PROGRAMA DE PESQUISA E DESENVOLVIMENTO TECNOLÓGICO
CICLO 2003/2004

Nº DO CONTRATO: 4500052325 **Nº DO PROJETO: 128**

INÍCIO: 28/09/2005 **DURAÇÃO: 24 meses**

A. Instituição executora:

B. Título do Projeto:

Modernização da Área de Automação de Processos das Usinas Hidroelétricas de Balbina e Samuel

C. Coordenador do Projeto:

Nome:	Alberto José Álvares				
Fone:	(61)-3307-2314	Cel:	(61)-99679435	E-Mail:	alvares@AlvaresTech.com

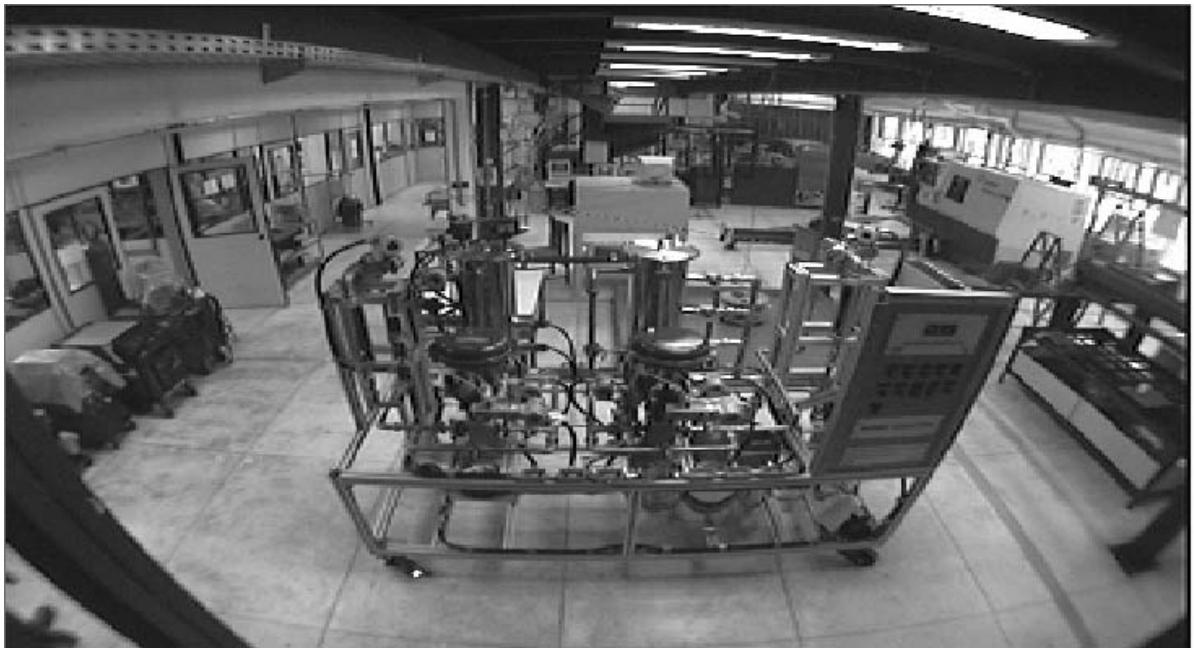
RELATÓRIO DE PRODUTOS GERADOS: ETAPA 3 – Ano1 & ETAPA 4 – Ano 2

ETAPA 3 – ANO 1: CONSTRUÇÃO MODELOS DE COMPONENTES PARA O SISTEMA DE TOMADA DE DECISÃO BASEADO EM 7 CAMADAS

SISTEMA I-KERNEL: UM KERNEL INTELIGENTE PARA O SIMPREBAL SISTEMA DE MANUTENÇÃO PREDITIVA DE BALBINA

ETAPA 4 – ANO 2: USO PLANTA DIDÁTICA SMAR PD III PARA TESTE DO DESENVOLVIMENTO DO SISTEMA I-KERNEL

(DEZEMBRO 2006)



Motivação

Este texto se destina a documentar a Especificação do Sub-sistema I-Kernel, um kernel de sistema inteligente a ser desenvolvido em Java, para dar suporte ao desenvolvimento do SIMPREBAL: Sistema Inteligente de Manutenção Preditiva de Balbina (Alvares, 2006).

O sistema SIMPREBAL vem sendo desenvolvido dentro do programa de pesquisa e desenvolvimento tecnológico da Eletronorte, por meio do contrato 4500052325 – projeto 128, e tem como responsável técnico o Prof. Alberto Álvares da UNB.

O sub-sistema I-Kernel vem a ser um dos componentes fundamentais do SIMPREBAL, sendo responsável pela captura dos sinais de monitoramento dos equipamentos, seu processamento inteligente e realimentação do banco de dados com recomendações de manutenção.

Neste documento, apresenta-se a especificação do Sub-sistema I-Kernel.

Compreensão do Problema

O sistema SIMPREBAL, atualmente em desenvolvimento, visa dar suporte às atividades de manutenção da Usina Hidroelétrica de Balbina, gerenciada pela Manaus Energia, utilizando-se de técnicas de sistemas inteligentes para detectar situações de manutenção preditiva e alertar o usuário de potenciais falhas em iminência de acontecer, de tal forma que atividades de manutenção dos equipamentos possam ser consideradas antes que tais falhas aconteçam de fato. A concepção do SIMPREBAL é fundamentada na idéia de manutenção centrada em confiabilidade, a partir de grandezas monitoradas pelo sistema de supervisão e controle da Usina.

Os sinais adquiridos dos equipamentos e do sistema de supervisão e controle da usina devem passar por um processamento que envolve sete camadas (1-sensor, 2-processamento de sinal, 3-monitoração de condição, 4-avaliação de saúde, 5-prognósticos, 6-tomada de decisão e 7-apresentação), de acordo com (Álvares, 2006). Parte deste processamento (camadas 1 e 2, e também parte das camadas 3 e 4), serão processadas em equipamentos de aquisição (Fieldbus, Rockwell) e sistemas de aquisição (Assetview/SMAR). A outra parte do processamento (parte das camadas 3 e 4, além das camadas 5, 6 e 7) deverão ser processadas dentro do SIMPREBAL.

Desta forma, para que o SIMPREBAL possa ser construído, vislumbrou-se a necessidade de um kernel de sistemas inteligentes, capaz de capturar os dados do sistema de supervisão (seja a partir do banco de dados do sistema, seja diretamente dos equipamentos sendo monitorados), processar estes dados utilizando técnicas de sistemas inteligentes, e atualizar os bancos de dados com os dados resultantes desse processamento. Este kernel, deveria ser capaz de acessar diretamente os dados, independente de sua origem e disponibilizá-los na forma de variáveis. Essas variáveis, devem poder ser processadas por um conjunto de regras (convencionais ou fuzzy) e/ou por redes neurais. Este processamento deve ser possível em N camadas, de tal forma a atender a arquitetura OSA-CBM que orienta o SIMPREBAL. O resultado desse processamento deve ser armazenado novamente no banco de dados, e daí poderem ser utilizados para gerar alertas aos controladores do sistema da Usina, na forma de mensagens “popup”, mensagens de e-mail ou algum outro tipo de alerta visual. Da mesma maneira, devem ser armazenados na forma de registros perenes, para compor “logs” dos eventos diagnosticados pelo sistema, funcionando como registros históricos desses diagnósticos, que poderão ser aproveitados futuramente para dotar o sistema de aprendizagem (seja por meio de redes neurais ou não).

Proposta de Solução do Problema

Para promover a solução do problema levantado acima, propõe-se o desenvolvimento de um sub-sistema de aquisição de dados e processamento inteligente que designaremos aqui como I-Kernel. Apesar do sub-sistema I-Kernel visar um apoio à construção do SIMPREBAL, sua concepção é a de um sub-sistema genérico, a ser desenvolvido em linguagem Java, capaz de realizar a aquisição de

dados a partir de bancos de dados e equipamentos acessíveis via OPC (OLE for Process Control), e seu processamento por meio de regras convencionais, regras fuzzy e redes neurais, em N camadas.

De modo a tornar o desenvolvimento do I-Kernel uma atividade mais rápida, ao invés de se desenvolver todo o sistema a partir do início, nossa proposta é a de se integrar diversos componentes já existentes, promovendo um desenvolvimento orientado ao reuso. Da mesma forma, o I-Kernel, dada sua concepção na forma de um componente de software, poderá ser reutilizado em projetos futuros que demandem soluções de processamento inteligentes envolvendo o processamento de regras, regras fuzzy ou redes neurais.

Dentre os componentes que se almeja integrar, estão as ferramentas Jess – Java Expert System Shell, desenvolvido pelo Sandia National Laboratories, para promover o processamento de regras compondo um sistema baseado em regras, o Fuzzy-Jess, desenvolvido pelo National Research Council do Canada's Institute for Information Technology, para o processamento de regras fuzzy, de modo análogo ao Jess, o JDBC para acesso aos bancos de dados e o JNI para o acesso aos servidores OPC que dão acesso direto aos equipamentos. O módulo de redes neurais deverá ser desenvolvido a partir das próprias equações e modelos dos diferentes tipos de redes neurais existentes na literatura científica.

Uma visão geral desta proposta, modelada em termos de um diagrama UML de subsistemas pode ser observada na figura 1 abaixo:

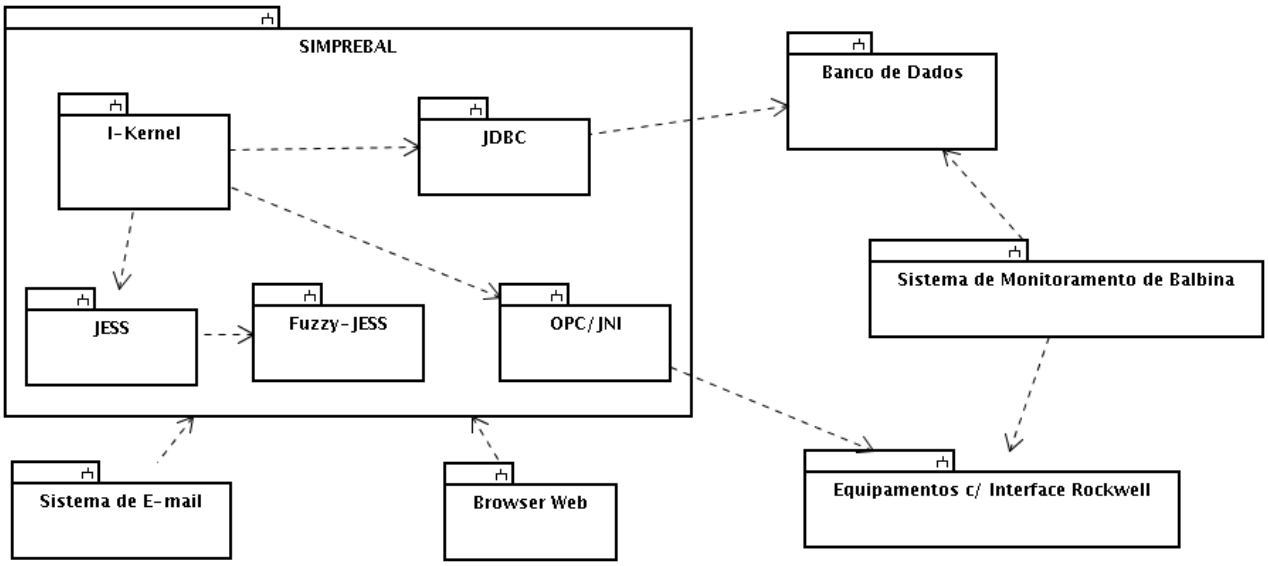


Figura 1: Visão Geral do Projeto SIMPREBAL, demonstrando o I-Kernel e sua interação com outros módulos

De acordo com (Sommerville, 2003), uma maneira útil de se estruturar os requisitos é dividí-los entre requisitos de usuário e requisitos de sistema. Os requisitos de usuário devem capturar as demandas mais gerais dos usuários do sistema. Normalmente, estes requisitos são descritos em linguagem natural e não devem entrar em detalhes mais técnicos. Os requisitos de sistema são descrições mais detalhadas dos requisitos de usuário. Neste documento, estaremos adotando este formato recomendado por Sommerville, listando os requisitos de usuário na forma de uma lista de requisitos em linguagem natural, e os requisitos de usuário serão detalhados de maneira mais técnica, utilizando-se diagramas UML. A metodologia de desenvolvimento a ser utilizada é uma adaptação do processo Unificado de modelagem (Jacobson et.al. 1999).

Requisitos de Usuário

Requisitos Funcionais

Requisito FU1: O sistema deve acessar os dados da Usina de Balbina a partir dos Bancos de Dados utilizados pelo Sistema de Monitoramento de Balbina, OU diretamente dos equipamentos de controle, por meio de um servidor OPC que disponibiliza as informações on-line dos equipamentos.

Requisito FU2: O sistema deve processar esses dados nas seguintes formas:

- Na forma de um sistema especialista baseado em regras
- Na forma de um sistema de regras fuzzy
- Na forma de redes neurais

Requisito FU3: O sistema deve alertar o usuário por meio de mensagens de e-mail quando possíveis falhas puderem ser diagnosticadas.

Requisito FU4: O sistema deve alertar o usuário por meio de um alerta visual, quando possíveis falhas puderem ser diagnosticadas.

Requisito FU5: O sistema deve propiciar a edição de um sinótico contendo certo conjunto de variáveis sendo monitoradas, escolhidas pelo usuário e compondo uma possível tela de apresentação.

Requisito FU6: O sistema deve exibir o valor on-line das variáveis sendo monitoradas que foram selecionadas para compor um determinado sinótico, apresentando-as em uma tela própria previamente desenvolvida.

Requisito FU7: O sistema deve implementar algum mecanismo de aprendizagem, de tal forma que o histórico de falhas e defeitos anteriores possa ser utilizado para prevenir o surgimento de novas falhas.

Requisito FU8: O processamento das informações se dará na forma de um ciclo operacional fechado, que seguirá a seguinte sequência:

1 . Verificação dos Dados a serem adquiridos
2 . Aquisição de Dados do Banco de Dados
3 . Aquisição de Dados via OPC
4 . Armazenamento provisório de todos os dados em variáveis do JESS
5 . Para cada uma de N camadas possíveis de processamento
    5.1 Processamento das regras via JESS
    5.2 Processamento das Regras Fuzzy, via Fuzzy-JESS
    5.3 Processamento dos dados via Redes Neurais
6 . Atualização dos Dados no Banco de Dados
7 . Atualização dos Dados via OPC.

Requisitos Não-Funcionais

Requisito NFU1: O sistema deve ser desenvolvido na linguagem Java.

Requisito NFU2: As regras do sistema não devem ser armazenadas diretamente em código-fonte, mas devem ser editáveis e estar disponíveis externamente em um arquivo modificável.

Requisito NFU3: O sistema deve possuir uma interface web de acesso, por meio da qual seja possível ao usuário configurar o sistema, editar regras e parâmetros e monitorar as variáveis sendo processadas pelo sistema.

Requisito NFU4: O sistema deve ser conectável a bancos de dados SQL genéricos, desde que exista um driver JDBC para o respectivo banco de dados.

Requisito NFU5: Para processar as regras na forma de sistemas especialistas, o sistema deve utilizar o pacote JESS.

Requisito NFU6: Para processar as regras fuzzy, o sistema deve utilizar o pacote Fuzzy-JESS.

Requisito NFU7: O sistema deve ser concebido de tal forma que as regras clássicas, regras fuzzy e redes neurais possam ser usadas de modo intercambiável para cada uma das camadas de processamento do SIMPREBAL (3-monitoração de condição, 4-avaliação de saúde, 5-prognósticos, 6-tomada de decisão)

Requisitos de Sistema

Seguindo o processo de desenvolvimento Unificado (Jacobson et.al. 1999), os requisitos do sistema devem ser especificados na forma de casos de uso. Na figura 2, a seguir, temos um diagrama UML de casos de uso, que nos propicia uma visão geral do sistema, segundo a perspectiva do sistema.

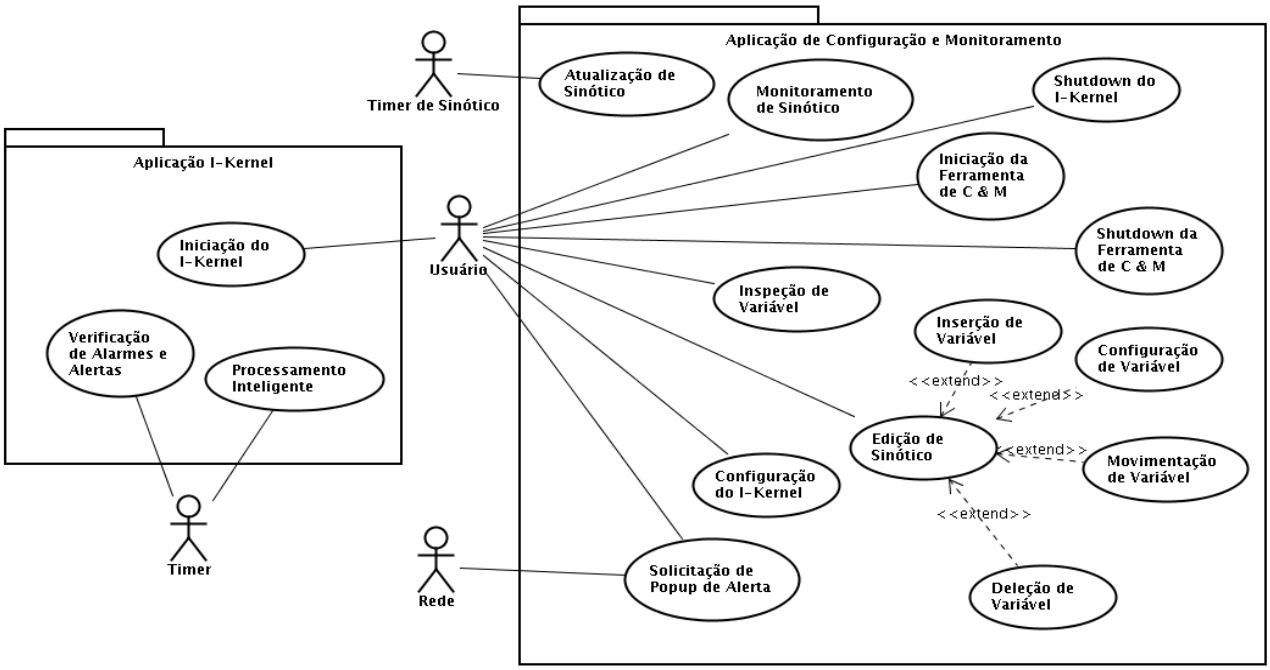


Figura 2 – Casos de Uso do Sistema

Detalhamento dos Casos de Uso

O sistema é composto de duas aplicações que são executadas de maneira independente:

- A aplicação I-Kernel
- A aplicação de Configuração e Monitoramento

A aplicação I-Kernel é um servidor que não possui interface com o usuário. Desta forma, a única coisa que o usuário pode fazer é iniciar a aplicação. Além disso, um timer previamente programado envia "ticks" periódicos que iniciam um ciclo operacional de processamento inteligente e da mesma forma fazem a verificação de alarmes e alertas. A aplicação I-Kernel está detalhada nos casos de uso de 1 a 3.

