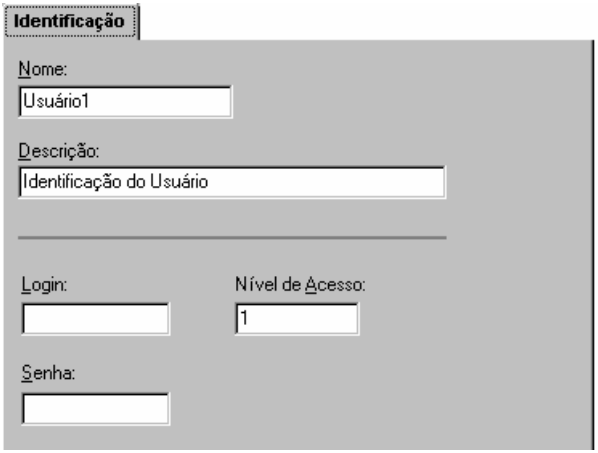


Figura 122: Propriedades do Usuário

Propriedades do Usuário

OPÇÃO	DESCRIÇÃO
Nome	Define o nome do usuário.
Descrição	Uma breve descrição sobre o usuário.
Login	String de identificação do usuário.
Senha	Campo para cadastrar a senha do usuário.
Nível de Acesso	Define o nível de acesso do usuário (de 1 a 100). A maior prioridade é o número 1.

Feito isto, você deve definir o nível de acesso para cada tela da aplicação conforme a tabela a seguir:

NÍVEL	ACESSO
0	livre acesso para todos
1	super usuário
maior que 1	outros usuários

Um usuário pode acessar somente as telas que possuem prioridade zero ou maior igual a sua. Como padrão, todas as telas possuem nível de acesso 0 (livre acesso).

13.1. Funções e Atributos

As funções a seguir são específicas para o *login*:

Aplicação.Login()
Chama uma caixa de diálogo para a identificação de um novo usuário. Retorna verdadeiro (diferente de zero) se o usuário foi identificado com sucesso (foi "logado") ou falso (zero) se não foi. Em caso de erro, nenhuma mensagem será mostrada. A função atualiza o atributo global lastError, colocando 0 (zero) se o login for cancelado pelo usuário ou 1 (um) se o usuário ou a senha for inválida.

Aplicação.Logout()
Retira um usuário logado da aplicação. Fecha todas as telas que possuem prioridade diferente de zero.

Aplicação.UserAdministration()
Esta função mostra uma de duas caixas de diálogo conforme o nível de acesso do usuário:

- Se User-level = 1 (administrador), ele poderá criar, modificar e remover os atributos de todos os usuários, inclusive trocar a senha.
- Se User-level > 1, ele somente poderá trocar a sua senha.

Quando você pressionar OK na caixa de diálogo a mensagem "Senha Trocada" deverá aparecer. Se aparecer a mensagem "Nova senha não confirma", você provavelmente reescreveu uma senha diferente da sua nova.

A aplicação possui os seguintes atributos para as funções de login:

ATRIBUTO	DESCRIÇÃO
UserName	Nome do usuário logado na aplicação (vazio, quando não há usuários logados).
UserAccessLevel	Nível do usuário logado na aplicação (100, quando não há usuários logados).

13.2. Scripts de Login

É possível associar scripts a eventos gerados pelo login de usuários.

Scripts Disponíveis
Você pode associar scripts a dois eventos de *login* da aplicação:

EVENTOS	DESCRIÇÃO
OnUserLogin	executado quando um usuário é logado na aplicação
OnUserLogout	executado quando um usuário sai da aplicação

Observações:

- Não é permitido mais de um usuário logado ao mesmo tempo na aplicação.
- Quando um usuário loga na aplicação, os dois scripts são executados: OnUserLogout (se já existe um usuário logado) e OnUserLogin;
- Quando a aplicação termina é executado um Logout;
- Você não pode acessar ou modificar qualquer propriedade de usuários (nome, descrição etc.) em tempo de execução. Propriedades de usuários não podem ser associadas a nenhum objeto de tela e não podem ser usadas em scripts.

Capítulo

14

14. Aplicações Remotas

As Aplicações Remotas são usadas quando se deseja conectar dois ou mais Elipse SCADA que estejam ligados via rede, modem ou cabo serial. Para usar esta característica, você deve definir uma estação Servidora e outra Cliente e adicionar o driver de rede que deseja usar em ambas.

Atualmente, só driver TCPIP está implementado para CE.

Você pode adicionar um driver de rede às suas aplicações Servidora e Cliente selecionando a opção Drivers na árvore da aplicação no Organizer e pressionando o botão Novo à direita da lista de drivers de rede. Cada novo driver que você adiciona na aplicação, aparece abaixo da opção Drivers e ao selecionar um Driver específico, suas propriedades são mostradas ao lado direito da árvore. Pressione o botão Configurar... para abrir a janela de configuração do driver de rede, que varia conforme o driver selecionado. (Ver o capítulo sobre drivers).

Na aplicação Cliente, depois de adicionar o driver de rede, você precisa criar uma Aplicação Remota e configurar seus parâmetros.

Para criar ou editar uma Aplicação Remota, você precisa selecionar a opção Aplicações Remotas na árvore da aplicação no Organizer. A página abaixo será mostrada contendo uma lista de todas as Aplicações Remotas existentes na aplicação. Você pode criar uma nova Aplicação Remota usando o botão Novo à direita da página ou remover uma existente selecionando-a na lista e pressionando o botão Deletar.

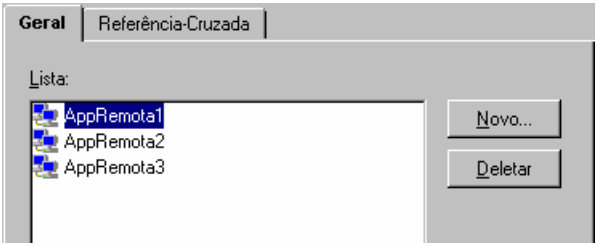


Figura 123: Aplicações Remotas

Propriedades da Lista de Aplicação Remota

OPÇÃO	DESCRIÇÃO
Lista	Mostra uma lista de todas as Aplicações Remotas existentes na sua aplicação, elas aparecem na árvore do Organizer logo abaixo da opção Remote Applications e quando selecionadas permitem a edição de suas propriedades.
Novo	Cria uma nova Aplicação Remota.
Deletar	Remove da lista a Aplicação Remota selecionada.

Depois de adicionar uma Aplicação Remota, você precisa configurar os parâmetros do Servidor conforme o driver selecionado. Para isso, pressione o botão Configurar… na página de Propriedades Gerais da Aplicação Remota (veja a próxima seção).

14.1. Propriedades Gerais

Cada Aplicação Remota que você cria para a aplicação, aparece abaixo da opção Aplicações Remotas na árvore da aplicação no Organizer. Ao selecionar uma Aplicação Remota específica, suas propriedades são mostradas ao lado direito da árvore.

A página de propriedades Gerais da Aplicação Remota aparece quando selecionada a aba Geral no topo das páginas da Aplicação Remota. Esta página é mostrada abaixo e seus respectivos campos são descritos na tabela que segue.

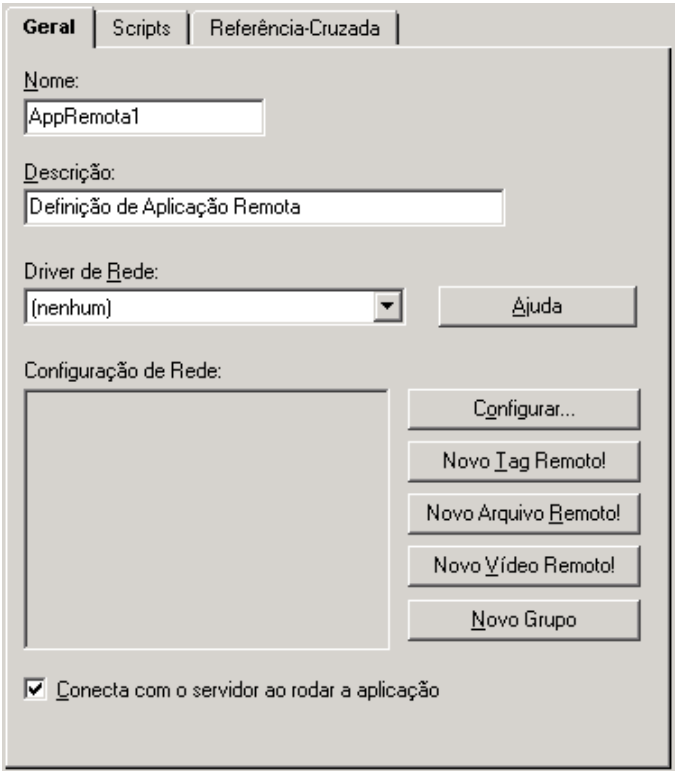


Figura 124: Propriedades Gerais

Propriedades Gerais de Aplicações Remotas

OPÇÃO	DESCRIÇÃO
Nome	Define o nome da Aplicação Remota que será usada na árvore do Organizer e nos Scripts.
Descrição	Uma breve descrição sobre a Aplicação Remota.
Driver de rede	Permite a seleção de um Driver de Rede para a Aplicação Remota corrente. Os drivers devem ser instalados na opção Drivers da árvore da aplicação a fim de que estejam disponíveis.
Configuração de rede	Mostra a configuração de rede usada pelo driver selecionado, por exemplo: porta remota, endereço de rede remoto, endereço do nodo remoto. Nenhuma edição é permitida neste campo.
Conectar com o servidor ao rodar a aplicação	Habilita a conexão com a Aplicação Remota (Servidor) no início da execução da aplicação no Elipse.
Ajuda	Mostra a Ajuda do Driver selecionado.
Configuração...	Abre uma janela que permite configurar, os parâmetros da Aplicação Servidora (remota), por exemplo, quando usando um driver MIRROR/SPX: - Porta Servidora: define a porta de comunicação do servidor (de 1 a 32767) - Endereço da Rede: define o endereço de rede do Servidor (8 caracteres hexa) - Endereço do Nodo: define o endereço do nodo Servidor (12 caracteres hexa)
Novo Tag Remoto!	Cria um novo Tag Remoto que aparece abaixo da Aplicação Remota na árvore do Organizer. Cada Tag Remoto criado na aplicação Cliente está associado a um Tag na aplicação Servidora (remota).
Novo arquivo Remoto!	Cria um novo Arquivo Remoto que aparece abaixo da Aplicação Remota na árvore do Organizer. O Arquivo Remoto é usado para obter um arquivo da aplicação Servidora, o que deve ser feito usando-se as funções dos Arquivos Remotos.
Novo Grupo	Cria um novo grupo de Tags Remotos que aparece abaixo da Aplicação Remota ou de um outro grupo deTags Remotos, na árvore do Organizer.

14.2. Scripts de Aplicações Remotas

Scripts de Aplicações Remotas geralmente estão associados a uma conexão remota. Isto significa, por exemplo, que eles podem ser executados ao iniciar ou terminar uma conexão.

Os Scripts disponíveis para as Aplicações Remotas são descritos na tabela abaixo. Maiores detalhes a respeito do uso de Scripts, veja no capítulo específico.

Scripts Disponíveis

Você pode associar scripts a Aplicações Remotas executando-os na situação a seguir:

EVENTOS	DESCRIÇÃO
OnDisconnect	Executa o Script ao terminar a conexão.

Capítulo

15. Elipse Web

O módulo adicional Elipse Web permite disponibilizar as telas da aplicação na Web através de um servidor Web qualquer como por exemplo o PSW (Personal Web Server) ou o IIS (Internet Information Services) do Windows. O servidor Web precisa estar instalado na máquina onde está rodando a aplicação Elipse, e esta deve possuir um IP fixo caso se queira acessar a aplicação pela Internet.

Para configurar o Elipse Web, siga os seguintes procedimentos:

- Verifique se seu computador possui um servidor Web (PWS ou IIS). Em caso negativo, instale-o.
- O Elipse Web é um módulo adicional, portanto verifique se o hardkey possui este módulo.
- No Elipse SCADA, acesse o Organizer e no item Aplicação, clique na aba Web.
- Habilite o item "Habilitar servidor de dados para web", conforme a figura abaixo:

Figura 125: Aba Web

- Clique no botão [Localizar...] e especifique o diretório padrão do servidor Web. No caso do PWS ou IIS, o diretório padrão é `C:\Inetpub\wwwroot`.
- Clique no botão [OK].
- No item Porta, defina qual porta TCP/IP será utilizada para comunicação entre os applets e o supervisório.
- Feitas estas configurações, copie o arquivo "Applet.jar" da pasta `C:\...\Elipse SCADA\Applet\` para a pasta padrão do servidor Web.
- Em cada tela que você deseja visualizar pela Web, é necessário habilitar a criação da página HTML. Para tanto, acesse as propriedades da tela e clique na aba Web, conforme figura abaixo:

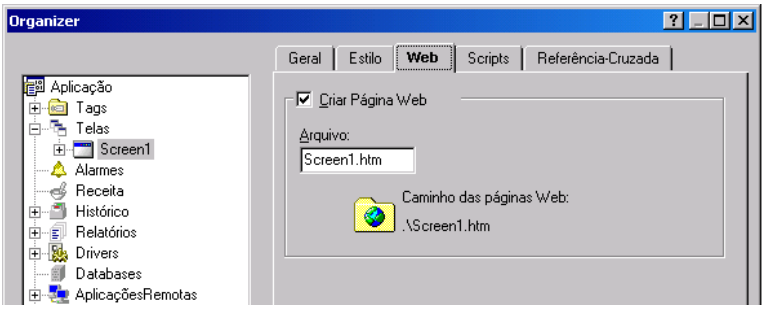


Figura 126: Aba Web acessada através das propriedades da Tela

- Habilite a opção Criar Página Web e especifique o nome da página html a ser criada. Estas páginas serão criadas no diretório padrão do servidor Web.
- Para visualizar a aplicação, basta digitar na barra de endereços do browser o endereço do servidor e o nome do documento HTML configurado (por exemplo: `http://NomedoServidor/Screen1.htm`).

NOTAS:

- O Elipse Web permite apenas a supervisão da aplicação. Não é possível controlar, enviar dados ou interagir com a aplicação.
- Será acessível somente a tela que estiver ativa na aplicação.

Capítulo

16

16. Watcher

O Elipse Watcher é um plug-in do Elipse SCADA que oferece captura, armazenamento e transmissão digital de imagens. Com ele, é possível visualizar imagens dentro de uma aplicação e trabalhar com elas como for desejado.

Cada placa de aquisição ligada ao Elipse SCADA é representada por um objeto do Watcher. Para acrescentar um equipamento, basta acrescentar um objeto correspondente na lista de objetos.

É possível configurar os parâmetros de cada um dos objetos independentemente. Estes objetos são posteriormente ligados a um objeto de tela (Preview, AVI ou Video) para sua utilização dentro da aplicação ou ligados a um objeto AVI Recorder, que permite a geração de um arquivo .AVI. O Elipse Watcher possui drivers para uma variedade de equipamentos do mercado. Consulte nosso departamento técnico para verificar a compatibilidade do seu com o Elipse SCADA.

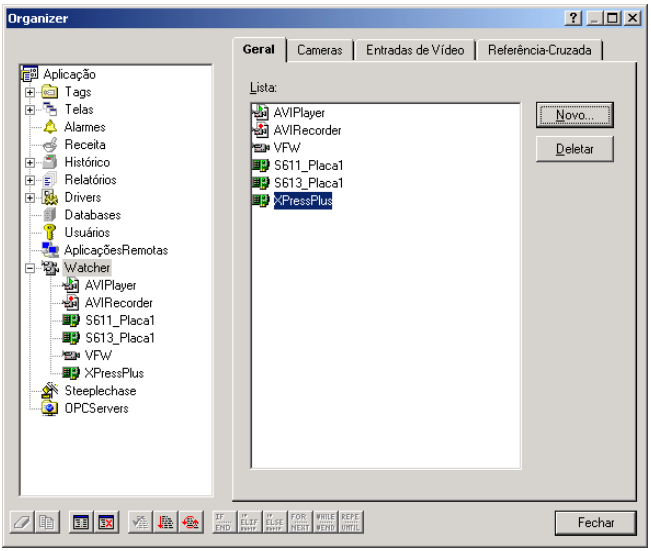


Figura 127: Watcher

16.1. Objetos do Watcher

Cada objeto do Watcher representa um dispositivo ou placa de aquisição conectado no sistema. Os objetos de captura de vídeo são ligados a "câmeras" (listadas na aba Cameras do Watcher), que representam a entrada de vídeo do objeto em questão. Quando configuramos objetos de tela para a apresentação de vídeo, temos que indicar a qual câmera o objeto estará ligado, isto é, qual entrada de vídeo ele irá receber.

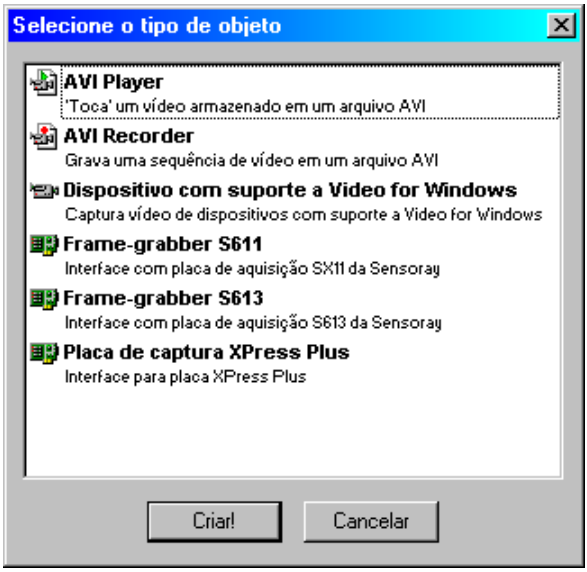


Figura 128: Tipos de objetos do Watcher

AVI Player

O objeto AVI Player permite a reprodução de um arquivo .AVI no sistema. Este objeto do Watcher é normalmente associado a um objeto de tela AVI.

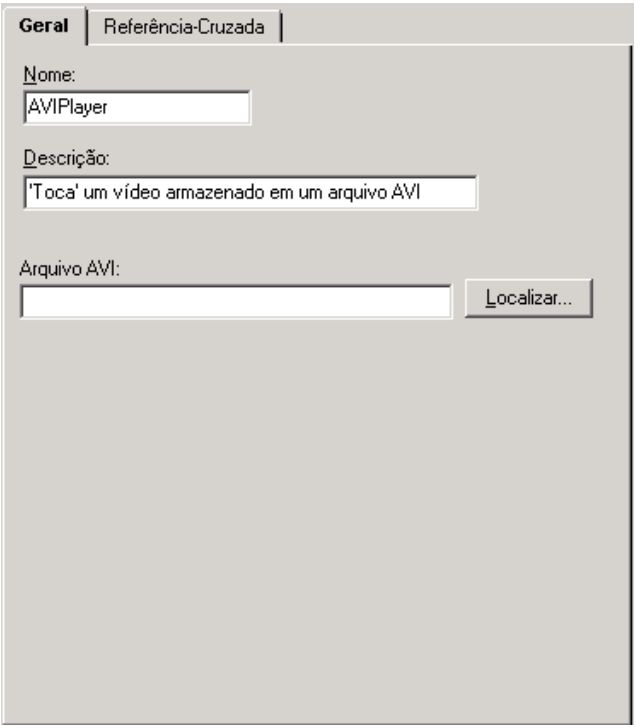


Figura 129: Propriedades do AVI Player

Propriedades do AVI Player

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do objeto.
Descrição	Uma breve descrição do objeto.
Arquivo AVI	Determina o caminho do arquivo AVI.
Localizar	Permite navegar pelo disco para indicar o arquivo AVI de origem.

AVI Recorder

O objeto AVI Recorder permite a gravação de uma entrada de vídeo (uma placa de aquisição, por exemplo) em um arquivo formato .AVI.

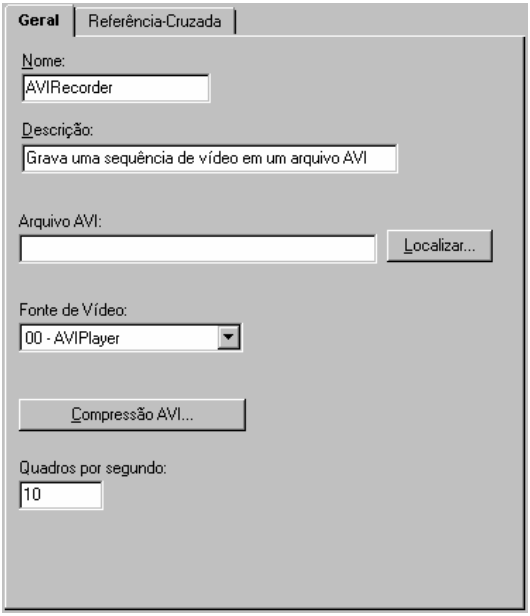


Figura 130: Propriedades do AVI Recorder

Propriedades do AVI Recorder

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do objeto.
Descrição	Uma breve descrição do objeto.
Arquivo AVI	Determina a localização e o nome do arquivo AVI a ser gerado.
Localizar	Permite navegar pelo disco para indicar o arquivo AVI de destino.
Fonte de Vídeo	Origem do vídeo a ser gravado.
Compressão	Determina como será feita a compressão do AVI, permitindo a escolha do compressor, qualidade e outras configurações.
Quadros por segundo	Determina a taxa de atualização com a qual será feita a gravação.

Frame Grabber S611

Através desta opção, é possível configurar as especificações referentes a interface com placa de aquisição (frame-grabber) S611 da Sensoray.

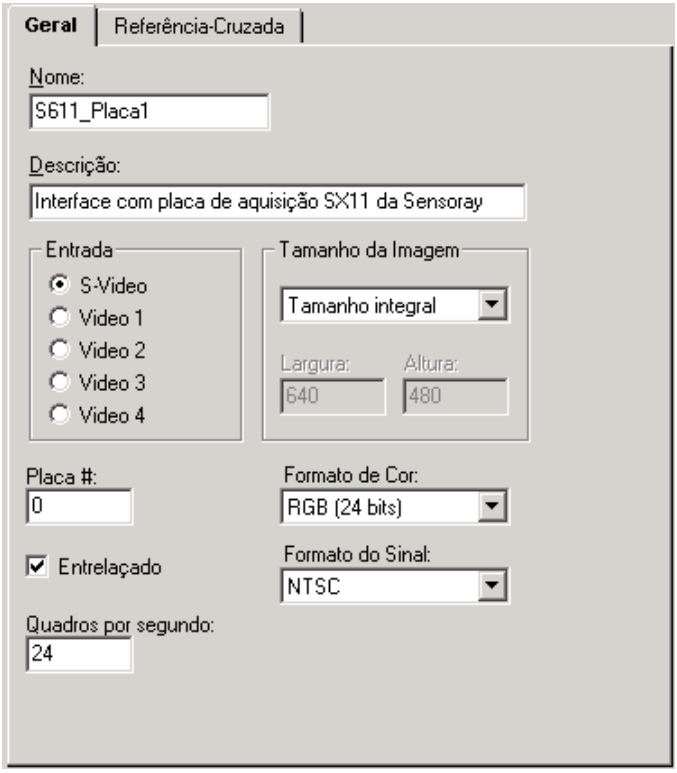


Figura 131: Propriedades da interface com a placa S611

Propriedades da interface com a placa S611

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do objeto.
Descrição	Uma breve descrição do objeto.
Entrada	Determina o tipo de entrada do video: S-Video, Video 1, Video 2, Video 3 ou Video 4.
Tamanho da imagem	Determina o tamanho da imagem: Tamanho integral, 3/4 do tamanho, 1/2 do tamanho, 1/4 do tamanho ou Avançado (determinando largura e altura).
Largura	Determina a largura da imagem. Esta opção só é válida se o ítem Avançado do Tamanho da imagem estiver habilitado.
Altura	Determina a altura da imagem. Esta opção só é válida se o ítem Avançado do Tamanho da imagem estiver habilitado.
Placa #	Determina as especificações da placa.
Formato de Cor	Determina o formato da cor do video: (RGB 24 bits ou Mono 8 bits)
Entrelaçado	Habilita a opção entrelaçado no sistema.
Formato do Sinal	Determina o formato do sinal do video: NTSC, NTSC Japão, PAL, PAL-M, PAL-N.
Quadros por segundo	Determina quantos quadros por segundo serão gravados no objeto.

Frame Grabber S613

Através desta opção, é possível configurar as especificações referentes a interface com placa de aquisição S613 da Sensoray.

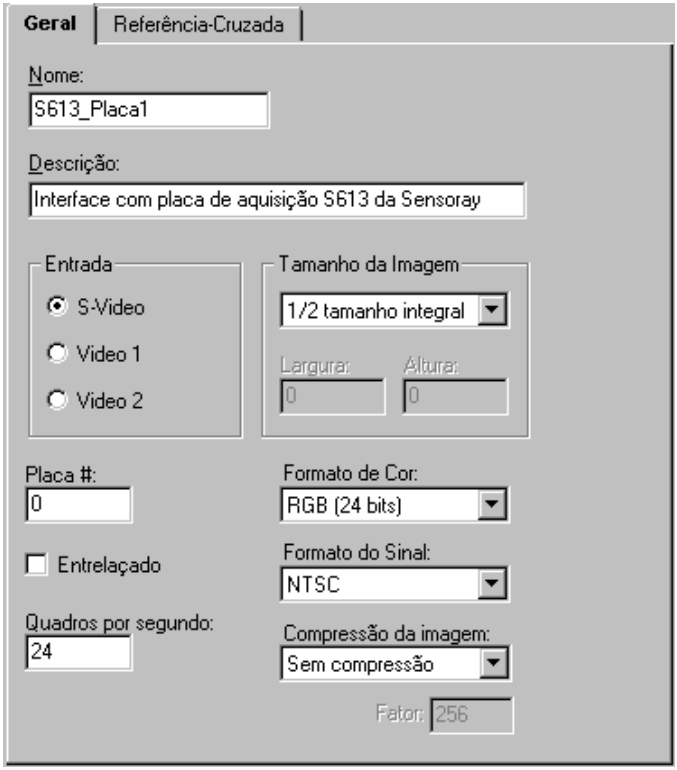


Figura 132: Propriedades da interface com a placa S613

Propriedades da interface com a placa S613

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do objeto.
Descrição	Uma breve descrição do objeto.
Entrada	Determina o tipo de entrada do video: S-Video, Video 1 ou Video 2.
Tamanho da imagem	Determina o tamanho da imagem: Tamanho integral, 3/4 do tamanho, 1/2 do tamanho, 1/4 do tamanho ou Avançado (determinando largura e altura).
Largura	Determina a largura da imagem. Esta opção só é válida se o ítem Avançado do Tamanho da imagem estiver habilitado.
Altura	Determina a altura da imagem. Esta opção só é válida se o ítem Avançado do Tamanho da imagem estiver habilitado.
Placa #	Determina as especificações da placa.
Formato de Cor	Determina o formato da cor do video: (RGB 24 bits ou Mono 8 bits)
Entrelaçado	Habilita a opção entrelaçado no sistema.
Formato do Sinal	Determina o formato do sinal do video: NTSC, NTSC Japão, PAL, PAL-M, PAL-N.
Quadros por segundo	Determina quantos quadros por segundo serão gravados no objeto.
Compressão da imagem	Determina como a imagem será comprimida: Sem compressão ou compressão M JPEG.
Fator	Determina o fator do MJPEG (Esta opção só é habilitada, se for selecionada o ítem MJPEG na opção acima).

Dispositivos com Suporte a Video for Windows

Através de esta opção, é possível configurar as especificações referentes a entrada de video através de dispositivos com suporte ao padrão Video for Windows.

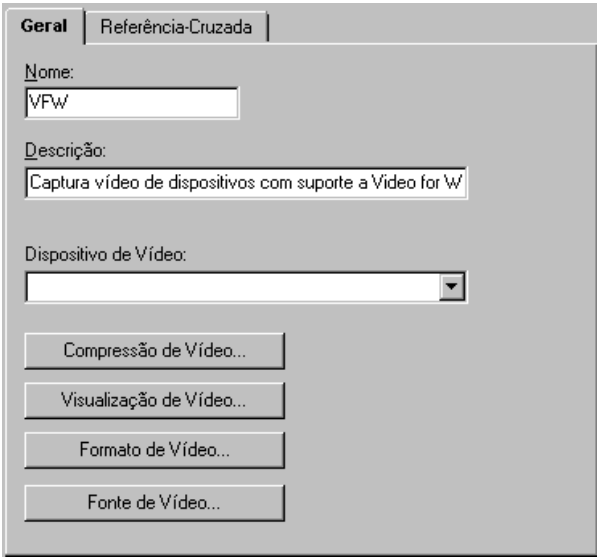


Figura 133: Propriedades do Dispositivo com Suporte a Video for Windows

Propriedades do Dispositivo com Suporte a Video for Windows

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do objeto
Descrição	Uma breve descrição do objeto.
Dispositivo de Video	Determina a o dispositivo de video habilitado no sistema.
Compressão de video	Habilita a compressão do video
Visualização de video	Habilita a visualização do video
Formato de video	Habilita a configuração do formato do video
Fonte de video	Determina as especificações referentes a fonte do video.

Placa de Captura Xpress Plus

Através desta opção, é possível configurar as especificações referentes à placa de captura XPressPlus da IntegralTech. Esta placa permite o controle de até 32 câmeras, que podem ser configuradas separadamente.

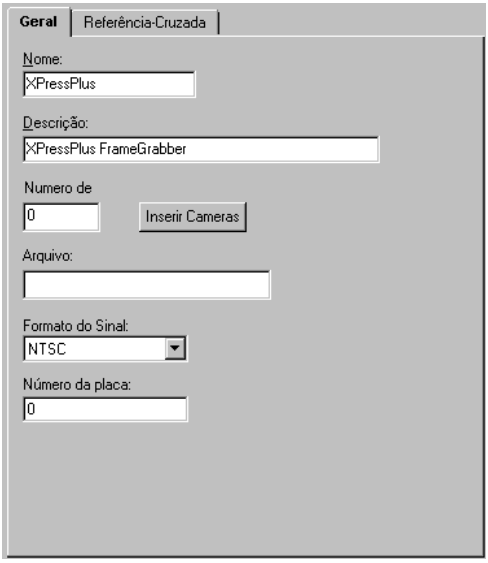


Figura 134: Propriedades da Plca a de captura Xpress Plus

Propriedades da Plca a de captura Xpress Plus

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do objeto.
Descrição	Uma breve descrição do objeto.
Número de	Especifica o número de câmeras que serão inseridas pelo botão Inserir Câmeras. Permite criar objetos que representarão as câmeras conectadas à placa.
Inserir Câmeras	Insere o número de objetos "Camera" especificado em Número de. A placa XPressPlus tem um limite de 32. As excedentes serão ignoradas.
Arquivo	Nome do arquivo (com o caminho completo) onde o vídeo será gravado.
Formato do Sinal	Formato do sinal que está sendo recebido.
Número da Placa	Número de identificação da placa.

Câmeras

Com a placa XPressPlus é possível gerenciar e ajustar opções de todas as câmeras conectadas, individualmente. As configurações são feitas através de um objeto Camera, que representa a câmera em questão. Estes objetos são criados através do botão Inserir Cameras, mostrado no item anterior. Cada câmera tem três conjuntos de opções, a saber: Geral, Opções de Gravação e Máscara.

a) Propriedades Gerais

Figura 135: Propriedades gerais de câmeras Xpress Plus

Propriedades gerais de câmeras Xpress Plus

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do objeto.
Descrição	Uma breve descrição do objeto.
Câmera	Determina a câmera em foco.
Brilho	Determina o brilho da imagem da câmera.
Saturação	Determina a saturação da imagem da câmera.
Contraste	Determina o contraste da imagem da câmera.
Hue	Determina o Hue da imagem da câmera.
Foco	Determina o foco da imagem da câmera.

b) Opções de Gravação

Através deste item é possível ajustar as opções de gravação da placa de captura.