A aplicação de Configuração e Monitoramento é uma aplicação web, destinada a promover a configuração de parâmetros do I-Kernel, bem como o monitoramento das variáveis sob controle do sistema e a solicitação de shutdown do I-Kernel. Este monitoramento pode ser um monitoramento direto do estado das variáveis, selecionando-se a variável dentre todas aquelas disponíveis, ou pode ser um monitoramento por sinótico. No monitoramento direto de variáveis, escolhe-se a variável a ser

monitorada, e o sistema exibe seu estado diretamente. Este tipo de monitoramento visa efetuar uma inspeção isolada no estado de alguma variável do sistema. No monitoramento por sinótico, é necessário antes a edição de um sinótico, por meio da configuração de uma tela com um conjunto de variáveis que se deseja monitorar, e uma vez que um sinótico tenha sido editado, pode-se proceder ao monitoramento de sinótico. Além destes casos de uso, pode surgir uma solicitação via rede, normalmente originada pela aplicação I-Kernel, para que um popup de alarme/alerta seja exibido para o usuário. Essa solicitação de popup normalmente é gerada a partir do caso de uso de verificação de alarmes e alertas do I-Kernel. A aplicação de Configuração e Monitoramento encontra-se detalhada por meio dos casos de uso de 4 a 13.

Caso de Uso 1: Iniciação do I-Kernel

Neste caso de uso, o usuário solicita ao sistema operacional que a aplicação I-Kernel seja carregada, e o sistema operacional carrega a aplicação e a inicia. Um detalhamento destas operações encontra-se na figura 3 a seguir:

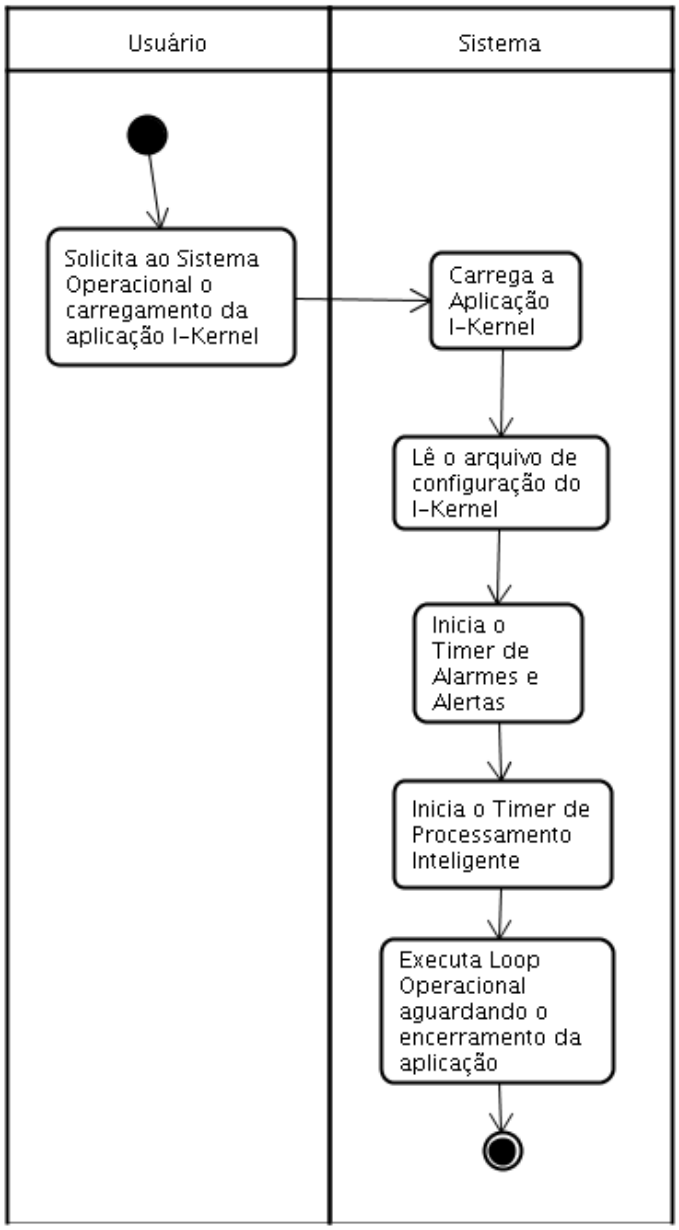


Figura 3 – Caso de Uso: Iniciação do Sistema

Caso de Uso 2: Processamento Inteligente

Nesse caso de uso, o Timer de Processamento Inteligente iniciado durante o caso de uso 1 envia "ticks" de relógio periódicos, iniciando um ciclo de processamento inteligente. Esse caso de uso se encontra detalhado na figura 4 a seguir:

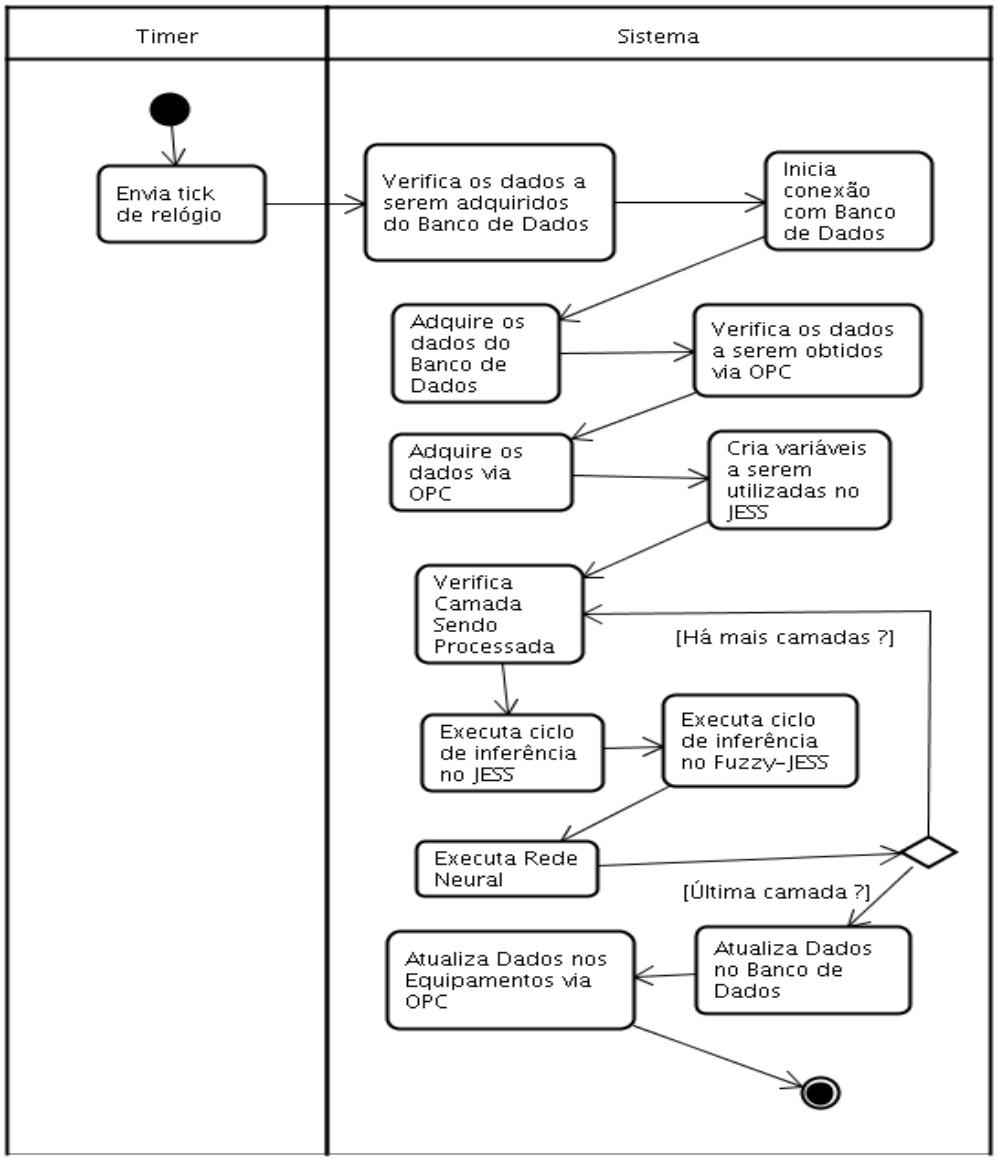


Figura 4 – Caso de Uso: Processamento Inteligente

Caso de Uso 3: Verificação de Alarmes e Alertas

Neste caso de uso, o Timer de Verificação de Alarmes e Alertas envia um tick de relógio, que procede à verificação de um conjunto de variáveis sendo monitoradas, e caso alguma delas esteja marcada, procede ao tipo de alarme ou alerta associado.

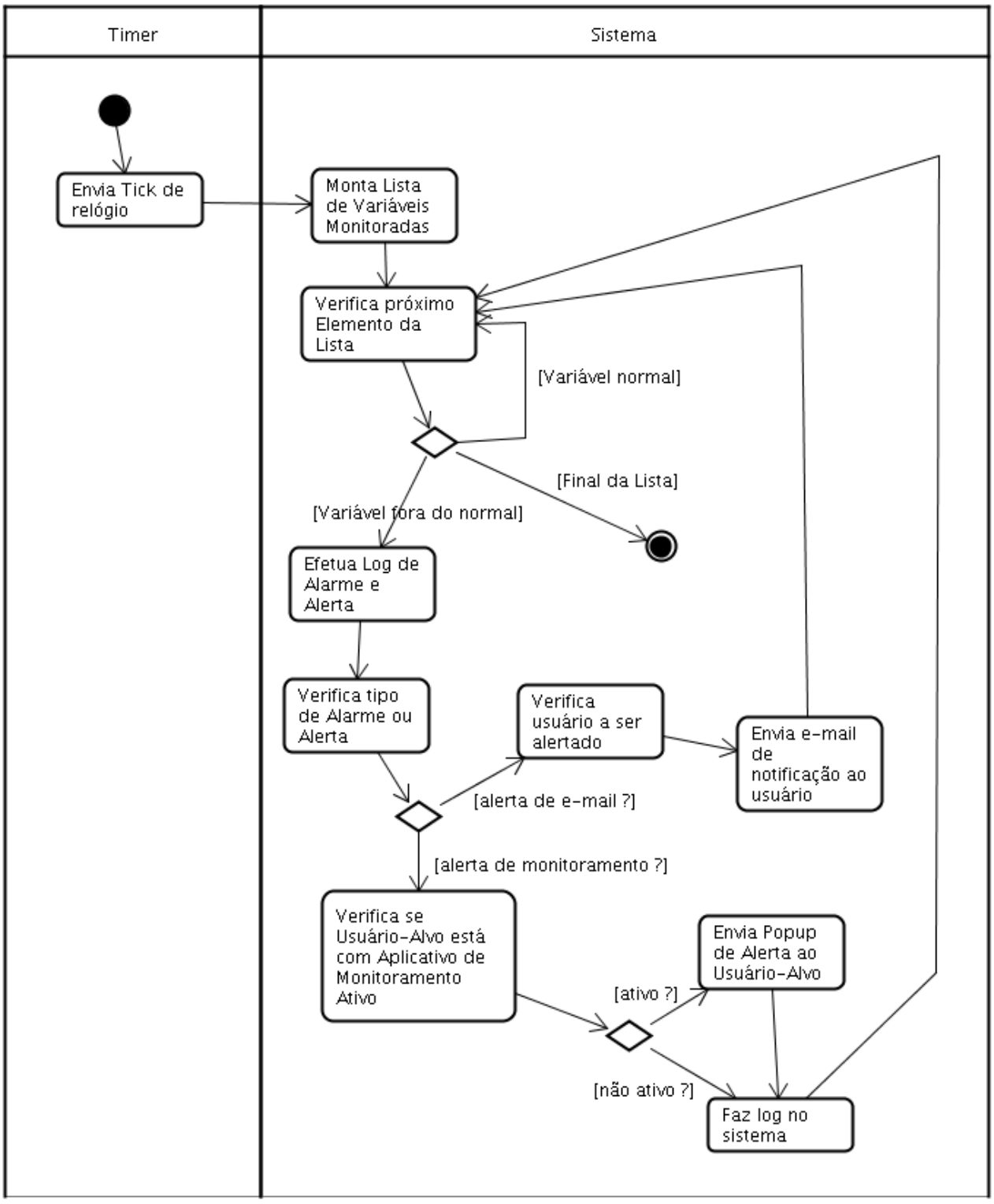


Figura 5 – Caso de Uso: Verificação de Alarmes e Alertas

Cada variável sendo acessada pelo I-Kernel pode ser marcada como uma variável a gerar alerta por e-

mail ou alerta de monitoramento. No alerta por e-mail o sistema deve especificar o endereço de e-mail para o qual uma mensagem, previamente armazenada em um template deve ser enviada. No alerta por monitoramento, o sistema deve especificar um usuário alvo (específico ou qualquer usuário), e levantar a lista de usuários monitorando o sistema, e gerar uma solicitação de popup de alerta com uma mensagem previamente armazenada em um template.

Caso de Uso 4: Shutdown do I-Kernel

Neste caso de uso, uma vez que o sistema de C & M tenha sido iniciado, o usuário solicita o encerramento do sistema I-Kernel. O sistema deve verificar se o I-Kernel está de fato operacional, e se estiver deve encerrar sua operação. Caso o sistema não esteja operacional, deve alertar o usuário de que ele não está operacional.

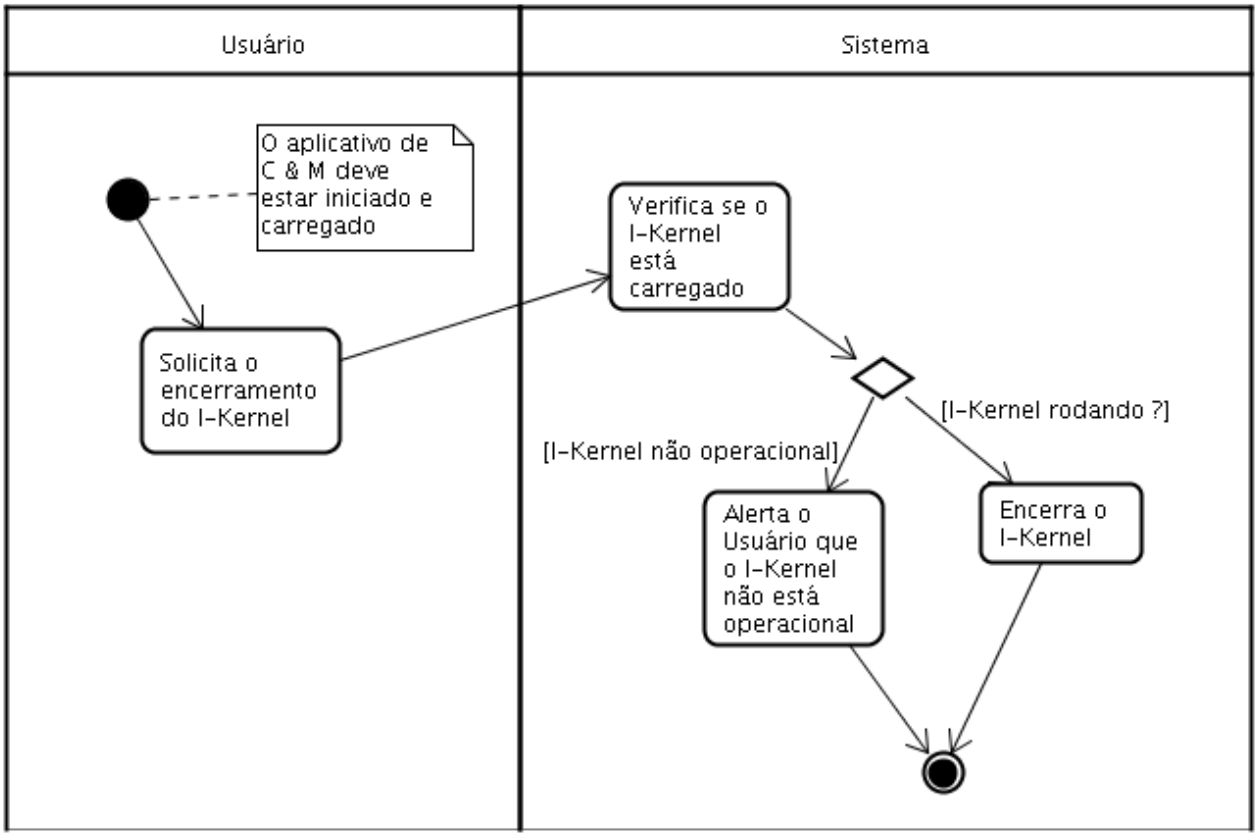


Figura 6 – Caso de Uso: Shutdown do I-Kernel

Caso de Uso 5: Iniciação da Ferramenta de C & M

Neste caso de uso, o Usuário carrega em um browser Web uma URL que corresponde ao endereço da Ferramenta de Configuração e Monitoramento. A partir daí, o sistema deve iniciar a ferramenta de C & M e abrir sua janela principal de interação com o usuário.

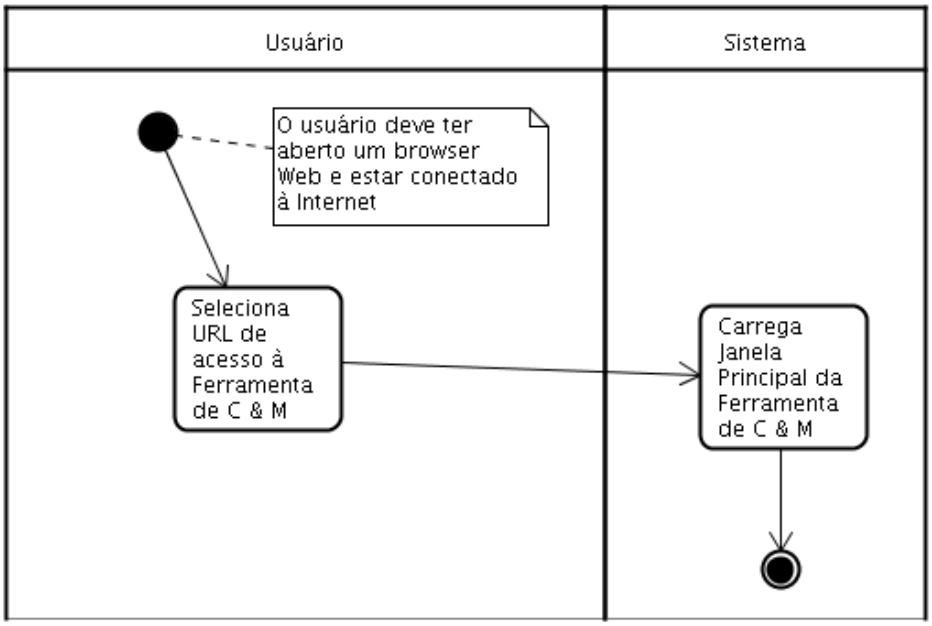


Figura 7 – Caso de Uso: Iniciação da Ferramenta de C & M

Caso de Uso 6: Shutdown da Ferramenta de C & M

Neste caso de uso, uma vez que a Ferramenta de C & M esteja aberta e operacional, o Usuário solicita que a mesma seja encerrada. A partir daí, o sistema deve encerrar a ferramenta.

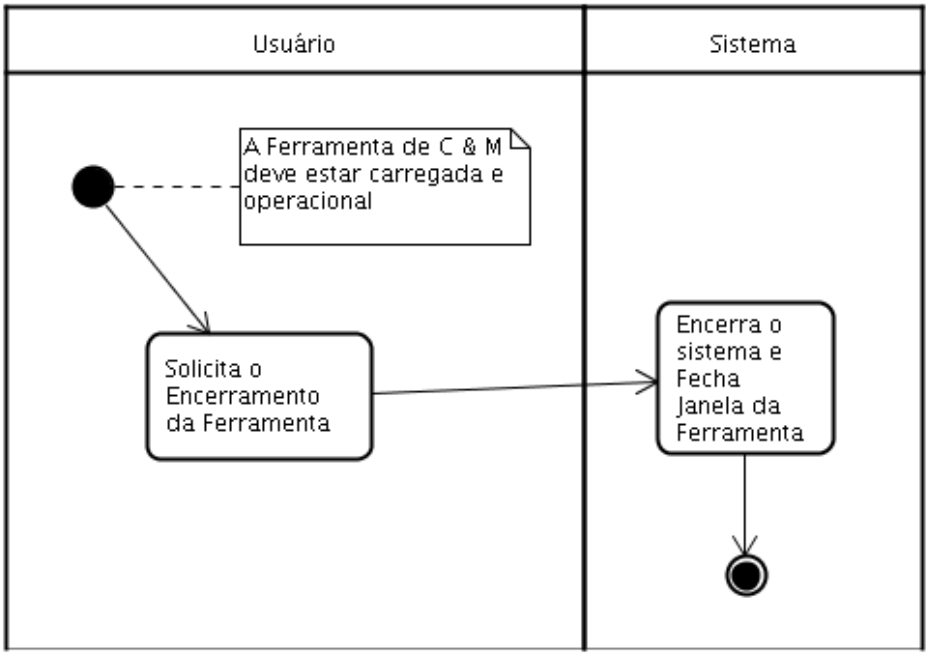


Figura 8 – Caso de Uso: Shutdown da Ferramenta de C & M

Caso de Uso 7: Edição de Sinótico

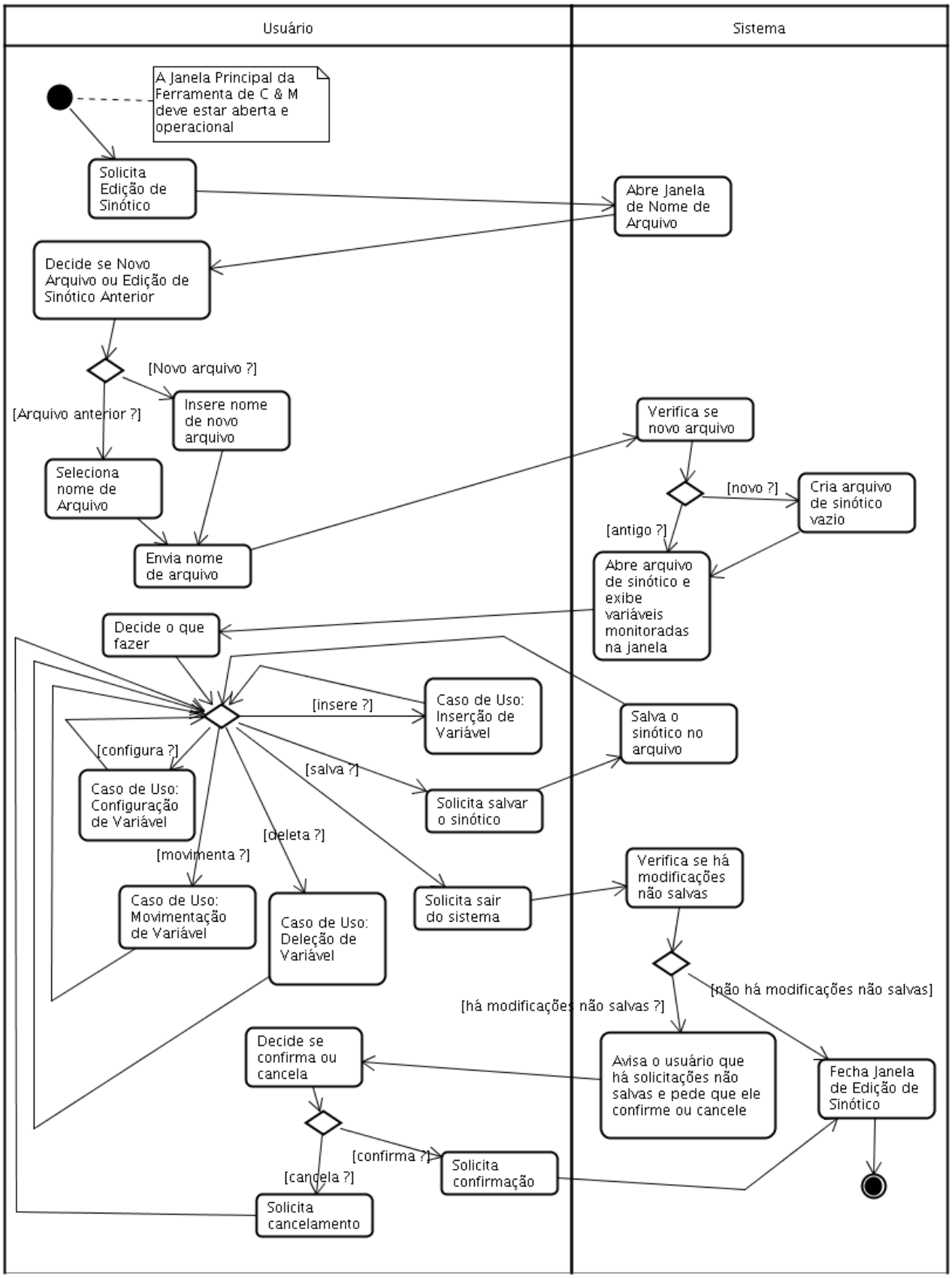


Figura 9 – Caso de Uso: Edição de Sinótico

Este caso de uso é um caso de uso bastante complexo, que se serve de 4 outros casos de uso que estendem sua funcionalidade. É por meio deste caso de uso que o usuário edita um novo sinótico que funcionará como template para o caso de uso de Monitoração de Sinótico.
Basicamente, o usuário informa o nome do sinótico a ser editado (pode ser um novo sinótico ou um sinótico editado anteriormente), e o sistema abre esse sinótico para edição. Com o sinótico aberto, o usuário tem várias opções que podem ser escolhidas em qualquer ordem:

- Inserção de Variável
- Deleção de Variável
- Movimentação de Variável
- Configuração de Variável
- Salvar Sinótico
- Finalizar Edição de Sinótico

Devido a sua complexidade, esse caso de uso prevê 4 pontos de extensão, onde os seguintes casos de uso podem estender as funcionalidades aqui especificadas:

- Caso de Uso: Inserção de Variável
- Caso de Uso: Deleção de Variável
- Caso de Uso: Movimentação de Variável
- Caso de Uso: Configuração de Variável

Caso o usuário tenha feito todas as edições necessárias, ele pode decidir salvar o sinótico e/ou encerrar a edição de sinótico. Os detalhes deste procedimento estão na figura 9 acima.

Caso de Uso 8: Inserção de Variável

Esse caso de uso estende o caso de uso Edição de Sinótico, permitindo que o usuário insira uma nova variável a ser monitorada no sinótico sendo editado. Basicamente o usuário seleciona a inserção de nova variável, posiciona o mouse onde deseja que a mesma seja inserida e clica no mouse, para que a mesma possa ser efetivamente inserida. O sistema insere a nova variável sendo monitorada e redesenha a tela.

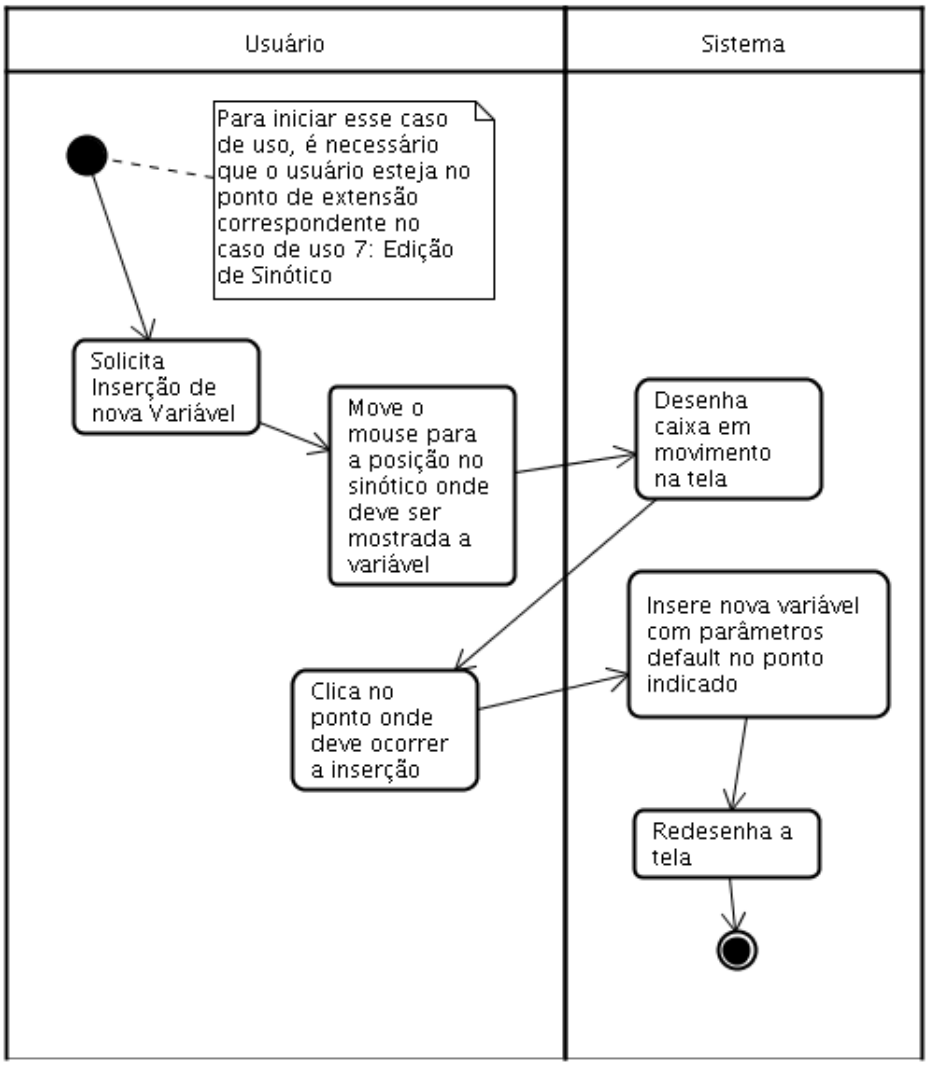


Figura 10 – Caso de Uso: Inserção de Variável

Caso de Uso 9: Deleção de Variável

Este caso de uso estende o caso de uso Edição de Sinótico, permitindo que o usuário delete uma variável sendo monitorada no sinótico sendo editado. Basicamente, o usuário seleciona a variável a ser deletada e pede que ela seja deletada. Ele pode ainda desistir de deletar a variável, e nesse caso o sistema simplesmente volta à situação anterior.

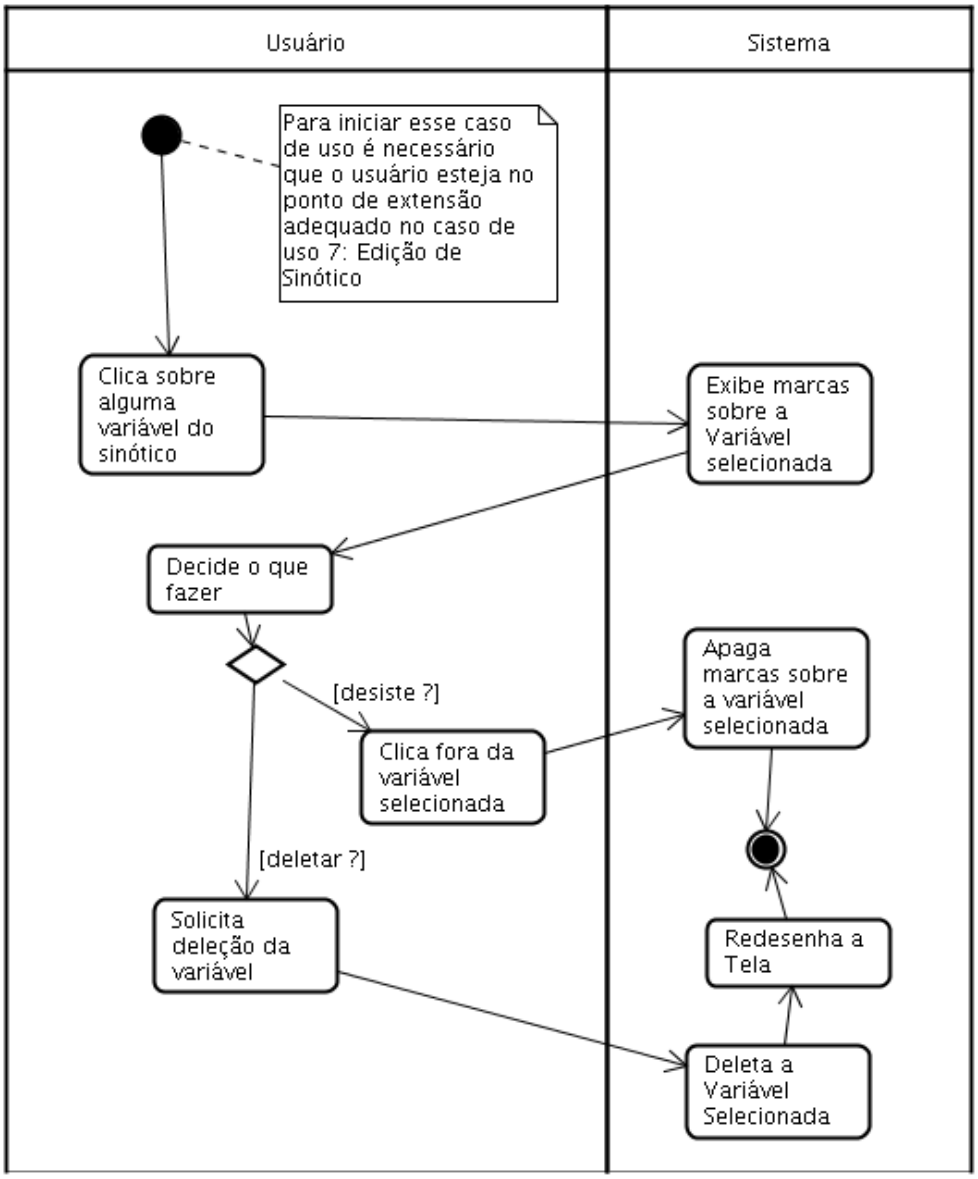


Figura 11: Caso de Uso: Deleção de Variável

Caso de Uso 10: Movimentação de Variável

Este caso de uso estende o caso de uso Edição de Sinótico, permitindo que o usuário movimente uma variável sendo monitorada no sinótico sendo editado. Basicamente, o usuário clica sobre a variável para marcá-la, e depois disso clica nela novamente, sem soltar o botão do mouse e a movimenta para a posição que deseja. Ao soltar o botão do mouse, o sistema movimenta a variável para a posição desejada.