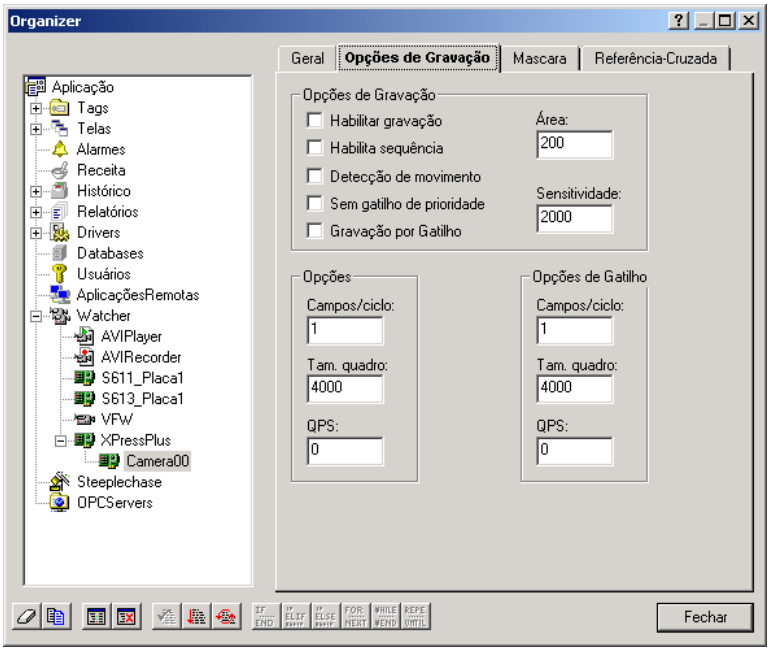


Figura 136: Propriedades das Opções de Gravação

Propriedades das Opções de Gravação

OPÇÃO	DESCRIÇÃO
Opções de gravação	Configura as opções de gravação da câmera: - Habilitar gravação: habilita a gravação na câmera selecionada. - Habilitar seqüência: habilita a gravação em sequência na câmera. - Detecção de movimento: habilita a gravação na câmera a cada detecção de movimento. - Sem gatilho de prioridade: habilita a gravação sem gatilho de prioridade. - Gravação por Gatilho: habilita a gravação por gatilho.
Área	Determina a área de gravação da câmera.
Sensitividade	Determina a sensitividade da gravação da câmera.
Campo/ciclo	Determina o campo/ciclo da gravação da câmera.
Tam.quadro	Determina o tamanho do quadro de gravação da câmera.
QPS	Determina o QPS da gravação da câmera.
Opções de gatilho	Determina as opções de gatilho da gravação da câmera.

c) Máscara

Definir uma máscara é indicar áreas de sensibilidade, isto é, detecção de movimento de cada entrada de vídeo. As áreas em vermelho representam os lugares não-sensíveis. As áreas em verde-claro, as áreas sensíveis.

Capítulo

17. Steeplechase

O objeto Steeplechase configura as especificações referentes ao Steeplechase, que é um SoftPLC ao qual o Elipse SCADA dá suporte. Através deste objeto, é possível a comunicação com este dispositivo.

Através de suas propriedades, é possível configurar o caminho pelo qual o sistema buscará as informações no servidor local ou da rede. Após configuradas as especificações, o sistema verifica a autorização da licença e ativa ou não a comunicação. Somente em algumas versões do Elipse SCADA esta opção é disponível.

As propriedades para esta opção são os seguintes:

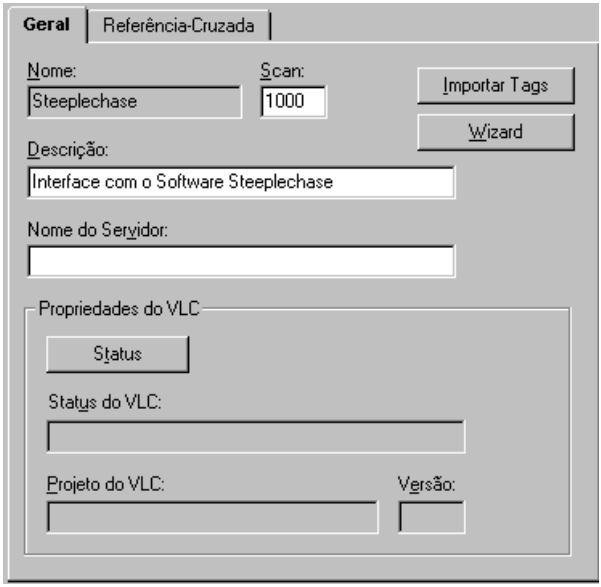


Figura 137: Steeplechase

Propriedades do Steeplechase

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do objeto.
Scan	Determina o valor do scan.
Importar Tags	Permite importar tags.
Wizard	Abre um wizard de configuração para o steeplechase.
Descrição	Uma breve descrição sobre o objeto.
Nome do Servidor	Determina o nome do servidor do steeplechase.
Status	Determina o status do objeto.
Status do VLC	Determina o status do VLC (Virtual Logic Controler).
Projeto do VLC	Determina o projeto do VLC (Virtual Logic Controler).
Versão	Determina a versão do objeto.

Capítulo

18. OPC Server

O objeto OPCServer é um cliente OPC (*OLE for Process Control*) que possibilita a comunicação com um determinado equipamento ou dispositivo, utilizando o protocolo OPC. O objeto OPCServer é a representação de um servidor OPC DA (*Data Access*) dentro do Elipse SCADA, o que permite o envio e recebimento de dados de tempo real (tags).

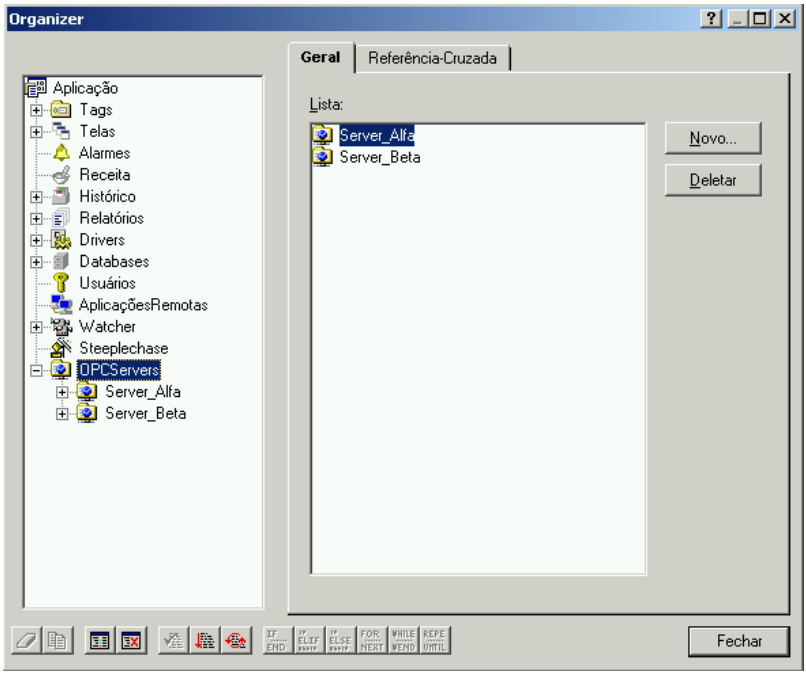


Figura 138: Objetos OPCServer no Organizer

18.1. Propriedades Gerais do OPC Server

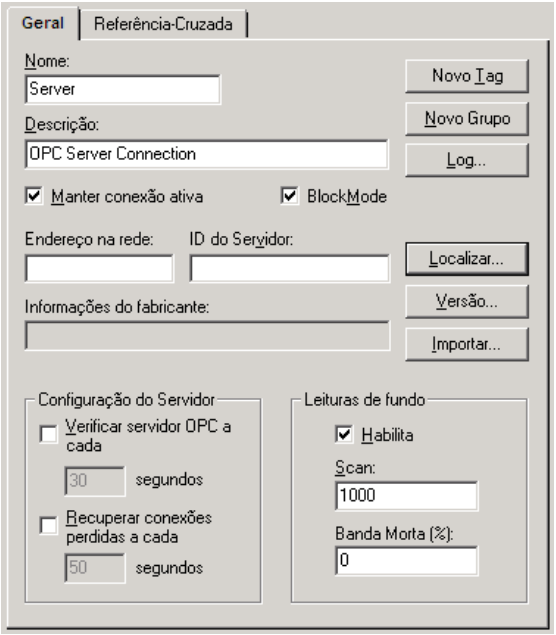


Figura 139: Propriedades Gerais do OPC Server

Propriedades Gerais do OPC Server

OPÇÃO	DESCRIÇÃO
Nome	Nome do objeto.
Descrição	Uma breve descrição sobre o objeto.
Manter conexão ativa	Quando habilitado, a conexão com o servidor é estabelecida uma vez e mantida até o encerramento da aplicação. Quando desabilitado, o Elipse SCADA irá estabelecer conexões com o servidor OPC toda vez que precisar solicitar um dado.
BlockMode	Quando habilitado, permite que tags OPC sejam adicionados ou removidos em blocos, aumentando a performance.
Novo Tag	Permite criar manualmente um novo tag OPC.
Novo Grupo	Permite agrupar tags em grupos, a fim de facilitar a sua organização.

OPÇÃO	DESCRIÇÃO
Log...	Permite habilitar o registro (log) do servidor e definir o nome e caminho do arquivo de log.
Endereço na rede	Endereço do servidor OPC na rede. Este campo permite a atribuição direta, quando não for possível navegar na rede para selecionar o servidor.
ID do servidor	Nome do servidor OPC.
Informações do fabricante	Apresenta as informações sobre o servidor OPC, registradas pelo fabricante.
Localizar...	Permite selecionar o servidor OPC dentre os servidores ativos.
Versão...	Informa a versão do servidor OPC.
Importar...	Permite a importação de tags ou grupos de tags definidos no servidor OPC. Obs.: este recurso pode não estar disponível no servidor OPC.
Verificar servidor OPC a cada ... segundos	Habilita a verificação do servidor OPC no período (em segundos) determinado no campo.
Recuperar conexões perdidas a cada ... segundos	Habilita a recuperação de conexões perdidas no período (em segundos) determinado no campo. Obs.: só é válida se a opção Verificar servidor OPC a cada ... segundos estiver habilitada.
Leituras de Fundo/Habilita	Habilita a leitura dos tags em segundo plano. É o modo mais eficiente de leitura, com um mínimo de comunicação entre o Elipse SCADA e o servidor OPC. Os dados só são atualizados quando ocorrerem variações maiores que a Banda Morta definida pelo usuário (ver opção Banda Morta).
Scan	Indica o tempo da atualização dos tags em segundo plano. Este valor poderá ser aproximado pelo servidor, de acordo com suas características internas. Um valor zero (0) requisita ao servidor um tempo de atualização menor possível.
Banda Morta (%)	Percentual que especifica a faixa de variação considerada não-significativa. Isto é, variações dentro desse percentual não são notificadas ao sistema pelo servidor. Um valor zero (0) por exemplo, mandaria qualquer variação. Obs.: a utilização dessa característica depende diretamente do servidor OPC utilizado. Consulte a documentação do servidor.

18.2. Tags OPC

Os tags OPC permitem a troca de informações com servidores OPC. Através deles, é possível enviar e receber dados ao servidor OPC.

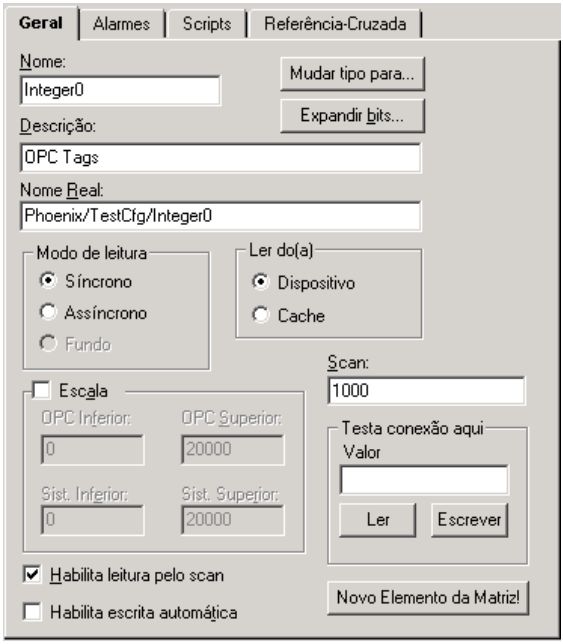


Figura 140: Propriedades de tags OPC

Propriedades de Tags OPC

OPÇÃO	DESCRIÇÃO
Nome	Determina o nome do tag OPC.
Descrição	Breve descrição sobre o tag.
Nome Real	Identificador do tag dentro do servidor OPC (caminho).

OPÇÃO	DESCRIÇÃO
Modo de Leitura	Especifica o modo de leitura dos tags no servidor OPC. Síncrono - faz a solicitação de leitura do tag e aguarda o retorno do servidor. Fundo (ou Segundo Plano) - os dados só são atualizados quando ocorrerem variações significativas (fora da Banda Morta). Veja as propriedades do servidor OPC.
Ler do (a)	Define qual a origem dos dados lidos do servidor OPC. Dispositivo - força a leitura o dado diretamente do dispositivo conectado ao servidor. Cache - solicita o valor armazenado no cache do servidor.
Escala	Marcando esta opção, os valores do Tag serão convertidos para uma nova escala de valores, determinada pelo usuário conforme os limites definidos em OPC Inferior, OPC Superior, Sist. Inferior e Sist. Superior.
OPC Inferior	Define o valor mínimo a ser lido do servidor OPC.
Sistema Inferior	Define o novo valor mínimo para a conversão dos valores lidos.
OPC Superior	Define o valor máximo a ser lido do servidor OPC.
Sistema Superior	Define o novo valor máximo para a conversão dos valores lidos.
Scan	Determina o valor do Scan.
Testar conexão aqui	Permite que você teste a configuração do Tag OPC, tanto para escrita quanto para leitura. Uma mensagem pode indicar um erro de conexão ou o valor recebido pelo item configurado.
Habilita a leitura pelo scan	Habilita a leitura periódica do tag OPC. O período é definido no campo Scan.
Habilita a escrita automática	Habilita escrita automática para o tag OPC.
Novo Elemento da Matriz!	Acrescenta elementos a um tag OPC tratado como uma matriz (*array*). Permite a leitura de variáveis definidas como matrizes no servidor.

Página de Alarmes

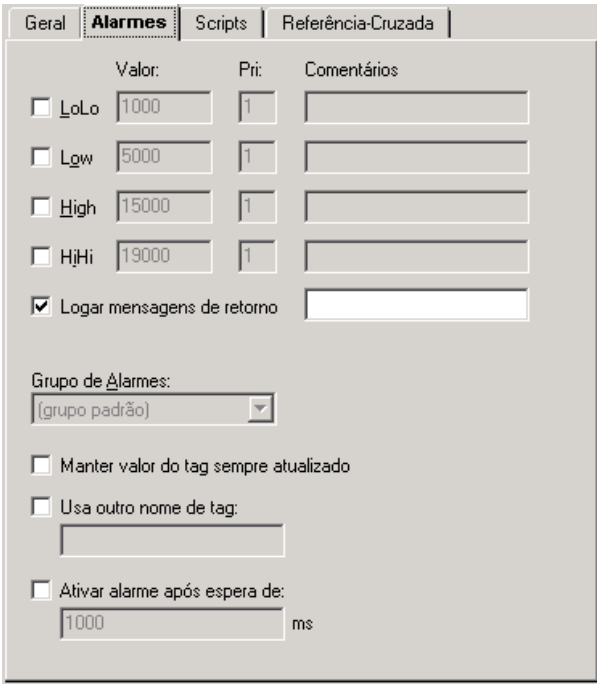


Figura 141: Página de alarmes

Propriedades da Página de Alarmes

OPÇÃO	DESCRIÇÃO
LoLo	Alarme Baixo Crítico. Define um intervalo de valores (menor igual) onde o Tag OPC é considerado em um estado de Alarme Baixo Crítico. É usado quando o valor do Tag OPC está abaixo de um mínimo, ou seja, extremamente baixo.
Low	Alarme Baixo. Define um intervalo de valores (menor igual) onde o Tag OPC é considerado em estado de alarme baixo. É usado quando o valor do Tag OPC está abaixo do normal.
High	Alarme Alto. Define um intervalo de valores (maior igual) onde o Tag OPC é considerado em estado de Alarme Alto. É usado quando o valor do Tag OPC está mais alto do que o normal.

OPÇÃO	DESCRIÇÃO
HiHi	Alarme Alto Crítico. Define um intervalo de valores (maior igual) onde o Tag OPC é considerado em estado de Alarme Alto Crítico. É usado quando o valor do Tag OPC é está acima de um máximo, ou seja, extremamente alto.
Valor	Define os limites para cada situação possível de alarme (lolo, low, hi, hihi).
Pri	Define a prioridade para cada situação de alarme. Números pequenos indicam alta prioridade (a prioridade deve ser um número entre 0 e 999). Para um melhor controle os alarmes de maior prioridade irão aparecer em primeiro plano na janela de alarmes (objeto de tela Alarme).
Comentários	Um comentário ou mensagem pode ser definido para cada alarme.
Logar mensagens de retorno	Habilita o registro (log) das mensagens de retorno de alarme.
Grupo de Alarmes	Define o grupo de Alarmes do tag corrente. O grupo de Alarmes deve ser definido na opção Alarmes do Organizer.
Manter o valor do tag sempre atualizado	Define que o sistema irá supervisionar o Tag OPC constantemente, mesmo que ele não esteja sendo utilizado em nenhum lugar da aplicação, a fim de não perder nenhum alarme deste tag.
Usa outro nome de tag	Permite definir um nome alternativo para o tag.
Ativar alarme após espera de	Ativa o alarme após o tempo (em milissegundos) determinado.

18.3. Grupo OPC

Os grupos OPC permitem uma organização visual dos tags OPC, facilitando sua manipulação. Em muitos casos, estes grupos são criados automaticamente durante a importação de tags do servidor, refletindo a estrutura hierárquica do próprio.

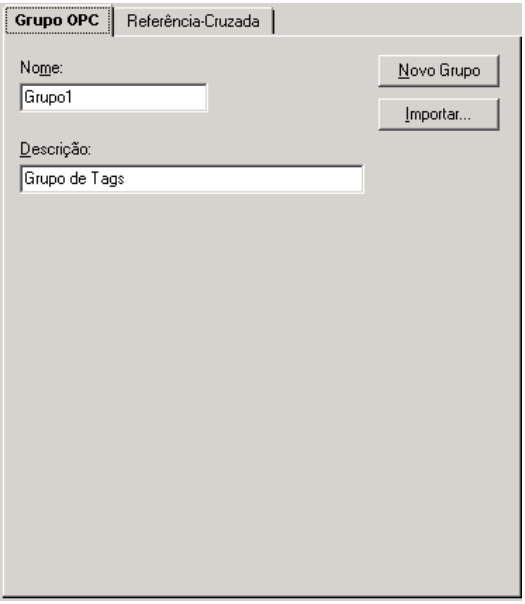


Figura 142: Propriedades de Grupos OPC

Propriedades do Grupo OPC

OPÇÃO	DESCRIÇÃO
Nome	Especifica o nome do objeto.
Descrição	Uma breve descrição do objeto.
Novo Grupo	Insere um novo Grupo OPC no objeto OPCServer.
Importar	Permite importar tags ou grupos de tags definidos no servidor OPC para o Grupo OPC em questão.

Quando é acionado o botão **Importar tags**, a janela abaixo é mostrada. A partir daí, basta arrastar o tag desejado para o grupo selecionado.

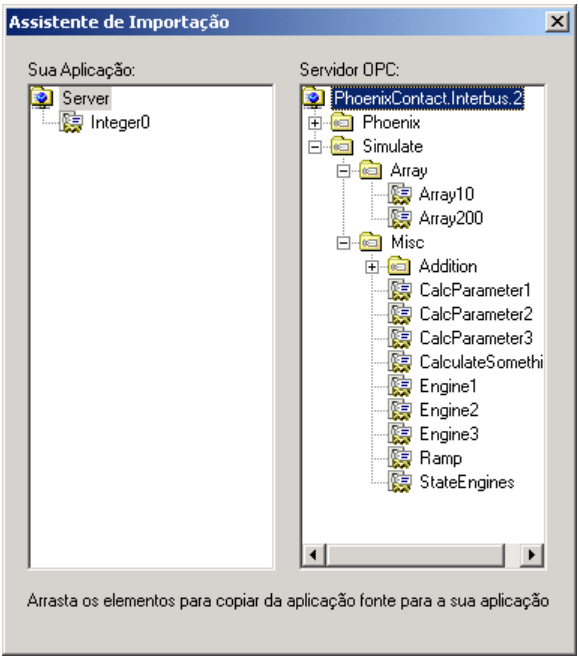


Figura 143: Assistente de importação

18.4. Qualidade

Os campos de qualidade representam o estado de qualidade do valor do item. É formado de uma palavra de 16 bits, sendo que os primeiros 8 bits são definidos na forma de 3 campos de bits: Campo Qualidade, Campo Substatus e Campo Limite.

Os outros 8 bits (de maior ordem) estão disponíveis para uso de cada fabricante. Se esses bits forem utilizados, os bits de qualidade padrão ainda são utilizados para indicar quais deduções pode-se fazer sobre os dados retornados. Assim, é de responsabilidade do cliente interpretar o campo de informações específicas de qualidade para garantir que o servidor que está provendo essa informação usa as mesmas "regras" que o cliente.

Campo Qualidade

QUALIDADE	CAMPO QUALIDADE	DESCRIÇÃO
0 – 63	Ruim	O valor não é usável pelas razões indicadas no substatus.
64 – 127	Incerta	A qualidade do valor é incerta pelas razões indicadas no substatus.
128 – 191	(Reservado)	Não é usado pelo padrão OPC.
192 – 255	Boa	A qualidade do valor é Boa.

Um servidor que não suporta informação de qualidade retorna 192 sempre (Bom). Também é aceitável que um servidor retorne simplesmente Ruim ou Bom (0 ou 192) e sempre retornar 0 para o Substatus e limite.

Capítulo

19. Scripts

Os **Scripts** são pequenos procedimentos escritos em linguagem de programação própria – Elipse Basic – que permitem uma maior flexibilidade na sua aplicação. O scripts são sempre associados a eventos, isto é, eles são iniciados no momento da ocorrência deste evento.

19.1. Considerações Gerais

Em qualquer linguagem de programação, é necessária a criação de métodos, de modo a especificar e ordenar a execução das instruções desejadas. A própria estrutura dos scripts do Elipse SCADA já organiza de certa maneira esta ordem, pois são orientados a **eventos.**

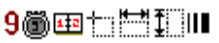
Os eventos são ocorrências relacionadas a um objeto, que podem ser tratadas de modo a se realizar uma ação específica. Eles podem ser **físicos**, como por exemplo, alguma ação no teclado ou no mouse. Em cada caso, temos diversas informações relevantes como a tecla pressionada ou a posição do cursor e o status dos botões. Os eventos podem ser **internos**, como a mudança do valor de uma variável. Estes eventos podem também ter associações físicas, como a mudança de uma temperatura de uma câmara de 10 para 11 graus quando temos um tag que recebe os valores dessa temperatura. O Elipse SCADA já tem diversos eventos pré-definidos disponíveis para a ligação ou associação de scripts. Exemplos de alguns desses eventos são listados a seguir.

Eventos comuns em aplicações no ElipseSCADA

OPÇÃO	DESCRIÇÃO
OnKeyPress	Quando uma tecla é pressionada
OnKeyRelease	Quando uma tecla é liberada
OnSetFocus	Quando um objeto recebe o foco de edição
OnLButtonDown	Quando o botão esquerdo é pressionado sobre um objeto
OnPress	Quando um objeto botão é pressionado
OnRelease	Quando um objeto botão é liberado
WhileRunning	Enquanto uma aplicação está executando
OnAlarm	Quando ocorre qualquer tipo de alarme

A linguagem utilizada nos módulos de script, o **Elipse Basic**, é bastante similar às linguagens C, porém com recursos de programação visuais como os encontrados no Visual Basic. Algumas características da linguagem:

- *Não é necessário a declaração de variáveis ou funções no início do Script*. As variáveis devem ser tags, objetos ou atributos previamente criadas ou importadas de outras aplicações. O Elipse SCADA já possui algumas variáveis de sistema pré-definidas.
- *O tipo de dado que se atribui a um tag é livre*. O valor suportado pode ser desde um inteiro de 8 bits até um tipo real de 64 bits ou ainda um string (texto). Em comunicação de dados com equipamentos externos, a conversão é feita automaticamente, de acordo com os tipos suportados pelo equipamento. No caso de propriedades, um ícone ao lado de cada uma (visualizado através do AppBrowser) indica o tipo de dado suportado:



Atributo numérico (número inteiro)

Atributo numérico (número real)

Atributo string (texto)

Atributo booleano (0 ou 1)

As **variáveis** e **constantes** são os objetos básicos manipulados num Script. Os **operadores** especificam o que será realizado com os mesmos. As **expressões** combinam variáveis e constantes para produzir novos valores. Para facilitar a edição de scripts ou de tags expressão, podem ser usadas as ferramentas AppBrowser e Referência Cruzada.

19.2. AppBrowser e Referência Cruzada

O **AppBrowser** permite navegar facilmente pela aplicação. Quando você seleciona um objeto na árvore ao lado esquerdo da janela, seus atributos e funções correspondentes são listados à direita.

Você pode usar o AppBrowser como referência durante a edição de um script. Uma característica bastante útil é a possibilidade de selecionar um objeto, atributo ou função que você deseja utilizar e copiar diretamente para o script pressionando o botão Copia no Script ->.

A ferramenta **Referência Cruzada** possui a mesma estrutura do AppBrowser com a diferença que quando você seleciona um objeto na árvore ao lado esquerdo da janela, suas respectivas referências é que são listadas à direita. Dê um duplo-clique sobre uma referência para ir ao objeto referido.

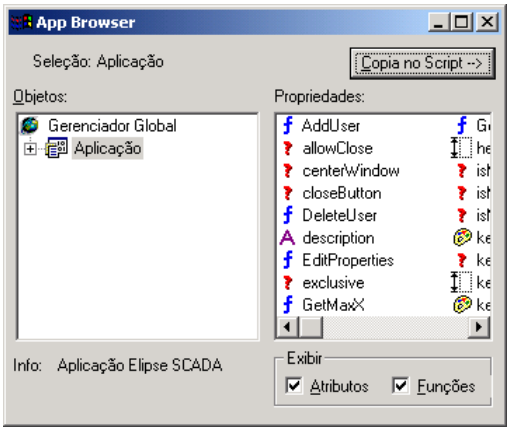


Figura 144: AppBrowser

Opções disponíveis no App Browser

OPÇÃO	DESCRIÇÃO
Seleção	Mostra o nome do objeto, atributo e função selecionado, da mesma forma que será copiado para o script.
Objetos	Lista dos objetos em ordem hierárquica.
Info	Mostra uma descrição do item selecionado.
Propriedades	Lista dos atributos e funções do objeto selecionado.
Exibir	Permite filtrar a informação (atributos ou funções) listada na janela de propriedades.
Copia no Script	Copia a referência do atributo ou função selecionado para o script.

19.3. Operadores e Constantes

Listamos as constantes (com suas notações) e operadores que podem ser utilizados nos scripts.

Constantes

TIPOS	EXEMPLO
Inteiros (32 bits, dec)	1234, 1234d, -993
Inteiros (32 bits, bin)	11001110b (não permite sinal)
Inteiros (32 bits, octal)	7733o (não permite sinal)
Inteiros (32 bits, hex)	0A100h, 3B8h (não permite sinal) (se o primeiro dígito é A-F, coloque um zero na frente)
Números reais (64 bits)	133.443, 344.939 (não tem notação científica)
Strings	"Temperatura", "pressão"

Operadores Aritméticos

OPERADOR	EXEMPLO
+ (adição)	tag001 + 34
+ (concatenação de strings)	slider1.Frame.title + "<- PLC1"
- (subtração)	tag001 - screen1.x
* (multiplicação)	screen1.width * 3.141592
/ (divisão)	tag001 / tag002
% (resto da divisão)	tag001 % tag002
** (exponenciação)	tag001 ** 2 (tag001 ao quadrado)

Operadores lógicos

OPERADOR	EXEMPLO
& (E bit-a-bit)	flags & 0F000h
| (OU bit-a-bit)	flags | 0F000h
^ (OU exclusivo bit-a-bit)	flags ^ 0F000h
~ (NÃO bit-a-bit)	~flags
<< (SHIFT à esquerda)	flags << 2 (desloca o valor de "flags" dois bits para a esquerda)
>> (SHIFT à direita)	flags >> 2 (desloca o valor de "flags" dois bits para a direita)
AND (E lógico)	tagOnOff AND (tag001 > 200)
OR (OU lógico)	tagOnOff OR (tag001 > 200)
XOR (OU exclusivo)	tagOnOff XOR (tag001 > 200)
NOT (negação)	NOT tagOnOff

Precedência de Operadores (Ordem de Execução)

A tabela abaixo lista as regras para precedência e associação de todos os operadores.

+ - ~ NOT (operadores unários)
**
/ %
+ -
>> <<
>= < <=
== <>
&
^
AND, XOR e OR
= += -= *= /= %= &= |= ^= **= <<= >>=

19.4. Controle de Fluxo

A fim de controlar a ordem na qual as instruções são processados o Elipse Basic disponibiliza uma série de comandos para fazer desvios e condições. Estes comandos são tratados a seguir.

19.4.1. Comando If...Else...ElseIf...EndIf

Permite a tomada de decisões durante a execução de um script.

Sintaxe:

```
If <condição1>
    <bloco de instruções 1>
Else
    <bloco de instruções n>
EndIf

If <condição1>
    <bloco de instruções 1>
ElseIf <condição2>
    <bloco de instruções 2>
Else
    <bloco de instruções n>
EndIf
```

As condições especificadas são expressões que podem ser avaliadas como verdadeiras (diferente de zero) ou falsas (zero ou string vazio ""). Se a condição for verdadeira o bloco de instruções é executado até o comando EndIf.

Quando este comando é executado, a condição do If (condição1) é avaliada primeiro, caso verdadeira o bloco de instruções 1 é executado encerrando o comando com o EndIf. Se a condição1 for falsa a condição do ElseIf (condição2) é avaliada e o bloco de instruções 2 é executado caso esta seja verdadeira. Se for falsa a próxima condição ElseIf será avaliada e assim por diante. Se nenhuma condição ElseIf for verdadeira o bloco de instruções do Else será executado. O programa continua sua execução com o comando após o EndIf.

Blocos de instruções ElseIf e Else são opcionais. Você pode especificar tantas cláusulas ElseIf desejar, entretanto elas nunca poderão estar após uma cláusula Else. Qualquer bloco de instruções pode conter comandos If…EndIf aninhados.

19.4.2. Comando For...Next

Repete um bloco de instruções um determinado número de vezes. A esta repetição damos o nome de “Laço” ou “Loop”.

Sintaxe:

```
For <contador> = <início> To <fim>
    <bloco de instruções>
Next
```

19.4.3. Comando While...Wend

Executa um bloco de instruções enquanto uma determinada condição é verdadeira.

Sintaxe:

```
While <condição>
    <bloco de instruções>
Wend
```

A condição especificada é uma expressão que pode ser avaliada como verdadeira (diferente de zero) ou falsa (zero ou string vazio ""). Se a condição for verdadeira o bloco de instruções é executado até a instrução Wend, quando, então, a condição é avaliada novamente. O bloco de instruções será repetido até que a condição seja falsa, quando o comando depois da instrução Wend será executado. Laços While...Wend podem ser aninhados.

19.4.4. Comando Repeat...Until

Executa um bloco de instruções até que determinada condição seja verdadeira.

Sintaxe:

```
Repeat
    <bloco de instruções>
Until <condição>
```

A condição especificada é uma expressão que pode ser avaliada como verdadeira (diferente de zero) ou falsa (zero ou string vazio ""). O bloco de instruções é executado e após a condição é avaliada. Se for falsa, o bloco de instruções é repetido e a condição é avaliada novamente. O bloco de instruções será repetido até que a condição seja verdadeira, quando então, o comando depois da instrução Until será executado. Laços Repeat...Until podem ser aninhados.

19.4.5. Comando Return

Encerra a execução imediatamente, retornando ao ponto de onde foi chamado o script.

19.5. Funções Especiais

O Elipse SCADA possui uma série de funções especiais pré-definidas que auxiliam na edição de scripts, facilitando a execução de tarefas mais complexas e permitindo uma melhor configuração do seu sistema.

Através da ferramenta AppBrowser podemos ver as diversas funções especiais disponíveis para cada objeto durante a edição de scripts.

Destacamos o objeto Gerenciador Global, que traz funções de utilidade geral, como funções de datas e do relógio de tempo-real, manipulação de strings e conversões numéricas, arquivos, multimídia e outras.

Além deste, temos diversos outros objetos que possuem funções específicas relacionadas: Aplicação, Tags, Telas, Objetos de Tela, Alarmes, Receitas, Históricos, Relatórios, Consulta, Plotagem, Drivers, Databases e Aplicações Remotas.

Este capítulo apresenta a lista completa de todas as funções disponíveis no Elipse SCADA, divididas por objeto, em ordem alfabética. Cada entrada indica o protótipo da função com seus parâmetros entre parênteses. Os parâmetros entre colchetes são opcionais. Seguindo a entrada da função, há a explicação de que ela faz e os seus parâmetros.

19.5.1. Funções do Gerenciador Global

Abs(x)
Retorna o valor absoluto de X.

Exemplo:

```
tag001 = Abs(-5.14) // Retorna 5.14
```

ACos(x)
Retorna arco cosseno (ângulo) do valor x no intervalo [0, π] radianos. O parâmetro x deve estar contido no intervalo [-1, 1], caso contrário a função retorna zero.

Asc(string)
Retorna o código ASCII para o primeiro caractere de string.

Exemplo:

```
tag001 = Asc("screen") // Retorna 115
```

ASin(x)
Retorna arco seno (ângulo) do valor x no intervalo [-π/2, π/2] radianos. O parâmetro x deve estar contido no intervalo [-1, 1], caso contrário a função retorna zero.

ATan(x)
Retorna arco tangente (ângulo) do valor x no intervalo [-π/2, π/2] radianos.

Blue(Cor)
Retorna o valor do componente azul de Cor.

Ver função **Red()**.

CaptureAppScreen([nomedoarquivo],[nomedatela],[tipo])

Captura a tela especificada no parâmetro nomedatela e salva-a em um arquivo de formato BMP com nome especificado em nomedoarquivo. Se o parâmetro nomedoarquivo a for em branco (" "), a captura é colocada na área de transferência.