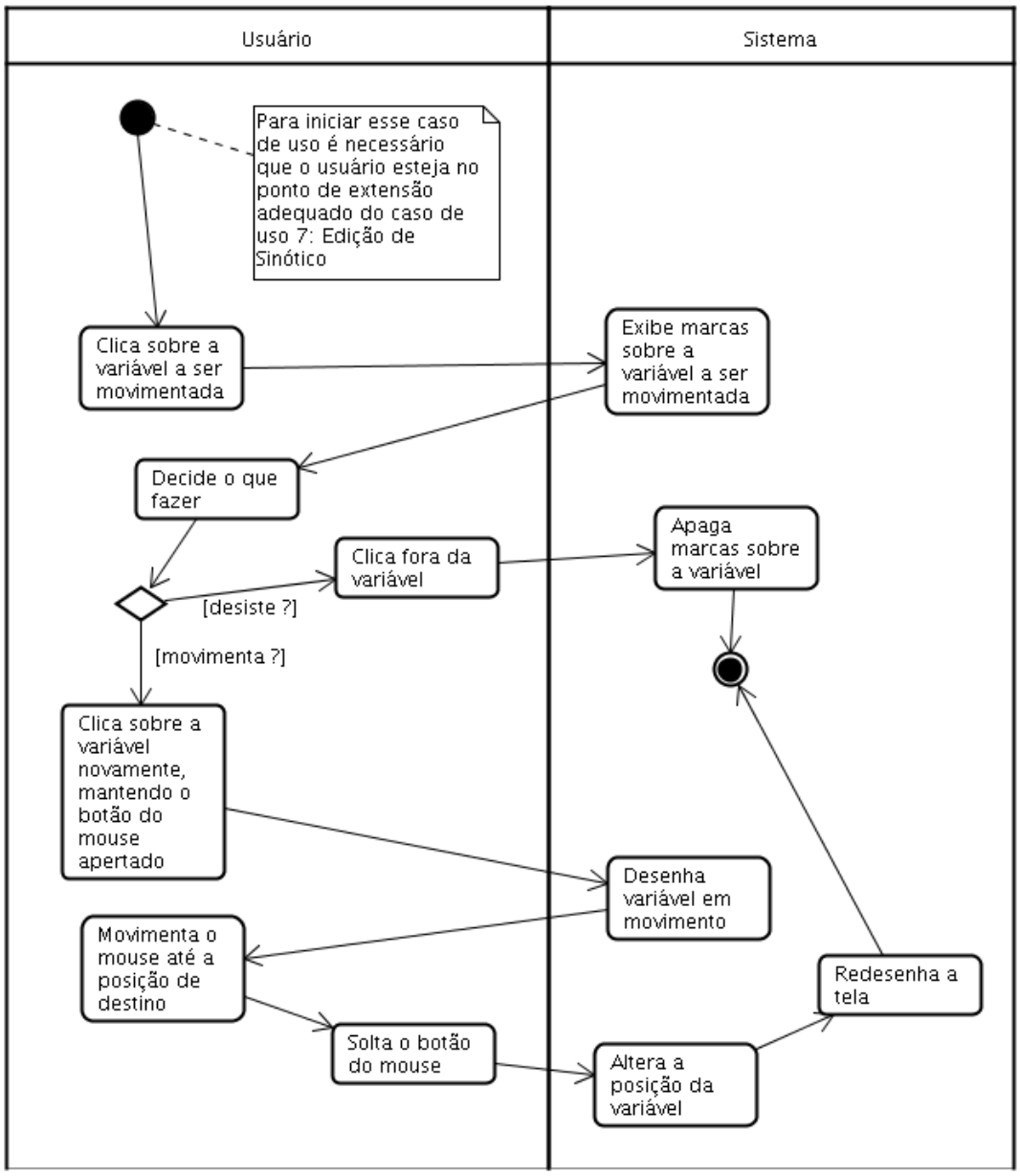


Figura 12 - Caso de Uso: Movimentação de Variável

Caso de Uso 11: Configuração de Variável

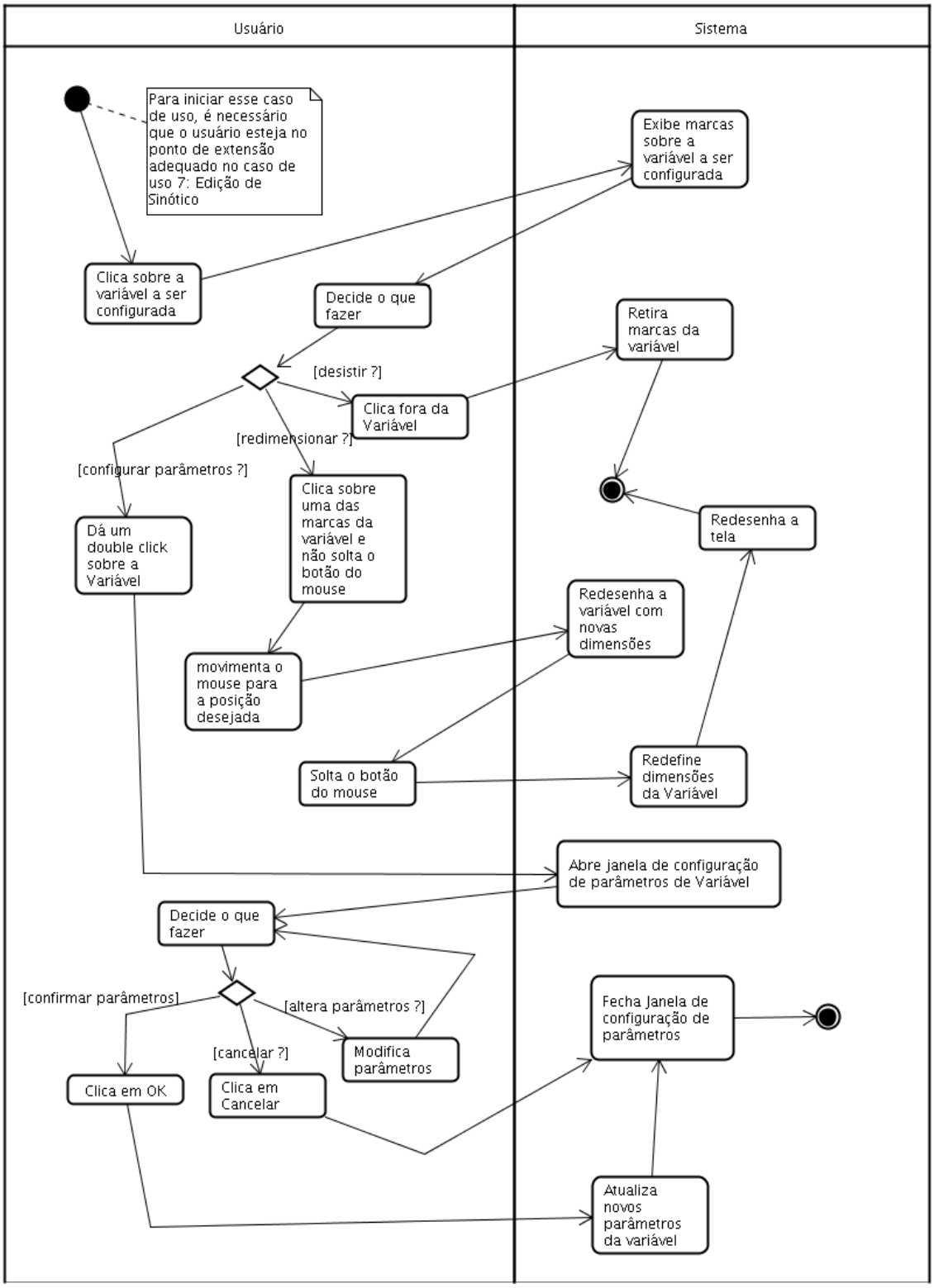


Figura 13 - Caso de Uso: Configuração de Variável

Este caso de uso estende o caso de uso Edição de Sinótico, permitindo que o usuário configure uma variável sendo monitorada no sinótico sendo editado. Basicamente o usuário tem duas opções: ou ele pode redimensionar a variável, ou ele pode modificar os parâmetros de configuração da variável. Se quiser redimensionar a variável, ele clica em uma das marcas da variável e a move para o novo ponto

desejado. O sistema deverá redimensionar a variável considerando esse novo ponto. Alternativamente, o usuário pode dar um “double click” na variável, e uma nova janela com os parâmetros de configuração irá aparecer. Nela, o usuário poderá editar todos os parâmetros da variável que deseja, e após terminar a configuração, ele poderá confirmar as alterações ou cancelar a alteração.

Caso de Uso 12: Monitoramento de Sinótico

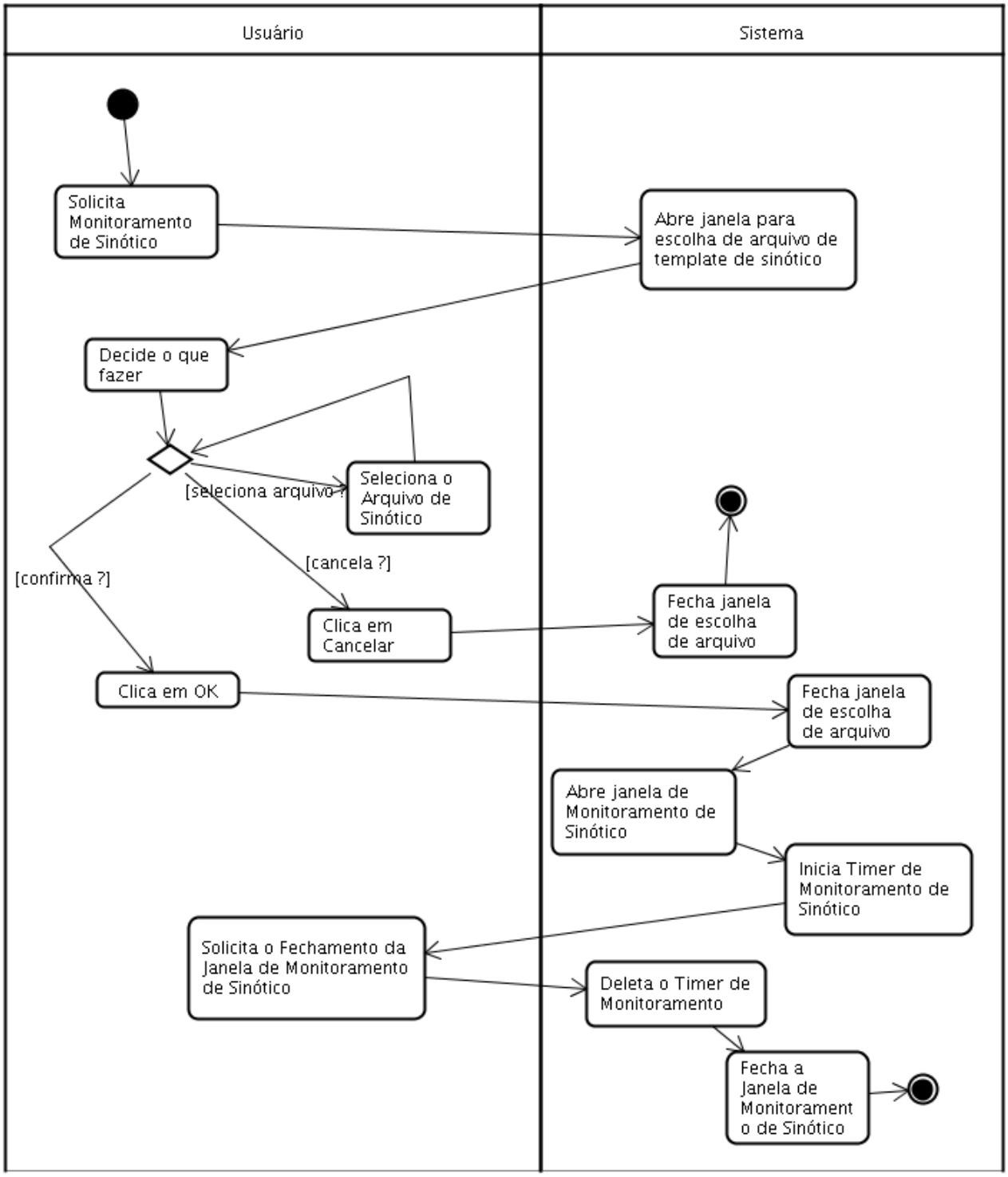


Figura 14 – Caso de Uso: Monitoramento de Sinótico

Neste caso de uso, o template de monitoração de sinótico editado no caso de uso 7: Edição de Sinótico, é utilizado para que o usuário obtenha o monitoramento de um conjunto de variáveis em uma janela definida no template. O usuário basicamente escolhe o arquivo com o template do sinótico, e o sistema abre a janela de monitoramento e inicia o Timer que fará a atualização da tela. Após isso, o sistema fica aguardando o usuário solicitar o fechamento da janela, e quando isso ocorre,

ele deleta o Timer e fecha a janela de monitoramento.

Caso de Uso 13: Atualização de Sinótico

Este caso de uso complementa o caso de uso 12: Monitoramento de Sinótico. Basicamente, quando uma janela de sinótico está sendo monitorada, ela está associada a um Timer de Sinótico que envia ticks periódicos de relógio. A cada tick, o sistema captura as variáveis do sinótico por meio do OPC e do Banco de Dados (conforme tenha sido configurada por meio do caso de uso Configuração de Variável), atualiza seu valor e redesenha tela para espelhar os novos valores.

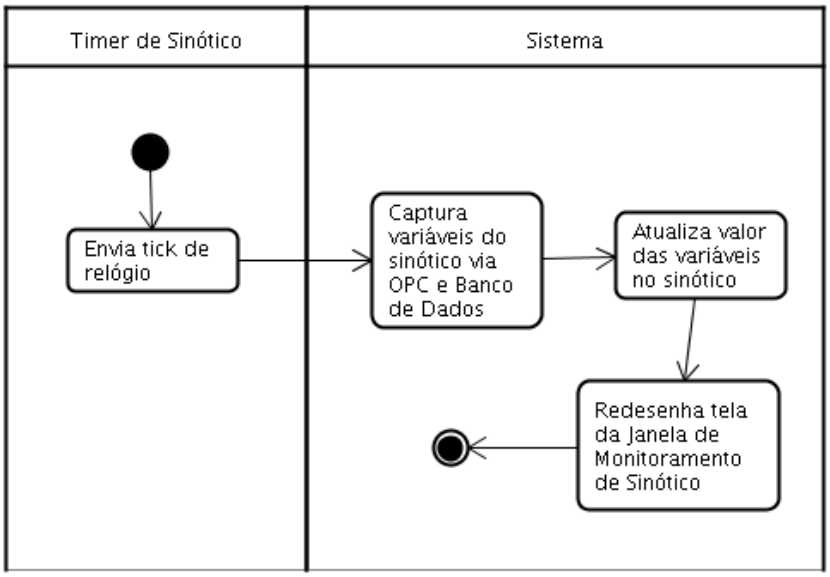


Figura 15 – Atualização do Sinótico

Caso de Uso 14: Inspeção de Variáveis

Nesse caso de uso, o usuário tem a opção de inspecionar uma ou mais variáveis, uma a uma. O usuário seleciona a inspeção de variáveis, e o sistema abre uma janela de inspeção de variáveis. Nela, o usuário pode selecionar qual variável deseja inspecionar, e o sistema busca essa variável no lugar adequado e exibe seu valor. O usuário pode então decidir inspecionar outra variável ou então finalizar a operação.

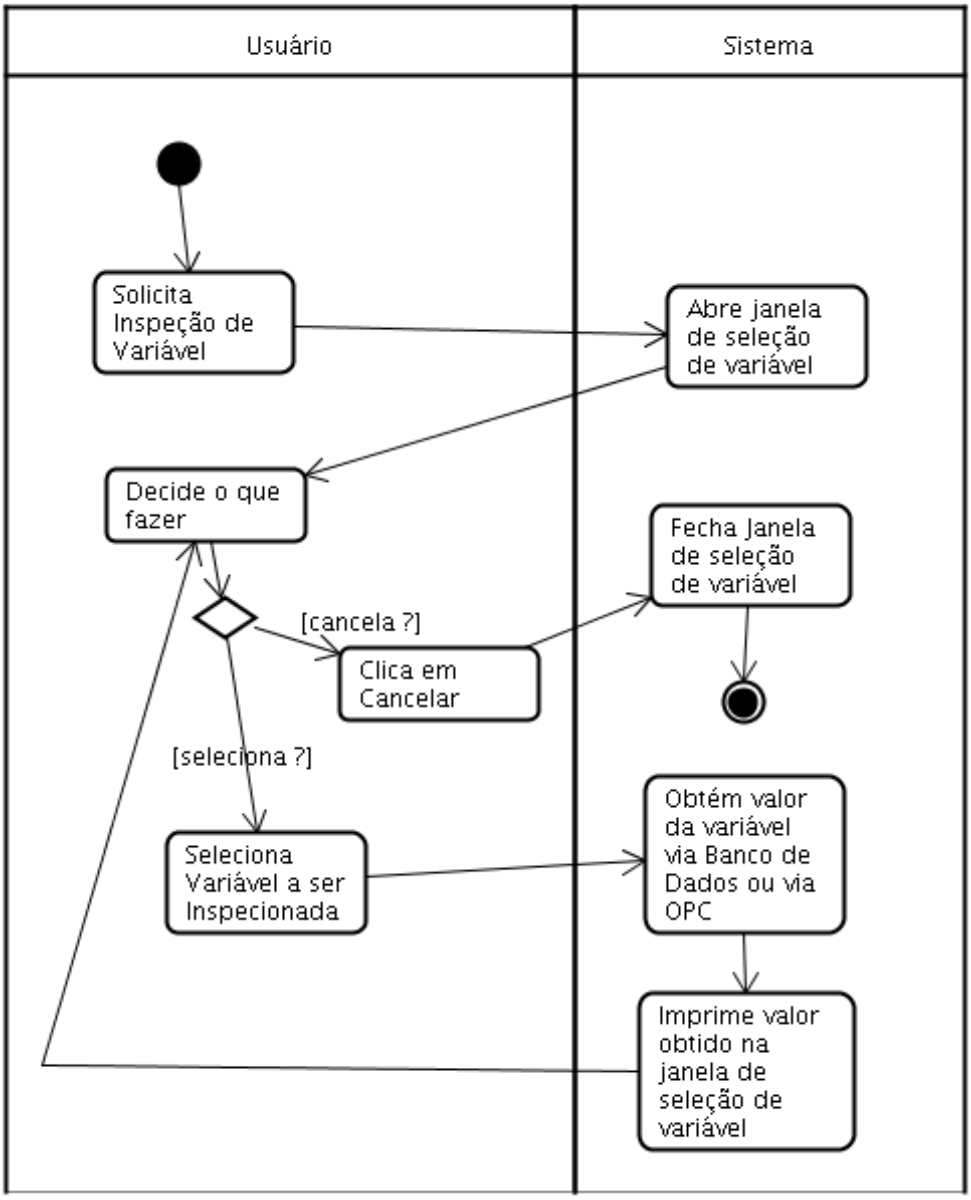


Figura 16 – Caso de Uso: Inspeção de Variável

Caso de Uso 15: Configuração do I-Kernel

Nesse caso de uso, o usuário pode atualizar os parâmetros de configuração do I-Kernel. Basicamente ele solicita ao sistema a configuração do I-Kernel, e o sistema abre uma janela de configuração, contendo todos os parâmetros configuráveis. O usuário pode então efetuar as alterações que desejar, e em seguida pode confirmar as mudanças ou sair cancelando a operação, caso em que nenhuma mudança será implementada.