O parâmetro tipo determina como a tela será capturada: se o parâmetro for 0, toda a tela é capturada, inclusive a barra de tarefas do Windows; se for 1 (ou qualquer valor diferente de 0), apenas a área que abrange os objetos da tela é capturada (o tamanho fica sendo o contorno ao redor de todos os objetos de tela).

Exemplo:

```
IF CaptureAppScreen("tela.bmp", "Tela1",1)
    MessageBox("Tela1 capturada no arquivo tela.bmp")
ELSE
    MessageBox("Erro capturando a tela!")
ENDIF
IF CaptureAppScreen("", "Tela1",1)
    MessageBox("Tela1 capturada para a área de transferência")
ELSE
    MessageBox("Erro capturando a tela!")
ENDIF
```

CaptureScreen([nomeArquivo])

Captura a toda a tela do sistema colocando-a na área de transferência (se nenhum arquivo for especificado no parâmetro nomeArquivo) ou em um arquivo com nome nomeArquivo, no formato BMP.

Exemplo:

```
IF CaptureScreen("tela.bmp")
    MessageBox("Tela capturada no arquivo tela.bmp")
ELSE
    MessageBox("Erro capturando a tela!")
ENDIF
```

Chr(n)

Retorna o caractere representado pelo código ASCII especificado.

Exemplo:

```
tag001 = Chr(115) // retorna "s"
```

CopyFile(arqFonte, arqDestino)
Copia os conteúdos do arquivo arqFonte para o arqDestino. A função retorna 1 (um) se foi bem sucedida e 0 (zero) caso contrário. Os atributos globais lastError e lastErrorStr são atualizados por essa função.

Exemplo:

```
IF CopyFile("source.dat", "dest.dat")
    MessageBox ("A cópia foi bem sucedida.")
ELSE
    MessageBox("Erro no comando CopyFile!")
ENDIF
```

CopyObject(strSrcName, strDestName [, strDestFolder])
Cria uma cópia do objeto strSrcName com o nome strDestName. O novo objeto é criado na mesma pasta do objeto-fonte como padrão, a não ser que o destino seja indicado no parâmetro opcional strDestFolder. A função retorna True se for bem sucedida ou False caso contrário.

Exemplo:

```
// Cria uma cópia do objeto Display1 na pasta
// Screens.Screen1; o novo objeto será chamado Display2
CopyObject ("Screens.Screen1.Display1", "Display2")

// Cria uma cópia de Display1 na pasta Screens.Screen1
// o novo objeto será chamado Display2 e será colocado
// na pasta Screens.Screen2
CopyObject ("Screens.Screen1.Display1",
    "Display2", "Screens.Screen2")
```

Cos(angle)
Retorna o cosseno do ângulo especificado. O ângulo angle é expresso em radianos.

CreateDir(dir)
Cria um novo diretório, especificado no parâmetro dir. Retorna TRUE se for bem sucedida ou FALSE se falhar. Os atributos globais lastError e lastErrorStr são atualizados por essa função.

CreateFile(nomeArquivo, string)
Cria um arquivo chamado nomeArquivo, escreve o texto string no arquivo e fecha o arquivo. Se o arquivo já existir, ele é destruído e substituído pelo novo conteúdo. Retorna FALSE em erro ou TRUE caso contrário.

DDEExecute(servidor, tópico, comando)

Envia um comando DDE para outra aplicação. O parâmetro servidor é o nome do servidor DDE, normalmente o nome de uma aplicação, o parâmetro topico é um documento aberto na aplicação e o parâmetro comando é o comando que você quer executar.

Exemplo:

```
DDEExecute("Excel", "Sheet1", '[SELECT("R1C1:R5C1")]')
DDEExecute("Excel", "Sheet1",
"[PRINT(1;;;1;FALSE;FALSE;1;FALSE;1;360;360)]")
```

O primeiro exemplo seleciona a região a ser impressa, o segundo executa o comando de impressão. O primeiro parâmetro é sempre o nome da aplicação. O segundo parâmetro é a planilha ou documento que você quer acessar, incluindo o caminho e a extensão do arquivo. O terceiro parâmetro é o comando.

Deg(angle)

Retorna o ângulo angle, expresso em graus, convertido para radianos, segundo a fórmula degrees = (radians / 3.141592) * 180.

Exemplo:

```
tag001 = Deg(ACos(x)) // Arcosseno de x expresso em graus
```

DeleteFile(arquivo)

Apaga o arquivo especificado no parâmetro arquivo. Retorna TRUE se for bem sucedida ou FALSE se falhar. Os atributos globais lastError e lastErrorStr são atualizados por essa função.

DeleteObject(strNomeObjeto)

Apaga o objeto strNomeObjeto da aplicação.

EditProperties()

Abre uma janela que permite a edição das propriedades do Gerenciador Global.

Execute(CmdLinha)

Executa o programa especificado no parâmetro CmdLinha. Se o programa não está localizado no diretório do \WINDOWS ou \WINDOWS\SYSTEM, você deve especificar a localização do arquivo. Parâmetros do programa também podem ser incluídos na linha de comando CmdLinha.

Exemplo:

```
Execute("C:\WINDOWS\WINCALC.EXE")
Execute("C:\UTIL\PKZIP.EXE DATA C:\DATA\*.*")
```

Esta função não aceita parâmetros no CE. Use a função **ShellExecute()**.

Exp(x)
Retorna a função exponencial de x.

Exemplo:

```
tag001 = Exp(2.302585093)    // retorna 10.0
```

FileSelectWindow([filepath][, extension][, type])
Mostra uma janela para a seleção de um arquivo. Retorna um string contendo o nome do arquivo selecionado ou um string vazio, se falhar ou for cancelado pelo usuário. O parâmetro filepath especifica o arquivo ou diretório que estará previamente marcado. O parâmetro extension especifica os tipos de arquivos que poderão ser selecionados. Sua sintaxe é:

```
filterName1 | mask1 | …  filterNameN | maskN
```

Cada filtro é uma seqüência de dois strings separados por um '|'. O primeiro string é o nome do filtro que será mostrado na caixa combo e o segundo string é a máscara de seleção de arquivos (*.dat, por exemplo). A máscara pode conter uma seqüência de máscaras, separadas por ponto-e-vírgula (por exemplo, *.bmp;*.jpg;*.gif). Os filtros devem estar separados por '|'.

Exemplo:

```
dim strDir
strdir = FileSelectWindow("C:\temp\*.dat",_
    "Arquivos de Historico (*.dat)|*.dat")
```

O parâmetro type é um inteiro que especifica a maneira que o arquivo será selecionado, de acordo com a tabela:

VALOR	DESCRIÇÃO
0	Abre uma janela de seleção de arquivo; não permite que o usuário digite nada, só escolher os arquivos existentes.
1	Permite o usuário criar um novo arquivo, digitando o nome do arquivo a ser criado.
2	Permite o usuário digitar o nome do arquivo desejado e retorna o caminho (path) completo; não cria o arquivo.

FillString(string, nCount)
Retorna um string contendo o parâmetro string tantas vezes quantas especificadas em nCount.

Exemplo:

```
// retorna "PasswordPassword" em String
String = FillString("Password", 2)
```

FindFirstFile(nome[, atributos])

Procura pelo primeiro arquivo a partir de um determinado padrão de nome e atributos específicos, retornando o nome do arquivo encontrado. O atributo nome determina o padrão de arquivo a ser procurado (por exemplo: "\DATA*.DAT") e o parâmetro atributos é opcional e determina a soma dos atributos desejados no arquivo, segundo a tabela:

VALOR	ATRIBUTO
0	Normal, sem atributos
1	Somente leitura (read only)
2	Oculto (hidden)
4	Arquivo de sistema (system)
16	Diretório (directory)
32	Arquivo-morto (archive)

Por exemplo, para colocar o atributo System (arquivo do sistema) e Hidden (oculto), deve atribuir o numeral 6, isto é: system (4) + hidden (2) = 4+2 = 6.

A função retorna o nome do primeiro arquivo que satisfaz os parâmetros ou vazio ("") caso nenhum arquivo seja encontrado. Após chamar FindFirstFile() a função FindNextFile() pode ser chamada para encontrar outros arquivos que satisfaçam essas condições.

Se o parâmetro atributos não for especificado, os atributos não são considerados.

FindNextFile()

Procura pelo próximo arquivo que obedece os parâmetros especificados na função FindFirstFile(). Retorna o nome do arquivo que satisfaz os parâmetros, ou vazio ("") caso nenhum arquivo seja encontrado.

Exemplo:

```
// Este exemplo faz uma cópia de segurança de todos os
// arquivos *.DAT no diretório C:\DADOS que tenham o
// atributo arquivo-morto (archive) para C:\BACKUP
DIM strFile
strFile = FindFirstFile ("C:\Dados\*.dat",32)
WHILE strFile<>""
    CopyFile ("C:\Data\"+strFile,"C:\Backup\"+strFile)
    strFile = FindNextFile()
WEND
```

FolderSelectWindow([filepath]) 

Abre uma janela para seleção de um diretório (pasta), retornando um string com o nome do diretório selecionado ou um string vazio, se for cancelada. O parâmetro filepath especifica o diretório inicial a ser selecionado. Se não for especificado, a seleção é o diretório-raiz do drive corrente.

Exemplo:

```
strFile = FolderSelectWindow ("C:\Dados")
```

Format(strFormato, valor)

Retorna um valor no formato data/hora de acordo com a string strFormato. O parâmetro valor é o próprio valor a ser formatado.

As opções para esta função são as seguintes:

Formatos Data/Hora para a função Format

FORMATO	SIGNIFICADO
w	Dia da semana (Dom – Sab)
W	Dia da semana (DOM – SAB)
ww	Dia da semana (Domingo – Sábado)
WW	Dia da semana (DOMINGO – SÁBADO)
d	Dia (1 – 31)
dd	Dia (01 – 31)
m	Mês (1 – 12)
mm	Mês (01 – 12)
mmm	Mês (Jan – Dez)
MMM	Mês (JAN – DEZ)
mmmm	Mês (Janeiro – Dezembro)
MMMM	Mês (JANEIRO – DEZEMBRO)
mmmmm	Mês (j – d)
MMMMM	Mês (J – D)
yy	Ano (00 – 99)
YY	Ano (1970 – 9999)
h	Hora (0 – 23)
hh	Hora (00 – 23)
m	Minuto (0 – 59) (deve estar acompanhando horas ou seguido de segundos)
mm	Minuto (00 – 59) (deve estar acompanhando horas ou seguido de segundos)
s	Segundo (0 – 59)
ss	Segundo (00 – 59)
AM/PM	Hora no formato 12 horas, mostra AM (manhã) e PM (tarde).
am/pm	Hora no formato 12 horas, mostra am e pm.
A/P	Hora no formato 12 horas, mostra A e P.
a/p	Hora no formato 12 horas, mostra a e p.
0	Décimos de segundo (deve ser precedido de segundos)
00	Centésimos de segundo (deve ser precedido de segundos)
000	Milésimos de segundo (deve ser precedido de segundos)
|	Nova linha (CR + LF)

Exemplo:

```
DIM date = GetTime()
// Retorna "November 21, 2000"
str = Format("mmmm d, yyyy",date)
// Retorna "11:41:32.612"
str = Format("hh:mm:ss.000",date)
```

GetAbsoluteFilename(arquivo)
Retorna o nome completo do arquivo.

Exemplo:

```
// Retorna "C:\WINDOWS\SYSTEM\VGA.DRV"
strArquivo = GetAbsoluteFilename("VGA.DRV")
```

GetAppDir() 
Retorna o diretório da aplicação.

GetCurDir() 
Retorna o diretório corrente.

GetDay(time)
Obtém o dia a partir de um valor de tempo absoluto.

Exemplo:

```
day = GetDay(GetTime())
// Exemplo abaixo retorna 30
day = GetDay(MakeTime(30,10,1996,17,25,56))
```

GetDayOfWeek(dataHora)
Obtém o dia da semana a partir de um valor de tempo absoluto (1 para domingo, 2 para segunda, …, 7 para sábado).

Exemplo:

```
day = GetDayOfWeek(GetTime())
// Exemplo abaixo retorna 5
day = GetDayOfWeek(MakeTime(30,10,1997,17,25,56))
```

GetDayOfYear(dataHora)
Obtém o dia do ano a partir de um valor de tempo absoluto em dias (1 para 1o. de jan, 2 para 2 de jan, …, 32 para 1o. de fev, 33 para 2 de fev, …).

Exemplo:

```
day = GetDayOfYear(GetTime())
day = GetDayOfYear(MakeTime(9,2,1997,17,25,56))
// retorna 40
```

GetDiskFreeSpace(strDiretório[,bEspacoLivreUsuario])
Retorna o número total de Kbytes disponíveis no caminho strDiretório especificado. O parâmetro bEspacoLivreUsuario é opcional e indica ao Elipse SCADA que ele deve considerar a quota do usuário, quando aplicável.

GetFileAttributes(arquivo[, versão])
Retorna a soma dos atributos de *arquivo*, de acordo com a seguinte tabela:

ATTRIBUTOS	VALOR
Erro	-1
Normal	0
Somente Leitura (Read Only)	1
Oculto (Hidden)	2
Sistema (System)	4
Volume	8
Diretório (Directory)	16
Arquivo-morto (Archive)	32

O parâmetro *versão* é opcional. Se ele for especificado (*versão* = 1), os seguintes atributos NTFS estarão disponíveis:

ATTRIBUTOS	VALOR
Arquivo comprimido	2048
Arquivo off-line	4096
Arquivo não-indexado pelo serviço de indexação	8192
Arquivo encriptado	16384

GetFileCreationTime(arquivo)
Retorna a hora em que o arquivo foi criado, através fo parâmetro arquivo.

GetFileLastAccessTime(arquivo)
Retorna a hora em que o arquivo foi acessado pela última vez, através do parâmetro arquivo.

GetFileLastModifiedTime(arquivo)
Retorna a hora da última modificação no arquivo através do parâmetro arquivo.

GetFileSize(arquivo)
Retorna o tamanho do arquivo em bytes, através do parâmetro arquivo.

GetHaspVar(strNomeVar) 
Lê o valor de uma variável interna armazenada em uma chave de hardware (Hasp). Esta variável só pode ser programada pela Elipse Software no Hasp. A função retorna uma string contendo o valor da variável ou em vazio ("") se ocorrer algum erro ou a variável não existir.

Exemplo:

```
tagSerialNo = GetHaspVar("SerialNo")
```

GetHour(time)
Obtém a hora a partir de um valor de tempo absoluto.

Exemplo:

```
hour = GetHour(GetTime())
hour = GetHour(MakeTime(30,10,1996,17,25,56))
// Retorna 17
```

GetLastError()
Retorna o código de erro da última operação no arquivo (o mesmo valor do atributo global lastError).

GetLicenseString()
Retorna uma string única que representa a licença do Elipse SCADA que está sendo utilizada.

GetMinute(time)
Obtém os minutos a partir de um valor de tempo absoluto.

Exemplo:

```
minute = GetMinute(GetTime())
minute = GetMinute(MakeTime(30,10,1996,17,25,56))
// retorna 25
```

GetMonth(time)
Obtém o mês a partir de um valor de tempo absoluto.

Exemplo:

```
month = GetMonth(GetTime())
month = GetMonth(MakeTime(30,10,1996,17,25,56))
// retorna 10
```

GetSecond(time)
Obtém os segundos a partir de um valor de tempo absoluto.

Exemplo:

```
second = GetSecond(GetTime())
second = GetSecond(MakeTime(30,10,1996,17,25,56))
// retorna 56
```

GetTime()
Retorna a hora atual do sistema como um valor de tempo absoluto medido em segundos desde 00:00 do dia 1o. de janeiro de 1970. O valor também expressa milisegundos nas três primeiras casas decimais.

Exemplo:

```
// Este exemplo conta quantos segundos leva para
// fazer uma cópia de um arquivo
DIM timeInicio, timeDuracao
timeInicio = GetTime()
CopyFile("C:\Dados\Temp.dat","C:\Dados\Backup\Temp.dat")
timeDuracao = GetTime() – timeInicio
MessageBox ("A cópia durou "
    + Str (timeDuracao,8,3) + " segundos.")
```

GetYear (time)
Obtém o ano a partir de um valor de tempo absoluto.

Exemplo:

```
year = GetYear(GetTime())
year = GetYear(MakeTime(30,10,1996,17,25,56))
// retorna 1996
```

Green (cor)
Retorna o valor do componente verde de Cor. Ver *Red()*.

HashString (strEntrada, strSenha)
Codifica a string passada como parâmetro (*strEntrada*) utilizando a senha fornecida (*strSenha*). O resultado é uma string única para a combinação (*strEntrada, strSenha*). O cálculo é irreversível: não é possível obter a string original e nem a senha a partir da string retornada.

Hex (n)
Retorna uma string com a representação hexadecimal de n.

Exemplo:

```
tag001 = Hex(31) // retorna "1F"
```

HexToDec (stringHexa)
Retorna um número inteiro com o valor correspondente a stringHexa.

Exemplo:

```
tag001 = HexToDec("1F") // retorna "31"
```

Int (x)
Retorna a parte inteira de x, obtida por truncagem.

Exemplo:

```
tag002 = -9,9
tag001 = Int (tag002) // retorna -9
```

IsNumeric (expressão)
Verifica se o resultado da expressão informada no parâmetro expressão é um número. Retorna TRUE (diferente de zero) se for um número ou FALSE (zero) se não for.

Exemplo:

```
tag001 = IsNumeric(1000)        // retorna TRUE (1)
tag001 = IsNumeric("teste")     // retorna FALSE (0)
tag001 = IsNumeric(GetTime())   // retorna TRUE (1)
```

IsString (expressão)
Verifica se o resultado da expressão informada no parâmetro expressão é um string. Retorna TRUE (diferente de zero) se for um número ou FALSE (zero) se não for.

Exemplo:

```
tag001 = IsString(1000)         // retorna FALSE (0)
tag001 = IsString("teste")      // retorna TRUE (1)
tag001 = IsString(GetTime())    // retorna FALSE (0)
```

Left (string, nCount)
Retorna os nCount caracteres mais à esquerda do parâmetro string.

Exemplo:

```
String = Left('Password', 4) // retorna 'Pass'
```

Len (string)
Retorna o número de caracteres de um string.

Exemplo:

```
Integer = Len('Password')  // retorna 8
```

Log(x)
Retorna o logaritmo natural do número x.

Exemplo:

```
tag001 = Log(tag002)
```

Log10(x)
Retorna o logaritmo base-10 do número x.

Exemplo:

```
tag001 = Log10(tag002)
```

MakeLower(string)
Retorna o parâmetro string em letras minúsculas.

Exemplo:

```
String = MakeLower('Password') // Retorna 'password'
```

MakeReverse(string)
Inverte um string, de forma que o primeiro caractere se torne o último e vice-versa.

Exemplo:

```
String = MakeReverse('Password') // Retorna 'drowssaP'
```

MakeTime (day, month, year, hour, minute, second)
Retorna um valor de tempo absoluto a partir dos parâmetros especificados. O intervalo de valores para cada parâmetro é:

Intervalo	Descrição
day	1 a 31 (pode ser 28, 29 ou 30 conforme o mês e o ano)
month	1 a 12
year	1970 a 2039
hour	0 a 23
minute	0 a 59
second	0 a 59

Se algum dos parâmetros for inválido a função retorna 0.

Exemplo:

```
myTime = MakeTime(30,10,1996,17,25,56)
```

MakeUpper (string)
Retorna o parâmetro string em letras maiúsculas.

Exemplo:

```
String = MakeUpper('Password')  // retorna 'PASSWORD'
```

Max (x)
Compara x e y e retorna o maior valor entre as duas expressões numéricas.

Exemplo:

```
tag001=Max(tag002,tag003)
```

MessageBox (texto[, título[, Estilo]][,x][,y])
Mostra uma caixa de mensagem que pode ser configurada pelo usuário. A função retorna um valor correspondente a um botão pressionado na caixa de mensagem ou no "Keypad".

Parâmetros da Função

O parâmetro Text é obrigatório, os parâmetros Title e Style são opcionais e seus valores default (0000h) determinam uma caixa de mensagem padrão.

texto: Deve ser um string ou um Tag contendo um string. Deve ser expresso entre aspas simples e aparecerá centralizado na caixa de mensagem logo acima dos botões.

Exemplo:

```
MessageBox ("Como está a Caixa de Mensagem?")
// Um string e' atribuido a um Tag
Mensagem = "Como está a Caixa de Mensagem?"
MessageBox(strMensagem)
```

Título: Deve ser um string da mesma forma que o parâmetro Text e irá aparecer na barra de título da caixa de mensagem. Exemplo:

```
MessageBox ("Como está a caixa de Mensagem?",
     "Teste da Caixa de Mensagem")
// Dois strings abaixo sao atribuidos a Tags
strText="Como está a caixa de Mensagem?"
strTitle="Teste da Caixa de Mensagem"
MessageBox(strText,strTitle)
```

x, y: Permite especificar a posição em que o diálogo irá aparecer.

Estilo: Permite modificar o estilo da caixa de mensagem e deve ser um valor em hexadecimal conforme as características que você deseja. As seguintes características podem ser configuradas.

Tipos da janela

TIPO	HEXA	DESCRIÇÃO
Application Modal	0000h	Você deve responder a caixa de mensagem para seguir usando a janela que a chamou. Entretanto, você pode ir para qualquer outra janela.
System Modal	1000h	Todas as aplicações ficam suspensas até que você responda esta caixa de mensagem. Ë usada para mensagens muito importantes.
Desk Modal	2000h	Semelhante ao Application Modal só que suspende todas as telas da aplicação até que a mensagem seja respondida.

Botões da caixa de mensagem

BOTÕES	HEXA	DESCRIÇÃO
OK	0000h	Aparecerá somente o botão de OK na caixa de mensagem
OK, Cancel	0001h	Aparecerão os botões OK e Cancel.
Abort, Retry, Ignore	0002h	Aparecerão os botões Abort, Retry, e Ignore.
Yes, No, Cancel	0003h	Aparecerão os botões Yes, No, e Cancel.
Yes, No	0004h	Aparecerão os botões Yes e No
Retry, Cancel	0005h	Aparecerão os botões Retry e Cancel.

Default para os botões

BOTÕES	HEXA	DESCRIÇÃO
Botão1 como default	0000h	Dá o foco para o Botão 1.
Botão2 como default	0100h	Dá o foco para o Botão 2.
Botão3 como default	0200h	Dá o foco para o Botão 3.

Ícone

TIPO	HEXA	DESCRIÇÃO
Pare	0010h	O ícone é um sinal de pare
Pergunta	0020h	O ícone é um ponto de interrogação
Exclamação	0030h	O ícone é um ponto de exclamação
Informação	0040h	O ícone é o caractere 'i' dentro de um círculo

Valores de retorno

VALOR	BOTÃO	MENSAGEM
1	OK	OK
2	Cancel	Cancelar
3	Abort	Abortar
4	Retry	Tentar de novo
5	Ignore	Ignorar
6	Yes	Sim
7	No	Não

Configurando uma Caixa de Mensagem

Para configurar o estilo de uma Caixa de Mensagem você deve fazer um OR lógico de cada característica que você deseja (Tipo, botões default, ícone, botões). Você pode escolher apenas uma das características disponíveis (veja tabelas acima). Por exemplo, para especificar uma caixa de Mensagem com o seguinte estilo:

```
Tipo System Modal (1000h) + Botão 2 como Default (0100h) +
Ícone de exclamação (0030h) + Botões Yes e No (0004h) = 1134h
```

Exemplo:

```
strText = "Como está a caixa?"
strTitle = "Teste da Caixa de Mensagem"
MessageBox(strText, strTitle, 1134h)
```

Mid (strText, nFirst, [nCount])

Retorna parte de strText conforme os parâmetros nFirst e nCount. nFirst especifica a posição, no string, do primeiro caractere do substring que será retornado (a contagem inicia em zero) e nCount é o número de caracteres do substring. Exemplo:

```
strSenha = Mid('Password', 0, 3)  // retorna 'Pas'
strSenha = Mid('Password', 3, 4)  // retorna 'swor'
strSenha = Mid('Password', 2)     // retorna 'ssword'
```

Min (x,y)

Compara x e y e retorna o menor valor entre as duas expressões numéricas.

Exemplo:

```
tag001 = Min (2.4,28.3)          // tag001 = 2.4
```

MoveFile (arqFonte, arqDestino)

Move o arquivo especificado para uma nova localização. Retorna TRUE se bem sucedida ou FALSE em caso de erro. Os atributos globais lastError e lastErrorStr são atualizados por essa função.

Exemplo:

```
MoveFile("c:\data\turtle.dat","c:\newdata\rabbit.dat")
```

PadC (string, tamanho [, caractere])

Retorna a string strString centralizada com o caractere, parâmetro caractere, completando o tamanho do string resultante especificado pelo parâmetro tamanho. Se caractere não é especificado, espaços são introduzidos. Somente o primeiro caractere do parâmetro caracter é usado.

Exemplos:

```
PadC("abc",7)          // Resulta "  abc  "
PadC("abcdefghij",5)   // Resulta "abcde"
PadC("abc",8,"-")      // Resulta "--abc---"
```

PadL (string, tamanho [, caractere])

Retorna string posicionado no lado direito com o caractere caractere completando o tamanho do string resultante, indicado por tamanho. Se caractere não é especificado, espaços são introduzidos. Somente o primeiro caractere de caractere é usado.

Exemplos:

```
PadL("abc",5)          // Resulta "  abc"
PadL("abcdefghij",5)   // Resulta "abcde"
PadL("abc",8,"-")      // Resulta "-----abc"
```

PadR (string, tamanho [, caracter])

Retorna string posicionado no lado esquerdo com o caractere especificado no parâmetro caracter, completando o tamanho do string resultante, indicado por tamanho. Se caracter não é especificado, espaços são introduzidos. Somente o primeiro caractere de strPadChar é usado.

Exemplos:

```
PadR("abc",7)          // Resulta "abc    "
PadR("abcdefghij",5)   // Resulta "abcde"
PadR("abc",8,"-")     // Resulta "abc-----"
```

PasswordDlg(text, title, [x], [y])

Abre uma caixa de diálogo para entrada de uma password. O parâmetro title é um string a ser mostrado na barra de título e text é um string com uma mensagem para o usuário. Retorna a password digitada pelo usuário. Os parâmetros opcionais x e y indicam a posição do canto superior esquerdo da caixa de mensagem a ser mostrada. A função retorna a senha digitada pelo usuário.

Exemplo:

```
strPwd = PasswordDlg ("Digite a senha","Segurança")
strPwd = PasswordDlg
("Digite sua senha","Login",100,100)
```

PlaySound (soundIndex)

Toca um som especificado no parâmetro soundIndex (inteiro entre 0 e 5) de acordo com a tabela abaixo:

ÍNDICE	DESCRIÇÃO
0	Bipe usando o alto-falante interno do computador
1	Asterisco (observação)
2	Exclamação
3	Mão (atenção)
4	Questão
5	Alerta padrão

Este som é definido no painel de controle do Windows. A função faz a solicitação do toque do som ao sistema e retorna. O som é tocado assincronamente.

PlayWave (filename, [startTime], [stopTime])

Toca um arquivo formato WAV. Os parâmetros startTime e stopTime especifica o intervalo do arquivo que será tocado (em milisegundos). Se estes parâmetros forem

omitidos, todo o arquivo será tocado. O processo pode ser parado chamando a função StopWave().

Exemplo:

```
PlayWave("tada.wav",2000,5000)
// Toca o arquivo do segundo 2 até o 5
```

Esta função não aceita os parâmetros startTime e stopTime no CE.

PrintString (numeroLPT, string)

Envia a string para a impressora numeroLPT. A string é enviada aguardando um pré processamento (raw bytes). Se a formatação é requerida, é possível acessar as configurações da impressora. Os caracteres CR e LF (Chr(13) + Chr(10)) deve ser fornecido para mudar uma nova linha.

Exemplo:

```
PrintString(1, "Hello World…")
```

Rad (angle)

Retorna o ângulo angle (expresso em graus) convertido para radianos, segundo a fórmula: radianos = (angle / 180) * 3.141592.

ReadFromFile (strArquivo, [posInicio], [bytesALer)

Lê o arquivo strArquivo, iniciando por posInicio (do início do arquivo, se não for especificado). Se bytesALer não for especificado, a função lê todos os bytes do arquivo. Recomenda-se que o arquivo contenha somente dados de texto/caractere. Dados binários podem ser lidos, mas não podem ser manipulados no Elipse SCADA.

Exemplo:

```
tag001 = ReadFromFile("Help.txt")
// Lê todo o arquivo HELP.TXT para tag001
```

ReadIniNumber (strArquivo, strSeção, strEntrada, [nValorDefault])

Retorna o número de uma entrada no arquivo de configuração (.INI) strArquivo. Os parâmetros strSeção e strEntrada especificam a seção e a entrada do arquivo respectivamente. Se a entrada não for encontrada, o valor nValorDefault é retornado. Se nValorDefault não for especificado, a função retorna zero. O arquivo deve ser formatado com a seguinte sintaxe:

```
[secao1]
entrada1 = valor1
entrada2 = valor2
<outras entradas>
entradaN = valorN
[secao2]
<outras entradas>
[secaoN]
<outras entradas>
```

Exemplo:

```
// Arquivo MYAPP.INI com o seguinte conteúdo:
// [General]
// Total = 1000
// UserName = Admin
ReadIniNumber("MYAPP.INI","General","Total",-1)
// retorna 1000
ReadIniString("MYAPP.INI","General","UserName")
// retorna "Admin"
```

NOTA: Se houver um outro programa que modifica o arquivo .INI, as mudanças poderão não ser vistas imediatamente porque Windows mantem uma cópia do arquivo em memória. Você pode forçar a atualização chamando WriteIni (strFile,"","","") antes de ler o arquivo .INI. Seja um arquivo VALUES.INI que está sendo atualizado por um outro programa. Pode-se ler os valores atualizados assim:

```
WriteIni("VALUES.INI","","","")

// Força o Windows a descarregar o cache

ReadIniNumber("VALUES.INI", "Secao", "Entrada")

// Lê o valor de Entrada na seção Secao
```

ReadIniString(strArquivo, strSeção, strEntrada, [strValorPadrão]) 

Retorna o string contendo o valor da entrada no arquivo strArquivo .INI. Os parâmetros strSeção e strEntrada especificam a seção e a entrada do respectivo arquivo. Se a entrada não for encontradaI, o strValorPadrão é retornado, ou um string nulo ("") se strValorPadrão não for especificado.

Real (string)

Converte um string numérico ou um número para um número real.

Exemplo:

```
Numeric = Real('30')  // retorna 30,0
```

Red(cor)

Estas funções permitem obter uma das três cores componentes de um valor 24 bits.de cor: vermelho, verde e azul. O valor retornado estará sempre entre 0 (intensidade mínima) e 255 (intensidade máxima).

Exemplo:

```
color = Display.backgroundColor
Display.backgroundColor = RGB(Red(color)/2,
Green(color)/2, Blue(color)/2)
```

O exemplo acima configura a cor de fundo de um Display para a metade da intensidade da cor original, para cada uma das três componentes de cores.

ReleaseMouseCapture ()

Libera a captura do mouse efetuada pela função SetMouseCapture(). Um objeto que tem uma captura de mouse ativa recebe todos os eventos do mouse, independente do mouse estar ou não sobre o objeto.

Exemplo: o script OnMouseOver a seguir muda a cor de fundo de um objeto de tela quando o mouse está sobre ele e retorna à cor original quando o mouse sai.

```
// verifica se já não há uma captura de mouse
IF NOT Display.HasMouseCapture()
     // o mouse entrou no objeto, capture
     Display.SetMouseCapture()
     // colocar uma cor de destaque
     Display.backgroundColor = RGB(220,220,220)
ELSEIF NOT Display.IsMouseInside()
     // se o mouse saiu do objeto, coloque a cor original
Display.backgroundColor = RGB(192,192,192)
     // e cancele a captura
     ReleaseMouseCapture()
ENDIF
```

RemoveDir (dir)

Remove um diretório (pasta) existente. No Windows, por restrição do sistema, o diretório deve estar vazio e não deve ter sub-diretórios, para ser apagado. Não é possível apagar o diretório corrente, o diretório de trabalho do sistema ou o diretório-raiz. Retorna o valor TRUE se bem sucedida ou FALSE (zero) se não. Os atributos globais lastError e lastErrorStr são atualizados por essa função.

RenameFile (arqFonte, arqDestino)
Muda o nome do arquivo arqFonte para arqDestinoe. Retorna o valor TRUE se bem sucedida ou FALSE (zero) se não. Os atributos globais lastError e lastErrorStr são atualizados por essa função.

RGB (vermelho, verde, azul)
Retorna um valor de cor 24 bits para as três cores componentes especificadas: vermelho, verde e azul. Cada componente é um inteiro entre 0 (intensidade mínima) e 255 (intensidade máxima). O valor retornado pode ser atribuído a qualquer atributo de cor de um objeto. Exemplo:

```
// Vermelho forte
Display1.backgroundColor = RGB(255,0,0)
```

Right (string, nCount)
Retorna os nCount caracteres mais à direita do parâmetro string.

Exemplo:

```
str = Right("Password",4)  // retorna "word"
```

Rnd ()
Retorna um número real aleatório entre 0 e 1 (inclusive).