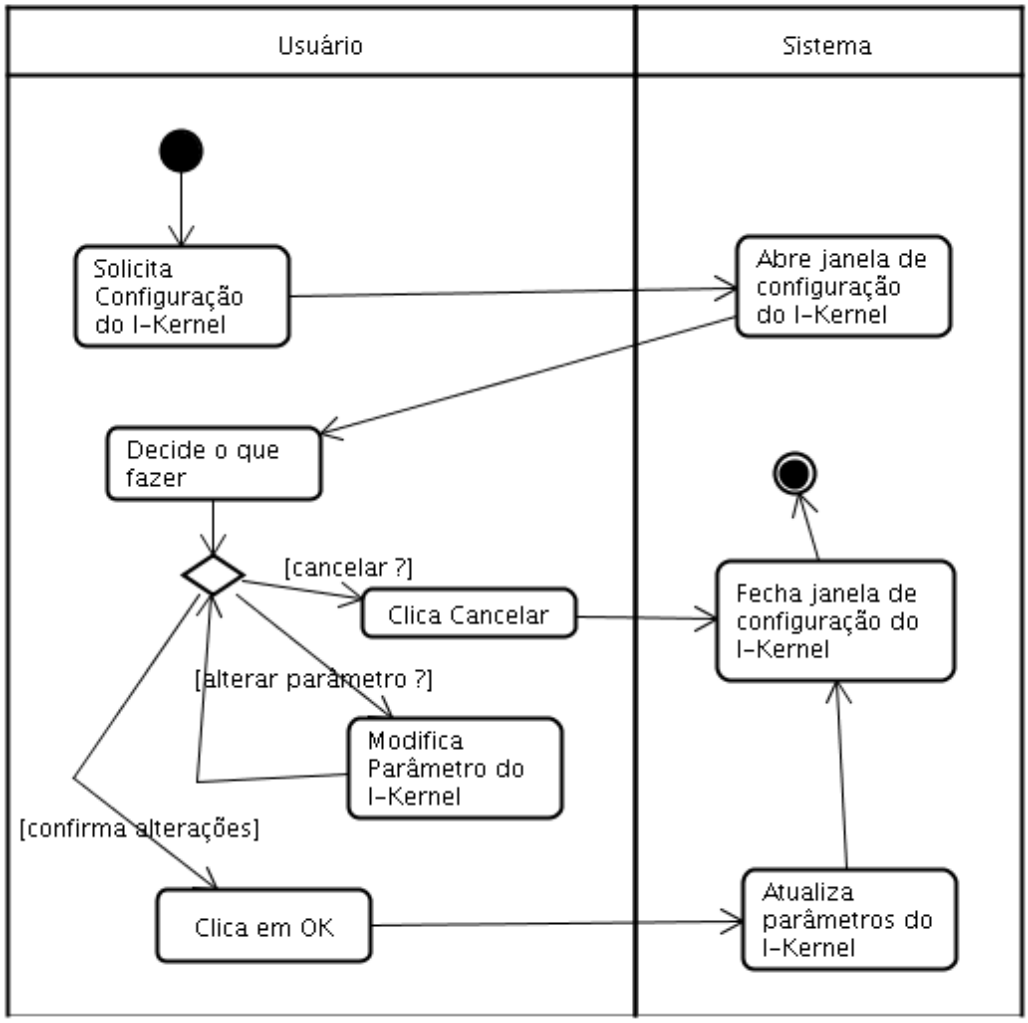


Figura 17 – Caso de Uso 15: Configuração do I-Kernel

Caso de Uso 16: Solicitação de Popup de Alerta

Esse caso de uso é utilizado em conjunto com o caso de uso 3: Verificação de Alarmes e Alertas. No caso de uso 3, o sistema I-Kernel envia uma solicitação via rede para a exibição de um popup de alerta caso alguma variável assim o seja programada. No presente caso de uso, a solicitação chega via rede, e o sistema C&M cria a janela de popup, mostrando a mensagem desejada ao usuário. Após ler a mensagem o usuário pode solicitar o fechamento da janela de popup.

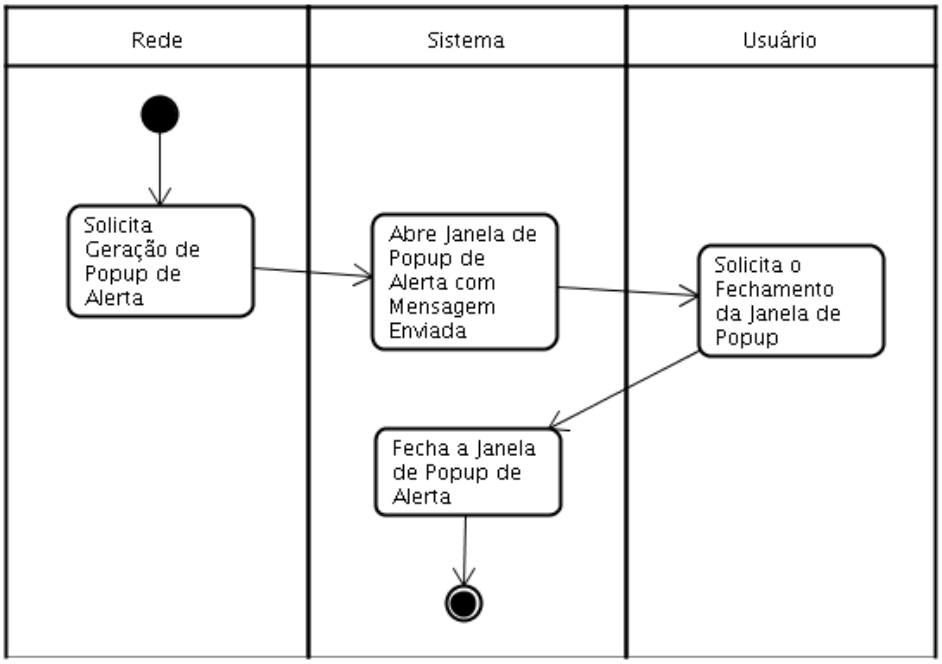


Figura 18 – Caso de Uso: Solicitação de Popup de Alerta

Referências

(Alvares, 2006) Alberto Alvares - SIMPREBAL: Metodologia Do Sistema De Manutenção Preditiva Da Usina De Balbina Baseado Nos Dados Monitorados Do Sistema De Supervisão E Controle Smar E Rockwell, Relatório Técnico de Pesquisa, UNB, Julho de 2006.

(Jacobson et. al. 1999) Ivar Jacobson, Grady Booch, James Rumbaugh - The Unified Software Development Process – Addisson Wesley, 1999.

(Sommerville, 2003) Ian Sommerville - Engenharia de Software, 6a. edição, Addison-Wesley/Pearson Education.

Sistema I-Kernel:
Um Kernel Inteligente para o SIMPREBAL - Sistema de Manutenção Preditiva de Balbina

PROJETO DO SISTEMA

Ricardo R. Gudwin
DCA-FEEC-UNICAMP

Campinas, Dezembro de 2006

Motivação

Este texto se destina a documentar o Projeto do Sub-sistema I-Kernel, um kernel de sistema inteligente a ser desenvolvido em Java, para dar suporte ao desenvolvimento do SIMPREBAL: Sistema Inteligente de Manutenção Preditiva de Balbina (Alvares, 2006).

O sistema SIMPREBAL vem sendo desenvolvido dentro do programa de pesquisa e desenvolvimento tecnológico da Eletronorte, por meio do contrato 4500052325 – projeto 128, e tem como responsável técnico o Prof. Alberto Álvares da UNB.

O sub-sistema I-Kernel vem a ser um dos componentes fundamentais do SIMPREBAL, sendo responsável pela captura dos sinais de monitoramento dos equipamentos, seu processamento inteligente e realimentação do banco de dados com recomendações de manutenção.

Neste documento, apresenta-se o projeto do Sub-sistema I-Kernel.

Inicialmente, apresentamos uma previsão de configuração para o Sistema, distribuindo os diferentes módulos em diferentes máquinas onde os mesmos possam ser instados.

Em seguida, apresentamos um projeto de arquitetura para o sistema, discriminando um conjunto de classes iniciais que darão suporte para os dois módulos executáveis do sistema, o módulo ConfMonitToolApp e o módulo I-KernelApp.

Por fim, apresenta-se o projeto de realização de cada caso de uso identificado na Especificação do Sistema (Gudwin 2006), a partir da colaboração entre objetos disponíveis na arquitetura, ressaltando-se a interação entre eles em diagramas de colaboração UML, conforme preconiza a metodologia Unificada de desenvolvimento (Jacobson et.al. 1999).

Previsão de Configuração do Sistema

Na figura 1 a seguir, temos uma previsão da configuração final do sistema, de acordo com uma possível instalação.

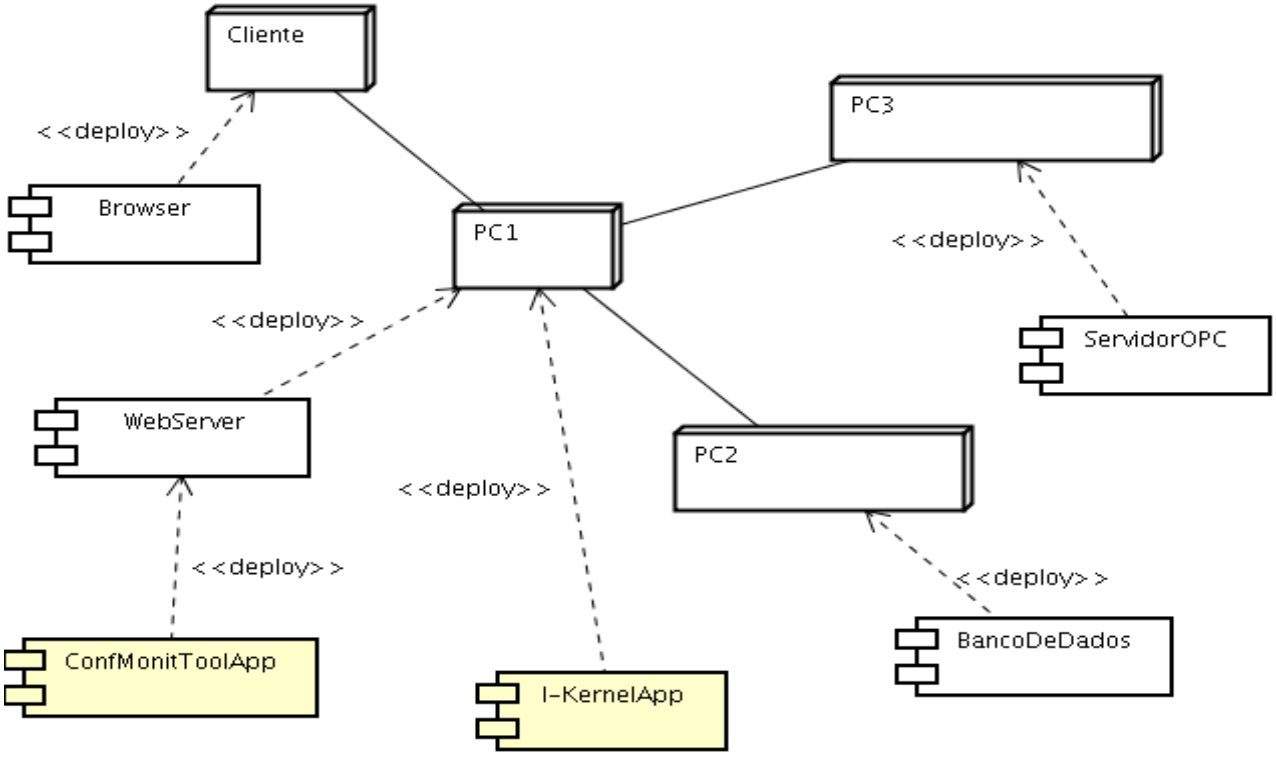


Figura 1 – Previsão de Configuração do Sistema

Na estrutura geral do sistema, temos uma série de computadores do tipo PC, onde diferentes serviços estão instalados. Apesar do diagrama mostrar diferentes máquinas, poderia-se assumir, sem prejuízo do conceito, que estes serviços estejam instalados todos em uma mesma máquina. A única restrição que se apresenta neste sentido é que tanto o módulo ConfMonitToolApp como o módulo I-KernelApp necessariamente precisam estar instalados na mesma máquina. Essa restrição ocorre pois o módulo ConfMonitToolApp será um Applet Java que se utilizará da rede para se comunicar com o módulo I-KernelApp. Como por restrições de segurança um Applet Java só pode se comunicar com a mesma máquina de que foi carregado, acaba sendo necessário que o módulo I-KernelApp seja instalado na mesma máquina que abriga o servidor Web, pois caso contrário o Applet não conseguirá se comunicar com o módulo I-KernelApp. Os serviços não marcados no diagrama não fazem parte do desenvolvimento sendo projetado neste documento, sendo considerados como reúso de software. Entretanto, nosso sistema deve interagir com tais sistemas, o que justifica sua indicação neste diagrama. Dentre os serviços que serão incorporados por reúso, estão um servidor Web, um Banco de Dados compatível com o JDBC e um servidor OPC que deve ser acessível por meio de JNI. Os dois componentes de software que serão produzidos neste projeto são os aplicativos I-KernelApp e ConfMonitToolApp. Estes aplicativos serão disponibilizados na forma de arquivos JAR e deverão ser executados independentemente por meio de uma máquina virtual Java. O módulo I-KernelApp é uma aplicação Java *standalone*, responsável pela aquisição de dados dos equipamentos de Balbina, por meio do banco de dados e dos equipamentos via OPC, seu processamento inteligente de forma a detectar situações de manutenção preventiva, e eventualmente a tomada de ações de atuação no sistema. O módulo ConfMonitToolApp é um Applet Java, carregado a partir de uma página Web armazenada no servidor Web, e é responsável pelas tarefas de configuração de parâmetros e regras de decisão do I-Kernel, bem como o monitoramento ativo das variáveis consideradas importantes, exibidas por meio de um sinótico.

Arquitetura do Sistema

A arquitetura do sistema pode ser visualizada nas figuras 2, 3 e 4 a seguir:

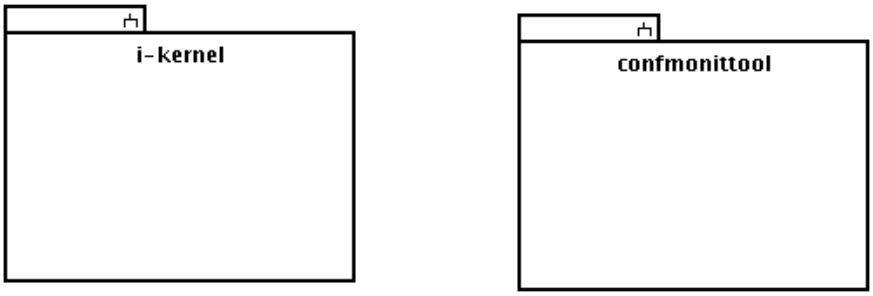


Figura 2 – Arquitetura do Sistema: Diagrama Raíz

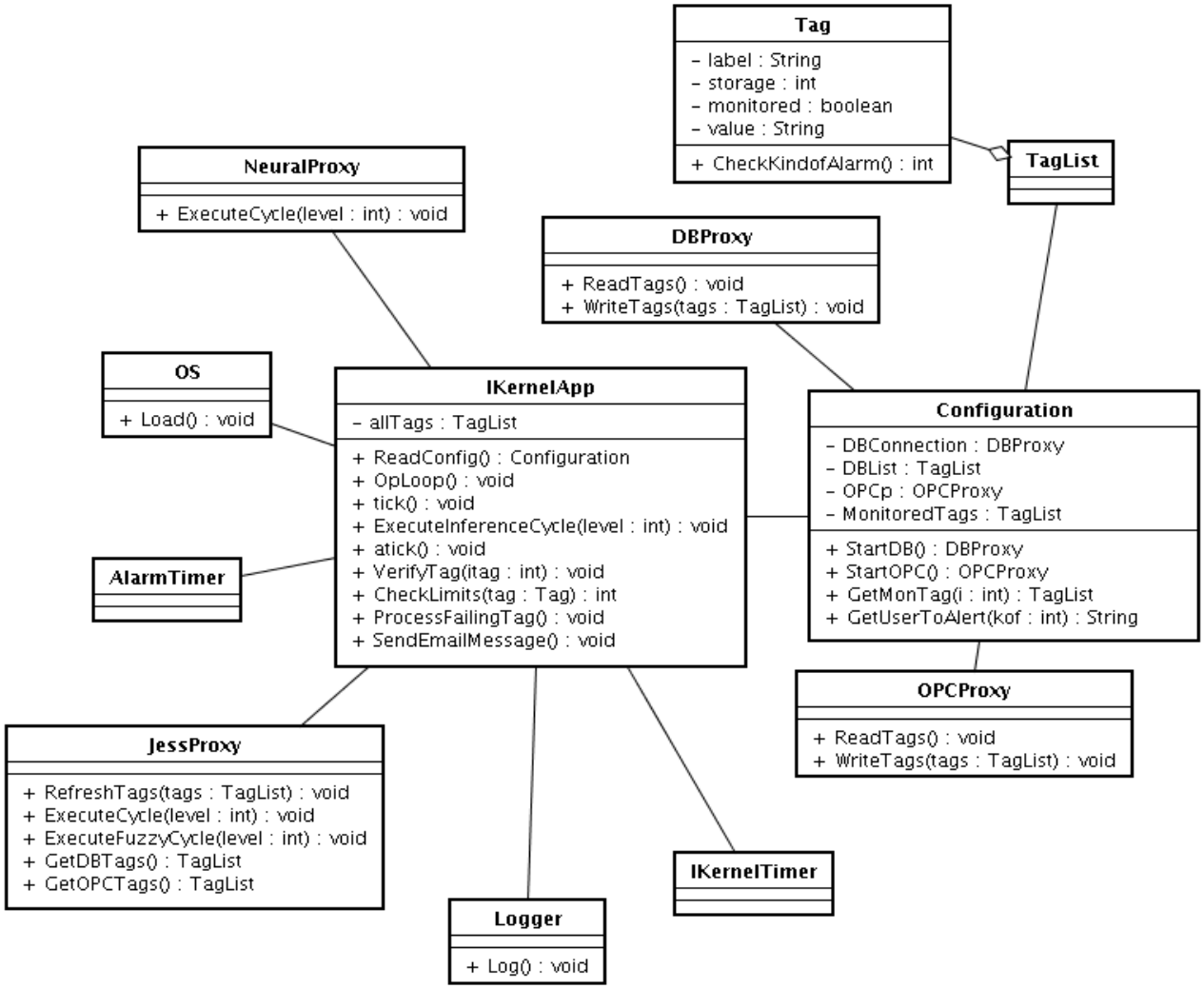


Figura 3 – Arquitetura do Sistema: Pacote i-kernel

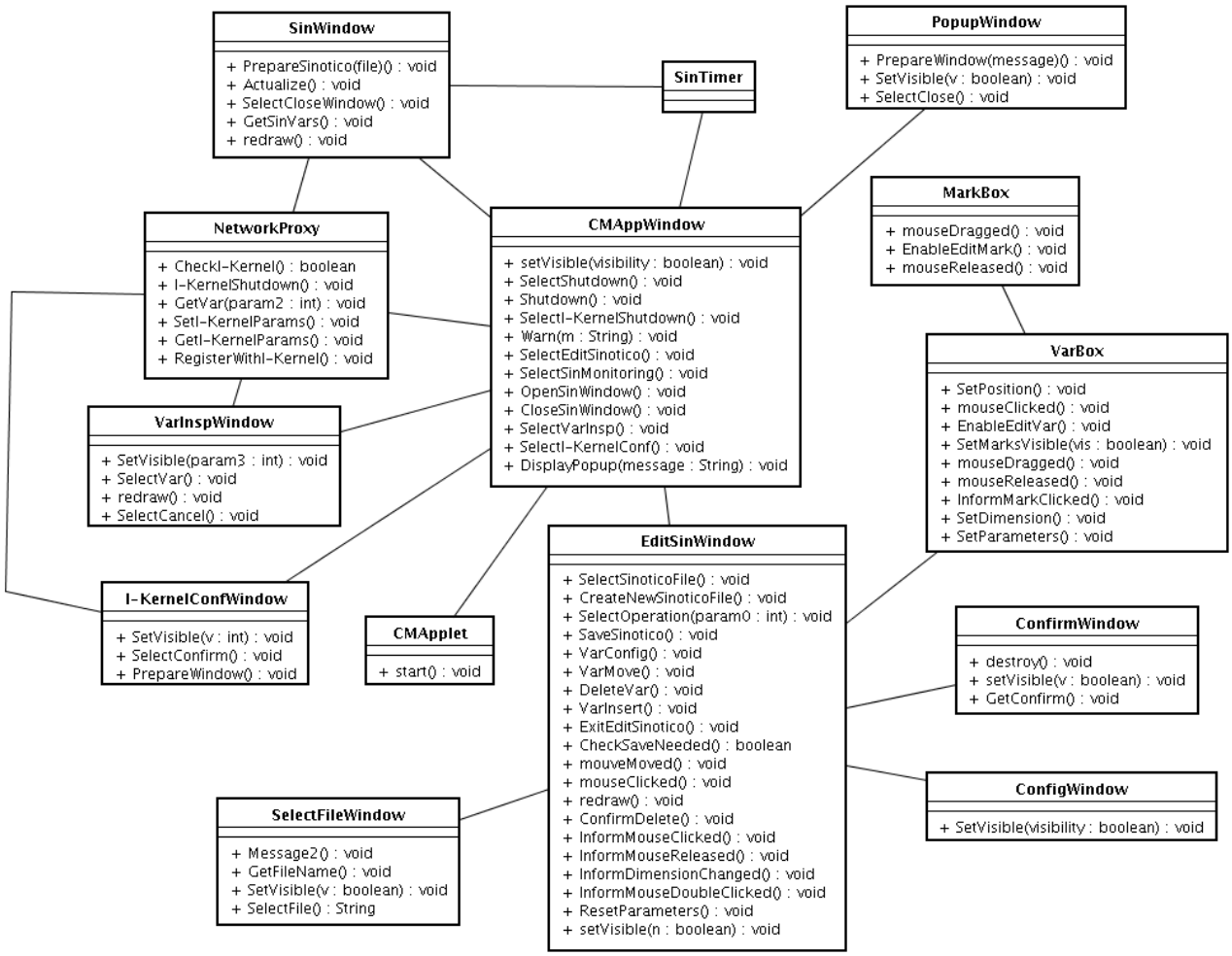


Figura 4 – Arquitetura do Sistema: Pacote confmonittool

A figura 2 apresenta o diagrama Raíz para todo o sistema, que basicamente subdivide a arquitetura em 2 pacotes, o pacote i-kernel e o pacote confmonittool.