Round (x)
Retorna o número x arredondado para o inteiro mais próximo.

Exemplo:

```
tag001 = Round(14,1)    // o resultado é 14.0
tag002 = Round(13,9)    // o resultado é 14.0
```

RunMacro (strExpressão)
Compila e executa a expressão contida no string strExpressão passado como parâmetro. Retorna o valor da expressão executada ou zero se nenhum erro de compilação ou execução ocorrer.

Exemplo:

```
// executa a expressão "Screen<activeScreen>.Activate",
// onde activeScreen é o número da tela.
RunMacro("Screen"+Str(activeScreen,1,0)+".Activate()")
```

ScriptWindow () 
Abre uma janela para debug no modo Runtime que permite supervisionar todos os scripts que estão rodando. Também mostra quanto tempo ou quantas vezes um script for executado.

SetFileAttributes (arquivo, attrNovo)
Modifica os atributos do arquivo file para attrNovo. Veja a função GetFileAttributes para a lista de atributos que o parâmetro arquivo pode receber. A função retorna TRUE se bem sucedida ou FALSE (zero) se falhar. Os atributos globais lastError e lastErrorStr são atualizados por esta função.

Exemplo:

```
SetFileAttributes("turtle.dat",3)
// somente leitura (1) + oculto (2) = 3
```

SetSystemTime (novaHora)
Ajusta a hora do sistema para o parâmetro novaHora. O parâmetro novaHora é um valor absoluto de datahora. Use a função MakeTime() para compilar este número. A função retorna 1 (TRUE) se obtiver sucesso, 0 (FALSE) se falhar.

Exemplo:

```
SetSystemTime(MakeTime(10,8,2000,17,25,56))
```

NOTA: A hora interna mantida pelo Elipse SCADA não é ajustada automaticamente. Para que isso ocorra, pode-se utilizar dois métodos: Adicionar a chave `HKEY_CURRENT_USER\Software\Elipse Software\Elipse32\Options\DisableInternalClock` com o valor 1 ao Editor de Registro do Windows, ou então iniciar o Elipse SCADA via linha de comando utilizando a opção `-disableInternalClock`. Caso nenhum dos dois métodos seja utilizado, o sistema faz pequenos incrementos/decrementos em seu tempo interno até que alcance a hora do relógio do sistema em uma taxa máxima de 20% (incrementa/decrementa no máximo 2 segundos a cada 10 segundos).

Sgn (x)
Retorna o sinal de x (-1 se negativo, 0 se zero, 1 se positivo).

Exemplo:

```
tag001 = Sgn(-123.98)   // retorna -1
```

ShellExecute (strOperação, strArquivo, strParametros, strDir, nCmdVisual))
Executa um comando específico sobre um arquivo (programa ou documento) especificado. O parâmetro strOperação indica o comando ser executado, a saber: "edit" (editar); "explore" (explorar); "find" (buscar); "open" (abrir); “print" (imprimir); ou "properties" (propriedades). O parâmetro strArquivo indica o caminho completo do arquivo que irá executar a ação. strParametros é um string que especifica os parâmetros que devem ser passados na chamada do comando ou o arquivo que sofrerá a ação indicada. strDir define o diretório padrão a ser utilizado. nCmdVisual especifica como a aplicação deve ser mostrada quando for aberta, a saber: 0 = esconder a janela; 1 = mostrar a janela; 2 = mostrar a janela minimizada; 3 = mostrar a janela maximizada.

Exemplo:

```
'Executa comandos FTP salvos no arquivo filename.txt
'sem mostrar nenhuma janela
ShellExecute("open", "ftp", "-s:filename.txt", "", 0)
```

Sin (angle)
Retorna o seno do ângulo especificado. O ângulo angle é expresso em radianos.

Sleep (nSegundos)
Faz uma pausa na execução do script durante nSeconds segundos (é possível especificar uma fração, por exemplo: Sleep (0.5)). A interface do usuário não responderá enquanto a pausa estiver sendo executada. Processos rodando em segundo plano que estiverem rodando em thread separados (E/S, drivers de rede, recepção de vídeo) continuarão funcionando, mas toda a lógica da aplicação (scripts) será pausada.

Sqrt (x)
Retorna a raiz quadrada de x.

Exemplo:

```
tag001 = Sqrt(81)      // returns 9
```

StartSound (soundIndex, [frequency (ms)]) 
Toca o som especificado em soundIndex na freqüência especificada em frequency (em milisegundos) até um comando StopSound ser chamado. soundIndex deve ser um inteiro entre 0 e 5 (veja a tabela de sons disponíveis consultando a função PlaySound()). A freqüência mínima é 100ms; se for especificado um número menor, o valor 100 será assumido. Note que o parâmetro frequency designa de quanto em quanto tempo o som será tocado.

Exemplo:

```
StartSound(2,2000)
// toca o som de exclamação a cada 2 segundos
```

StopSound () 
Pára um som que está tocando, iniciado pela função StartSound().

StopWave ()
Pára a reprodução de um arquivo WAV iniciada pela função PlayWave().

Str (value, size, precision)
Converte um número inteiro ou real para um string. O parâmetro value é o número a ser convertido, o parâmetros size e precision determinam o tamanho e precisão do string. Quando o parâmetro value for um string ele primeiro é convertido para um número e depois formatado conforme os parâmetros size e precision.

Exemplo:

```
String = Str(30.95,6,2)  // Retorna " 30.95"
```

StrZero (Valor, tamanho, prec)
Converte um número inteiro ou real para um string preenchendo com zeros a esquerda até o tamanho especificado. O parâmetro valor é o número a ser convertido, o parâmetros tamanho e prec determinam o tamanho e precisão do string. Quando o parâmetro value for um string ele primeiro é convertido para um número e depois formatado conforme os parâmetros tamanho e prec.

Exemplo:

```
String = StrZero(30.95,6,2)  // Retorna "030.95"
```

Tan (angle)
Retorna a tangente do ângulo especificado. O ângulo angle é expresso em radianos.

WaitCursor(bHabilita)
Habilita (bHabilita=1) ou desabilita (bHabilitae=0) o cursor de espera (ampulheta).

Exemplo:

```
WaitCursor (1)
<alguns comandos que demoram tempo>
WaitCursor (0)
```

WatchWindow() 
Abre uma janela de depuração no módulo Runtime que permite visualizar os valores das propriedades de um item selecionado na árvore da aplicação.

WriteIni (strArquivo, strSeção, strEntrada, valor) 
Escreve o valor (numérico ou string) no arquivo .INI strArquivo. O parâmetro strSeção e strEntrada especificam a seção e a entrada do arquivo respectivo.

NOTA: Se o arquivo for modificado INI file, tas transformações não serão imediatamente modificadas, porque o Windows copia um arquivo na memória. Você pode forçar a atualização utilizando o parâmetro WriteIni (strFile, "", "", "") depois de ler o arquivo .INI.

O exemplo abaixo escreve as seguintes linhas em MYAPP.INI:

```
[General]
Total = 1500
UserName = Admin
```

Exemplo:

```
WriteIni("MYAPP.INI","General","Total",1500)
WriteIni("MYAPP.INI","General","UserName","Admin")
```

WriteToFile(arquivo, texto, [operação])
Escreve o texto no arquivo. Retorna 1 se for bem sucedida ou 0 se ocorrer algum erro. O parâmetro operação é opcional e possui as seguintes especificações:

MODO	DESCRIÇÃO
0	(default) Cria o arquivo se não existir, colocando a string no fim do arquivo.
1	Falha se o arquivo não existir, colocando a string no final do arquivo.
2	Cria o arquivo se não existir, substituindo o arquivo por uma nova string.
3	Falha se o arquivo não existir, substituindo o arquivo por uma nova string.

```
Exemplo:
WriteToFile("test.txt", "Essa linha será
    acrescentada para um arquivo existente", 1)
```

19.5.2. Funções da Aplicação

AddUser (nome, descrição, login, senha, nivelAcesso)
Adiciona um novo usuário na lista de usuários da aplicação. Retorna 1 (um) se for bem sucedida ou 0 (zero) caso contrário.

Exemplo:

```
Application.AddUser
    ("João Machado", "Gerente", "JoaoM", "1234", 0 )
```

DeleteUser (login)
Apaga um usuário da lista de usuários. Retorna 1 (um) se for bem sucedida ou 0 (zero) caso contrário.

Exemplos:

```
Application.DeleteUser("JoaoM")
Application.DeleteUser(strUserName)
```

EditProperties() 
Abre uma janela que permite a edição das propriedades da Aplicação.

GetMaxX()
Retorna a lagura da janela da aplicação em pixels.

GetMaxY()
Retorna a altura da janela da aplicação em pixels.

Login ([x][,y][,habilitaTitulo][,permiteFechar][,strTitulo])
Chama uma caixa de diálogo para a identificação (login) de um usuário. Retorna verdadeiro (diferente de zero) se o usuário for logado com sucesso ou falso (zero) se não for. Em caso de erro nenhuma mensagem ou janela será mostrada. Os parâmetros opcionais X e Y indicam as coordenadas da posição do canto superior esquerdo da janela de login. O parâmetro habilitaTitulo determina se a janela de login terá barra de título (*habilitaTitulo* = 1, *default*) ou não (*habilitaTitulo* = 0). O parâmetro permiteFechar é um booleano que indica se o botão de fechar a janela deve aparecer (padrão é 1, visível). O parâmetro strTitulo permite especificar um título para a janela diferente do padrão. Em caso de erro, o atributo global lastError também atualizado: 0 (zero) se o login for cancelado pelo usuário ou 1 (um) se o usuário ou a senha forem inválidos.

Exemplo:

```
IF Application.Login()
    SetPointScreen.Activate()
ENDIF
```

Logout()
Executa o logout (saída) de um usuário da aplicação. Fecha todas as telas que possuem prioridade diferente de zero.

Exemplo:

```
IF MessageBox("Você quer dar logout?","Logout",4) == 6
    Application.Logout( )
ENDIF
```

MaximizeApp() 
Maximiza a janela da aplicação.

MinimizeApp()
Minimiza a janela da aplicação.

ReloadApp([strAppName])
Termina a execução da aplicação corrente, recarrega-a do disco e reinicia-a. O parâmetro opcional strAppName permite passar o nome de outra aplicação a ser reiniciada, ao invés da corrente. Esta função não está disponível no modo Demo (demonstração) do Elipse SCADA.

RestoreApp()
Restaura o tamanho original da janela da aplicação, antes de ela ter sido maximizada ou minimizada.

ShutDownWindows(modoDesligamento)
Permite reiniciar ou desligar o computador via scripts. O parâmetro modoDesligamento pode ser: 1 (desligar, sem forçar o encerramento das aplicações), 2 (desligar, forçando o encerramento das aplicações), 3 (reiniciar, sem forçar o encerramento das aplicações) ou 4 (reiniciar, forçando o encerramento das aplicações). Um desligamento forçado ou um reinício forçado indicam que todos os outros programas em execução (exceto o Elipse SCADA) serão imediatamente terminados e não será possível salvar nenhuma alteração.

Exemplo:

```
// Parada programada normal do Windows
Application.ShutdownWindows(1)
```

Nota: Quando a aplicação é rodada no modo Configurador, ele simplesmente pára a aplicação. Quando a aplicação é rodada no modo Runtime, o aplicativo é parado e o computador reiniciado.

StopRunning()
Finaliza a aplicação corrente.

UserAdministration([x][,y][,enableTitle][,allowClose][,customTitle])
Permite a um usuário modificar sua senha. Se o usuário é administrador (nível de acesso 1) ele poderá criar, modificar e remover os atributos de todos os usuários.

19.5.3. Funções de Tags

Funções Comuns

EditProperties()
Abre uma janela que permite a edição das propriedades do tag.

Funções de Tags Matriz

Avg (r1, c1, r2, c2)
Retorna a média dos valores no retângulo definido por linha1 (r1) coluna1 (c1) e linha2 (r2) coluna2 (c2).

Exemplo:

```
tagSum = tagMatriz.Avg(2,1,3,3)
// retorna a média das células
// (2,1), (2,2), (2,3), (3,1), (3,2), (3,3)
```

ExchangeColumns (coluna1, coluna2)
Troca o conteúdo de duas colunas de uma matriz.

Exemplo:

```
// Troca os conteúdos das colunas 3 e 4
TagMTX1.ExchangeColumns(3,4)
```

ExchangeRows (linha1, linha2)
Troca o conteúdo de duas linhas de uma matriz.

Exemplo:

```
// Troca os conteúdos das linhas 3 e 4
TagMTX1.ExchangeRows(3,4)
```

GetCell (linha, coluna)
Retorna o valor (numérico ou string) de uma determinada célula da matriz especificada pela linha row e columa column.

Exemplo:

```
// Retorna o valor da célula da linha 3, coluna 5
TagMTX1.GetCell(3,5)
```

MapCellsToTags(startColumn, startRow, endColumn, endRow)
Mapeia as informações associada com as propriedades ou tags.

Exemplo:

```
TagMTX1.MapCellsToTags(1,1,5,5)
```

OrderBy(linhaSuperior, linhaInferior, coluna, ascendente)
Ordena as linhas de uma matriz entre a linha inicial linhaSuperior e a final linhaInferior pela coluna coluna, de maneira ascendente (se ascendente = 1) ou descendente (se ascendente = 0).

Exemplo:

```
// Ordena as linhas 1 a 8 da matriz de forma ascendente,
// conforme os valores da coluna 1
TagMTX1.OrderBy(1,8,1,1)
```

SetCell (linha, coluna, valor)
Atribui um valor value a uma determinada célula da matriz especificada pela linha especificada pelo parâmetro linha e coluna especificada pelo parâmetro coluna.

Exemplo:

```
// Atribui o valor 9 à célula da linha 3, coluna 5
TagMTX1.SetCell(3,5,9)
```

Sum (r1, c1, r2, c2)
Retorna o somatório dos valores contidos no retângulo definido por linha1 (r1) coluna1 (c1) e linha2 (r2) coluna2 (c2).

Exemplo:

```
// Soma os conteúdos das células
// (2,1), (2,2), (2,3), (2,4), (3,1), (3,2), (3,3), (3,4)
TagMTX1.Sum(2,1,3,4)
```

Funções de Tags Crono

Reset()
Zera o acumulador.

Funções de Tags Bloco

GetAt(índice)
Pega o valor armazenado em um elemento de bloco. O atributo índice indica o índice do elemento do tag bloco. Atenção: esta função não pode ser usada em versões limitadas (versões Lite) do Elipse SCADA.

Exemplo:

```
// Soma todos os elementos do bloco
DIM index, sum
sum = 0
FOR index = 0 TO Block1.size-1
    sum += Block1.GetAt(index)
NEXT
```

Read()
Lê o bloco do PLC imediatamente e retorna 1 se a leitura foi bem sucedida e 0 se a leitura falhar. Para maiores informações sobre blocos PLC, verifique o item **Tag Bloco** do capítulo sobre Tags neste manual.

ATENÇÃO: Esta é uma função síncrona, ou seja, espera até que a operação de leitura esteja terminada antes de retornar para a próxima instrução, o que pode degradar seriamente o desempenho da aplicação, conseqüentemente você deve usá-la com cuidado e somente quando for absolutamente necessário.

SetAt (índice, valor)
Seta o valor de um elemento de bloco. Retorna verdadeiro se a operação for bem sucedida, caso contrário retorna falso. O atributo índice indica o índice do elemento do tag bloco e o atributo valor é o valor a ser enviado para o elemento.

Atenção: Esta função não pode ser usada em versões limitadas (versões Lite) do Elipse SCADA.

Exemplo:

```
// Este exemplo ajusta todos os elementos do bloco para zero
DIM index
FOR index = 0 TO Block1.size-1
    Block1.SetAt(index, 0)
NEXT
```

SetSize (size)
Ajusta o número dos elementos do bloco, indicado pelo parâmetro size. Os elementos do bloco serão adicionados se o bloco crescer, ou removidos se o bloco diminuir.

Write ()
Escreve os valores atuais do bloco no PLC imediatamente e retorna 1 se a escrita foi bem sucedida e 0, se a escrita falhar. Para maiores informações sobre escrita em serviços de I/O, verifique o item **Tag Bloco** do capítulo sobre Tags neste manual.

Funções de Elementos de Bloco

Read()
Lê o elemento de um tag bloco PLC. Retorna 1 (um) se a leitura foi bem sucedida, e 0 (zero) se a leitura falhar.

Atenção: esta é uma função síncrona, ou seja, espera até que a operação de leitura esteja terminada antes de retornar para a execução da próxima instrução. Ela pode degradar seriamente o desempenho da aplicação, conseqüentemente deve ser usada quando for absolutamente necessário.

Write()
Escreve o valor atual do elemento do bloco no PLC imediatamente, retornando 1 (um) se a escrita foi bem sucedida, ou 0 (zero) se a leitura falhar. Para maiores informações sobre escrita em serviços de I/O, verifique o item **Elemento de Bloco** do capítulo sobre Tags neste manual.

Atenção: Esta é uma função síncrona, ou seja, espera até que a operação de escrita esteja terminada antes de retornar para a execução da próxima instrução. Ela pode degradar seriamente o desempenho da aplicação, conseqüentemente deve ser usada quando for absolutamente necessário.

WriteEx(valor)
Escreve o valor para o driver I/O, especificando as informações para o tag. Esta função é usada para especificar as informações de varredura (scan) do tag.

Funções de Tags OPC

Read()
Lê o valor do servidor OPC.

Write()
Escreve o valor no tag para o servidor OPC.

Funções de Tags PLC

Read()
Lê o Tag PLC imediatamente e retorna 1 se a leitura foi bem sucedido e 0 se a leitura falhar. Para maiores informações sobre tags que lêem dados de servidores de I/O, consulte o item **Tag PLC** do capítulo sobre Tags neste manual.

Atenção: esta é uma função síncrona, ou seja, espera até que a operação de leitura esteja terminada antes de retornar para a execução da próxima instrução. Ela pode degradar seriamente o desempenho da aplicação, conseqüentemente deve ser usada quando for absolutamente necessário.

Write()
Escreve o valor atual do Tag no PLC imediatamente. Retorna 1 se a escrita foi bem sucedida e 0 se a escrita falhar. Para maiores informações sobre tags que escrevem dados em servidores de I/O, consulte o item **Tag PLC** do capítulo sobre Tags neste manual.

Atenção: esta é uma função síncrona, espera até que a operação de escrita esteja terminada antes de retornar para a execução da próxima instrução. Ela pode degradar seriamente o desempenho da aplicação, conseqüentemente deve ser usada quando for absolutamente necessário.

WriteEx(valor[,timeStamp])
Escreve o valor diretamente no I/O, sem atribuí-lo ao Tag PLC. Esta função usa a escala definida no Tag PLC. O parâmetro numérico opcional *timeStamp* permite forçar uma data/hora para a escrita do valor.

Exemplo:

```
Block1.elm000.WriteEx(12)
```

19.5.4. Funções de Tela

Activate()
Semelhante à função *Show()*.

EditProperties() 
Abre uma janela que permite a edição das propriedades do Tela.

GetMouseX()
Retorna a coordenada X atual do mouse.

GetMouseY()
Retorna a coordenada Y atual do mouse.

Hide()
Esconde (fecha) uma tela.

SendTab (nTabs)
Envia caracteres [Tab] para a aplicação de modo a mudar o foco do teclado entre os objetos de tela. Valores positivos de nTabs simulam a tecla [Tab], valores negativos simulam a combinação de teclas [Shift]+[Tab].

Exemplo:

```
Screen1.SendTab(4)
```

Show()
Mostra uma tela. A tela não recebe o foco quando mostrada.

19.5.5. Funções dos Objetos de Tela

Funções Comuns

BringToFront()
Coloca o objeto no primeiro plano da tela (na frente dos demais objetos).

EditProperties()
Abre uma janela que permite a edição das propriedades do objeto.

HasFocus()
Retorna verdadeiro (diferente de zero) se o objeto tem o foco de teclado ou falso (zero) se não tem.

Exemplo:

```
// Muda a cor de Botao se ele tem o foco do teclado
IF Botao.HasFocus()
    Botao.foregroundColor = RGB(255,255,0)
ENDIF
```

HasMouseCapture()
Retorna verdadeiro (diferente de zero) se o objeto está com o mouse capturado ou falso (zero) se não está.

Exemplo:

```
Botao.SetMouseCapture()
tag1 = Botao.HasMouseCapture()      // retorna 1
ReleaseMouseCapture()
tag1 = Botao.HasMouseCapture()      // retorna 0
```

IsMouseInside()
Retorna verdadeiro (diferente de zero) se o mouse está sobre o objeto ou falso (zero) se não está.

Exemplo:

```
// Muda a cor de Botao se ele tem o mouse sobre ele
IF Botao.IsMouseInside( )
    Botao.foregroundColor = RGB(255,255,0)
ENDIF
```

MoveTo(x,y)
Move um objeto para as coordenadas x e y especificadas.

Exemplo:

```
// Nesse exemplo, quando o mouse passar por cima do objeto,
// ele é movido para o canto esquerdo superior da tela.
IF Quadrado.IsMouseInside()
    Quadrado.MoveTo(0,0)
ENDIF
```

SendToBack()
Coloca o objeto no último plano da tela (atrás dos demais objetos).

SetFocus()
Define o foco de teclado para o objeto.

SetMouseCapture()
Captura o mouse para o objeto que chama a função. Todas as mensagens de mouse serão direcionadas a este objeto até que a função global ReleaseMouseCapture( ) seja chamada.

SetTag(nomeTag)
Associa um novo Tag (nomeTag) ao objeto. Retorna verdadeiro (diferente de zero) se o Tag foi adicionado com sucesso ou falso (zero) se não foi. Se nomeTag é um string vazio (" "), então o Tag associado é removido do objeto. Esta função somente pode ser usada para objetos com apenas um único Tag, ou seja, não é válida para os objetos Tendência e Barra.

Funções do Objeto Alarme

AckSelection()
Envia um sinal de reconhecimento (ACK) para todas as mensagens de alarme selecionadas no objeto. Retorna o número de mensagens reconhecidas.

Exemplo:

```
DIM nAcks
nAcks = Alarm1.AckSelection()
IF nAcks > 0
    MessageBox(Str(nAcks) + " message(s) acknowledged!")
ELSE
    MessageBox("No messages acknowledged!")
ENDIF
```

GetColorZoneInfo(campo, zona)
Retorna informações sobre uma zona de alarmes não-reconhecidos. zona é o número da zona (0, 1 ou 2). campo pode ser um dos seguintes campos:

MODO	DESCRIÇÃO
CheckValue	Retorna 0 se a zona está desabilitada, 1 se a zona é permitida
ForeColor	Retorna a cor de primeiro plano (texto) da zona
BackColor	Retorna a cor de fundo da zona
InitialValue	Retorna a menor prioridade para a zona
SecondValue	Retorna a maior prioridade para a zona

Exemplos:

```
// Ajusta a cor da zona 0 para branco se está habilitada
IF Alarm.GetColorZoneInfo("CheckValue", 0)
    Alarm.SetColorZoneInfo("ForeColor", 0, RGB(255,255,255))
ENDIF
```

SetColorZoneInfo(campo, zona, valor)
Modifica o valor de um campo de uma zona de alarmes não-reconhecidos. zona e campo têm o mesmo significado que em GetColorZoneInfo(). valor é o novo valor a ser atribuído ao campo.

Funções de Objetos AVI

End()
Move a posição do filme para o final.

Home()
Move a posição do filme para o início.

OpenAVI()
Abre um arquivo AVI sem iniciar a reprodução.

PauseAVI()
Pausa o filme na posição atual.

PlayAVI([nFrom [, nTo]])
Abre um arquivo AVI e inicia a execução do avi. Se nFrom for especificado, o início será a partir do quadro nFrom. Se nTo for especificado, a reprodução será até o quadro nTo. Se o AVI estiver configurado para tocar ao contrário, estes parâmetros serão ignorados e o filme vai ser tocado por inteiro. Se o atributo timeFormat do AVI for 1, então os parâmetros nFrom e nTo serão interpretados como milisegundos.

Step(nframes)
Move a posição do filme nframes quadros para frente. Se nframes for negativo, a posição será movida para trás. O filme será pausado depois da chamada da função. Se o atributo timeFormat do AVI for 1, então o parâmetro nframes será interpretado como milisegundos.

StopAVI()
Pára a reprodução do filme e fecha o arquivo.

Funções da Barra

O parâmetro barIndex usado nas funções abaixo é o número que identifica a barra no gráfico de barras, sendo a primeira, identificada com o número zero, a segunda, um, a terceira, dois, e assim por diante.

CheckLoLoLimit(barIndex, bEnable)

CheckLowLimit(barIndex, bEnable)

CheckHighLimit(barIndex, bEnable)

CheckHiHiLimit(barIndex, bEnable)
Estas funções habilitam ou desabilitam uma zona em um Bar Gauge.

Exemplo:

```
// Habilita a zona HiHi
// se tag1.HiHi.verify for verdadeiro.
Bar1.CheckHiHiLimit(2, tag1.HiHi.verify)
```

GetCheckLoLoLimit(barIndex)

GetCheckLowLimit(barIndex)

GetCheckHighLimit(barIndex)

GetCheckHiHiLimit(barIndex)
Estas funções retornam 1 se a zona correspondente estiver habilitada em um Bar Gauge.

GetLoLoColor(barIndex)

GetLowColor(barIndex)

GetNormalColor(barIndex)

GetHighColor(barIndex)

GetHiHiColor(barIndex)
Retorna a cor da zona específica em um Bar Gauge.

GetLoLoLimit(index) CE

GetLowLimit(index) CE

GetHighLimit(index) CE

GetHiHiLimit(index) CE
Retorna o limite da zona específica em um Bar Gauge.

SetIndexBarColor(barIndex, color)
Ajusta a cor de um gráfico de barra normal. Para um Bar Gauge, é feita uma configuração diferente.

Exemplo:

```
// Este exemplo muda a cor das barras 1 a 4 para vermelho
// no gráfico de barras Bar1.
DIM index
FOR index = 1 to 4
    Bar1.SetIndexBarColor(index, RGB(255,0,0) )
NEXT
```

SetLoLoColor(barIndex, color)

SetLowColor(barIndex, color)

SetNormalColor(barIndex, color)

SetHighColor(barIndex, color)

SetHiHiColor(barIndex, color)
Ajusta a cor de uma zona específica em um Bar Gauge.

SetLoLoLimit(barIindex, value)

SetLowLimit(barIndex, value)

SetHighLimit(barIndex, value)

SetHiHiLimit(barIndex, value)
Ajusta o limite de uma zona específica em um Bar Gauge.

Funções do Browser

GetField(nomeDoCampo)
Retorna o valor do campo indicado por nomeDoCampo na linha atual (apontada por curSel). Se chamada no script OnDrawRow(), retorna o valor do campo na linha que vai ser desenhada.

SetLabel(indiceColuna, titulo)
Permite mudar o título de uma coluna em runtime. O parâmetro indiceColuna indica o número da coluna, começando por 0. O número da coluna é atribuído obedecendo a ordem que aparece na aba Banco de Dados, mesmo para colunas desabilitadas. O parâmetro titulo indica o novo texto que deve ser configurado para a coluna.

SetRecordRange(primeiroReg, últimoReg)
Define um intervalo de registros a ser carregado no Browser.

SetTempRowColor(cor)
Permite ajustar a cor de fundo da linha que vai ser desenhada.

SetTempRowTextColor(cor)
Permite ajustar a cor do texto da linha que vai ser desenhada.

UpdateQuery()
Atualiza a consulta do Browser, renovando os valores das linhas mostradas.

Funções de Objetos Texto

SetZoneText(iZone, Text)
Muda o texto da zona iZone para Text. Se iZone for 0 o texto de todas as zonas é modificado.

Funções de Objetos Tendência

AddData(time)
Adiciona um novo valor a todas as penas da Tendência.

BreakPen()
Quebra a pena, fazendo com que os novos dados adicionados não estejam conectados com os pontos que já estavam na pena.

ClearData()
Apaga os dados da tendência ou somente de uma pena da tendência. Exemplo:

```
// Apaga todos os dados da tendência Fornos
Fornos.ClearData()
// Apaga só os dados da pena Temp1 da tendência Fornos
Fornos.Temp1.ClearData()
```

LoadHistoricData()
Carrega dados do histórico relacionado à tendência, segundo a consulta especificada.

Funções do VideoIn

Pause ()
Pausa a reprodução do video.

Play ()
Liga a reprodução do video.

Snapshot (nomedoarquivo, tipo)
Grava o quadro (frame) atual de um vídeo em um arquivo com o nome e caminho nomedoarquivo. O nome do arquivo deverá ter a extensão incluída, a função não adiciona-a. tipo designa o tipo de arquivo e pode ser:

TIPO	DESCRIÇÃO
0	Arquivo "RAW" (salva o quadro atual no disco, sem nenhuma conversão.
1	Formato BMP.
2	Formato GIF (limitado a uma paleta de 256 cores).
3	Formato JPEG.

Stop ()
Pára a reprodução do vídeo.

19.5.6. Funções de Alarmes

AckAllAlarms([tagName])
Reconhece todos os alarmes pertencentes ao tag tagName. Se tagName não for especificado, todos os alarmes de todos os tag do alarme serão reconhecidos.

Exemplo:

// Reconhece todas as mensagens para o tag Heat1

Alarms.AckAllAlarms(Heat1)

// Reconhece tudo

Alarms.AckAllAlarms()

ClearHistoricalData()

Apaga todas as mensagens de alarmes e também limpa o arquivo de log de alarmes.

GetInfo(campo, zona)

Retorna informações a respeito de uma mensagem de alarmes. Se zona não for especificado, a função retorna informações sobre a última mensagem criada no grupo de alarmes. O parâmetro campo é o nome do campo que se deseja saber o conteúdo. Os valores possíveis são:

Valores possíveis

CAMPO	DESCRIÇÃO
Comment	Retorna um string com o "Comentário" da mensagem.
DateTime	Retorna a marcação de tempo (timestamp) do alarme como um valor "datetime" (número de segundos desde 01.01.1970).
Event	Retorna o tipo de evento, a saber: "ALM" para mensagem de alarme e "RTN" para mensagens de retorno.
Limit	Retorna um inteiro com o limite do alarme.
Priority	Retorna um inteiro com a prioridade da mensagem de alarme.
State	Retorna um string com o estado do alarme, a saber: "ACK" para reconhecido e "UNACK" para não-reconhecido.
Tagname	Retorna o nome do tag associado à mensagem de alarme.
Type	Retorna um string com o tipo de alarme ("HIHI", "HIGH", "LOW", "LOLO", "RET")
Value	Retorna o valor do alarme.
Handle	Retorna um "handle" da mensagem de alarme. Este "handle" pode ser armazenado e usado para recuperar as informações sobre o alarme posteriormente.
User	Retorna o usuário responsável pela mensagem de alarme.

É possível usar abreviações para os nomes dos campos, com pelo menos, dois caracteres. Isto é, o campo "Value" pode ser abreviado por "VA", o campo "Priority" pode ser abreviado por "PR" e assim por diante.

Exemplo:

```
// Alarms.OnAlarm() script
// guarda o handle de alarme das mensagens
// de tags que começam "TEMP"
IF Left(Alarms.GetInfo("Tagname"), 4) == "TEMP"
     lastHandle = Alarms.GetInfo("Handle")
ENDIF
```

GetNextActiveAlarmHandle(lastHandle)
Retorna o handle da próxima mensagem de alarmes ativos. Mensagens de alarmes ativos são mostradas no Alarme Resumido. lastHandle pode ser: -1 para buscar o primeiro handle de alarme ativo ou o número do handle anterior, retornado pela função GetNextActiveAlarmHandle. Se não há mais mensagens de alarmes ativos, a função retorna -1. O handle retornado pode ser usado na função Alarms.GetInfo().

Exemplo:

```
// conta o número de mensagens de alarmes
// de tags que começam "TEMP"
// e mostra em uma Caixa de Mensagem (MessageBox)
DIM hAlarm, strTagName, nTotal
nTotal = 0
hAlarm = Alarms.GetNextActiveAlarmHandle(-1)
While hAlarm <> -1
     strTagName = Alarms.GetInfo("TA", hAlarm)
     IF Left(strTagName, 4) == "TEMP"
        nTotal = nTotal + 1
     EndIf
     hAlarm = Alarms.GetNextActiveAlarmHandle(hAlarm)
Wend
MessageBox("Existem " + Str(nTotal) + " alarmes ativos!")
```

RemoveFromSummary(messageID)
Remove do Alarme Resumido uma mensagem de alarme criada pela função SimulateAlarm com a ID messageID.

SimulateAlarm(id, timeStamp, strNomeTag, strComentário, valor, limite, pri, tipo, evt, estado)
Cria uma nova mensagem de alarme. Retorna 1 se bem-sucedida, ou 0 caso contrário. Os parâmetros disponíveis nesta função são os seguintes:

Parâmetros

CAMPO	DESCRIÇÃO
id	Identificador da mensagem do alarme. Se o valor for 0, a mensagem de alarme será escrita somente no registro de alarmes. Se o valor for maior que 0, a mensagem será escrita no registro de alarmes e poderá ser mostrada no objeto Alarme da Tela. Você deve passar o mesmo valor deste parâmetro a função RemoveFromSummary para remover a mensagem do sumário.
timeStamp	TimeStamp do alarme.
strNomeTag	String que contém o nome do tag. Caso não exista tag, este campo é puramente informativo.
strComentário	String que contém a mensagem de comentário do alarme.
valor	Valor do alarme.
limite	Determina o limite máximo excedido pelo alarme.
pri	Nível de prioridade do alarme, que pode ser de 1 até 999
tipo	Tipo de mensagem do alarme, que pode ser: "RET", "HIHI", "HIGH", "LOLO", "LOW" or ""
evt	Tipo de evento: "EVT", "ACK", "ALM", "RTN" or ""
estado	Determina o estado de reconhecimento do alarme: "ACK", "UNACK" ou ""

Exemplo:

```
// Este exemplo grava um evento do sistema no registro
// de alarmes. A mensagem não vai para o Alarme Resumido
// porque a ID é 0.
Alarms.SimulateAlarm(0, GetTime(),_
     "System", "Comm Error",0, 0, 1, "", "EVT", "UNACK")
// Este exemplo simula um alarme High
Alarms.SimulateAlarm(230, GetTime(), "pressure1",_
     "Pressao 1 ALTA", 140, 120, 10, "HIGH", "ALM", "UNACK")
// A linha seguinte remove a mensagem simulada do Resumido
// e grava uma mensagem de retorno de alarme falsa no
// registro de alarmes
Alarms.RemoveFromSummary(230)
Alarms.SimulateAlarm(0, GetTime(), "pressure1",_
     "Pressure normal",110, 120, 10, "RTN", "ALM", "ACK")
```

19.5.7. Funções das Receitas

ChooseRecipe(título,registroSel[,x][,y][,largura][,altura])
Abre um diálogo que permite selecionar uma receita a partir de uma lista, retornando seu índice no arquivo. O parâmetro título é um string a ser mostrado na barra de título do diálogo (coloque " " para uma janela sem título), registroSel é o índice da receita a ser selecionada quando o diálogo for aberto (coloque -1 para não selecionar nenhuma), x, y, largura and altura são opcionais e indicam a posição e o tamanho da janela; se não forem informados, o diálogo é aberto no centro da tela ou na última posição em que foi aberto, e uma borda permite que seu tamanho seja modificado.

Exemplo:

```
// tagReceita irá receber o índice da receita escolhida.
// O Diálogo será aberto no centro da tela, em uma janela
// redimensionável, com o título "Lista de Receitas".
tagReceita = Receitas.ChooseRecipe("Lista de receitas",1)
```

No CE, a posição e a dimensão da janela são fixas.

CopyRecord(indice)
Copia um registro selecionado com os mesmos valores do registro original, especificados em indice. Esta função retorna o número de novos registros ou 0 se falhar.

```
Exemplo:
// Copiando uma receita
DIM nRecord
nRecord = Recipe1.ChooseRecipe("Recipes", -1)
IF nRecord > 0
    Recipe1.CopyRecord(nRecord)
ENDIF
```

CreateNewRecord(description)
Cria um novo registro no arquivo de Receitas conforme o parâmetro description, retornando o índice do registro no arquivo.

DeleteRecipe(registro)
Apaga o registro de número registro em uma receita. Retorna verdadeiro (diferente de zero) se a operação for efetuada com sucesso ou falso (zero) se não for.

EditProperties() 
Abre uma janela que permite a edição das propriedades da Receita;

EditRecipe()
Abre uma caixa de diálogo para editar um arquivo de Receita.

FindRecipe(descrição)
Procura por um registro de uma receita que possui o string description na sua descrição. Retorna o número do registro ou zero caso não encontre nenhum.

GetRecCount()
Retorna o número de registros do arquivo de receitas.

GetRecDescription(record)
Retorna a descrição de um registro record no arquivo de Receitas. O parâmetro record deve estar entre 1 <= record <= GetRecCount().

LoadRecipe(record)
Carrega uma receita record do arquivo de Receitas para os respectivos tags relacionados. O parâmetro record deve estar entre 1 <= record <= *GetRecCount()*. A função retorna True se a receita foi carregada corretamente; False, se algum erro ocorrer.

Exemplo:

```
// Carrega o quarto registro de Receitas
Receitas.LoadRecipe(4)
```

SaveRecipe(record)
Salva o registro record. A função retorna True se a receita foi gravada corretamente; False, se algum erro ocorrer.

SetRecDescription(record, description)
Muda para description a descrição do registro record no arquivo de Receitas.

19.5.8. Funções de Históricos

Analysis (x, y [, width [, height]])
Abre a janela da análise histórica gerando o seu respectivo gráfico. Os parâmetros x e y determinam as coordenadas horizontal e vertical da janela da análise, respectivamente. O parâmetro width determina a largura da janela da análise histórica. O parâmetro height determina a altura da janela da análise histórica.

Average (src, type, startTime [, endTime])
Retorna a média de um conjunto de dados do histórico. O parâmetro src determina o nome do tag que é gravado no histórico. O parâmetro type determina o intervalo de tempo a ser considerado para calcular a média, que pode ser o seguinte:

Intervalos de tempo

VALOR	ATRIBUTO
0	Ano
1	Mês
2	Semana
3	Dia
4	Hora
5	Minuto

O parâmetro startTime determina o tempo inicial da média. O parâmetro endTime (opcional) determina o tempo final da média. Se ele for utilizado, o parâmetro Type é ignorado.

Close()
Fecha um arquivo de histórico.

Edit()
Edita o registro corrente sem escrevê-lo em disco. Para escrevê-lo use a função Update().

EditProperties() 
Abre uma janela que permite a edição das propriedades do histórico.

FindTime(time)
Retorna o índice do primeiro registro que possui data e hora maior ou igual à especificada em time.

FinishBatchProcess()
Termina um histórico tipo batelada.

GetFirstRec()
Retorna o número do primeiro registro do arquivo de histórico.

GetLastRec()
Retorna o número do último registro do arquivo de histórico.

GetRecCount()
Retorna o número total de registros do arquivo de histórico.

GetRecno()
Retorna a posição atual do ponteiro para o arquivo de histórico.

GoTo(recno)
Move o ponteiro de registros do arquivo de histórico para o registro de índice recno sem ler o registro.

IsBOF()
Verifica se o ponteiro de registros aponta para o início do arquivo de histórico. Retorna verdadeiro (diferente de zero) se o ponteiro estiver no início ou falso (zero) se não.

IsEOF()
Verifica se o ponteiro de registros aponta para o fim do arquivo de histórico. Retorna verdadeiro (diferente de zero) se o ponteiro estiver no fim ou falso (zero) se não.

Move(n)
Move o ponteiro de registros do arquivo de histórico n registros para frente (números positivos) ou para trás (números negativos). O registro é lido e seus valores carregados nos respectivos campos do histórico.

MoveFirst()
Move o ponteiro de registros do arquivo de histórico para o primeiro registro. O registro é lido e seus valores carregados nos respectivos campos do histórico.

MoveLast()
Move o ponteiro de registros do arquivo de histórico para o último registro. O registro é lido e seus valores carregados nos respectivos campos do histórico.

MoveNext()
Move o ponteiro de registros do arquivo de histórico para o próximo registro. O registro é lido e seus valores carregados nos respectivos campos do histórico.

MovePrev()
Move o ponteiro de registros do arquivo de histórico para o registro anterior. O registro é lido e seus valores carregados nos respectivos campos do histórico.

Open()
Abre um arquivo de histórico.

RestartLastBatch()
Reinicia a última batelada.

SPC() 
Executa a análise do SPC.

StartBatchProcess()
Inicia um histórico por batelada.

Update()
Escreve o registro corrente.

WriteRecord()
Escreve um registro no arquivo de histórico.

19.5.9. Funções da Análise Histórica

Analysis(x, y[,width[,height]])
Abre a janela da análise histórica gerando o seu respectivo gráfico. Os parâmetros x e y determinam as coordenadas horizontal e vertical da janela da análise, respectivamente. O parâmetro width determina a largura da janela da análise histórica. O parâmetro height determina a altura da janela da análise histórica.

CloseAnalysis()
Fecha a janela da análise histórica.

RequeryAnalysis()
Reaplica a consulta sobre o arquivo de dados da análise histórica, carregando novamente os dados na janela da análise.

SPC()
Executa o CEP (Controle Estatístico de Processos) sobre o arquivo de dados da análise histórica.

19.5.10. Funções do CEP

Recalc([bMostraJanelaProgresso=1])
Recalcula todos os dados do CEP. Esta função juntamente com SetDatField() permite calcular o CEP automaticamente sem ter que abrir as janelas de configuração e gráfico, especificada através do parâmetro bMostraJanelaProgresso=1.

SetDatField(nomeCampo)
Ajusta o nome do campo do arquivo histórico que será utlizado nos cálculos do CEP.

19.5.11. Funções de Relatórios

Funções Comuns

EditProperties()
Abre uma janela que permite a edição das propriedades do relatório.

LoadCfg(nomeArquivo)
Carrega a configuração da impressora, margens e fontes do relatório de um arquivo em disco salvo pela função SaveCfg().

Print(bShowErrorMessages,[strHeaderBmp],[strFooterBmp], [bStretchHeader], [bStretchFooter])
Imprime o relatório conforme a configuração especificada. Retorna verdadeiro (diferente de zero) se o Relatório for impresso com sucesso, ou falso (zero) se não for. O parâmetro bShowErrorMessages indica se o comando deverá mostrar mensagens de erro (1) ou deverá executar em modo "silencioso", ou *silent mode* (0).

Os parâmetros opcionais strHeaderBmp e strFooterBmp indicam os nomes do arquivos com as imagems a serem incluídas no cabeçalho e no rodapé, respectivamente. Já os parâmetros opcionais bStretchHeader e bStretchFooter habilitam , cada um por vez, o stretch no cabeçalho e no rodapé.

SaveCfg(nomeArquivo)
Salva a configuração da impressora, margens e fontes do relatório em um arquivo;

SetupPrinter()
Abre a janela de configuração (setup) da impressora.

Relatório Texto

AddFilter (campo, valorMin, valorMax)
Filtra o valor de um campo. O parâmetro campo é um string especificando o nome do campo numérico a ser filtrado. Os parâmetros valorMin e valorMax especificam um intervalo de valores para a busca. Registros que possuem valores fora deste intervalo serão excluídos do relatório. É recomendável configurar o filtro antes da impressão e removê-lo depois. Mais de um campo podem ser filtrados e somente os registros que satisfaçam todos os filtros serão incluídos no relatório.

Exemplo:

```
TextRep1.AddFilter("code", 0, 23)
```

PrintToFile (arquivo, [bImprimeCabeçalho], [separador], [bIndicarProgress], [bInserirAspas])
Imprime os dados de um arquivo Histórico ou de Alarmes para um arquivo texto especificado pelo nome no parâmetro fileName.

O parâmetro bImprimeCabeçalho habilita (1) ou desabilita (0) a impressão do cabeçalho (este parâmetro é opcional e o valor default é 0). Quando você seleciona uma Batelada específica para ser impressa com o parâmetro bImprimeCabeçalho

como 1 e o atributo BImprimeCabeçalho do Relatório como 1, a função também irá imprimir os dados do cabeçalho da batelada.

O parâmetro separador é um string contendo o caractere que será utilizado para separar os campos (é opcional e o valor default é " ", ou seja, um espaço em branco). Retorna verdadeiro (diferente de zero) se o relatório foi impresso com sucesso ou falso (zero) se não foi. Nenhum cabeçalho é impresso.

O parâmetro bIndicarProgress habilita (1) ou desabilita (0) mostrar uma barra de progresso enquanto o relatório é impresso (este parâmetro é opcional e o valor default é 0).

Se for utilizado um caractere separador, o parâmetro bInserirAspas define se colunas do tipo string e data/hora terão os valores envolvidos por aspas (bInserirAspas = 1). Se o parâmetro receber 0, nenhum tipo de coluna será envolvido por aspas, mesmo quando um caractere limitador é usado. Este parâmetro é opcional e o valor default é 1.

Exemplo:

```
Relatorio.PrintToFile ("turtle.dat", 1, "-", 1)
```

RemoveAllFilters()
Remove os filtros de todos os campos do relatório.

RemoveFilter(campo)
Remove um filtro de um campo field.

19.5.12. Funções de Consultas

FindTime(time, firstReg, lastReg)
Retorna o número do primeiro registro entre firstReg e lastReg que possue a data e hora maior ou igual a time. Se não existirem registros no intervalo especificado com a data maior ou igual a time a função retorna -1.

Exemplo:

```
Query1.FindTime(MakeTime(2,3,1995,13,45,30), 20, 50)
```

ReloadStructure()
Força o objeto consulta a recarregar a lista de campos. Isto está requerido se você mudar a especificação de fonte para a consulta e a estrutura da especificação for diferente do campo precedente.

Exemplo:

```
ReportTxt.Query.filename = "C:\Data\MON12.DAT"
ReportTxt.Query.ReloadStructure()
```

19.5.13. Funções da Plotagem

SaveBitmap (nomeArquivo, largura, altura, escalaFonte) CE
Grava o gráfico em um arquivo .BMP. O parâmetro nomeArquivo determina o nome (e o caminho) do arquivo .BMP. largura e altura especificam a largura e altura do bitmap, respectivamente. escalaFonte determina o tamanho da fonte do texto do bitmap, de acordo com a fórmula `tamanhoDaFonte * (escalaFonte/1000)`. Se zero, o tamanho da fonte não é alterado.

Exemplo:

```
Trend1.Plotter.SaveBitmap("C:\SNAPSHOT.BMP", 320, 200, 0)
```

TimeInHAxis()
Determina que o eixo X em um gráfico XY expresse tempo.

TimeInVAxis()
Determina que o eixo Y em um gráfico XY expresse tempo.

Funções do Cursor/Marca/Pena

CE O cursor não está habilitado no CE.

GetXAxis()
Retorna o nome do tag/campo do eixo x.

GetYAxis()
Retorna o nome do tag/campo do eixo y .

SetXAxis(strNome)
Modifica o tag/campo do eixo x através do parâmetro strNome.

SetYAxis(strNome)
Modifica o tag/campo do eixo y através do parâmetro strNome.

19.5.14. Funções de Drivers

Funções Comuns

EditProperties()
Abre uma janela que permite a edição das propriedades do driver.

LoadCfg(nomeArquivo)
Carrega um arquivo, especificado por nomeArquivo, com a configuração do driver.

SaveCfg(nomeArquivo)
Salva a configuração do driver em um arquivo especificado por nomeArquivo.

StartComm()
Inicia a comunicação com o driver. Retorna verdadeiro (diferente de zero) se a comunicação foi iniciada com sucesso, ou falso (zero) se não foi.

StopComm()
Encerra a comunicação com o driver. Retorna verdadeiro (diferente de zero) se a comunicação foi encerrada com sucesso, ou falso (zero) se não foi.

Drivers PLC

AddFilter(filter)
Adiciona um filtro de comunicação filter ao Driver. filter permite desabilitar leitura ou escrita nos Tags PLC ou Bloco que possuam os mesmos parâmetros especificados no filtro.

Exemplo:

```
// desabilita todas comunicações
// com N1 ou B1=1 e N2 ou B2 = 30
Driver1.AddFilter("1,30")
// desabilita todas as leituras com N4 ou B4 = 40
Driver1.AddFilter("R,,,40")
```

filter é um string composto por 4 números separados por vírgulas, correspondendo aos parâmetros N1/B1, N2/B2, N3/B3, N4/B4. O primeiro parâmetro pode ser precedido por uma letra como segue:

"R" desabilita leituras

"W" desabilita escritas

“B" desabilita Tags Bloco

"T" desabilita Tags PLC

Se nenhuma letra for especificada todas as leituras e escritas nos tags PLC e Bloco serão afetadas pelo filtro. A função retorna zero (0) caso ocorra algum erro ou 1 se o filtro foi adicionado com sucesso.

GetErrorInfo(param)

Retorna informação sobre o último erro; param define qual informação será retornada.

Se param for 0, a função vai retornar o tipo de comunicação que causou o erro, como segue:

1 = leitura de um tag PLC;

2 = leitura de um tag Bloco;

3 = escrita em um tag PLC;

4 = escrita em um tag Bloco;

Se param for 1, 2, 3 ou 4, retorna os parâmetros N1 a N4 do tag PLC ou B1 a B4 do tag bloco que causou o erro.

Exemplo:

```
// Script Driver1.OnCommError()
// Filtra todos tags N1/B1 do último erro
DIM strFilter
strFilter = Str(Driver1.GetErrorInfo(1))
// Evita adicionar o mesmo filtro mais de uma vez
// removendo o filtro anterior para o mesmo PLC
Driver1.RemoveFilter(strFilter)
Driver1.AddFilter(strFilter)
```

Reload()

Recarrega o driver do disco. Não retorna nenhum valor caso ocorra um erro.

RemoveAllFilters()

Remove todos os filtros de um driver. Não retorna nenhum valor caso ocorra um erro.

RemoveFilter(filter)

Remove um filtro de um driver, adicionado com AddFilter(). Retorna verdadeiro (diferente de zero) se o filtro foi removido com sucesso ou falso (zero) se ele não existir ou for inválido. O parâmetro filter deve ser especificado da mesma forma que em AddFilter().

StartComm()
Inicia a comunicação com o driver. Retorna verdadeiro (diferente de zero) se a comunicação foi iniciada com sucesso ou falso (zero) se não foi.

StopComm()
Encerra a comunicação com o driver. Retorna verdadeiro (diferente de zero) se a comunicação foi encerrada com sucesso ou falso (zero) se não foi.

Drivers de Rede

Configure()
Abre uma caixa de diálogo para configuração dos parâmetros do driver de rede.

StartDriver()
Carrega o driver de rede habilitando a comunicação remota. Retorna verdadeiro (diferente de zero) se o driver foi carregado com sucesso ou falso (zero) se não foi.

StopDriver()
Pára o driver de rede desabilitando qualquer comunicação remota. Retorna sempre verdadeiro (diferente de zero) indicando que o driver foi parado com sucesso.

19.5.15. Funções de Database

AddRecord (bUpdateQuery)
Adiciona um novo registro na database. Retorna TRUE (diferente de zero) se foi adicionado com sucesso ou FALSE (zero) se algum dos seguintes erros ocorrer: disco cheio, banco de dados é somente para leitura, uma chave primária com um valor já existente ou não foram feitas modificações na database. É necessário preencher todos os campos do registro antes de chamar a função. O parâmetro bUpdateQuery indica se a consulta (query) deve ser atualizada depois de adicionar-se o registro. Se o parâmetro estiver em 0 (zero) então o registro adicionado só estará visível após a atualização da consulta. O valor padrão é 1 (um).

Exemplos:

```
Table1.ID = tag002
Table1.Valor = tag003
Table1.AddRecord(1)
```

BeginTrans()
Inicia uma transação com a database (base de dados). Retorna verdadeiro se a transação for começada, falso se falhar ou se as transações da sustentação da base de dados não forem localizadas.

Exemplo:

```
if (BD1.CanTransact())
     BD1.BeginTrans()// inicia uma transação
     BD1.GotoRecord(0)// vai para o início
     while (not BD1.IsEOF())// se não é o fim do arquivo
        BD1.DeleteRecord()// apaga um registro
        BD1.MoveNext()// vai para o próximo
     wend
     BD1.CommitTrans()// mesmo que BD1.EndTrans(1)
endif
```

CanTransact()
Retorna verdadeiro (diferente de zero) se a database está disponível para transações ou falso (zero) se não está. Exemplo: veja em BeginTrans().

CloseConnection()
Fecha uma conexão ODBC. Esta operação libera o banco de dados para operações externas, como por exemplo, fazer um backup da database que está sendo usada pelo Elipse ou mudar algum parâmetro de conexão.

CommitTrans()
Termina uma transação salvando todos os novos dados.

Exemplo: veja em BeginTrans().

DeleteRecord()
Apaga o registro corrente do Banco de Dados. Retorna verdadeiro (diferente de zero) se o registro foi removido ou falso (zero) se algum dos seguintes erros ocorrer: o banco de dados é somente para leitura, o ponteiro de registros está apontando para a marca de início de arquivo (BOF), o ponteiro de registros está apontando para a marca de fim de arquivo (EOF). Esta função apaga todos os registros duplicados, ou seja, que possuam os mesmos dados, caso um único registro duplicado seja apagado.

Exemplo: veja o exemplo em BeginTrans().

EditProperties()
Abre uma janela que permite a edição das propriedades do Banco de Dados.

EditRecord()
Edita o registro corrente sem escrevê-lo em disco. Para escrevê-lo use a função Update(). Estas duas funções estão obsoletas e são mantidas para permitir compatibilidade com versões anteriores do software. Para editar um registro, atualmente, deve-se apenas modificar os valores dos campos, uma vez estando sobre os mesmos.

EndTrans(n)
Encerra uma transação conforme o parâmetro n. Se n é um (1) é chamada a função CommitTrans() se é zero (0) é chamada a função RollBack(). Exemplo: veja em BeginTrans().

Find(strCondition [, bMoveFirst=0])
Procura o primeiro registro da condição strCondition. Se bMoveFirst for 1 a busca partirá do primeiro registro, se bMoveFirst for 0 (opção) a busca partirá do registro atual. strCondition é uma expressão de texto usada encontrar o registro (como cláusula Where em uma indicação do SQL sem a palavra Where). (Apenas em DAO.)

GetConnectionString()
Obtém o string de conexão usado pelo driver ODBC.

Exemplo:

```
// Obtém o string ODBC da conexão de BD1
strConexao = BD1.GetConnectionString()
```

GetEditMode()
Retornam o estado da edição para o registro atual. (Apenas em DAO.) Pode ser um dos seguintes valores:

VALOR	ATRIBUTO
-1	A tabela não está conectada (fechada).
0	Nenhuma edição em progresso.
1	O registro está sendo modificado.
2	O registro está sendo adicionado.

GetLastError()
Retorna a última mensagem de erro ocorrida durante uma transação com o Banco de Dados.

GetODBCParameter(strParam)
Retornam o valor do parâmetro strParam da string da conexão ODBC. A string da conexão é uma seqüência dos parâmetros e dos valores de parâmetro como no ODBC;DSN=C:\Data.DB;Timeout=500.

Exemplo:

```
strDSN = Table1.GetODBCParameter(.DSN.)
```

GetRecordNumber()
Retorna o número do registro no Banco de Dados. O primeiro registro é o número zero.

GetTableName()
Obtém o nome da tabela do Banco de Dados. (Apenas em ODBC.)

GetTotalNumberOfRecords()
Retorna o número total de registros no Banco de Dados conforme a Consulta especificada na função SQLQuery().

Exemplo:

```
// Retorna o número total de registros,
// pois não tem um consulta definida.
BD1.SQLQuery(" ", " ")
TagRam = BD1.GetTotalNumberOfRecords()
```

GotoRecord(n)
Move o ponteiro de registros para o registro n, conforme a Consulta especificada. Retorna verdadeiro (diferente de zero) se a movimentação foi feita com sucesso ou falso (zero) se for achado uma marca de fim de arquivo EOF.

Exemplo:

```
// vai para o 13o. registro, uma vez que o primeiro
// registro tem sempre índice 0 (zero)
Database1.GotoRecord(12)
```

IsBOF()
Verifica se o ponteiro de registros está no início do arquivo. Retorna verdadeiro (diferente de zero) se está ou falso (zero) se não está.

IsEOF()
Verifica se o ponteiro de registros está no final do arquivo. Retorna verdadeiro (diferente de zero) se está ou falso (zero) se não está.

Exemplo: veja em BeginTrans().

Locate(strCondição [, bMoveParaInício])
Procura pelo próximo registro que atenda à expressão informada no parâmetro string strCondição. O parâmetro bMoveParaInício é um valor booleano que define se a procura deve começar a partir do primeiro registro do banco de dados (bMoveParaInicio = 1) ou do registro atual (bMoveParaInicio = 0).

Exemplo:

```
// Localiza o primeiro registro que atenda a expressão
// a partir do registro atual
Database1.Locate("temp == tag001 * tag002", 0)
```

Move(n)
Move o ponteiro do banco de dados n registros para frente relativos a posição atual do ponteiro. Retorna verdadeiro (diferente de zero) se a movimentação foi feita com sucesso ou falso (zero) se for achado uma marca de fim de arquivo EOF.

Exemplo:

```
// Posiciona o ponteiro no 11o. registro, uma vez que
// o primeiro registro tem sempre índice 0 (zero)
Database1.GotoRecord(10)
// Posiciona o ponteiro no próximo registro (12o. registro)
Database1.Move(1)
// Volta dois registros (10o. registro)
Database1.Move(-2)
```

MoveFirst()
Move o ponteiro de registros para o primeiro registro do banco de dados. Retorna verdadeiro (diferente de zero) se a movimentação foi feita com sucesso ou falso (zero) se não foi.

MoveLast()
Move o ponteiro de registros para o último registro do banco de dados. Retorna verdadeiro (diferente de zero) se a movimentação foi feita com sucesso ou falso (zero) se não foi.

MoveNext()
Move o ponteiro de registros para o próximo registro. Retorna verdadeiro (diferente de zero) se a movimentação foi feita com sucesso ou falso (zero) se for achado uma marca de fim de arquivo EOF.

MovePrev()
Move o ponteiro de registros para o registro anterior. Retorna verdadeiro (diferente de zero) se a movimentação foi feita com sucesso ou falso (zero) se for achado uma marca de início de arquivo BOF.

OpenConnection()
Reabre uma conexão ODBC. Deve ser usada somente após uma função CloseConnection() já ter sido usada, uma vez que os bancos de dados usados no Elipse SCADA são abertos juntamente com a aplicação. Retorna verdadeiro (diferente de zero) se a operação obteve sucesso ou falso (zero) se não.

Requery()
Atualiza o Banco de Dados. É útil quando mais de um usuário está usando o BD. Esta função não é necessária quando você está usando a função Update(1), já que esta própria função atualiza o BD.

RollBack()
Desfaz todas as ações feitas durante uma transação. Retorna verdadeiro (diferente de zero) se as ações foram desfeitas com sucesso ou falso (zero) se não.

SetODBCParameter(param, valor)
Muda um parâmetro dentro do string de conexão ODBC.

Exemplo:

```
// muda a string de conexão ODBC para usar o banco
// de dados Access de nome Suporte que se encontra
// no raiz do disco C:
Database1.CloseConnection()
Database1.SetODBCParameter("DBQ", "C:\suporte.mdb")
Database1.OpenConnection()
```

SetTableName(nomeTabela)
Define um novo nome para a tabela corrente.

Exemplo:

```
Table1.CloseConnection()
Table1.SetTableName("Producao")
Table1.OpenConnection()
```

SQLQuery(Filtro, OrdenarPor)
Define um filtro a ser usado na consulta ao Banco de Dados, conforme os parâmetros Filtro e OrdenarPor.

Filtro: Define um filtro a ser usado na consulta ao Banco de Dados conforme os campos e as opções do usuário. Para usar este parâmetro você precisa especificar um valor ou expressão a ser usado para pesquisa de um campo específico. Quando não existe um filtro específico para o parâmetro Filtro, você deve entrar com um string vazio.

Os operadores aceitos em expressões neste parâmetro são: =, <> (diferente), <, >, <=, >=, !< (não menor do que), !> (não maior do que), AND, OR, NOT, LIKE (semelhante ou igual), IN, BETWEEN, IS NULL, IS NOT NULL.

OrdenarPor: Define uma ordem para a pesquisa no Banco de Dados conforme o campo especificado, que pode ser ascendente (default) ou descendente. Este campo é obrigatório, mas pode ser especificada um string vazio.

A função retorna o número de registros encontrados na consulta.

Exemplos:

```
// Procura todos os registros que possuem "Maria" no campo
// nome, em ordem ascendente e retorna a quantidade de
// registros resultantes em Quantos
quantos = Database1.SQLQuery("nome = 'Maria'","nome")

// Nenhum filtro. Note que as aspas do string
// podem ser duplas ou simples.
Database1.SQLQuery(" ")
// Procura todos os registros que possuem no
// campo name um string maior que 'Mary' e no campo
// salary um número maior que 1000. Ambos os filtros
// devem ser satisfeitos.
Database1.SQLQuery("name > 'Mary' AND salary > 1000")

// Você pode usar o valor de um Tag para criar
// um string usando a concatenação.
Database1.SQLQuery("name = '"+TagRam1+"'")

// Procura todos os registros que possuem no
// campo name um string que começa por 'ma'.
// Por exemplo: Maria, Mario, Manoela, Marcelo…
Database1.SQLQuery("name LIKE 'ma%'")

// Procura todos os registros que possuem no
// campo vendas um número entre 40000 e 100000.
Database1.SQLQuery("vendas BETWEEN 40000 AND 100000")

// Procura todos os registros que possuem no
// campo opcional um espaço em branco, ordenado em
// ordem descendente pelo campo nome.
Database1.SQLQuery("opcional IS NULL","nome DESC")
// Faz a consulta por data
Database1.SQLQuery("Date = CDate('10/09/2006')")
```

NOTA: este último exemplo é válido para o banco de dados MDB. Outros bancos podem não aceitar este formato de consulta.

Update(bReconsulta)
Atualiza as mudanças feitas no registro. O parâmetro bReconsulta determina se deve ser feita uma consulta após atualização dos registros. Se este parâmetro for 1, a consulta é feita e os registros são atualizados e se for 0, os registros adicionados só estarão visíveis após a atualização da consulta. Esta função está obsoleta e foi mantida para permitir compatibilidade com versões anteriores do software. Atualmente, as gravações no banco de dados são feitas automaticamente quando necessárias.

19.5.16. Funções de Aplicações Remotas

Configure() 
Abre uma caixa de diálogo do driver para configuração dos parâmetros da Aplicação Remota.

Connect()
Estabelece conexão com a Aplicação Remota. Retorna verdadeiro (diferente de zero) se a conexão foi estabelecida com sucesso ou falso (zero) se não.

Disconnect()
Encerra a conexão com a Aplicação Remota. Retorna verdadeiro (diferente de zero) se a conexão foi encerrada com sucesso ou falso (zero) se não.

EditProperties() 
Abre uma janela que permite a edição das propriedades da Aplicação Remota.

LoadCfg(nomeArquivo)
Carrega de um arquivo, especificado por nomeArquivo, a configuração da Aplicação Remota.

SaveCfg(nomeArquivo)
Salva a configuração da Aplicação Remota em um arquivo, especificado por nomeArquivo.

Funções para Arquivos Remotos

Cancel()
Aborta uma transferência de arquivo que esteja em andamento.

GetFile(serverFile, clientFile)
Obtém um arquivo do servidor com o nome serverFile e faz uma cópia deste para um arquivo local com nome clientFile.

19.5.17. Funções do OPCServer

Connect()
Estabelece conexão com o servidor OPC.

Disconnect()
Encerra a conexão com o servidor OPC.

NOTA: Para saber se o servidor OPC está conectado ou não, deve-se monitorar a propriedade *ServerStatus* (ver Propriedades do OPCServer).

EditProperties()
Abre uma janela que permite a edição das propriedades do objeto.

19.6. Atributos

Atributos são dados associados a um objeto que determinam suas características e a maneira com que ele irá se comportar. Normalmente existe uma janela onde você pode modificar os atributos de um objeto. Cada atributo tem um tipo associado representado por seu ícone conforme segue:

Atributo numérico

Atributo string (texto)

Atributo booleano (verdadeiro/falso, true/false

Atributo de sistema, usado internamente pelo Elipse SCADA.