O pacote i-kernel é ilustrado na figura 3, e inclui basicamente as classes necessárias para implementar o módulo I-KernelApp.

O pacote confmonittool é ilustrado na figura 4, e inclui basicamente as classes necessárias para implementar o módulo ConfMonitToolApp.

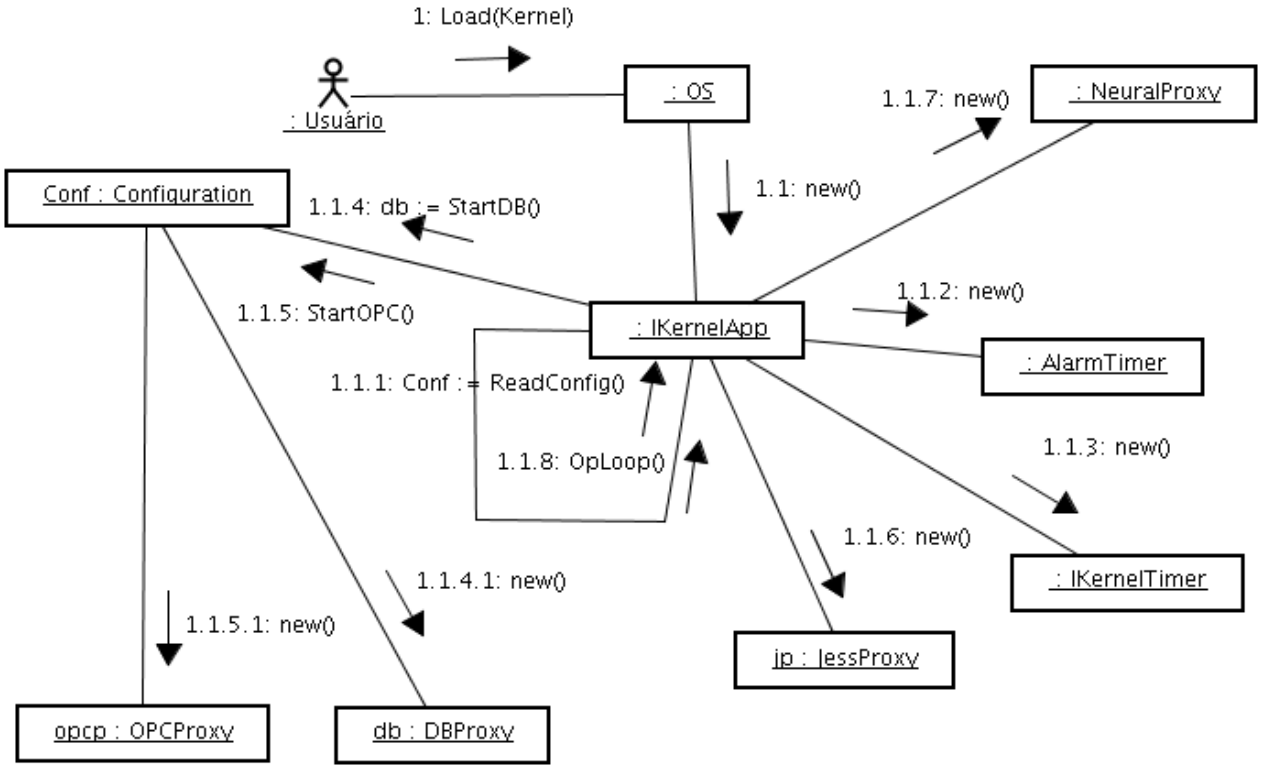
Essa arquitetura visa projetar um conjunto de classes básica para a implementação das aplicações, mas não se propõe a ser uma documentação exaustiva de todas as classes a serem implementadas no projeto. Durante a implementação, certamente diversas outras classes deverão ser incluídas na arquitetura do sistema, dependendo-se dos detalhes técnicos que poderão surgir em tempo de geração de código. A presente arquitetura visa somente apresentar uma referência para a implementação, sendo que correções e/ou adições certamente serão necessárias na fase seguinte, de implementação. Detalhes de implementação não foram aqui considerados.

Detalhamento dos Casos de Uso

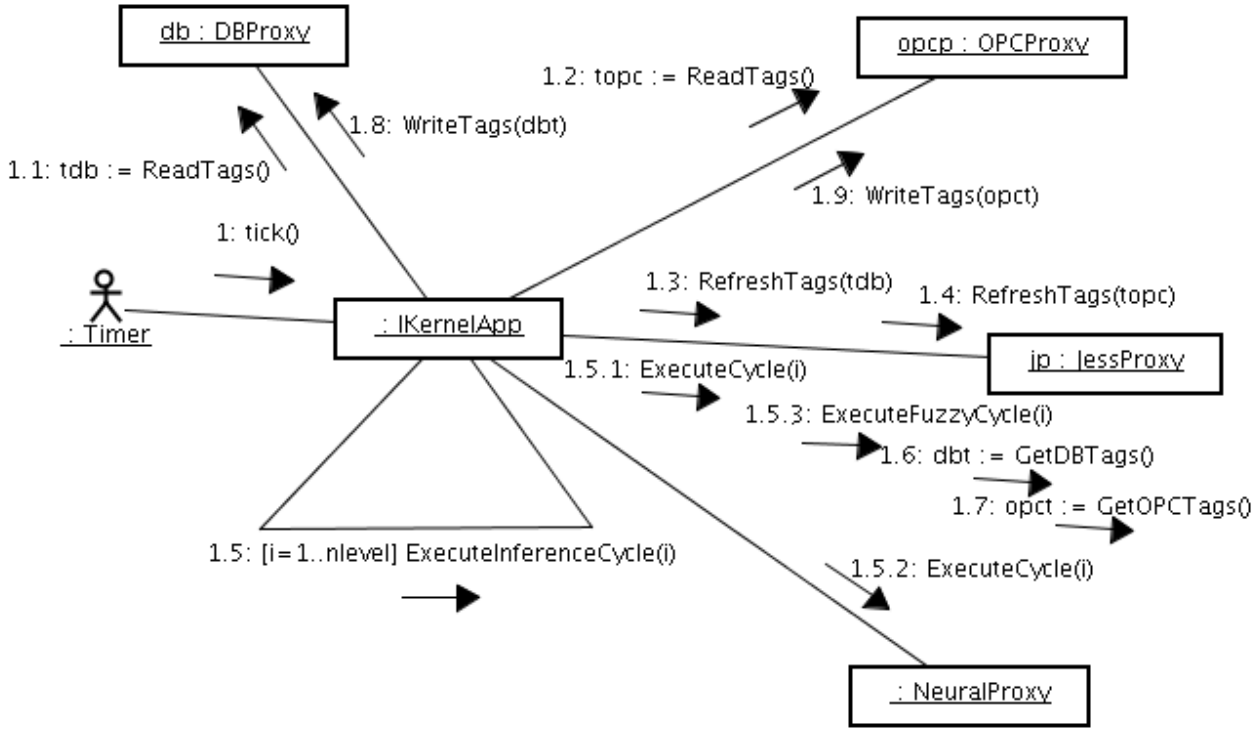
Nas seções a seguir, apresentamos um conjunto detalhado de projetos de realização de casos de uso, considerando-se a arquitetura de software apresentada anteriormente. Para cada caso de uso identificado na Especificação (Gudwin 2006), desenvolveu-se uma possível colaboração entre objetos da arquitetura, que visa realizar o caso de uso em termos de objetos do sistema.

Estas colaborações visam dar uma idéia ao implementador do sistema, para que o código dos métodos das classes possa ser implementado. Entretanto, essas colaborações devem ser vistas como realizações de referência, podendo ser modificadas na etapa posterior de implementação, caso isso se mostre necessário. Detalhes maiores das realizações foram omitidos, e deixados a encargo do responsável pela implementação.

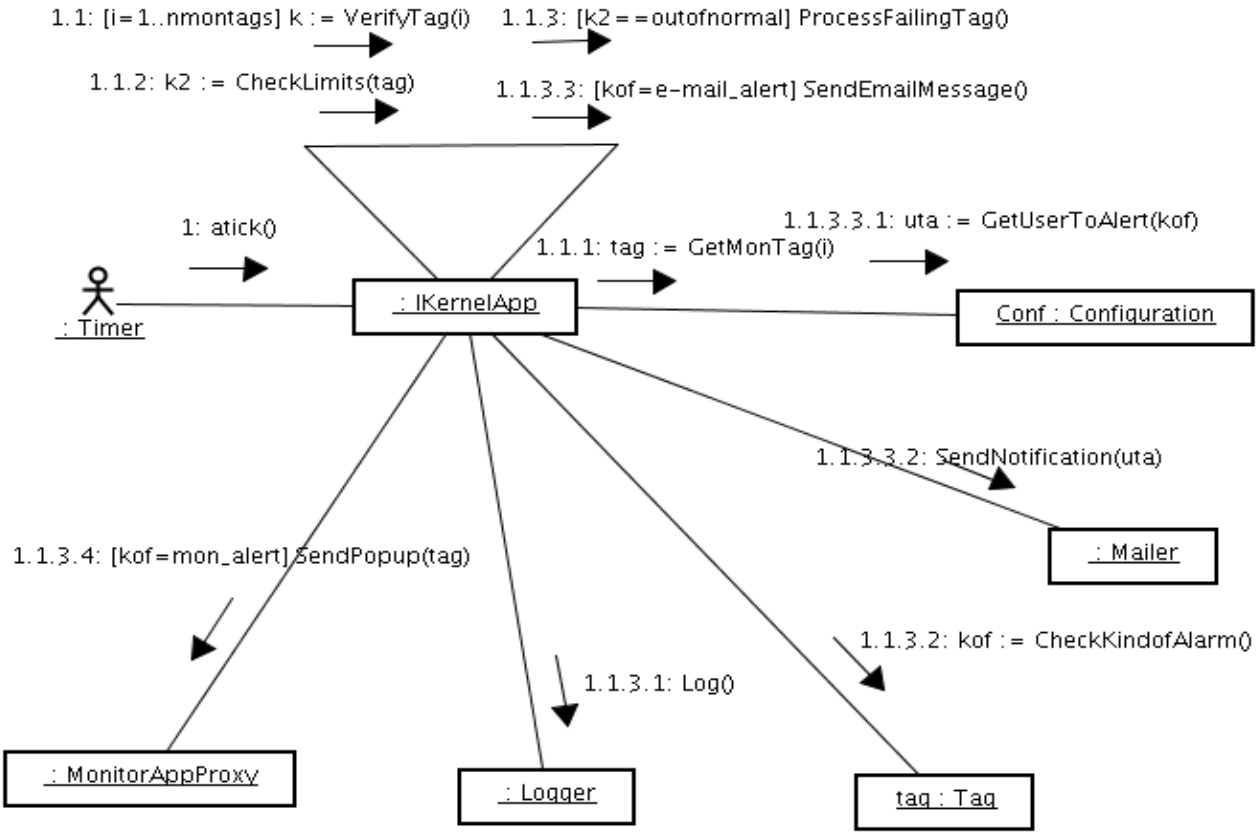
Caso de Uso 1: Iniciação do I-Kernel



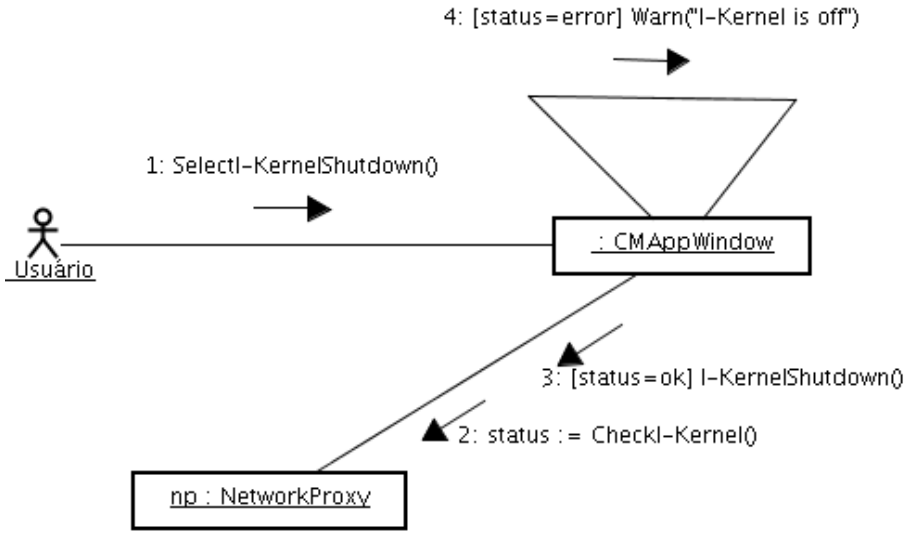
Caso de Uso 2: Processamento Inteligente



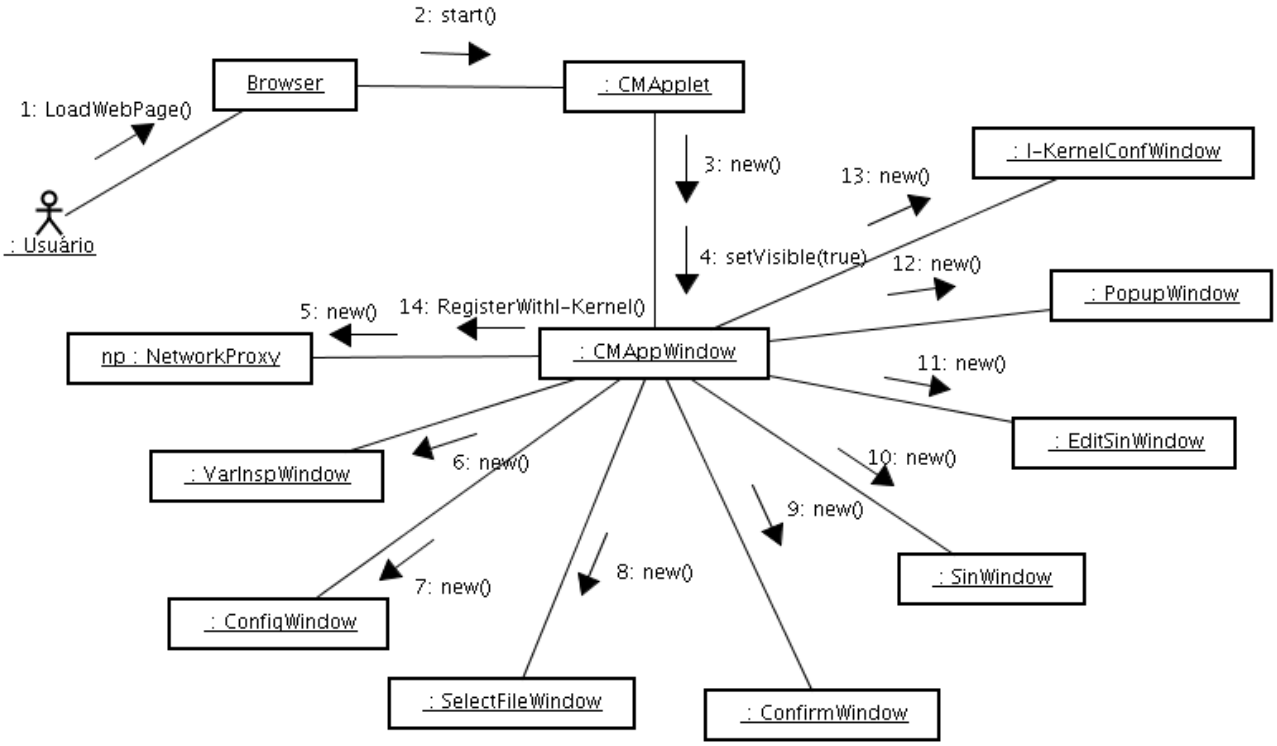
Caso de Uso 3: Verificação de Alarmes e Alertas



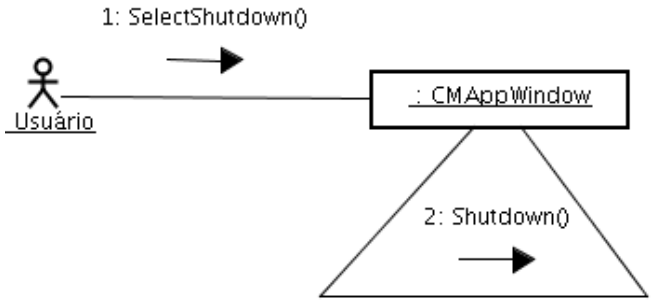
Caso de Uso 4: Shutdown do I-Kernel



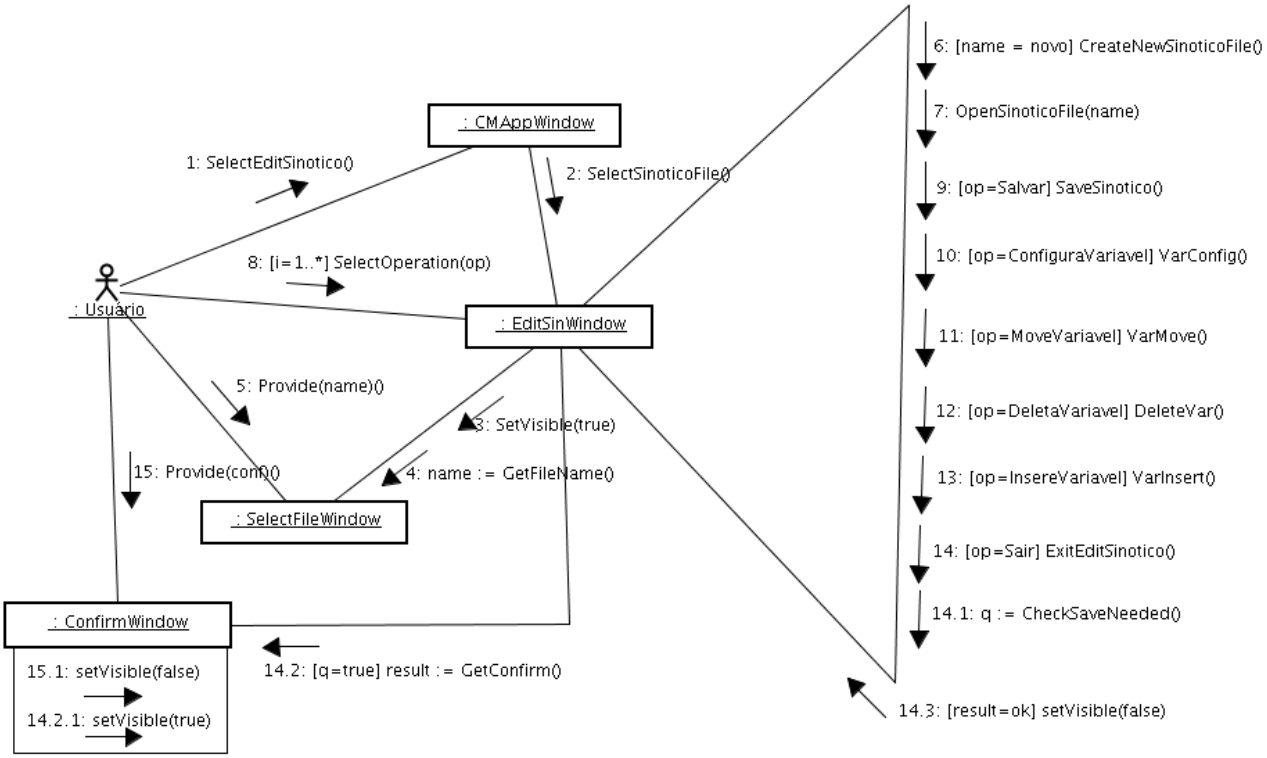
Caso de Uso 5: Iniciação da Ferramenta de C & M



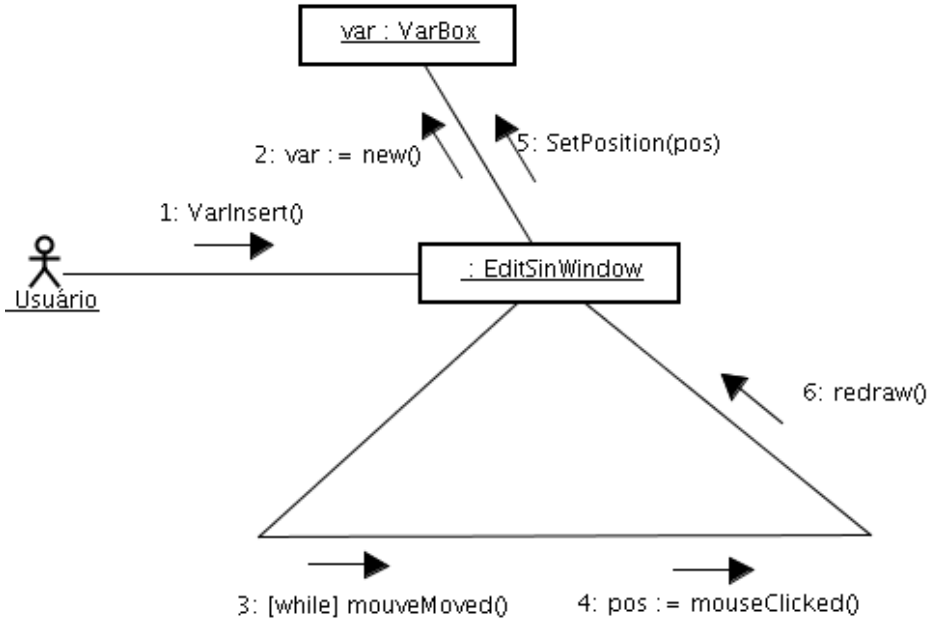
Caso de Uso 6: Shutdown da Ferramenta de C & M



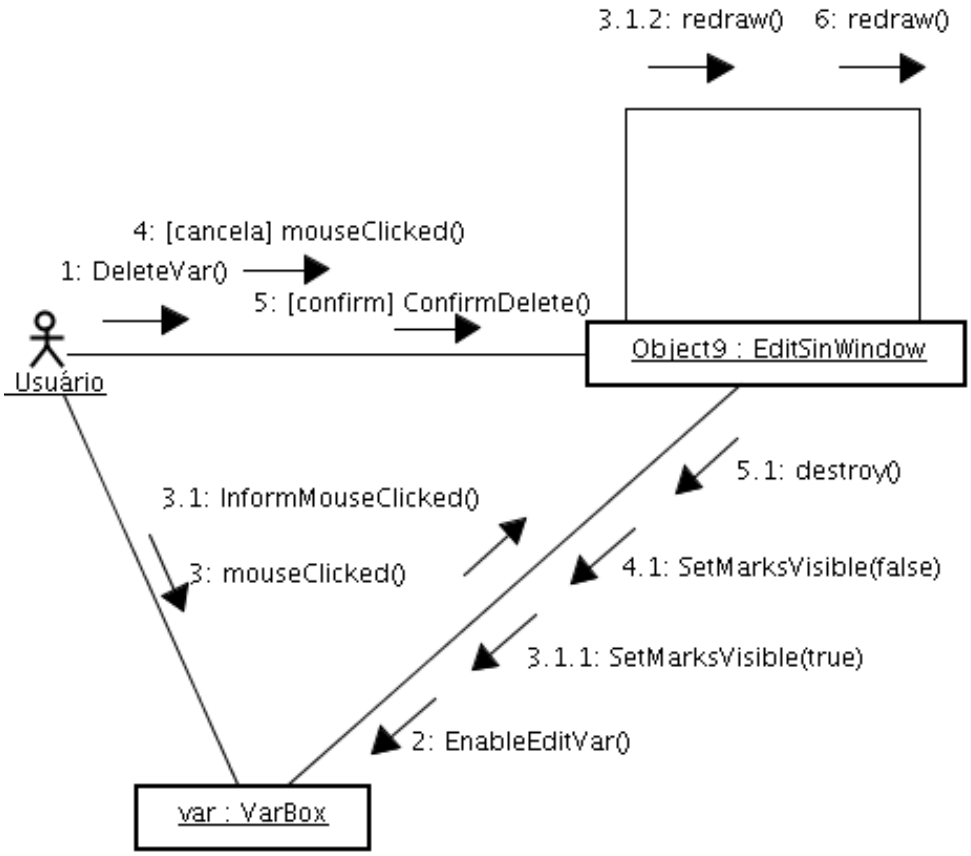
Caso de Uso 7: Edição de Sinótico



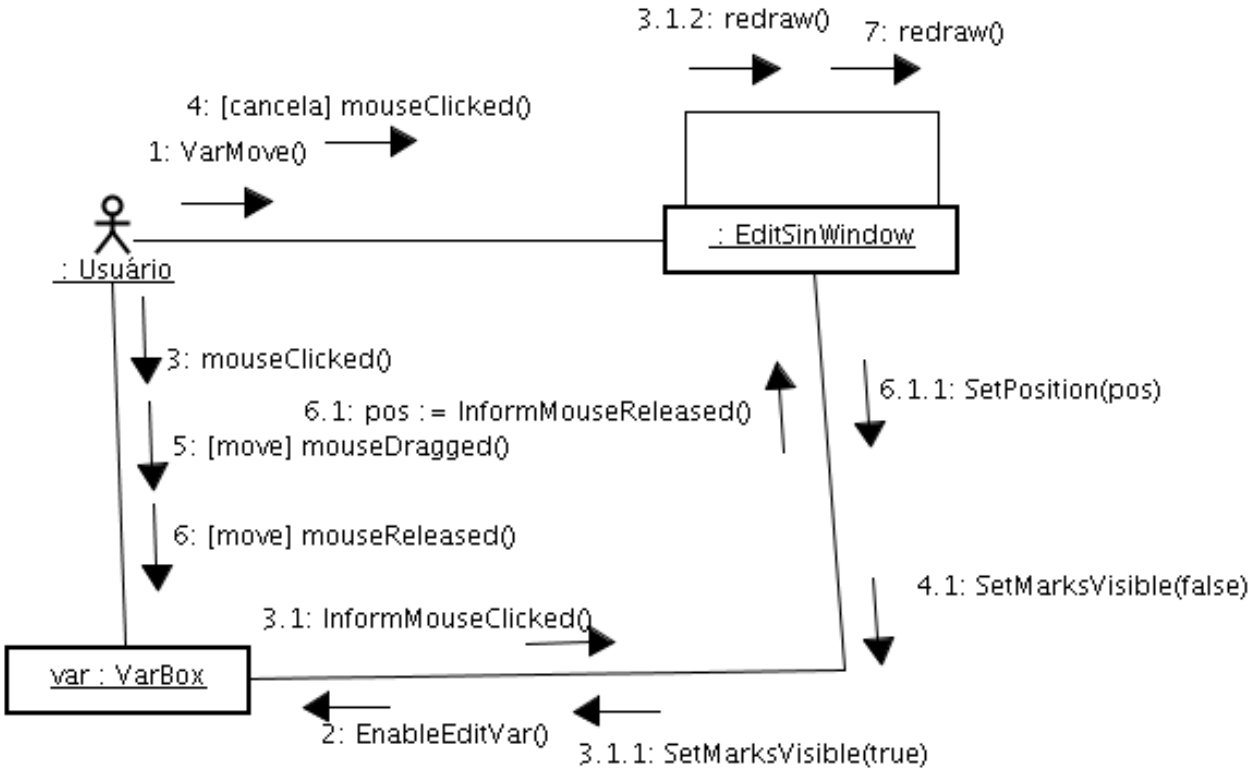
Caso de Uso 8: Inserção de Variável



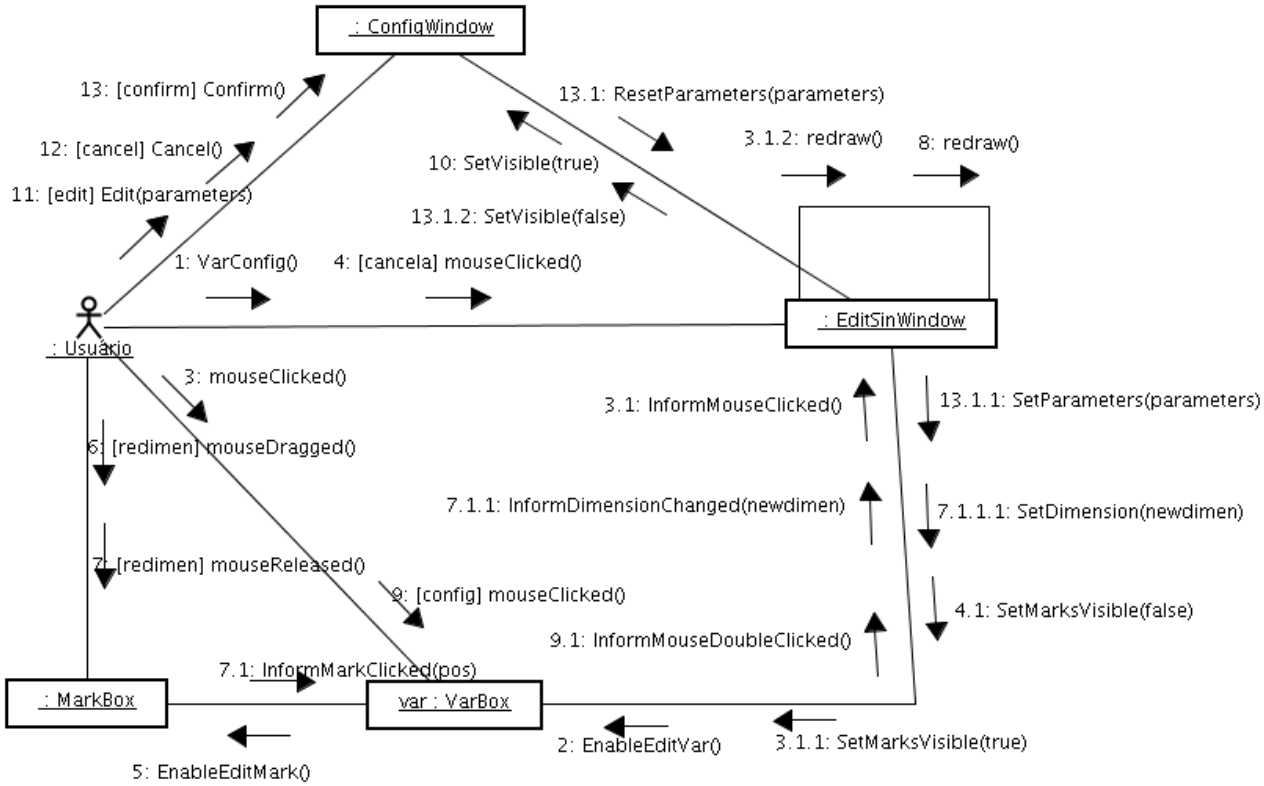
Caso de Uso 9: Deleção de Variável



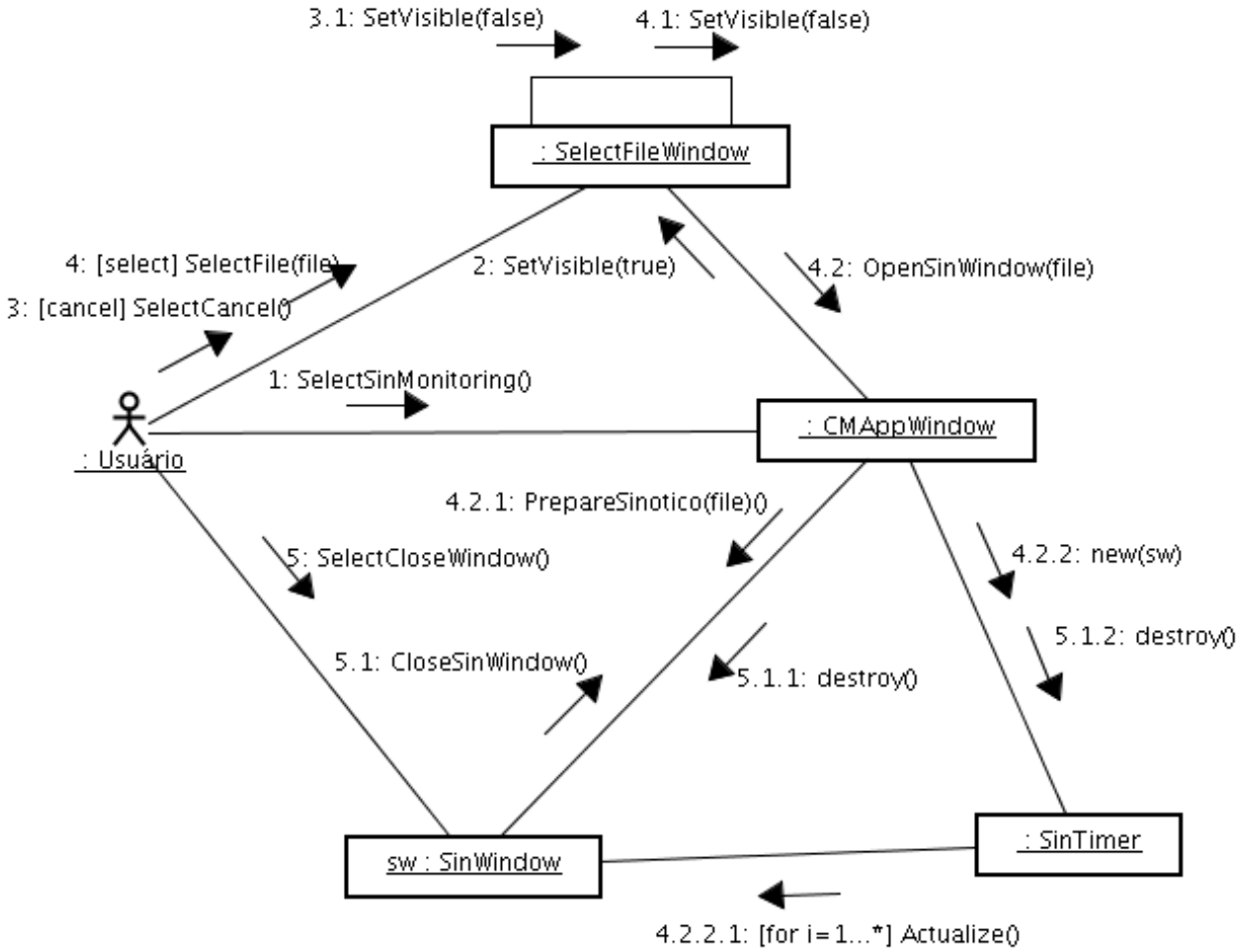
Caso de Uso 10: Movimentação de Variável



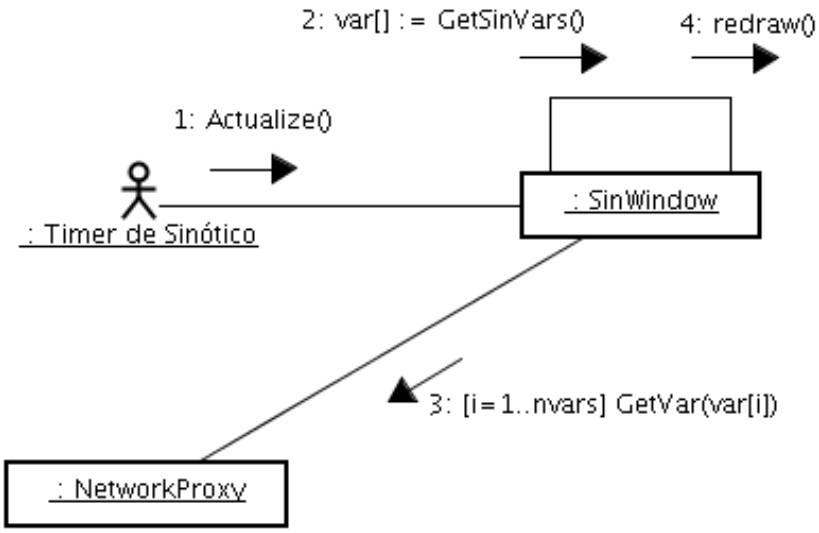
Caso de Uso 11: Configuração de Variável