As seções a seguir irão descrever os seguintes atributos:

Atributos Globais

Atributos da Aplicação

Atributos dos Tags

Atributos da Tela

Atributos do Alarme

Atributo de Usuários

Atributo de Watcher

Atributo de OPCServer

Atributos da Receita

Atributos do Histórico

Atributos do Relatório

Atributos do Driver

Atributos de Databases

Atributo de Aplicação Remota

Atributo de Steeplechase

19.6.1. Atributos do Gerenciador Global

Estes atributos são atributos Globais do Elipse SCADA e podem ser modificados de qualquer ponto da aplicação. Porém, deve-se ter cuidado ao modificar qualquer um destes atributos já que eles podem mudar a funcionalidade de um Script ou de um objeto.

currentTime: Um inteiro sem sinal (somente leitura) contendo a data e hora atual do sistema para ser utilizado em expressões ou em displays com formato data/hora. É atualizado uma vez por segundo. Não deve ser usado como contador de tempo em scripts pois não é atualizado durante a execução de scripts. Neste caso, use a função gettime() que é atualizada sempre que é chamada;

day: Um inteiro contendo o dia do sistema;

dayOfWeek: um inteiro sem sinal de 1 to 7 (somente leitura) contendo o número do dia da semana corrente, conforme segue: 1 - Domingo, 2 - Segunda, 3 - Terça, 4 - Quarta, 5 - Quinta, 6 - Sexta, 7 - Sábado;

description: Um string contendo uma breve descrição sobre o objeto global;

hour: Um inteiro contendo a hora do sistema;

lastError: Um inteiro sem sinal de 0 a 65535 (somente leitura) contendo o código do último erro ocorrido em operações com arquivos ou diretórios;

lastErrorStr: Um string contendo a descrição do último erro ocorrido em operações com arquivos ou diretórios.

minute: Um inteiro contendo os minutos do sistema;

month: Um inteiro contendo o mês do sistema;

name: Um string contendo o nome do objeto global, usado como identificador para o mesmo;

performanceCount **:** Indica a performance da aplicação, que pode variar de 0 a 1000. O valor 0 determina que a aplicação está carregada e 1000 determina que a aplicação está com folga de processamento.

second: Um inteiro contendo os segundos do sistema;

year: Um inteiro contendo o ano do sistema;

19.6.2. Atributos da Aplicação

activeAlarms: Um inteiro que indica o número de alarmes ativos na aplicação.

allowClose **:** Um atributo booleano (somente leitura) determinando que o aplicativo possa ser fechado por outros meios que não a função StopRunning() como por exemplo, um clique do mouse no botão Close ou o comando Shutdown do Windows. Retorna verdadeiro (diferente de zero) se o botão está habilitado ou falso (zero) se não está;

centerWindow **:** Um atributo booleano usado para habilitar a centralização da janela da aplicação, no início da sua execução. Retorna True (sem zero) se a centralização da janela for habillitada e false (zero) caso contrário.

closeButton **:** Um atributo booleano (somente leitura) determinando que o botão de close esteja habilitado na janela da aplicação. Retorna verdadeiro (diferente de zero) se o botão está habilitado ou falso (zero) se não está;

description: Um string contendo uma breve descrição sobre a aplicação;

exclusive **:** Um atributo booleano determinando que o Elipse SCADA possua exclusividade para uso da CPU. Retorna verdadeiro (diferente de zero) se o Elipse SCADA é exclusivo ou falso (zero) se não é;

height: Um inteiro sem sinal (de 0 a 65535) determinando a altura da janela da Aplicação, em pixels. É usado juntamente com o atributo *width* para definir o tamanho da janela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) o parâmetro *height* pode variar de 0 a 480 pixels;

isMaximized: Um atributo booleano determinando se a janela da aplicação está maximizada ou não. Retorna verdadeiro (diferente de zero) se a janela está maximizada ou falso (zero) se não está;

isMinimized: Um atributo booleano determinando se a janela da aplicação está minimizada ou não. Retorna verdadeiro (diferente de zero) se a janela está minimizada ou falso (zero) se não está;

isNormal: Um atributo booleano determinando se a janela da aplicação está normalizada ou não. Retorna verdadeiro (diferente de zero) se a janela está normalizada ou falso (zero) se não está;

keyPadBackColor **:** Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo do Key Pad (Teclado em Tela). Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255.

keyPadHeight **:** Um inteiro sem sinal (de 0 a 65535) determinando a altura da janela do Key Pad, em pixels. É usado juntamente com o atributo *KeyPadWidth* para definir o tamanho da janela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) o parâmetro *height* pode variar de 0 a 480 pixels;

keyPadKeyColor **:** Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor das teclas do Key Pad (Teclado em Tela). Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255.

keyPadTextColor **:** Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto das teclas do Key Pad (Teclado em Tela). Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255.

keyPadTitleBar : Atributo booleano, indica se o Key Pad terá barra de título (um, default) ou não (zero).

keyPadVisible : Um atributo booleano determinando que o Key Pad seja visível. Retorna verdadeiro (diferente de zero) se a Key Pad é visível ou falso (zero) se não é.

keyPadWidth **:** Um inteiro sem sinal (de 0 a 65535) determinando a largura da janela do Key Pad, em pixels. É usado juntamente com o atributo *KeyPadHeight* para definir o tamanho da janela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) o parâmetro *height* pode variar de 0 a 640 pixels;

keyPadX **:** Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo da janela do Key Pad, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo da janela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) a sua coordenada *X* pode variar de 0 a 640 pixels;

keyPadY **:** Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo da janela do Key Pad, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo da janela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) a sua coordenada *Y* pode variar de 0 a 480 pixels;

largeButtons: Um atributo booleano habilitando botões grandes no Organizer, Análise Histórica, etc. para facilitar o uso do Touch Screen. Retorna verdadeiro (diferente de zero) se os botões grandes estão habilitados ou falso (zero) se não estão.

maximizeButton **:** Um atributo booleano(somente leitura) determinando que o botão maximizar esteja habilitado na janela da aplicação. Retorna verdadeiro (diferente de zero) se o botão está habilitado ou falso (zero) se não está.

minimizeButton **:** Um atributo booleano (somente leitura) determinando que o botão minimizar esteja habilitado na janela da aplicação. Retorna verdadeiro (diferente de zero) se o botão está habilitado ou falso (zero) se não está.

name: Um string contendo o nome da aplicação, usado como identificador para a mesma.

noMousePointer **:** Um atributo booleano usado para desabilitar (esconder) o ponteiro do mouse em tempo de execução (quando executando uma aplicação). Retorna verdadeiro (diferente de zero) se o mouse está desabilitado ou falso (zero) se não está.

noScreenSaver **:** Um atributo booleano usado para desabilitar o screen saver em tempo de execução (quando executando uma aplicação). Retorna verdadeiro (diferente de zero) se o screen saver está desabilitado ou falso (zero) se não está.

numPadCloseButton: Atributo booleano, indica se o botão para fechar o NumPad deverá ser apresentado (valor um, default) ou não (valor zero).

numPadTitleBar: Atributo booleano, indica se o Numeric Pad terá barra de título (um, default) ou não (zero).

numPadTitleBarText: Atributo string, permite indicar um texto para a barra de título do Numeric Pad (vazio, por default).

revision: Um inteiro sem sinal (somente leitura) indicando o número de revisão da aplicação. É automaticamente incrementado cada vez que a aplicação é salva.

saveKeyPadCoord: Um atributo booleano determinando que as coordenadas e o tamanho da janela do KeyPad (Teclado em tela) seja salvo para carregar na próxima abertura do mesmo. Retorna verdadeiro (diferente de zero) se o salvamento está habilitado ou falso (zero) se não está.

startMaximized: Um atributo booleano determinando que a janela da aplicação inicie maximizada. Retorna verdadeiro (diferente de zero) se a aplicação inicia maximizada ou falso (zero) se não.

startMinimized: Um atributo booleano determinando que a janela da aplicação inicie minimizada. Retorna verdadeiro (diferente de zero) se a aplicação inicia minimizada ou falso (zero) se não.

startNormal: Um atributo booleano determinando que a janela da aplicação inicie normalizada. Retorna verdadeiro (diferente de zero) se a aplicação inicia normalizada ou falso (zero) se não.

startScreen: Somente para uso do Elipse SCADA. Não são permitidas modificações por parte de usuários.

titleBar: Um atributo booleano determinando que a Barra de Título seja mostrada na janela da aplicação. Retorna verdadeiro (diferente de zero) se a Barra de Título é mostrada ou falso (zero) se não é.

useKeyPad: Um atributo booleano habilitando o uso do KeyPad (Teclado em tela) em tempo de execução. Retorna verdadeiro (diferente de zero) se o KeyPad está habilitado ou falso (zero) se não está.

userAccessLevel: Um inteiro sem sinal de 0 a 65535 (somente leitura) determinando o nível de acesso do usuário logado na aplicação.

userLogin: Um string (somente leitura) contendo o nome de usuário (login) corrente.

userName: Um string (somente leitura) contendo o usuário corrente logado na aplicação.

version: Um string (somente leitura) indicando a versão do Elipse que salvou a aplicação por último.

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura da janela da Aplicação, em pixels. É usado juntamente com o atributo *height* para definir o tamanho da janela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) o parâmetro *width* pode variar de 0 a 640 pixels.

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo da janela da Aplicação, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo da janela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) a sua coordenada *X* pode variar de 0 a 640 pixels.

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo da janela da Aplicação, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo da janela.

19.6.3. Atributos de Tags

Atributos Comuns

advised: (booleano, somente leitura) Usado para indicar se o item está em *advise*, isto é, se os objetos associados ao tag são "informados" sempre que o valor do tag mudar.

alarmDelay: (inteiro sem sinal) É o tempo mínimo (em milisegundos) que o tag deve permanecer em uma mesma condição de alarme antes deste tornar-se ativo. Se alarmDelay for modificado em tempo de execução, o contador de tempo é reiniciado. Este atributo só é válido se o atributo delayedAlarm for True (ver atributo delayedAlarm).

alternateName: (string) Nome alternativo do tag para ser usado no registro (log) de alarmes do Tag.

alwaysInAdvise: (boolean) Permite forçar um tag estar em *advise* (ver atributo advised).

currentAlarm: (inteiro sem sinal, apenas leitura) Indica o estado de alarme atual do tag, conforme segue: 0, sem alarme; 1, LoLo; 2, Low; 3, High e 4, HiHi.

delayedAlarm: (booleano) Se TRUE, o tag deve permanecer um tempo mínimo (especificado em alarmDelay) em uma mesma condição de alarme (Hi, HiHi, Lo, LoLo) para ativar (ou gerar) o alarme correspondente.

description: (string) Contém uma breve descrição sobre o tag.

enableAlternateName: (booleano) Habilita utilizar um nome alternativo (indicado no atributo alternateName) para o registro (log) de alarmes do tag.

enableReturnLog: (booleano) Usado para habilitar o log da mensagem de retorno do alarme. Retorna verdadeiro (diferente de zero) se o log está habilitado ou falso (zero) se não está.

name: (string) Contém o nome do tag, usado como identificador do tag.

returnMessage: (string) Contém a mensagem de retorno de alarme para o tag.

timeStamp: (real, somente leitura) Contém a última data e hora em que o valor do tag foi modificado.

Atributos de Grupos de Tags

totalAlarms: (inteiro sem sinal, apenas leitura) que indica o número de alarmes ativos nos tags e subgrupos do grupo.

Atributos de Tags OPC

quality: Um inteiro sem sinal (somente leitura), que indica a qualidade do tag OPC.

Atributos de Tags PLC

autoRead: Um atributo booleano usado para habilitar a leitura automática do PLC. Retorna verdadeiro (diferente de zero) se a leitura automática está habilitada ou falso (zero) se não está. Deve ser usado em conjunto com o atributo *enableScan* devendo ambos estarem marcados ou desmarcados;

autoWrite: Um atributo booleano usado para habilitar a escrita automática no PLC. Retorna verdadeiro (diferente de zero) se a escrita automática está habilitada ou falso (zero) se não está;

driverName: Um string contendo o nome do Driver do PLC conectado ao sistema, é usado como um identificador do Driver;

enableScan: Um atributo booleano usado para habilitar a leitura do PLC. Retorna verdadeiro (diferente de zero) se o scan está habilitado ou falso (zero) se não está. Deve ser usado em conjunto com o atributo *autoRead* devendo ambos estarem marcados ou desmarcados;

high1: Um número real configurando o limite superior do PLC. Ele é usado junto com o atributo *low1* para definir um intervalo de variação;

high2: Um número real configurando o limite superior do sistema. Ele é usado junto com o atributo *low2* para definir um intervalo de variação;

low1: Um número real configurando o limite inferior do PLC. Ele é usado junto com o atributo *high1* para definir um intervalo de variação;

low2: Um número real configurando o limite inferior do sistema. Ele é usado junto com o atributo *high2* para definir um intervalo de variação;

n1: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro N1 conforme a documentação do Driver;

n2: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro N2 conforme a documentação do Driver;

n3: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro N3 conforme a documentação do Driver;

n4: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro N4 conforme a documentação do Driver;

nFailReads: Um inteiro sem sinal (somente leitura) que indica o número total de leituras incorretas;

nFailWrites: Um inteiro sem sinal (somente leitura) que indica o número total de escritas incorretas;

nOKReads: Um inteiro sem sinal (somente leitura) que indica o número total de leituras feitas com sucesso;

nOKWrites: Um inteiro sem sinal (somente leitura) que indica o número total de escritas feitas com sucesso;

realScan: Um número real (somente leitura) indicando o tempo real de scan em que o Elipse está conseguindo ler o Tag.

scaling: Um atributo booleano usado para habilitar escalas de valores entre o PLC e o sistema. Retorna verdadeiro (diferente de zero) se a escala está habilitada ou falso (zero) se não está. Com este atributo habilitado você pode configurar os atributos low1, high1, low2 e high2;

scanTime: Um inteiro com sinal (de -32768 a 32767) configurando o intervalo de tempo entre duas leituras;

status: Um string (somente leitura) contendo o estado atual de comunicação do Tag: "Reading" (lendo), "Writing" (escrevendo), "Idle" (fazendo nada);

Atributos de Tags DDE

computer: Um string contendo o nome do computador a ser conectado usando DDE. É usado como identificador do computador em uma rede e pode ser mudado em tempo de execução.

high1: Um número real configurando o limite superior do PLC. Ele é usado junto com o atributo *low1* para definir um intervalo de variação;

high2: Um número real configurando o limite superior do sistema. Ele é usado junto com o atributo *low2* para definir um intervalo de variação;

item: Um string contendo o nome dos dados a serem monitorados pelo Tag DDE. É usado como identificador dos dados. Em uma conexão entre dois Elipse SCADA este atributo pode ser um Tag, em uma conexão com o Excel este atributo deve ser declarado como RxCy onde x é a linha e y a coluna de uma célula específica;

low1: Um número real configurando o limite inferior do PLC. Ele é usado junto com o atributo *high1* para definir um intervalo de variação;

low2: Um número real configurando o limite inferior do sistema. Ele é usado junto com o atributo *high2* para definir um intervalo de variação;

scaling: Um atributo booleano usado para habilitar escalas de valores entre o PLC e o sistema. Retorna verdadeiro (diferente de zero) se a escala está habilitada ou falso (zero) se não está. Com este atributo habilitado você pode configurar os atributos low1, high1, low2 e high2;

server: Um string contendo o nome da aplicação a ser conectada usando DDE. É usado como um identificador da aplicação no computador especificado para a conexão. Por exemplo, você deseja criar uma conexão DDE no computador *Elipse_Software*, que está executando o Elipse SCADA cujo nome DDE é *ELIPSE$*. Você pode mudar este atributo em tempo de execução;

topic: Um string contendo o nome da janela ou área de trabalho a ser conectada usando DDE. É usado como identificador de uma área de trabalho no servidor da aplicação;

Atributos de Tags Demo

delay: Um inteiro configurando o número de períodos entre cada geração de valor para o tag demo. Por exemplo, se for 1 um valor é gerado a cada período, se for 2, gera um valor a cada dois períodos, e assim por diante. É usado junto com o atributo period para controlar o intervalo de tempo para a variação dos dados. A fórmula que exprime a frequência de modificação do tag demo é period * delay.

enabled: Um atributo booleano usado para habilitar a simulação de dados do tag demo enquanto o tag está ativo. Retorna verdadeiro (diferente de zero) se a simulação de dados está habilitada ou falso (zero) se não está;

highLimit: Um número real configurando o limite alto do tag demo. É usado junto com o atributo lowLimit para definir o intervalo de variação do Tag;

increment: Um número real determinando o incremento a ser adicionado aos dados conforme a frequência de tempo especificada em period.

lowLimit: Um número real configurando o limite baixo do tag demo. É usado junto com o atributo highLimit para definir o intervalo de variação do Tag.

period: Um inteiro configurando o intervalo em milisegundos entre a geração de cada novo valor para o tag demo. É usado junto com o atributo delay para controlar o intervalo de tempo para a variação dos dados.

type: Um inteiro configurando o tipo de variação do tag demo. A variação é descrita por uma curva e pode ter os seguintes tipos:

Tipos de variação

NOME	VALOR	DESCRIÇÃO
 Randômico	0	Dados variando randomicamente
 Senoidal	1	Dados variando conforme uma função seno
 Onda Quadrada	2	Dados alternam entre dois valores
 Rampa de Subida	3	Um gráfico tipo dente de serra onde os dados variam de um limite mais baixo para um mais alto
 Triangular	4	Um gráfico tipo dente de serra onde os dados variam entre dois limites
 Rampa de Descida	5	Um gráfico tipo dente de serra onde os dados variam de um limite mais alto para um mais baixo

Atributos de Tags Bloco

autoRead: Um atributo booleano usado para habilitar a leitura automática do PLC. Retorna verdadeiro (diferente de zero) se a leitura automática está habilitada ou falso (zero) se não está. Deve ser usado em conjunto com o atributo *enableScan* devendo ambos estarem marcados ou desmarcados.

autoWrite: Um atributo booleano usado para habilitar a escrita automática no PLC. Retorna verdadeiro (diferente de zero) se a escrita automática está habilitada ou falso (zero) se não está.

b1: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro B1 conforme a documentação do Driver.

b2: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro B2 conforme a documentação do Driver.

b3: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro B3 conforme a documentação do Driver.

b4: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro B4 conforme a documentação do Driver.

driverName: um string contendo o nome do Driver do PLC conectado ao sistema, é usado como um identificador do Driver.

enableScan: Um atributo booleano usado para habilitar a leitura do PLC. Retorna verdadeiro (diferente de zero) se o scan está habilitado ou falso (zero) se não está. Deve ser usado em conjunto com o atributo *autoRead* devendo ambos estarem marcados ou desmarcados.

nFailReads: Um inteiro sem sinal (somente leitura) que indica o número total de leituras incorretas.

nFailWrites: Um inteiro sem sinal (somente leitura) que indica o número total de escritas incorretas.

nOKReads: Um inteiro sem sinal (somente leitura) que indica o número total de leituras feitas com sucesso.

nOKWrites: Um inteiro sem sinal (somente leitura) que indica o número total de escritas feitas com sucesso.

realScan: um número real (somente leitura) indicando o tempo real de scan em que o Elipse está conseguindo ler o Tag.

scanTime: Um inteiro com sinal (de -32768 a 32767) configurando o intervalo de tempo entre duas leituras.

size: Um inteiro determinando o tamanho do bloco a ser monitorado pelo Elipse SCADA.

status: Um string (somente leitura) contendo o estado atual de comunicação do Tag: "Reading" (lendo), "Writing" (escrevendo), "Idle" (fazendo nada).

Atributos de Elementos de Tags Bloco

high1: Um número real configurando o limite superior do PLC. Ele é usado junto com o atributo *low1* para definir um intervalo de variação.

high2: Um número real configurando o limite superior do sistema. Ele é usado junto com o atributo *low2* para definir um intervalo de variação.

index: Um inteiro sem sinal (de 0 ao número total de elementos - 1) determinando o índice (iniciando em zero) do elemento no bloco.

low1: Um número real configurando o limite inferior do PLC. Ele é usado junto com o atributo *high1* para definir um intervalo de variação.

low2: Um número real configurando o limite inferior do sistema. Ele é usado junto com o atributo *high2* para definir um intervalo de variação.

scaling: Um atributo booleano usado para habilitar escalas de valores entre o PLC e o sistema. Retorna verdadeiro (diferente de zero) se a escala está habilitada ou falso (zero) se não está. Com este atributo habilitado você pode configurar os atributos low1, high1, low2 e high2.

Atributos de Tags Matriz

nColumns: Um inteiro sem sinal configurando o número de colunas do Tag Matriz.

nRows: Um inteiro sem sinal configurando o número de linhas do Tag Matriz.

Atributos de Tags Crono

acum: Valor atual do cronômetro;

autoRestart: Reinicia automaticamente ao atingir valor de preset;

description: Descrição do ítem.

enable: Habilita/Desabilita cronômetro**.**

name: O identificador usado para referência do item.

resetAcum: Reseta acumulador quando habilitado (reiniciado)

saveAcum: Salva acumulador ao sair.

Atributos de Tags Expressão

expression: (string) Contém a própria expressão do tag, permitindo que ela seja modificada em tempo de execução.

Atributos de Tags RAM

initialValue: (string) Contém um valor inicial para o tag e permite que este valor seja modificado em tempo de execução.

19.6.4. Atributos da Tela

accessLevel: Um inteiro sem sinal de 0 a 65535 (somente leitura) determinando o nível de acesso da Tela.

background: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo da Tela. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255. Exemplo: `tela1.background = RGB(255,0,0)`

bitmapName: Um string contendo o nome de um bitmap usado como fundo em uma Tela. Exemplo: `screen.bitmapName = "Leaves.bmp"`

border : somente para telas janeladas e de tamanho fixo. Um atributo booleano determinando que a uma borda de 1 pixel de largura seja mostrada em volta da tela. Retorna verdadeiro (diferente de zero) se a borda é mostrada (default) ou falso (zero) se não é;

caption: Um atributo booleano determinando que a Barra de Título seja mostrada na Tela. Retorna verdadeiro (diferente de zero) se a Barra de Título é mostrada ou falso se não é;

clipChildren: Um atributo booleano, habilitando o uso do clipping para o redesenho dos objetos de tela relativo ao fundo da mesma. Retorna verdadeiro (diferente de zero) se o clipping está habilitado ou falso se não está. Este atributo deve ser habilitado somente se os objetos de tela não estão sobrepostos e não serão movidos, caso contrário poderá não surtir o efeito desejado;

closeButton: Um atributo booleano determinando que o botão de close seja mostrado na Tela. Retorna verdadeiro (diferente de zero) se o botão é mostrado ou falso (zero) se não é;

description: Um string contendo uma breve descrição sobre o Tag;

fullScreen: Um atributo booleano determinando o tipo da tela (cheia ou janelada). Retorna verdadeiro (diferente de zero) se a tela é cheia ou falso (zero) se é janelada;

height: Um inteiro sem sinal (de 0 a 65535) determinando a altura da Tela, em pixels. É usado juntamente com o atributo *width* para definir o tamanho da Tela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) o parâmetro *height* pode variar de 0 a 480 pixels;

maximizeButton: Um atributo booleano determinando que o botão maximizar seja mostrado na Tela. Retorna verdadeiro (diferente de zero) se o botão é mostrado ou falso (zero) se não é;

minimizeButton: Um atributo booleano determinando que o botão minimizar seja mostrado na Tela. Retorna verdadeiro (diferente de zero) se o botão é mostrado ou falso (zero) se não é;

modal: somente para telas janeladas (fullScreen=0). Um atributo booleano determinando que a tela seja, ou não, do tipo modal (ela deve ser fechada para que o

foco passe para outras telas abertas). Retorna verdadeiro (diferente de zero) se a Tela é modal ou falso (zero) se não é;

moveable: Um atributo booleano determinando que a Tela possa ser movida em tempo de execução. Retorna verdadeiro (diferente de zero) se a Tela pode ser movida ou falso (zero) se não pode;

name: Um string contendo o nome da Tela, usado como identificador da Tela;

popup: Somente para telas janeladas (fullScreen=0). Um atributo booleano determinando que a tela seja, ou não, do tipo popup (ela é automaticamente fechada quando perde o foco). Retorna verdadeiro (diferente de zero) se a Tela é popup ou falso (zero) se não é;

resizeable: Um atributo booleano determinando que a Tela possa ser redimensionada em tempo de execução. Retorna verdadeiro (diferente de zero) se a Tela pode ser redimensionada ou falso (zero) se não pode;

scrollBars: Um atributo booleano determinando que o scroll bar seja mostrado na Tela. Retorna verdadeiro (diferente de zero) se o scroll bar é mostrado ou falso (zero) se não é;

visible: Um atributo booleano determinando que a Tela seja visível. Retorna verdadeiro (diferente de zero) se a Tela é visível ou falso (zero) se não é;

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura da Tela, em pixels. É usado juntamente com o atributo *height* para definir o tamanho da Tela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) o parâmetro *width* pode variar de 0 a 640 pixels;

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo da Tela, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo da Tela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) a sua coordenada *X* pode variar de 0 a 640 pixels;

xOrigin: Um inteiro sem sinal (de 0 a 65535) determinando a coordenada *X* para a origem da Tela, em pixels. É usado juntamente com o atributo *yOrigin* para definir o tamanho da Tela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) o parâmetro *xOrigin* pode variar de 0 a 640 pixels;

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo da Tela, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo da Tela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma

janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) a sua coordenada *Y* pode variar de 0 a 480 pixels;

yOrigin: Um inteiro sem sinal (de 0 a 65535) determinando a coordenada *Y* para a origem da Tela, em pixels. É usado juntamente com o atributo *xOrigin* para definir o tamanho da Tela. Este atributo deve ser definido conforme a resolução usada no Windows para que se tenha uma janela sem scroll bar. Por exemplo, se você está usando uma resolução de 640x480 (VGA) o parâmetro *yOrigin* pode variar de 0 a 480 pixels;

19.6.5. Atributos dos Objetos de Tela

Os objetos de tela possuem atributos em comum, que aparecem em todos os objetos e atributos específicos, associados as suas características únicas. Listamos todos estes atributos a seguir.

Atributos Comuns

accessLevel: (inteiro) Permite atribuir um nível de segurança ao objeto. Se for 0 (zero), todos os usuários tem livre acesso ao objeto, podendo clicar e entrar dados neste. Se for 1 (um) ou mais, apenas os usuários com o nível de acesso menor ou igual podem usar o objeto.

enabled: Valor booleano que, quando verdadeiro (TRUE), habilita o acesso ao objeto via mouse ou teclado. Este atributo não afeta objetos que não provêem acesso via teclado ou mouse (como o Display ou o Gauge).

height: Inteiro sem sinal (de 0 a 65535) que Determina a altura do objeto, em pixels. É usado juntamente com o atributo width para definir o tamanho do objeto.

showTip: Valor booleano que habilita o objeto a mostrar uma pequena dica (tip) quando o mouse está sobre ele.

visible: Valor booleano que indica se o objeto está visível ao usuário (TRUE) ou não (FALSE).

width: Inteiro sem sinal (de 0 a 65535) Determinando a largura do objeto, em pixels. É usado juntamente com o atributo height para definir o tamanho do objeto.

x, **y**: Dois inteiros com sinal (de -32768 a 32767) Determinando a coordenada X e Y para o canto superior esquerdo do objeto, em pixels, a partir do canto superior esquerdo da Tela em que está o objeto.

Atributos do Alarme

ackALMColor: Número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto para uma mensagem de Alarme reconhecida.

Este valor também pode ser especificado pela função RGB (r, g, b) onde cada componente RGB pode variar de 0 a 255. Exemplo: `Alarm.ackAlmColor = RGB(255,0,255)`

alarmState: (booleano) Determina que a coluna de status dos alarmes seja mostrada no objeto. Retorna verdadeiro (diferente de zero) se a coluna de status é mostrada ou falso (zero) se não é.

alarmType: (booleano) Determina se a coluna de tipo de Alarme será mostrada no objeto: verdadeiro (diferente de zero) é mostrada; falso (zero) não é.

comment: (booleano) Determina que a coluna de comentários seja mostrada no objeto Alarme. Retorna verdadeiro (diferente de zero) se a coluna de comentários é mostrada ou falso (zero) se não é.

commentLength: Inteiro sem sinal (de 1 a 199) Determinando o comprimento do comentário que será mostrado. Somente disponível se o atributo *comment* está habilitado;

date: (booleano) Determina que a coluna de data seja mostrada no objeto Alarme. Retorna verdadeiro (diferente de zero) se a coluna de data é mostrada ou falso (zero) se não é.

description: String contendo uma breve descrição sobre o Alarme.

displayTitle: (booleano) Determina que a barra de título do Alarme seja mostrada. Retorna verdadeiro (diferente de zero) se a barra de título é mostrada ou falso (zero) caso contrário.

event: (booleano) Determina que a coluna de eventos seja mostrada no objeto Alarme. Retorna verdadeiro (diferente de zero) se a coluna de eventos é mostrada ou falso (zero) se não é.

finalPriority: Inteiro sem sinal (de 1 a 199) determinando a maior prioridade de mensagens de alarme a ser monitorada pelo objeto. Somente mensagens de Alarme com prioridade inferior a finalPriority serão mostrados no objeto.

initialPriority: Inteiro sem sinal (de 1 a 199) determinando a menor prioridade de mensagens de alarme a ser monitorada pelo objeto. Somente mensagens de Alarme com prioridade maior que initialPriority serão mostrados no objeto.

largeButtons: (booleano) Habilita botões grandes na toolbar do objeto Alarmes. Retorna verdadeiro (diferente de zero) se os botões grandes estão habilitados ou falso (zero) se não estão.

limit: (booleano) Determina que a coluna de limites dos alarmes seja mostrada no objeto. Retorna verdadeiro (diferente de zero) se a coluna de limites é mostrada ou falso (zero) se não é.

limitLength: Inteiro sem sinal (de 1 a 10) determinando o comprimento dos limites dos Alarmes que serão mostrados. Somente disponível se o atributo *limit* está habilitado;

limitPrec: Inteiro sem signal (de 0 a limitLength - 2) determinando o número de dígitos decimais para os limites que serão mostrados no Alarme.

name: String contendo o nome do Alarme, usado como identificador do Alarme.

priority: (booleano) Determina que a coluna de prioridades de alarmes seja mostrada no objeto. Retorna verdadeiro (diferente de zero) se a coluna de prioridades é mostrada ou falso (zero) se não é.

rtnColor: Número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da mensagem de retorno do Alarme. Este valor também pode ser especificado pela função RGB (r, g, b) onde cada componente RGB pode variar de 0 a 255.

Exemplo:

```
Alarm.rtnColor = RGB(255,255,0)
```

tagName: (booleano) Determina que a coluna de nomes dos Tags seja mostrada no objeto. Retorna verdadeiro (diferente de zero) se a coluna de nomes dos Tags é mostrada ou falso (zero) se não é.

tagNameLength: Inteiro sem signal (de 1 a 32) determinando o comprimento dos nomes dos Tags que serão mostrados. Somente disponível se o atributo *tagName* está habilitado;

time: (booleano) Determina que a coluna de hora seja mostrada no objeto Alarme. Retorna verdadeiro (diferente de zero) se a coluna de hora é mostrada ou falso (zero) se não é.

titleBarColor: Número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da linha de título do Alarme. Este valor também pode ser especificado pela função RGB (r, g, b) onde cada componente RGB pode variar de 0 a 255. Exemplo: Alarm.titleBarColor = RGB(255,255,0)

titleTextColor: Número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de texto do título do Alarme. Este valor também pode ser especificado pela função RGB (r, g, b) onde cada componente RGB pode variar de 0 a 255. Exemplo: Alarm.titleTextColor = RGB(255,255,0)

toolbar: (booleano) Determina que a barra de ferramentas do objeto Alarme seja mostrado. Retorna verdadeiro (diferente de zero) se a barra de ferramentas é mostrado ou falso (zero) se não é.

toolbarColor: Número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da barra de ferramentas do Alarme. Este valor também pode ser especificado pela função RGB (r, g, b) onde cada componente RGB pode variar de 0 a 255. Exemplo: Alarm.toolbarColor = RGB(255,255,0)

type:Inteiro sem sinal (de 0 a 1) definindo o tipo do Alarme, que pode ser resumido (0) ou histórico (1). Esta propriedade é somente de leitura.

unAckALMColor: Número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto para uma mensagem de Alarme não reconhecida. Este valor também pode ser especificado pela função RGB (r, g, b) onde cada componente RGB pode variar de 0 a 255.

userLength: (inteiro de 1 a 20) Indica o tamanho da coluna Usuário (user), em caracteres.

userName: (booleano) Se verdadeiro, indica que a coluna Usuário (user) deve ser mostrada.

value: (booleano) Determina que a coluna de valores dos Tags seja mostrada no objeto Alarme. Retorna verdadeiro (diferente de zero) se a coluna de valores dos Tags é mostrada ou falso (zero) se não é.

valueLength: Inteiro sem sinal (de 1 a 10) determinando o comprimento dos valores dos Tags que serão mostrados. Somente disponível se o atributo *comment* está habilitado;

valuePrec: Inteiro sem sinal (de 0 a valueLength - 2) determinando o número de dígitos decimais para os valores que serão mostrados no Alarme.

windowColor: Número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo do Alarme. Este valor também pode ser especificado pela função RGB (r, g, b) onde cada componente RGB pode variar de 0 a 255.

Atributos da Animação

backgroundColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo da Animação. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

blinkTime: Um inteiro sem sinal (de 0 a 100) determinando o tempo em ms de pisca-pisca da Animação, ou seja, o tempo em que a zona default irá piscar alternando com a zona atual.

border: Um atributo booleano determinando que a borda da Animação seja mostrada no objeto. Retorna verdadeiro (diferente de zero) se a borda é mostrada ou falso (zero) se não é;

description: um String contendo uma breve descrição sobre a Animação. Exemplo:

```
Animation.description = "Animation Object"
```

name: Um string contendo o nome da Animação, usado como identificador da Animação.

transparent: Um atributo booleano determinando que o fundo da Animação seja transparente. Retorna verdadeiro (diferente de zero) se a o fundo é transparente ou falso (zero) se não é;

Atributos da Barra

backgroundColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo do Gráfico de Barras. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

barColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da Barra no gráfico. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

bipolar: Um atributo booleano determinando que o Gráfico de Barras seja bipolar, desta forma as barras possuem dois lados divididos por um centro. As barras crescem a partir do centro em direção aos limites do gráfico. Retorna verdadeiro (diferente de zero) se o Gráfico de Barras é bipolar ou falso (zero) se não é;

bottomRange: Um número real configurando o valor mínimo para o Gráfico de Barras. É usado junto com o atributo *topRange* para definir um intervalo de valores para a Barra;

bottomRuler: Um atributo booleano determinando que uma régua de valores seja mostrada na parte inferior do Gráfico de Barras. Retorna verdadeiro (diferente de zero) se a régua inferior é mostrada ou falso (zero) se não é;

description: Um string contendo uma breve descrição sobre o Gráfico de Barras;

enable3D: Um atributo booleano usado para habilitar um efeito 3D na Barra. Retorna verdadeiro (diferente de zero) se o efeito 3D está habilitado ou falso (zero) se não está.

enableRuler: Um atributo booleano determinando que uma régua de valores seja mostrada no Gráfico de Barras. Retorna verdadeiro (diferente de zero) se a régua é mostrada ou falso (zero) se não é;

gridColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da grade e divisões do Gráfico de Barras. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

label: Um string contendo a unidade de medida a ser mostrada na régua. Exemplo: Barra1.label = "MHz"

labelEnable: Um atributo booleano determinando que o label da régua seja mostrado no Gráfico de Barras. Retorna verdadeiro (diferente de zero) se o label da régua é mostrado ou falso (zero) se não é;

leftRuler: Um atributo booleano determinando que uma régua de valores seja mostrada à esquerda do Gráfico de Barras. Retorna verdadeiro (diferente de zero) se a régua à esquerda é mostrada ou falso (zero) se não é;

middle: Um número real determinando o centro das barras bipolares. Somente disponível se o atributo *middle* está habilitado;

name: Um string contendo o nome do Gráfico de Barras, usado como identificador da Barra;

orientation: Um inteiro sem sinal determinando a orientação do Gráfico de Barras, conforme a tabela a seguir:

GAUGE	ORIENTAÇÃO
	Vertical, de cima para baixo
	Vertical, de baixo para cima
	Horizontal, da esquerda para direita
	Horizontal, da direita para esquerda

rightRuler: Um atributo booleano determinando que uma régua de valores seja mostrada à direita do Gráfico de Barras. Retorna verdadeiro (diferente de zero) se a régua à direita é mostrada ou falso (zero) se não é;

rulerColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da régua do Gráfico de Barras. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

rulerDivisions: Um inteiro sem sinal (de 0 a 100) determinando o número de divisões na régua do Gráfico de Barras. Somente disponível se o atributo *enableRuler* está habilitado;

spacing: Um inteiro sem sinal (de 0 a 10) determinando o espaço (em pixels) entre as barras do gráfico;

textColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto da régua. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

thickness: Um inteiro sem sinal configurando a largura do efeito 3D da barra (em pixels);

topRange: Um número real configurando o valor máximo para o Gráfico de Barras. É usado junto com o atributo *bottomRange* para definir um intervalo de valores para a Barra;

topRuler: Um atributo booleano determinando que uma régua de valores seja mostrada na parte superior do Gráfico de Barras. Retorna verdadeiro (diferente de zero) se a régua superior é mostrada ou falso (zero) se não é;

Atributos do Browser

ascending: Um atributo booleano determinando a ordem dos registros a serem mostrados no Browser. Retorna verdadeiro (diferente de zero) se a ordem é ascendente ou falso (zero) se é descendente. O default é ascendente;

backColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da área fora da tabela do Browser. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

completeRows: Um inteiro sem sinal (de 0 a 65535) determinando o número das linhas completamente visíveis, não incluindo a última linha se ela estiver parcialmente visível.

curSel: Um inteiro sem sinal (de 0 a 65535) determinando o número do registro selecionado (somente leitura). Retorna -1 se nenhuma linha está selecionada;

description: Um string contendo uma breve descrição sobre o Browser;

firstRec: Um inteiro sem sinal (de 0 a 65535) determinando o número do primeiro registro selecionado para ser visto (somente leitura).

gridColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor das linhas da grade do Browser. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

lastRec: Um inteiro sem sinal (de 0 a 65535) determinando o número do último registro selecionado para ser visto (somente leitura).

name: Um string contendo o nome do Browser, usado como identificador do Browser;

rowColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo das linhas não selecionadas do Browser. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

rowTextColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto das linhas não selecionadas do Browser. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

selColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo da linha corrente, selecionada no Browser. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

selTextColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto da linha corrente, selecionada no Browser. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

titleColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da linha de título do Browser. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

titleTextColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de texto do título do Browser. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

topRow: Um inteiro sem sinal (de 0 a 65535) determinando o número do registro mostrado na primeira linha visível (somente leitura).

visibleRows: Um Inteiro sem sinal (de 0 a 65535) determinando o número de linhas visíveis, incluindo a última mesmo que esteja aparecendo só parcialmente (somente leitura).

Atributos do Botão

backgroundColor0: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do Botão quando está em seu estado normal (não pressionado). Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

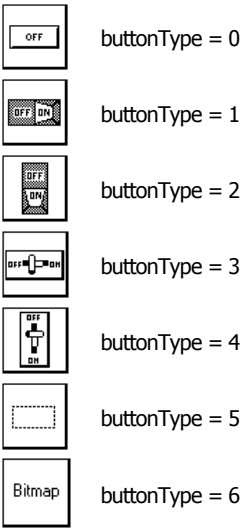
backgroundColor1: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do Botão quando está pressionado. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

bitmap0: Um string contendo o nome do bitmap a ser mostrado como fundo de um Botão quando está em seu estado normal (não pressionado);

bitmap1: Um string contendo o nome do bitmap a ser mostrado como fundo de um Botão quando está pressionado;

buttonAction: Um inteiro sem signal (de 0 a 2) determinando o tipo de ação do botão, que pode ser Momentâneo (*Momentary*) (0), Liga-desliga (*Toggle*) (1) ou Jog (2);

buttonType: Um inteiro sem signal (de 0 a 6) determinando o tipo do botão de 7 tipos disponíveis:



description: Um string contendo uma breve descrição sobre o Botão;

goToScreen: Somente para uso do Elipse SCADA. Não são permitidas modificações por parte de usuários.

name: Um string contendo o nome do Botão, usado como identificador do Botão;

text0: Um string contendo o texto que será mostrado no Botão quando ele está em estado normal (não pressionado);

text1: Um string contendo o texto que será mostrado no Botão quando ele está pressionado;

textColor0: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto do Botão quando está em seu estado normal (não pressionado). Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

textColor1: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto do Botão quando está pressionado. Este valor

também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

value0: Um inteiro sem sinal (de 0 a 65535) determinando o valor do Botão quando em estado normal (não pressionado);

value1: Um inteiro sem sinal (de 0 a 65535) determinando o valor do Botão quando pressionado;

visible: Um atributo booleano determinando que o Botão seja visível. Retorna verdadeiro (diferente de zero) se o Botão é visível ou falso (zero) se não é;

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura do Botão, em pixels. É usado juntamente com o atributo *height* para definir o tamanho do Botão;

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo do Botão, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo do Botão a partir da origem da Tela (0,0);

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo do Botão, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo do Botão a partir da origem da Tela (0,0);

Atributos do Display

backgroundColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo do Display. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

description: Um string contendo uma breve descrição sobre o Display;

enabled: Um atributo booleano usado para habilitar o acesso do teclado e mouse ao objeto Display. Retorna verdadeiro (diferente de zero) se o acesso está habilitado ou falso (zero) se não está;

height: Um inteiro sem sinal (de 0 a 65535) determinando a altura do Display, em pixels. É usado juntamente com o atributo *width* para definir o tamanho do Display;

horizontalAlign: Um inteiro sem sinal (de 0 a 2) determinando o alinhamento horizontal para os dados que serão mostrados no Display, conforme a tabela abaixo:

VALOR	ALINHAMENTO
0	Esquerda
1	Centro

2	Direita

multiLine: Um atributo booleano usado para habilitar múltiplas linhas no Display. Retorna verdadeiro (diferente de zero) se o uso de múltiplas linhas está habilitado ou falso (zero) se não está;

name: Um string contendo o nome do Display, usado como identificador do Display;

precision: Um inteiro sem sinal (de 0 a 254) determinando o número de dígitos decimais a serem mostrados no Display. Nunca poderá ser maior que o atributo *size*.

prefix: Um string contendo um prefixo a ser mostrado antes do valor do Tag associado ao Display;

size: Um inteiro sem sinal (de 0 a 255) determinando o tamanho dos dados a serem mostrados no Display.

showTip: Um atributo booleano habilitando o objeto a mostrar uma Tip (dica) quando o mouse está sobre ele. Retorna verdadeiro (diferente de zero) se a Tip está habilitada ou falso (zero) se não está;

suffix: Um string contendo um sufixo a ser mostrado depois do valor do Tag associado ao Display;

textColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto do Display. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

type: Um inteiro sem sinal (de 1 a 2) determinando o tipo de dados a ser mostrado no Display, que pode ser um string (1) ou um número (2).

verticalAlign: Um inteiro sem sinal (de 0 a 2) determinando o alinhamento vertical para os dados que serão mostrados no Display, conforme a tabela abaixo:

VALOR	ALINHAMENTO
0	Topo
1	Centro
2	Abaixo

visible: Um atributo booleano determinando que o Display seja visível. Retorna verdadeiro (diferente de zero) se o Display é visível ou falso (zero) se não é;

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura do Display, em pixels. É usado juntamente com o atributo *height* para definir o tamanho do Display;

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo do Display, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo do Display a partir da origem da Tela (0,0);

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo do Display, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo do Display a partir da origem da Tela (0,0);

Atributos do Gauge

backgroundColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo do Gauge. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

bulletsVisible: Um atributo booleano determinando que as marcas sejam mostradas em forma de bullets. Retorna verdadeiro (diferente de zero) se os bullets são mostrados ou falso (zero) se não são;

decimalPlaces: Um inteiro sem sinal (de 0 a 65535) determinando o número de casas decimais a serem utilizadas na formatação dos valores do Gauge.

description: Um string contendo uma breve descrição sobre o Gauge;

enabled: Um atributo booleano usado para habilitar o acesso do teclado e mouse ao objeto Gauge. Retorna verdadeiro (diferente de zero) se o acesso está habilitado ou falso (zero) se não está;

frameColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da moldura do Gauge. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

frameVisible: Um atributo booleano determinando que um frame seja mostrado ao longo do percurso da agulha do Gauge. Retorna verdadeiro (diferente de zero) se o frame é mostrado ou falso (zero) se não é;

height: Um inteiro sem sinal (de 0 a 65535) determinando a altura do Gauge, em pixels. É usado juntamente com o atributo *width* para definir o tamanho do Gauge;

hiColorLegend: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da legenda para o intervalo de valores especificados como High. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

hiDiv: Um número real configurando o valor do limite inferior do intervalo High;

hiHiColorLegend: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da legenda para o intervalo de valores especificados como High-High. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

hiHiDiv: Um número real configurando o valor do limite inferior do intervalo High-High;

hiLimit: Um número real no intervalo [0,1], configurando posição da borda superior da legenda. É usado junto com o atributo *lowLimit* para definir a espessura da legenda;

isHiHiLimitVisible: Um atributo booleano determinando que o intervalo de valores HighHigh do Gauge seja mostrado na legenda. Retorna verdadeiro (diferente de zero) se o intervalo HighHigh é mostrado ou falso (zero) se não é;

isHiLimitVisible: Um atributo booleano determinando que o intervalo de valores High do Gauge seja mostrado na legenda. Retorna verdadeiro (diferente de zero) se o intervalo High é mostrado ou falso (zero) se não é;

isLowLimitVisible: Um atributo booleano determinando que o intervalo de valores Low do Gauge seja mostrado na legenda. Retorna verdadeiro (diferente de zero) se o intervalo Low é mostrado ou falso (zero) se não é;

isLowLowLimitVisible: Um atributo booleano determinando que o intervalo de valores LowLow do Gauge seja mostrado na legenda. Retorna verdadeiro (diferente de zero) se o intervalo LowLow é mostrado ou falso (zero) se não é;

legendVisible: Um atributo booleano determinando que uma legenda seja mostrada no Gauge. Retorna verdadeiro (diferente de zero) se as marcas são mostradas ou falso (zero) se não são;

limitVisible: Um atributo booleano determinando que os limites do Gauge sejam mostrados. Retorna verdadeiro (diferente de zero) se os limites são mostrados ou falso (zero) se não são;

lowColorLegend: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da legenda para o intervalo de valores especificados como Low. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

lowDiv: Um número real configurando o valor do limite superior do intervalo Low;

lowLimit: Um número real no intervalo [0,1], configurando a posição da borda inferior da legenda do Gauge. É usado junto com o atributo *hiLimit* para definir a espessura da legenda;

lowLowColorLegend: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da legenda para o intervalo de valores

especificados como Low-Low. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

lowLowDiv: Um número real configurando o valor do limite superior do intervalo Low-Low;

maximum: Um número real configurando o valor máximo para o Gauge. É usado junto com o atributo *minimum* para definir um intervalo de valores para o Gauge;

minimum: Um número real configurando o valor mínimo para o Gauge. É usado junto com o atributo *maximum* para definir um intervalo de valores para o Gauge;

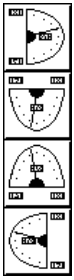
name: Um string contendo o nome do Gauge, usado como identificador do Gauge;

needleColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da agulha do Gauge. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

needleThickness: Um inteiro sem sinal (de 1 a 2) determinando a espessura da agulha do Gauge, que pode ser ponteiro fino (1) ou grosso (2);

normalColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da legenda para o intervalo de valores especificados como normais. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

orientation: Um Inteiro sem sinal (de 0 a 3) determinando a orientação do gráfico do Gauge, conforme segue:



Gauge1.orientation = 0

Gauge1.orientation = 1

Gauge1.orientation = 2

Gauge1.orientation = 3

points: Um inteiro sem sinal (de 0 a 65535) determinando o número de pontos na régua do Gauge. Alguns pontos não podem ser mostrados se forem definidos muitos pontos para um Gauge pequeno.

showTip: Um atributo booleano habilitando o objeto a mostrar uma Tip (dica) quando o mouse está sobre ele. Retorna verdadeiro (diferente de zero) se a Tip está habilitada ou falso (zero) se não está;

startAngle: Um número real configurando o ângulo inicial da agulha do Gauge;

subTicksColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor das sub-marcas do Gauge. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

subTicksVisible: Um atributo booleano determinando que sub-marcas (ticks) sejam mostradas entre as marcas principais. Retorna verdadeiro (diferente de zero) se as sub-marcas são mostradas ou falso (zero) se não são;

textColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto que será mostrado no Gauge. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

thickTicks: Um atributo booleano determinando que as marcas principais do Gauge sejam grossas. Retorna verdadeiro (diferente de zero) se as marcas são grossas ou falso (zero) se não são;

tickColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor das marcas principais do Gauge. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

ticksValues: Um atributo booleano determinando que os valores das marcas (ticks) sejam mostrados no Gauge. Retorna verdadeiro (diferente de zero) se os valores são mostrados ou falso (zero) se não são;

ticksVisible: Um atributo booleano determinando que as marcas principais (ticks) sejam mostradas no Gauge. Retorna verdadeiro (diferente de zero) se as marcas são mostradas ou falso (zero) se não são;

totalNumberOfSubticks: Um inteiro sem sinal (de 0 a 9) determinando o número total de sub-marcas a serem mostradas entre as marcas principais.

valueVisible: Um atributo booleano determinando que o valor do Gauge seja mostrado. Retorna verdadeiro (diferente de zero) se o valor do Gauge é mostrado ou falso (zero) se não é;

visible: Um atributo booleano determinando que o Gauge seja visível. Retorna verdadeiro (diferente de zero) se o Gauge é visível ou falso (zero) se não é;

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura do Gauge, em pixels. É usado juntamente com o atributo *height* para definir o tamanho do Gauge;

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo do Gauge, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo do Gauge a partir da origem da Tela (0,0);

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo do Gauge, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo do Gauge a partir da origem da Tela (0,0);

Atributos do Setpoint

autoSend: (booleano) Quando em 1 (um), automaticamente manda os dados entrados no controle para o tag associado quando este perder o foco.

backgroundColor: Número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo do SetPoint. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255.

checkMaximum: (booleano) Usado para habilitar a verificação do valor do SetPoint com o valor máximo especificado para o mesmo. Retorna verdadeiro (diferente de zero) se a verificação está habilitada ou falso (zero) se não está.

checkMinimum: (booleano) Usado para habilitar a verificação do valor do SetPoint com o valor mínimo especificado para o mesmo. Retorna verdadeiro (diferente de zero) se a verificação está habilitada ou falso (zero) se não está.

dateFormat: (string) Define um formato para mostrar e entrar dados de data/hora em um setpoint. Exemplo: "dd/mm/yy hh:mm:ss" será mostrado como "02/07/97 18:35:22".

Formatos Data/Hora para o atributo dateFormat

FORMATO	SIGNIFICADO
w	Dia da semana (Dom – Sab)
W	Dia da semana (DOM – SAB)
ww	Dia da semana (Domingo – Sábado)
WW	Dia da semana (DOMINGO – SÁBADO)
d	Dia (1 – 31)
dd	Dia (01 – 31)
m	Mês (1 – 12)
mm	Mês (01 – 12)
mmm	Mês (Jan – Dez)
MMM	Mês (JAN – DEZ)
mmmm	Mês (Janeiro – Dezembro)
MMMM	Mês (JANEIRO – DEZEMBRO)
mmmmm	Mês (j – d)
MMMMM	Mês (J – D)
Yy	Ano (00 – 99)
YY	Ano (1970 – 9999)
H	Hora (0 – 23)
Hh	Hora (00 – 23)
M	Minuto (0 – 59) (deve estar acompanhando hs ou seguido de seg.)
mm	Minuto (00 – 59) (deve estar acompanhando hs ou seguido de seg.)
S	Segundo (0 – 59)
Ss	Segundo (00 – 59)
AM/PM	Hora no formato 12 horas, mostra AM (manhã) e PM (tarde).
Am/pm	Hora no formato 12 horas, mostra am e pm.
A/P	Hora no formato 12 horas, mostra A e P.
a/p	Hora no formato 12 horas, mostra a e p.
0	Décimos de segundo (deve ser precedido de segundos)
00	Centésimos de segundo (deve ser precedido de segundos)
000	Milésimos de segundo (deve ser precedido de segundos)
|	Nova linha (CR + LF)

description: (string) Contém uma breve descrição sobre o SetPoint.

enabled: (booleano) Usado para habilitar o acesso do teclado e mouse ao objeto Setpoint. Retorna verdadeiro (diferente de zero) se o acesso está habilitado ou falso (zero) se não está.

height: um inteiro sem signal (de 0 a 65535) determinando a altura do SetPoint, em pixels. É usado juntamente com o atributo *width* para definir o tamanho do SetPoint.

horizontalAlign: (inteiro sem sinal de 0 a 2) Determina o alinhamento horizontal para os dados que serão mostrados no SetPoint, a saber: 0, à esquerda; 1, centro ou 2, à direita.

maximum: Um número real configurando o valor máximo para o SetPoint. É usado junto com o atributo *minimum* para definir um intervalo de valores para o SetPoint;

minimum: Um número real configurando o valor mínimo para o SetPoint. É usado junto com o atributo *maximum* para definir um intervalo de valores para o SetPoint;

multiLine: Um atributo booleano usado para habilitar múltiplas linhas no SetPoint. Retorna verdadeiro (diferente de zero) se o uso de múltiplas linhas está habilitado ou falso (zero) se não está;

name: (string) Contém o nome do SetPoint, usado como identificador do SetPoint.

precision: Um inteiro sem sinal (de 0 a 254) determinando o número de dígitos decimais a serem mostrados no SetPoint. Nunca poderá ser maior que o atributo *size*.

prefix: (string) Contém um prefixo a ser mostrado antes do valor do Tag associado ao SetPoint. Exemplo: `SetPoint.prefix = "Peso"`

refresh: Um atributo booleano determinando que o valor do SetPoint seja atualizado automaticamente se alguma modificação ocorrer no Tag associado. Retorna verdadeiro (diferente de zero) se a atualização automática está habilitada ou falso (zero) se não está.

selectAll: Um atributo booleano determinando que todos os caracteres do SetPoint sejam selecionados quando o objeto receber o foco. Retorna verdadeiro (diferente de zero) se a seleção de todos os caracteres está habilitada ou falso (zero) se não está.

showTip: Um atributo booleano habilitando o objeto a mostrar uma Tip (dica) quando o mouse está sobre ele. Retorna verdadeiro (diferente de zero) se a Tip está habilitada ou falso (zero) se não está.

size: Um inteiro sem sinal (de 0 a 255) determinando o tamanho dos dados a serem mostrados no SetPoint;

suffix: (string) Contém um sufixo a ser mostrado depois do valor do Tag associado ao SetPoint. Exemplo: `SetPoint.suffix = "Kg"`

textColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto que será mostrado no SetPoint. Este valor

também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

type: Um inteiro sem sinal (de 1 a 2) determinando o tipo de dados a ser mostrado no SetPoint, que pode ser um string (1) ou Um número (2).

verticalAlign: (inteiro sem sinal de 0 a 2) Determina o alinhamento vertical para os dados que serão mostrados no SetPoint, a saber: 0, pelo topo; 1, centro e 2, por baixo.

visible: Um atributo booleano determinando que o SetPoint seja visível. Retorna verdadeiro (diferente de zero) se o SetPoint é visível ou falso (zero) se não é;

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura do SetPoint, em pixels. É usado juntamente com o atributo *height* para definir o tamanho do SetPoint;

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo do SetPoint, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo do SetPoint a partir da origem da Tela (0,0);

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo do SetPoint, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo do SetPoint a partir da origem da Tela (0,0);

Atributos do Slider

description: um string contendo uma breve descrição sobre o Slider;

enabled: Um atributo booleano usado para habilitar o acesso do teclado e mouse ao objeto Slider. Retorna verdadeiro (diferente de zero) se o acesso está habilitado ou falso (zero) se não está;

frameColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da moldura do Slider. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

height: um inteiro sem sinal (de 0 a 65535) determinando a altura do Slider, em pixels. É usado juntamente com o atributo *width* para definir o tamanho do Slider;

name: um string contendo o nome do Slider, usado como identificador do Slider;

orientation: um inteiro sem sinal (de 0 a 1) determinando a orientação do Slider. Retorna 0 (zero) se o Slider é horizontal ou 1 (um) se vertical;

precision: um inteiro sem sinal (de 0 a 254) determinando o número de dígitos decimais a serem mostrados nos limites e no valor visível do Slider.

rangeMaximum: Um número real configurando o valor máximo para o Slider. É usado junto com o atributo *rangeMinimum* para definir um intervalo de valores para o Slider;

rangeMinimum: Um número real configurando o valor mínimo para o Slider. É usado junto com o atributo *rangeMaximum* para definir um intervalo de valores para o Slider;

showRange: Um atributo booleano determinando que os limites do Slider sejam mostrados. Retorna verdadeiro (diferente de zero) se os limites são mostrados ou falso (zero) se não são;

showTip: Um atributo booleano habilitando o objeto a mostrar uma Tip (dica) quando o mouse está sobre ele. Retorna verdadeiro (diferente de zero) se a Tip está habilitada ou falso (zero) se não está;

showValue: Um atributo booleano determinando que o valor do Slider seja mostrado. Retorna verdadeiro (diferente de zero) se o valor do Slider é mostrado ou falso (zero) se não é;

step: Um número real a ser incrementado ou decrementado do valor do Slider quando as setas do mesmo são pressionadas;

textColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto que será mostrado no Slider. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

updateOnRelease: Um atributo booleano determinando que o valor do Slider seja atualizado quando o botão do mouse for solto depois de mover a alavanca do Slider. Retorna verdadeiro (diferente de zero) se o Slider é atualizado quando sua alavanca é movida ou falso (zero) se não é.

visible: Um atributo booleano determinando que o Slider seja visível. Retorna verdadeiro (diferente de zero) se o Slider é visível ou falso (zero) se não é;

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura do Slider, em pixels. É usado juntamente com o atributo *height* para definir o tamanho do Slider;

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo do Slider, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo do Slider a partir da origem da Tela (0,0);

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo do Slider, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo do Slider a partir da origem da Tela (0,0);

Atributos do Texto

description: Um string contendo uma breve descrição sobre o Texto;

enabled: Um atributo booleano usado para habilitar o acesso do teclado e mouse ao objeto Texto. Retorna verdadeiro (diferente de zero) se o acesso está habilitado ou falso (zero) se não está;

height: um inteiro sem sinal (de 0 a 65535) determinando a altura do Texto, em pixels. É usado juntamente com o atributo *width* para definir o tamanho do Texto;

name: um string contendo o nome do Texto, usado como identificador do Texto;

showTip: Um atributo booleano habilitando o objeto a mostrar uma Tip (dica) quando o mouse está sobre ele. Retorna verdadeiro (diferente de zero) se a Tip está habilitada ou falso (zero) se não está;

visible: Um atributo booleano determinando que o Texto seja visível. Retorna verdadeiro (diferente de zero) se o Texto é visível ou falso (zero) se não é;

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura do Texto, em pixels. É usado juntamente com o atributo *height* para definir o tamanho do Texto;

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo do Texto, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo do Texto a partir da origem da Tela (0,0);

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo do Texto, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo do Texto a partir da origem da Tela (0,0);

Atributos da Tendência

description: Um string contendo uma breve descrição sobre a Tendência;

enabled: Um atributo booleano usado para habilitar o acesso do teclado e mouse ao objeto Tendência. Retorna verdadeiro (diferente de zero) se o acesso está habilitado ou falso (zero) se não está;

enableDataRec: Um atributo booleano usado para habilitar a Tendência a coletar novos valores sempre que os Tags associados a ela mudarem. Retorna verdadeiro (diferente de zero) se os valores são coletados ou falso (zero) se não são;

frozen: Um atributo booleano usado para habilitar/desabilitar o scroll automatico do Trend. Retorna verdadeiro (diferente de zero) se o Trend está congelado ou falso (zero) se o scroll está habilitado (default);

height: Um inteiro sem sinal (de 0 a 65535) determinando a altura da Tendência, em pixels. É usado juntamente com o atributo *width* para definir o tamanho da Tendência;

isXY: Um atributo booleano. Somente para uso do Elipse SCADA. Não são permitidas modificações por parte de usuários.

name: Um string contendo o nome do Tendência, usado como identificador da Tendência;

orientation: Um inteiro sem sinal (de 0 a 3) determinando a orientação do gráfico da Tendência, conforme a tabela a seguir:

VALOR	ORIENTAÇÃO
0	Horizontal, da direita para esquerda
1	Horizontal, da esquerda para direita
2	Vertical, de cima para baixo
3	Vertical, de baixo para cima

showTip: Um atributo booleano habilitando o objeto a mostrar uma Tip (dica) quando o mouse está sobre ele. Retorna verdadeiro (diferente de zero) se a Tip está habilitada ou falso (zero) se não está;

timeSpan: Um número real determinando o tempo a ser visualizado na Tendência em segundos.

visible: Um atributo booleano determinando se a tendência está visível. Retorna verdadeiro (diferente de zero) se a tendência está visível ou falso (zero) se não está.

width: Um inteiro sem sinal (de 0 a 65535) determinando a largura da Tendência, em pixels. É usado juntamente com o atributo *height* para definir o tamanho da Tendência;

x: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *X* para o canto superior esquerdo da Tendência, em pixels. É usado juntamente com o atributo *Y* para definir a posição do canto superior esquerdo da Tendência a partir da origem da Tela (0,0);

y: Um inteiro com sinal (de -32768 a 32767) determinando a coordenada *Y* para o canto superior esquerdo da Tendência, em pixels. É usado juntamente com o atributo *X* para definir a posição do canto superior esquerdo da Tendência a partir da origem da Tela (0,0);

Atributos da Moldura

borderColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da borda da Moldura. Este valor também pode ser

especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

borderEnabled: Um atributo booleano determinando que a borda da Moldura seja mostrada. Retorna verdadeiro (diferente de zero) se a borda é mostrada ou falso (zero) se não é;

borderThickness: Um inteiro sem sinal (de 0 a 255) determinando a espessura da borda da Moldura;

color: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da Moldura. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

description: Um string contendo uma breve descrição sobre a Moldura;

enable3D: Um atributo booleano usado para habilitar um efeito 3D na Moldura. Retorna verdadeiro (diferente de zero) se o efeito 3D está habilitado ou falso (zero) se não está.

frame: Um atributo booleano determinando que a Moldura seja mostrada. Retorna verdadeiro (diferente de zero) se a Moldura é mostrada ou falso (zero) se não é.

name: Um string contendo o nome da Moldura, usado como identificador da Moldura;

separator: Um atributo booleano determinando que uma linha de separação do título seja mostrada. Retorna verdadeiro (diferente de zero) se a linha é mostrada ou falso (zero) se não é;

set3DInset: Um atributo booleano determinando que o efeito 3D seja na parte de dentro ou de fora da Moldura. Retorna verdadeiro (diferente de zero) se o efeito 3D é mostrado por dentro ou falso (zero) se é por fora;

thickness3D: Um inteiro sem sinal (de 0 a 255) determinando a espessura do efeito 3D da Moldura.

title: Um string contendo o título da Moldura;

titleColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor do texto do título. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

titleEnabled: Um atributo booleano determinando que o título da Moldura seja mostrado. Retorna verdadeiro (diferente de zero) se o título é mostrado ou falso (zero) se não é;

19.6.6. Atributos da Plotagem

autoRangeX: Um atributo booleano determinando que os limites do eixo X do gráfico sejam automaticamente calculados. Retorna verdadeiro (diferente de zero) se o cálculo automático para os limites do eixo X está habilitado ou falso (zero) se não está;

autoRangeY: Um atributo booleano determinando que os limites do eixo Y do gráfico sejam automaticamente calculados. Retorna verdadeiro (diferente de zero) se o cálculo automático para os limites do eixo Y está habilitado ou falso (zero) se não está;

backgroundColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo da Plotagem. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

backRulerColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor de fundo da Plotagem. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

bottomLimit: Um número real configurando o limite inferior a ser mostrado na régua do eixo Y. É usado junto com o atributo *topLimit* para definir um intervalo de valores;

bottomRuler: Um atributo booleano determinando que uma régua de valores seja mostrada na parte inferior da Plotagem. Retorna verdadeiro (diferente de zero) se a régua inferior é mostrada ou falso (zero) se não é;

bShowMsec: Um atributo booleano determinando que os milisegundos sejam sejam mostrados na Plotagem. Retorna verdadeiro (diferente de zero) se os milisegundos são mostrados ou falso (zero) se não são;

description: Um string contendo uma breve descrição sobre a Plotagem;

enableXUnit: Um atributo booleano determinando que a unidade do eixo X seja mostrada na Plotagem. Retorna verdadeiro (diferente de zero) se a unidade do eixo X é mostrada ou falso (zero) se não é;

enableYUnit: Um atributo booleano determinando que a unidade do eixo Y seja mostrada na Plotagem. Retorna verdadeiro (diferente de zero) se a unidade do eixo Y é mostrada ou falso (zero) se não é;

gridColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da grade da Plotagem. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

gridVisible: Um atributo booleano determinando que a grade da Plotagem seja mostrada. Retorna verdadeiro (diferente de zero) se a grade é mostrada ou falso (zero) se não é;

gridX: Um inteiro sem sinal (de 0 a 65535) determinando o número de divisões no eixo X;

gridY: Um inteiro sem sinal (de 0 a 65535) determinando o número de divisões no eixo Y;

leftLimit: Um número real configurando o limite esquerdo a ser mostrado na régua do eixo X. É usado junto com o atributo *rightLimit* para definir um intervalo de valores;

leftRuler: Um atributo booleano determinando que uma régua de valores seja mostrada à esquerda da Plotagem. Retorna verdadeiro (diferente de zero) se a régua à esquerda é mostrada ou falso (zero) se não é;

name: Um string contendo o nome da Plotagem, usado como identificador da Plotagem;

rightLimit: Um número real configurando o limite direito a ser mostrado na régua do eixo X. É usado junto com o atributo *leftLimit* para definir um intervalo de valores;

rightRuler: Um atributo booleano determinando que uma régua de valores seja mostrada à direita da Plotagem. Retorna verdadeiro (diferente de zero) se a régua à direita é mostrada ou falso (zero) se não é;

rulerColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da régua da Plotagem. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

topLimit: Um número real configurando o limite superior a ser mostrado na régua do eixo Y. É usado junto com o atributo *bottomLimit* para definir um intervalo de valores;

topRuler: Um atributo booleano determinando que uma régua de valores seja mostrada na parte superior da Plotagem. Retorna verdadeiro (diferente de zero) se a régua superior é mostrada ou falso (zero) se não é;

xUnit: Um string contendo o nome da unidade a ser mostrada no eixo X;

yUnit CE: Um string contendo o nome da unidade a ser mostrada no eixo Y;

Atributos do Cursor CE

barCalc: Determina como o valor da barra é calculado, a saber: 0 = Média; 1 = Máxima; 2 = Mínima; 3 = Por Amostra.

barWidth: Define a largura da barra no eixo X quando for uma barra por amostra.

bottomWorld: Determina o ponto inferior dos dados do gráfico.

description: Possui a descrição do objeto.

leftWorld: Determina o ponto esquerdo dos dados do gráfico.

mode: Configura o modo de operação do cursor, a saber: 0 = Nenhum; 1 = Modo de Seleção; 2 = Modo de Zoom; 3 = Modo de Rolagem.

name: Contém o nome que identifica o objeto.

penColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da pena do cursor. Este valor também pode ser especificado pela função RGB (ver função RGB).

penStyle: Especifica o estilo da pena do cursor, a saber: 0 = Sólido; 1 = Tracejado; 2 = Pontilhado; 3 = Traço-Ponto; 4 = Traço-Ponto-Ponto.

penType: Especifica o tipo da pena do cursor a ser desenhado, a saber: 1 = Linha; 2 = Barra; 3 = Área.

penVisible: Atributo booleano que determina se a pena está sendo mostrada (diferente de zero) ou não (zero).

penWidth: Um inteiro sem sinal (de 0 a 65.535) determinando a largura da pena do cursor em pixels.

rangeType: Determina the seleção do cursor:

VALOR	ESTILO
0	Sem seleção.
1	A seleção é um ponto.
2	A seleção é um retângulo.

rangeX1: Determina o valor x do primeiro ponto da seleção (atributo rangeType deve ser 2).

rangeX2: Determina o valor x do segundo ponto da seleção (atributo rangeType deve ser 2).

rangeY1: Determina o valor y do primeiro ponto da seleção (atributo rangeType deve ser 2).

rangeY2: Determina o valor y do segundo ponto da seleção (atributo rangeType deve ser 2).

rightWorld: Determina o ponto direito dos dados do gráfico (atributo rangeType deve ser 2).

selLabel: Nome da pena selecionada.

selRecord: Determina o número do registro selecionado: Se o valor for -1, não há seleção.

selX: Determina o valor x do ponto selecionado (atributo rangeType deve ser 1).

selY: Determina o valor y do ponto selecionado (atributo rangeType deve ser 1).

topWorld: Determina o ponto superior dos dados do gráfico.

vLock: Atributo booleano que determina se os limites verticais do gráfico serão congelados quando em modo zoom ou rolagem (diferente de zero) ou não (zero).

Atributos das Penas

description: Um string contendo uma breve descrição sobre a Pena;

name: Um string contendo o nome da Pena, usado como identificador da Pena;

penColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da Pena. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

penStyle: Um inteiro sem sinal (de 0 a 4) determinando o estilo da Pena, conforme a tabela a seguir:

VALOR	ESTILO
0	Sólido
1	Tracejado
2	Pontilhado
3	Traço-Ponto
	Traço-Ponto-Ponto

penVisible: Atributo booleano que determina se a pena está sendo mostrada (diferente de zero) ou não (zero).

penWidth: Um inteiro sem sinal (de 0 a 65535) determinando a largura da Pena em pixels;

Atributos da Marca

description: Um string contendo uma breve descrição sobre a Marca;

name: Um string contendo o nome da Marca, usado como identificador da Marca;

markType: Um inteiro sem sinal (de 0 a 2) determinando o tipo da Marca, conforme a tabela a seguir:

VALOR	TIPO
0	Ponto
1	Vertical
2	Horizontal

penColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor da Marca. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

penStyle: Um inteiro sem sinal (de 0 a 4) determinando o estilo da Marca, conforme a tabela a seguir:

VALOR	ESTILO
0	Sólido
1	Tracejado
2	Pontilhado
3	Traço-Ponto
	Traço-Ponto-Ponto

penVisible: Um atributo booleano determinando que a Marca seja mostrada. Retorna verdadeiro (diferente de zero) se a Marca é mostrada ou falso (zero) se não é;

penWidth: Um inteiro sem sinal (from 0 to 65535) determinando a largura da Marca em pixels;

x: Um número real determinando a coordenada *X* da Marca em pixels. É usado juntamente com o atributo *Y* para definir a posição da Marca a partir da origem do gráfico (0,0);

y: Um número real determinando a coordenada *Y* da Marca em pixels. É usado juntamente com o atributo *X* para definir a posição da Marca a partir da origem do gráfico (0,0);

19.6.7. Atributos de Alarmes

description: Um string contendo uma breve descrição sobre o Alarme;

displayMessageBox : Um atributo booleano determinando que as mensagens de alarme sejam mostradas quando um alarme ocorrer. Retorna verdadeiro (diferente de zero) se as mensagens estão habilitadas ou falso (zero) se não estão;

filename: Um string contendo o nome do arquivo de log de Alarmes;

lastAlarmPri: Um inteiro sem sinal (de 0 a 65535) informando a prioridade do último alarme ativado. É um atributo somente de leitura.

logAlarms: Um atributo booleano determinando que os alarmes sejam gravados no arquivo de log de Alarmes. Retorna verdadeiro (diferente de zero) se a gravação está habilitada ou falso (zero) se não está;

name: Um string contendo o nome do Alarme, usado como identificador do Alarme;

nRecords: Um inteiro sem signal (de 0 a 65535) determinando o número de alarmes mais recentes a serem gravados no arquivo de log de alarmes.

soundType: Um inteiro sem signal (de 0 a 65535) determinando o tipo de som a ser tocado quando um alarme ocorrer. Está disponível somente se o atributo *warningSound* está habilitado.

warningSound: Um atributo booleano usado para habilitar um som de alerta quando um alarme ocorrer. Retorna verdadeiro (diferente de zero) se o som de alerta está habilitado ou falso (zero) se não está;

19.6.8. Atributos das Receitas

description: Um string contendo uma breve descrição sobre o Receita;

name: Um string contendo o nome da Receita, usado como identificador da Receita;

19.6.9. Atributos dos Históricos

batch: Um atributo booleano usado para habilitar um processo de batelada. Retorna verdadeiro (diferente de zero) se a batelada está habilitada ou falso (zero) se não está;

description: Um string contendo uma breve descrição sobre o Histórico;

enabled: Um atributo booleano usado para habilitar a gravação dos dados históricos no arquivo de Histórico. Retorna verdadeiro (diferente de zero) se a gravação está habilitada ou falso (zero) se não está;

filename: Um string contendo o nome do arquivo de Histórico;

isBatchRunning: Um atributo booleano (somente leitura) determinando se a batelada está sendo executada ou não. Retorna verdadeiro (diferente de zero) se a batelada está executando ou falso (zero) se não está;

name: Um string contendo o nome do Histórico, usado como identificador do Histórico;

networkSupport: Um atributo booleano usado para habilitar o suporte a rede para o Histórico, isto é, permite que o Histórico seja acessado (somente para leitura)

por outras aplicações Elipse na Rede, através de um Browser ou Relatório do tipo Análise Histórica. Retorna verdadeiro (diferente de zero) se o suporte a rede está habilitado ou falso (zero) se não está;

size: Um inteiro sem signal (de 0 a 65535) determinando o tamanho do arquivo de Histórico em número de registros;

writeTime: Um inteiro longo sem sinal (de 0 a 2^{32}) determinando a frequência com que os dados serão escritos no arquivo de Histórico;

19.6.10. Atributos da Análise Histórica

advised: Este atributo é somente para leitura e não faz sentido para a Análise Histórica;

description: Um string contendo uma breve descrição sobre a Análise Histórica;

maximizeButton: Um atributo booleano determinando que o botão maximizar esteja habilitado na janela da Análise Histórica. Retorna verdadeiro (diferente de zero) se o botão está habilitado ou falso (zero) se não está;

minimizeButton: Um atributo booleano determinando que o botão minimizar esteja habilitado na janela da Análise Histórica. Retorna verdadeiro (diferente de zero) se o botão está habilitado ou falso (zero) se não está;

modal: Um atributo booleano determinando que a caixa de diálogo da Análise Histórica seja modal (não permite acessar nenhuma outra janela antes de ser fechada) ou modeless (não necessita ser fechada para se acessar outras telas). Retorna verdadeiro (diferente de zero) se a página é modal ou falso (zero) se não é;

name: Um string contendo o nome da Análise Histórica, usado como identificador da Análise;

pageStyle: Um inteiro sem sinal (de 0 a 2) determinando quais as páginas da Análise Histórica que devem ser mostradas, conforme a tabela a seguir:

VALOR	PÁGINAS
0	Mostra todas as páginas
1	Mostra somente a página da Análise (Gráfico)
2	Mostra somente as páginas selecionadas

resizeable: Um atributo booleano determinando que a janela da Análise Histórica possa ser redimensionada em tempo de execução. Retorna verdadeiro (diferente de zero) se a janela pode ser redimensionada ou falso (zero) se não pode;

showGraphPage: Um atributo booleano determinando que a página *Gráfico* seja mostrada na janela da Análise Histórica. Retorna verdadeiro (diferente de zero) se a página é mostrada ou falso (zero) se não é;

showPenColorsPage: Um atributo booleano determinando que a página *Cores das Penas* seja mostrada na janela da Análise Histórica. Retorna verdadeiro (diferente de zero) se a página é mostrada ou falso (zero) se não é;

showPensPage: Um atributo booleano determinando que a página *Penas* seja mostrada na janela da Análise Histórica. Retorna verdadeiro (diferente de zero) se a página é mostrada ou falso (zero) se não é;

showPrintSetupPage: Um atributo booleano determinando que a página *Impressão* seja mostrada na janela da Análise Histórica. Retorna verdadeiro (diferente de zero) se a página é mostrada ou falso (zero) se não é;

showQueryPage: Um atributo booleano determinando que a página *Consulta* seja mostrada na janela da Análise Histórica. Retorna verdadeiro (diferente de zero) se a página é mostrada ou falso (zero) se não é;

showSettingsPage: Um atributo booleano determinando que a página *Configurações* seja mostrada na janela da Análise Histórica. Retorna verdadeiro (diferente de zero) se a página é mostrada ou falso (zero) se não é;

19.6.11. Atributos da Consulta

batchField: texto (string) que define qual o campo de cabeçalho a ser utilizado em uma consulta em um histórico por bateladas. Este valor é equivalente ao campo Field na página Settings nas propriedades do browser. Após configurada esta propriedade, deve-se chamar a função UpdateQuery() para visualizar os novos dados no browser.

criteria: um string contendo o nome da batelada que se deseja procurar.

description: um string contendo uma breve descrição sobre a Consulta.

filename: um string contendo o nome do arquivo sobre o qual vai ser aplicada a Consulta.

finalDay: um inteiro sem sinal (de 1 a 31) determinando o dia final quando o tipo de Consulta é por Intervalo de Tempo;

finalHour: um inteiro sem sinal (de 0 a 23) determinando a hora final quando o tipo de Consulta é por Intervalo de Tempo;

finalMinute: um inteiro sem sinal (de 0 a 59) determinando o minuto final quando o tipo de Consulta é por Intervalo de Tempo;

finalMonth: um inteiro sem sinal (de 1 a 12) determinando o mês final quando o tipo de Consulta é por Intervalo de Tempo;

finalSecond: um inteiro sem sinal (de 0 a 59) determinando o segundo final quando o tipo de Consulta é por Intervalo de Tempo;

finalYear: um inteiro sem sinal (de 1970 a 2039) determinando o ano final quando o tipo de Consulta é por Intervalo de Tempo;

name: um string contendo o nome da Consulta, usado como identificador da Consulta.

queryType: um inteiro sem sinal (de 0 a 2) determinando o tipo de Consulta conforme a tabela a seguir:

VALOR	TIPO DA CONSULTA
0	Intervalo de Tempo
1	Dados mais Recentes
2	Sem Critério

queryUnit: É usado juntamente com o atributo queryValue, quando este é igual a 1 (dados mais recentes ou *newest data*). É um inteiro sem sinal (de 0 a 5) determinando uma unidade de tempo para a busca dos dados mais recentes, conforme a tabela a seguir:

VALOR	FORMATO
0	Dia
1	Mês
2	Ano
3	Horas
4	Minutos
5	Segundos

queryValue: Somente disponível se o atributo queryType é 1 (Dados mais Recentes ou *newest data*). É um inteiro sem sinal determinando um período de tempo para a busca dos dados mais recentes, especificado no campo Last na página de Consulta. É usado juntamente com o atributo queryUnit.

startDay: um inteiro sem sinal (de 1 a 31) determinando o dia inicial quando o tipo de consulta é por Intervalo de Tempo;

startHour: um inteiro sem sinal (de 0 a 23) determinando a hora inicial quando o tipo de Consulta é por Intervalo de Tempo;

startMinute: um inteiro sem sinal (de 0 a 59) determinando o minuto inicial quando o tipo de Consulta é por Intervalo de Tempo;

startMonth: um inteiro sem sinal (de 1 a 12) determinando o mês inicial quando o tipo de consulta é por Intervalo de Tempo;

startSecond: um inteiro sem sinal (de 0 a 59) determinando o segundo inicial quando o tipo de Consulta é por Intervalo de Tempo;

startYear: um inteiro sem sinal (de 1970 a 2039) determinando o ano inicial quando o tipo de Consulta é por Intervalo de Tempo;

19.6.12. Atributos do CEP (SPC)

avg: Um número real determinando a média;

cp: Somente para uso do Elipse SCADA.;

cpk: Somente para uso do Elipse SCADA.

cr: Somente para uso do Elipse SCADA;

description: um string contendo uma breve descrição sobre o SPC;

dispersionChartType: um inteiro sem sinal (de 0 a 1) determinando o tipo do gráfico como Sigmas (0) ou Ranges (1);

dispLCL: Um número real configurando o limite inferior do gráfico de Dispersão. Ele é usado junto com o atributo *dispUCL* para definir um intervalo de variação;

dispUCL: Um número real configurando o limite superior do gráfico de Dispersão. Ele é usado junto com o atributo *dispLCL* para definir um intervalo de variação;

k: Somente para uso do Elipse SCADA;

lcl: Um número real configurando o limite inferior de Controle. Ele é usado junto com o atributo *ucl* para definir um intervalo de variação;

lel: Um número real configurando o limite inferior de Engenharia. Ele é usado junto com o atributo *uel* para definir um intervalo de variação;

maxValue: (somente leitura) Um número real indicando o valor máximo entre as amostras calculadas.

minValue: (somente leitura) Um número real indicando o valor mínimo entre as amostras calculadas.

name: Um string contendo o nome do SPC, usado como identificador do SPC;

nRecords: (somente leitura) um inteiro sem sinal (de 1 a 65535) indicando o número total de registros analisados.

pointsPerGroup: um inteiro sem sinal (de 0 a 65535) determinando o número de pontos por amostra;

stdDev: Um número real determinando o Desvio Padrão;

ucl: Um número real configurando o limite superior de Controle. Ele é usado junto com o atributo *lcl* para definir um intervalo de variação;

uel: Um número real configurando o limite superior de Engenharia. Ele é usado junto com o atributo *lel* para definir um intervalo de variação;

19.6.13. Atributos da Batelada

description: Um string contendo uma breve descrição sobre a Batelada;

fieldName: Um string contendo o nome do arquivo da Batelada;

fieldType: Um inteiro sem sinal determinando o tipo de Batelada (somente leitura).

name: Um string contendo o nome da Batelada, usado como identificador da Batelada;

19.6.14. Atributos dos Relatórios

bottomMargin: um inteiro sem sinal (de 0 a 65535) determinando a margem inferior do Relatório, em milímetros;

description: um string contendo uma breve descrição sobre o Relatório;

leftMargin: um inteiro sem sinal (de 0 a 65535) determinando a margem esquerda do Relatório, em milímetros;

name: um string contendo o nome do Relatório, usado como identificador do Relatório;

printHeader: Um atributo booleano usado para habilitar a impressão de uma página contendo informações do cabeçalho da batelada. Retorna verdadeiro (diferente de zero) se a impressão do cabeçalho está habilitada ou falso (zero) se não está;

printTitle: Um atributo booleano usado para habilitar a impressão de um título no Relatório. Retorna verdadeiro (diferente de zero) se a impressão do título está habilitada ou falso (zero) se não está;

rightMargin: um inteiro sem sinal (de 0 a 65535) determinando a margem direita do Relatório, em milímetros;

rowColor: Um número hexadecimal (de 0h a 1000000h) ou decimal (de 0 a 16777215) definindo a cor das linhas do Relatório. Este valor também pode ser especificado pela função RGB(r,g,b) onde cada componente (r,g,b) pode variar de 0 a 255;

topMargin: um inteiro sem sinal (de 0 a 65535) determinando a margem superior do Relatório, em milímetros;

19.6.15. Atributos dos Drivers

Driver PLC

abortOnError: Um atributo booleano usado para habilitar a interrupção da comunicação se algum erro ocorrer. Retorna verdadeiro (diferente de zero) se a comunicação está habilitada ou falso (zero) se não está;

busy: Um atributo booleano usado para informar se uma comunicação está ocorrendo. Retorna verdadeiro (diferente de zero) se uma comunicação está ocorrendo ou falso (zero) se nenhuma comunicação está sendo feita;

description: Um string contendo uma breve descrição sobre o Driver;

driverName: Um string contendo o nome do Driver conforme o fabricante, este nome será usado para a comunicação com o PLC;

driverPath: Um string contendo a localização (path) do Driver;

enableRead: Um atributo booleano usado para habilitar a leitura de valores e blocos do driver. Retorna verdadeiro (diferente de zero) se a leitura está habilitada ou falso (zero) se não está;

enableRetry: Um atributo booleano usado para habilitar a uma nova tentativa de comunicação se algum erro ocorrer. Retorna verdadeiro (diferente de zero) se a retentativa está habilitada ou falso (zero) se não está;

enableWrite: Um atributo booleano usado para habilitar a escrita de valores e blocos do driver. Retorna verdadeiro (diferente de zero) se a escrita está habilitada ou falso (zero) se não está;

hideMouse: Um atributo booleano determinando que o mouse seja escondido durante a comunicação. Retorna verdadeiro (diferente de zero) se o desaparecimento do mouse está habilitado ou falso (zero) se não está;

isLoaded: Um atributo booleano usado para informar se o driver está carregado em memória ou não. Retorna verdadeiro (diferente de zero) se o driver está carregado ou falso (zero) se não está;

isStarted: Um atributo booleano usado para informar se uma comunicação foi iniciada ou não. Retorna verdadeiro (diferente de zero) se uma comunicação foi iniciada ou falso (zero) se não foi;

name: Um string contendo o nome do Driver, usado como identificador do Driver;

p1: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro P1 conforme a documentação do Driver;

p2: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro P2 conforme a documentação do Driver;

p3: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro P3 conforme a documentação do Driver;

p4: Um inteiro com sinal (de -32768 a 32767) configurando o parâmetro P4 conforme a documentação do Driver;

retryCount: Um inteiro sem sinal (de 0 a 65535) determinando o número de tentativas, se o atributo *enableRetry* está habilitado;

Drivers de Rede

bytesReceived: Um inteiro sem sinal (de 0 a 65535) determinando o número de bytes recebidos pelo driver (somente leitura);

bytesSent: Um inteiro sem sinal (de 0 a 65535) determinando o número de bytes enviados pelo driver aos clientes (somente leitura);

description: Um string contendo uma breve descrição sobre o Driver;

name: Um string contendo o nome do Driver, usado como identificador do Driver;

nClients: Um inteiro sem sinal (de 0 a 65535) determinando o número de clientes atualmente conectados ao driver (somente leitura);

rx: Um atributo booleano (somente leitura) usado para informar se o driver está recebendo dados ou não. Retorna verdadeiro (diferente de zero) se o driver está recebendo dados ou falso (zero) se não está;

tx: Um atributo booleano (somente leitura) usado para informar se o driver está enviando dados ou não. Retorna verdadeiro (diferente de zero) se o driver está enviando dados ou falso (zero) se não está;

19.6.16. Atributos do Banco de Dados

description: Um string contendo uma breve descrição sobre o Banco de Dados.

directory: Determina o diretório do Database.

filter: Um string ou valor usado como filtro quando é feita uma consulta no Banco de Dados.

lastErrorStr: Um string contendo a descrição do último erro ocorrido em operações com o Banco de Dados.

name: Um string contendo o nome da database, usado como identificador do Banco de Dados.

originalTableName: Especifica as informações referentes a tabela do database.

recno: Um inteiro sem sinal (de 0 a 65535) determinando o número do registro corrente.

sort: Um string contendo o nome do campo no Banco de Dados usado para ordená-lo.

Atributos de campos de Bancos de Dados

advised: (booleano, somente leitura) Indica que o campo está em *advise*, ou seja, ele é atualizado toda vez que um objeto associado for referenciado.

originalFieldName: Determina o nome do campo no database.

type: Determina o tipo do campo.

19.6.17. Atributos da Lista de Usuários

description: Um string contendo uma breve descrição sobre o Usuário.

name: Um string contendo o nome do Usuário, usado como identificador do Usuário.

19.6.18. Atributos da Aplicação Remota

description: Um string contendo uma breve descrição sobre a Aplicação Remota;

isConnected: Atributo booleano usado para informar se a Aplicação Remota está conectada ou não. Retorna verdadeiro (diferente de zero) se está conectada ou falso (zero) se não está;

isConnecting: Atributo booleano usado para informar se a Aplicação Remota está em processo de conexão ou não. Retorna verdadeiro (diferente de zero) se está conectando ou falso (zero) se não está;

name: Um string contendo o nome da Aplicação Remota, usado como identificador da Aplicação Remota;

Atributos do Arquivo Remoto

bytesDone: Um inteiro sem sinal (de 0 a 2^{32}) que determina o número de bytes já comparados / transferidos (somente leitura).

description: Um string contendo uma breve descrição sobre o Arquivo Remoto;

lastError: Um inteiro sem sinal (de 0 a 5) que determina o código do erro ocorrido na última transferência de arquivo (somente leitura), conforme a tabela a seguir:

VALOR	EVENTO
0	Transferência OK
1	Não conseguiu abrir ou criar arquivo do cliente
2	Arquivo do servidor não foi encontrado.
3	Dados inválidos recebidos do servidor durante conexão
4	Bloco fora de sequência recebido do servidor
5	Erro de posicionamento ou de escrita no arquivo do cliente

name: Um string contendo o nome identificador do Arquivo Remoto;

result: Um inteiro com sinal (de -1 a 1) que determina o resultado da última transferência de arquivo (somente leitura), conforme a tabela a seguir:

VALOR	EVENTO
-1	Nenhuma transferência foi efetuada
0	Última transferência falhou
1	Última transferência teve sucesso.

state: Um inteiro sem sinal (de 0 a 3) indicando o estado da transferência de arquivo atual (somente leitura), conforme a tabela a seguir:

VALOR	EVENTO
0	Nenhuma transferência em andamento
1	Conectando
2	Comparando arquivos
3	Transferindo

totalLength: Um inteiro sem sinal (de 0 a 2^{32}) determinando o número de bytes a comparar / transferir (somente leitura).

Atributos do Tag Remoto

alarmDelay: (inteiro sem sinal) É o tempo mínimo (em milisegundos) que o tag deve permanecer em uma mesma condição de alarme antes deste tornar-se ativo. Se alarmDelay for modificado em tempo de execução, o contador de tempo é reiniciado. Este atributo só é válido se o atributo delayedAlarm for True (ver atributo delayedAlarm).

alternateName: Nome alternativo para este tag no log de alarmes.

alwaysInAdvise: Mantém o valor do tag sempre atualizado.

currentAlarm: Determina o estado do tag. As opções disponíveis são as seguintes:

Opções disponíveis

ÍNDICE	DESCRIÇÃO
0	Indica que não há alarmes ativos.
1	Alarme de nível LOLO
2	Alarme de nível LOW
3	Alarme de nível High
4	Alarme de nível HIHI

delayedAlarm: (booleano) Se TRUE, o tag deve permanecer um tempo mínimo (especificado em alarmDelay) em uma mesma condição de alarme (Hi, HiHi, Lo, LoLo) para ativar (ou gerar) o alarme correspondente.

enableAlternateName: Habilita o uso de outro nome para o tag.

enableReturnLog: Habilita/desabilita o log de mensagens de retorno de alarme.

initialValue: (string) Contém um valor inicial para o tag e permite que este valor seja modificado em tempo de execução.

remoteTag: Caminho do tag remoto.

remoteTagPath: (string) Indica o caminho do tag remoto dentro de uma aplicação remota.

returnMessage: Determina a mensagem de retorno de alarme do tag.

timeStamp: Data/hora em que o valor do tag foi gerado.

Atributos do Video Remoto

name: Nome do objeto.

remoteCameraId: Determina o ID da câmera na aplicação remota.

Atributos da Câmera do Video Remoto

bitsPerPixel: Determina o número de bits necessários para armazenar cada pixel da imagem.

brighness: Determina o ajuste do brilho da imagem.

cameraId: Identificador da câmera na aplicação (de 0 a 65535).

color: Ajusta a saturação de cores da imagem.

contrast: Ajusta o contraste da imagem.

curentFrameRate: Taxa de captura atual.

currentPos: Determina a posição atual (em segundos) na sequência do video.

firstPos: Determina a posição inicial (em segundos) da sequência de video.

frameSize: Determina o tamanho (em bytes) do último quadro capturado.

gamma: Ajuste de gama (de 0 a 1000).

 **imageHeight**: Determina a altura da imagem, em pixels.

 **imageWidth**: Determina a largura da imagem, em pixels.

isPaused: Indica se o video está pausado em um quadro.

isPlaying: Indica se o video está sendo reproduzido.

isStopped: Indica se a entrada de video está parada.

lastPos: Indica a posição final da sequência de video (em segundos).

name: Determina o nome da câmera.

sharpness: Ajusta o foco da imagem.

tint: Ajusta a coloração da imagem.

velocity: Indica a velocidade em que o video está sendo tocado. As opções disponíveis são as seguintes:

Opções disponíveis

ÍNDICE	DESCRIÇÃO
1	Velocidade normal à frente
-1	Velocidade normal para trás

19.6.19. Atributos do Watcher

Atributos Comuns

description: Um string contendo uma breve descrição sobre o Watcher.

name: Um string contendo o nome do Watcher, usado como identificador.

Atributos do Objeto AVI Player

duration: Determina o número total de frames do AVI.

fileName: Determina o nome do arquivo AVI.

position: Determina a posição do frame no arquivo AVI.

reverse: Retrocede um frame no arquivo AVI.

speed: Determina a velocidade dos frames do AVI.

timeFormat: Determina o formato de tempo do frame: 0 – duração e posição são especificados no número de frames e 1 – duração e posição são especificados por milisegundos.

volume: Determina o volume do som em 0,1% unidades do nível do som original (1000 = volume original, 500 = metade-volume etc.)

Atributos do Objeto AVI Recorder

description: Descrição do objeto.

fileName: Nome do arquivo AVI.

frameRate: Indica o número de quadros a gravar por segundo.

isRecording: Inicia/pára a gravação no arquivo AVI.

name: Nome do objeto.

Atributos da Placa S611

board: Indica o número da placa (de 0 a 7).

color: Determina o formato de cor da saída do video.

customHeight: Determina altura da imagem, empixels (para escala avançada).

- **custonWidth**: Determina a largura da imagem, em pixels (para escala avançada).
- **format**: Determina o formato da imagem (PAL-M, NTSC, etc).
- **frameRate**: Determina o número de quadros a capturar por segundo.
- **input**: Determina a entrada de video.
- **interlaced**: Determina o entrelaçamento de imagens. As opções disponíveis são as seguintes:

Opções disponíveis

ÍNDICE	DESCRIÇÃO
0	Imagem não entrelaçada
1	Imagem entrelaçada

- **scale**: Define o tamanho da imagem.

Atributos da Placa S613

- **board**: Determina o número da placa (de 0 a 7).
- **cfactor**: Determina o fator de compressão. As configurações são as seguintes:

Opções disponíveis

ÍNDICE	DESCRIÇÃO
0	Menor compressão
Até 65535	Maior compressão:

- **color**: Determina o formato de cor da saída do video.
- **compress**: Determina a compressão da imagem. As configurações são as seguintes:

Opções disponíveis

ÍNDICE	DESCRIÇÃO
0	Sem compressão
1	MJPEG

- **customHeight**: Determina a altura da imagem, em pixels (para escala avançada).
- **customWidth**: Determina a largura da imagem, em pixels (para escala avançada).

description: Descrição do objeto.

format: Determina o formato da imagem (PAL-M, NTSC, etc).

frameRate: Determina o número de quadros a capturar, por segundo.

input: Determina a entrada de video.

interlaced: Determina o entrelaçamento de imagem. Se o valor for 0, a imagem não é entrelaçada e se for 1, a imagem é entrelaçada.

name: Nome do objeto.

scale: Define o tamanho da imagem.

Atributos do Objeto VFW

deviceIndex: Determina o índice do dispositivo na listagem de dispositivo do sistema.

deviceName: Determina o nome do dispositivo com suporte a Video for Windows.

deviceVersion: Determina a versão do dispositivo com suporte a Video for Windows.

frameRate: Determina o número de quadros a capturar por segundo.

Atributos do Objeto XPresPlus Camera

fileName: Nome do arquivo de video.

format: Formato da imagem (PAL-M, NTSC, etc).

name: Nome do objeto.

nFrame: Determina o número do quadro atual.

numBoard: Determina o número da placa.

numCam: Define o número total de câmeras.

19.6.20. Atributos do Steeplechase

server: (string) Contém o nome do servidor Steeplechase.

scan: (inteiro) Tempo de varredura e atualização do Steeplechase (em milisegundos).

19.6.21. Atributos do OPCServer

- **serverStatus**: (inteiro) Indica o status atual do servidor OPC. Se a verificação do servidor estiver habilitada nas propriedades do OPCServer (opção "Verificar servidor OPC"), a propriedade *serverStatus* será atualizada na freqüência especificada na propriedade *vrftTime*. Caso a verificação do servidor estiver desabilitada, a propriedade poderá assumir os valores 0 (não-conectado) ou -1 (desconhecido).
- **totalAlarms**: (inteiro) Indica o número total de alarmes ativos em um grupo de tags OPC.
- **vrfyTime**: (inteiro) Especifica o período de tempo (em milissegundos) em que o sistema deve verificar se o servidor está ativo.

Capítulo

20. Suporte a DDE

O Elipse SCADA permite a troca de informações com outros programas através do suporte a tecnologia DDE (*Dynamic Data Exchange*). O sistema pode operar como servidor ou cliente de aplicativos como o Microsoft Excel, Microsoft Access e outros.

20.1. Elipse SCADA como Cliente

O Elipse SCADA trabalhando como Cliente DDE permite buscar dados de aplicações servidoras DDE. Para tal conexão, é necessário que a outra aplicação seja servidora DDE (no mesmo computador). A sintaxe DDE é definida na aplicação servidora. Como Cliente DDE, o Elipse SCADA deve ter um tag DDE configurado de acordo com a documentação de sintaxe fornecida pela aplicação servidora. Não é necessário criar qualquer definição na aplicação servidora, pois o Elipse SCADA pode encontrar as informações automaticamente. A configuração do tag DDE é feita na página geral do tag no Organizer, como pode ser visto abaixo.

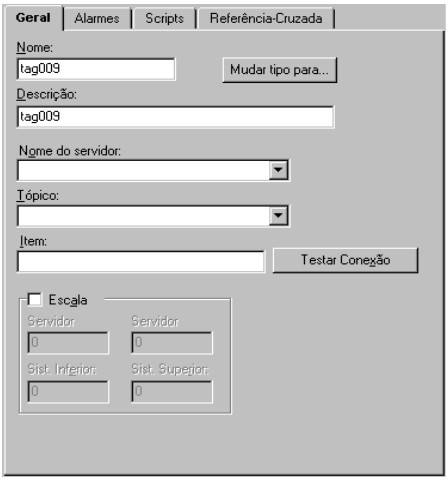


Figura 145: Página de configuração do tag DDE

Os campos Nome do Servidor, Tópico e Item são sempre definidos de acordo com a aplicação servidora. Se a aplicação estiver aberta no mesmo computador que o Elipse SCADA está rodando, esta é detectada automaticamente através dos itens Servidor e Tópico. É possível testar a conexão com a aplicação servidora através do botão [Testar Conexão].

Você pode enviar um comando DDE para uma aplicação servidora através de Scripts, usando a função DDEExecute(). Veja o capítulo sobre Scripts, em Funções Especiais, Gerenciador Global.

Exemplo: Elipse SCADA & Microsoft Excel

A configuração a seguir estabelece uma conexão DDE entre o Elipse SCADA, como um cliente, e o Microsoft Excel, como servidor, usando a célula A1 em uma planilha chamada Sheet1 na pasta de trabalho Book1. Ambos os programas estão rodando no mesmo computador.

Para esse exemplo, execute os seguintes procedimentos:

- Crie um Tag DDE.
- Faça as seguintes configurações no Tag DDE:

Configurações do Tag DDE

OPÇÃO	DESCRIÇÃO
Nome do Servidor	Excel
Tópico	[Plan1.xls]Plan1
Item	L1C1

NOTA: Para versões ou linguagens diferentes do software, tópicos e itens podem mudar.

- O botão **Testar Conexão** permite testar a conexão DDE. Uma vez feita a conexão, o tag DDE configurado pode ser usado em objetos de tela, scripts ou outras funções.

20.2. Elipse SCADA como Servidor

Elipse SCADA não necessita de nenhuma configuração especial para trabalhar como um servidor DDE, precisa apenas estar rodando com uma aplicação aberta no momento em que outra aplicação precisar acessar seus dados.

Para configurar a aplicação cliente basta especificar os três parâmetros para acessar o Elipse SCADA como servidor DDE: servidor, tópico e item.

Exemplo: Elipse SCADA & Microsoft Excel

Neste exemplo o Excel (Cliente) precisa acessar um tag no Elipse SCADA (Servidor) usando DDE.

Para esse exemplo, execute os seguintes procedimentos:

- Defina o tag que o Excel irá acessar no Elipse SCADA; por exemplo: tagDDE.
- Crie uma tabela no Excel e salve com qualquer nome; por exemplo: tabela.xls.
- No Excel, digite na célula desejada, digite a seguinte fórmula: =ELIPSE|Aplicação!Tags.tagDDE, onde: "ELIPSE" é o servidor DDE, "Aplicação" é o tópico e Tags.tagDDE é o item a ser acessado. Esse item pode ser qualquer tag ou atributo de qualquer objeto do aplicativo como por exemplo: "Hist1.DateTime".