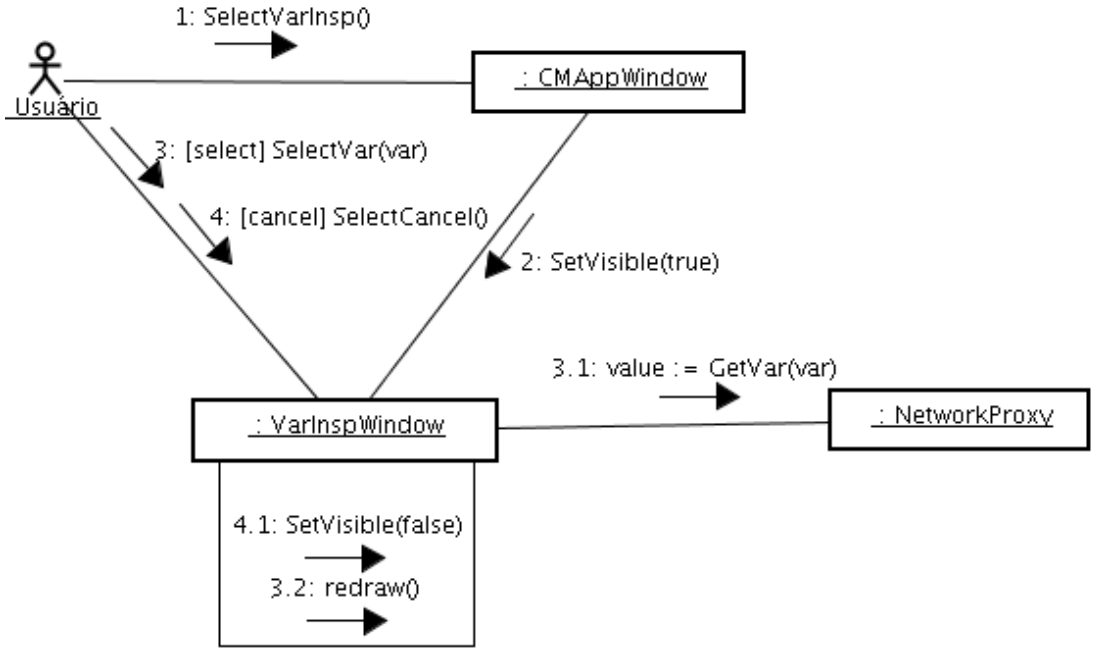
Caso de Uso 12: Monitoramento de Sinótico



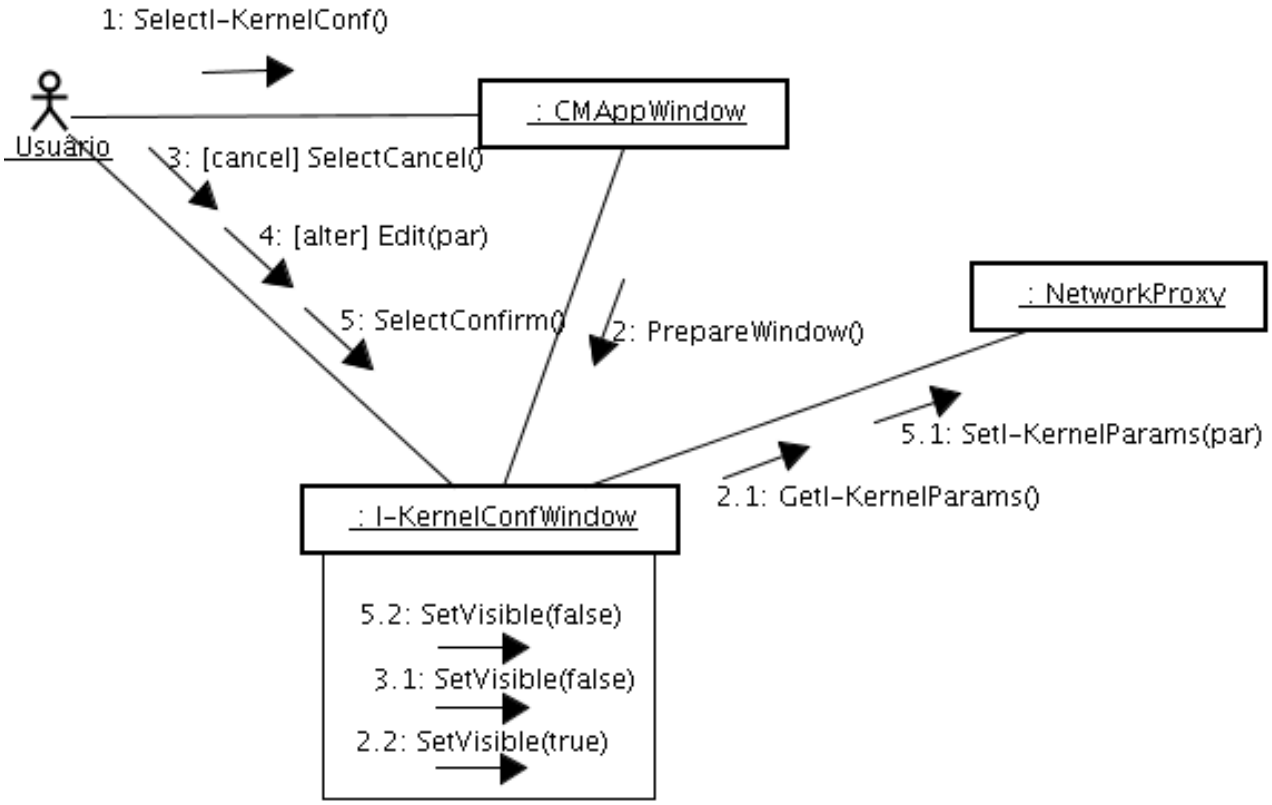
Caso de Uso 13: Atualização de Sinótico



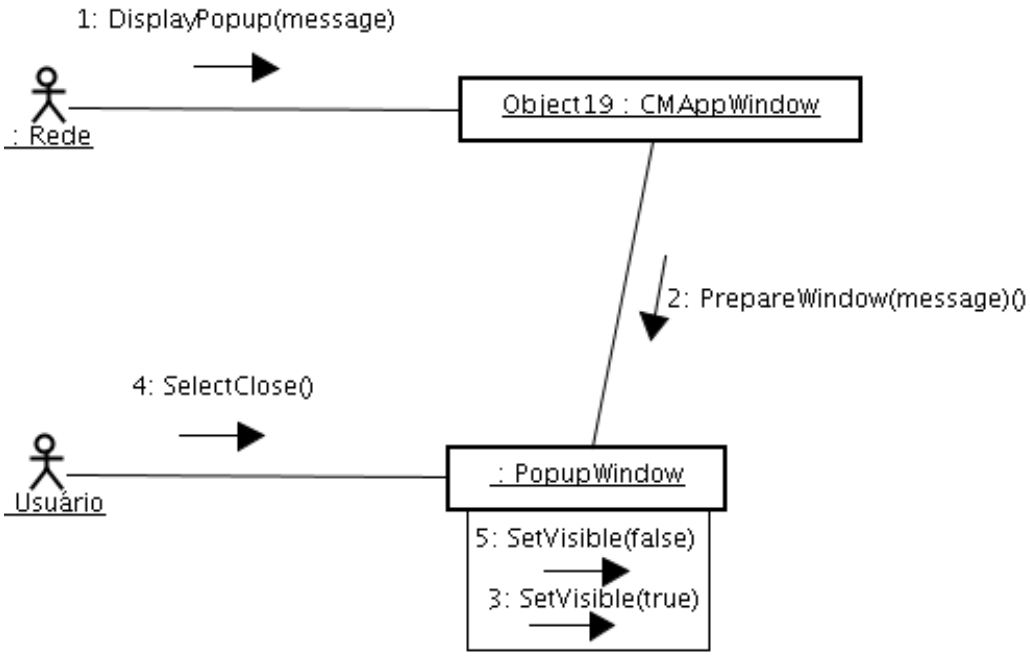
Caso de Uso 14: Inspeção de Variáveis



Caso de Uso 15: Configuração do I-Kernel



Caso de Uso 16: Solicitação de Popup de Alerta



Referências

(Alvares, 2006) Alberto Alvares - SIMPREBAL: Metodologia Do Sistema De Manutenção Preditiva Da Usina De Balbina Baseado Nos Dados Monitorados Do Sistema De Supervisão E Controle Smar E Rockwell, Relatório Técnico de Pesquisa, UNB, Julho de 2006.

(Jacobson et. al. 1999) Ivar Jacobson, Grady Booch, James Rumbaugh - The Unified Software Development Process – Addisson Wesley, 1999.

(Gudwin, 2006) Especificação do Sistema - Sistema I-Kernel: Um Kernel Inteligente para o SIMPREBAL - Sistema de Manutenção Preditiva de Balbina, Relatório Técnico, Novembro de 2006.

(Sommerville, 2003) Ian Sommerville - Engenharia de Software, 6a. edição, Addison-Wesley/Pearson Education.

CONTROLADOR NEURAL NA PLANTA SMAR PD-3

Edgar Amaya Simeón

Universidade de Brasília (UnB)
Departamento de Engenharia Mecânica
Grupo de Automação e Controle (GRACO)
Brasília, DF, Brasil
Emails: eamaya@unb.br

Abstract — In this paper, a neural controller was implemented in MATLAB® 7, controlling temperature on mix water tank acting on a Smar PD-3 didactic plant by means of OPC technology. The controller implementation was using perceptron multi layer network trained using o PID controller with back propagation algorithm performed.

Keywords — Control systems, Neural Networks, Fieldbus Foundation.

Resumo — Neste trabalho foi implementado um controlador “Neural” em MATLAB® 7, controlando a temperatura no tanque de mistura da agua atuando na planta didática PD3 da Smar pelo uso da tecnologia OPC. A implementação do controlador foi usando uma rede perceptron multicamada treinada como os valores do controlador PID usando o algoritmo backpropagation.

1 Introdução

As Redes Neurais Artificiais são importantes ferramentas para aplicação em controle de plantas não-lineares dadas suas características de mapeadores universais e pela capacidade de aprender por treinamento. Aplicações envolvendo controle adaptativo através da linearização do sinal de controle para sistemas SISO (Chen & Khalil, 1995) e controle por Modelo Interno (Hunt & Sbarbaro, 1991) são algumas das muitas utilizações de redes neurais em controle de processos.

A necessidade de se controlar sistemas e processos existe desde tempos remotos, enquanto que as metodologias para melhorar o treinamento de uma rede neural artificial são recentes (Bakshi, 1993),(Qin, 1992)(Leonard, 1992). A análise e projeto de controladores neurais não são triviais, devido as Redes Neurais Artificiais não serem modeladas diretamente através de equações diferenciais, o que impossibilita a utilização de métodos clássicos de análise e projeto.

Em geral é difícil obter expressões analíticas gerais devido à variedade de estruturas de redes e funções de ativação que podem ser usadas. Os modos mais freqüentes de usar redes neurais em sistemas de controle são: obtenção do modelo inverso do sistema através de aprendizado; mapeamento do omportamento de um controlador conhecido na rede; aprendizado de características de adaptação ou de modelos de referência. Assim, o treinamento de redes neurais on-line, até pouco tempo atrás, exigia uma velocidade considerável de processamento e requeria métodos de convergência bastante eficazes, o que demanda grande processamento computacional.

Com os avanços tecnológicos e a criação de novos métodos de otimização, tornou-se possível o treinamento on-line, sem pré-processamento ou modelo

do sistema, com isso, surgiu uma nova área para aplicações das redes neurais, fazendo com que uma grande quantidade de aplicações práticas tornasse possível.
As Redes Neural Artificiais (RNAs) são um aproximador genérico não-linear. Nem a estrutura e nem os parâmetros são necessários. Se os dados de treinamento são ricos em informação, o número de neurônios é suficiente, e temos um algoritmo de treinamento adequado, podemos esperar uma solução plausível. Uma solução ótima global raramente é necessária para problemas práticos. Quando uma faixa ampla de operação é esperada é mais difícil que uma única RNA tenha bom desempenho em toda esta faixa. Neste artigo propomos uma RNA treinada para uma faixa de operação.

Rede Neural Artificial
No modelo do neurônio artificial, Fig. 1, cada entrada recebe um estimulo, que é ponderado pelos pesos sinápticos. Todas as entradas são somadas, gerando uma resposta, a qual é posteriormente modulada por uma função matemática (função de ativação)

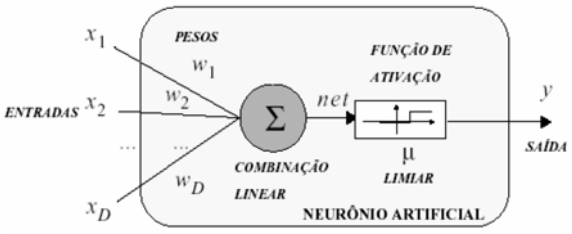


Fig. 1 - Modelo de um neurônio de McCulloch e Pitts

As Redes Neurais são compostas de vários elementos conectados entre si de alguma forma, possibilitando a sua operação em paralelo(Gabriel, 1996). Amaneira como os neurônios são organizados define a arquitetura da rede. A arquitetura utilizada é do tipo perceptron de múltiplas camadas descrito na Fig. 2.
A operação realizada pelo neurônio artificial sobre um vetor de entradas [*x0 ... xn*]*T* pode ser expressa pela equação:

$$y_s = f\left(\sum_{k=0}^{n} x_k w_k\right)$$

O vetor de pesos [*w0 ... wn*]*T* multiplica ponto a ponto cada uma das entradas [*x0 ... xn*]*T* e *f*(.) representa uma função de ativação do neurônio. As diversas camadas ou níveis podem ser obtidos interligando os neurônios de uma camada aos neurônios da camada seguinte.

A arquitetura neural utilizada está disposta da seguinte maneira: 3 neurônios na primeira camada, 20 neurônios na escondida e 1 na ultima camada. A função de ativação da camada escondida é tangente sigmoidal e da camada de saida Logaritmo Sigmoidal.

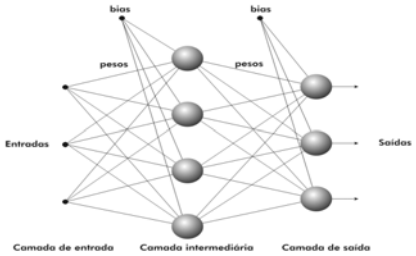


Fig. 2. Estrutura de um perceptron de múltiplas camadas.

Sitema de Control Neural

Treinamento do Controlador Neural
Para o treinamento do controlador neural utilizo-se os valores obtidos do controlador PID segun a Fig. 3

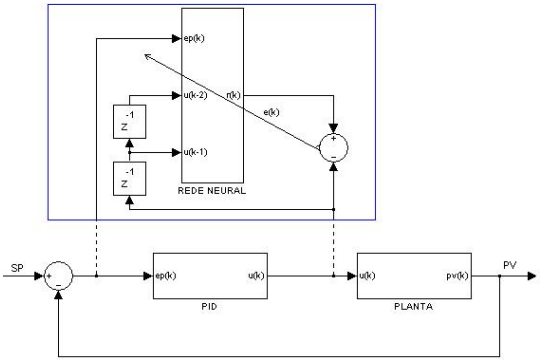


Fig. 3. Modelo de Treinamento do Controlador Neural.

O método *"Backpropagation"*, que foi a técnica utilizada neste trabalho, pertence à família dos métodos *Least Mean Square* (LMS), nos quais se procuram o valor mínimo da função erro quadrático, que é baseado no método do gradiente. O algoritmo de treinamento basicamente consiste em corrigir os pesos sinápticos, com base no erro ocorrido em cada saída da rede neural. A correção dos pesos é feita utilizando o método dos mínimos médios quadráticos, visando encontrar um valor para os pesos que minimize o erro na saída da rede.

Neuro-controlador

O neuro-controlador aprende inicialmente com os valores coletados da acão do controlador PID, fazendo-o acompanhar um sinal de controle u(k). Após o treinamento, espera-se que o Neuro-Controlador esteja pronto para gerar os sinais de controle que serão colocados a entrada da planta para fazer controlar. A estrutura do neuro-controlador deve calcular o sinal de controle dentro do período de amostragem do processo, que também inclui a leitura do sensor e a escrita do sinal de controle.

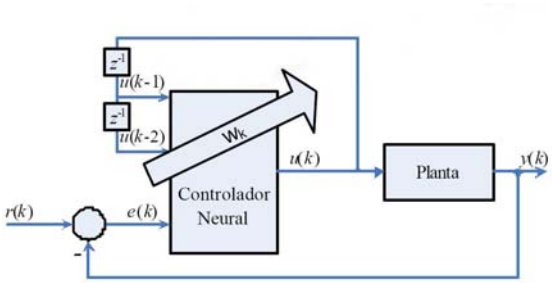


Fig. 3. Controlador Neural.

Sistema Supervisório

Tuvo-se a necessidade de centralizar as informações com o objetivo de se ter acesso ao máximo de informações no menor tempo possível para promover a interface homem/ máquina. O sistema supervisório utilizado foi feito no Matlab7 mostrado na Fig. 4 usando o GUI e dois active x para a vizualição das grandezas do proceso

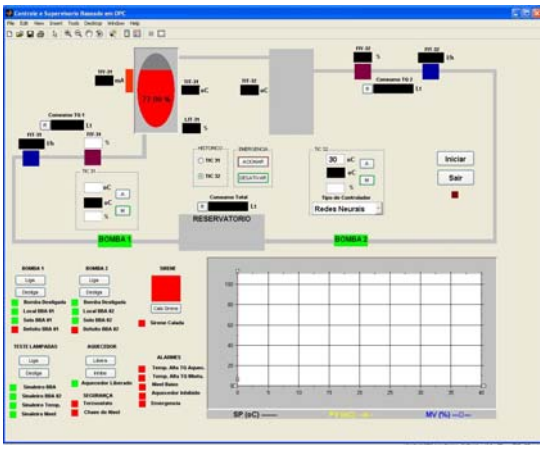


Fig. 4. Supervisorio de control OPC

Comunicação de Dados

O protocolo de comunicação OPC(*Ole for Process Control*) foi utilizado para a comunicação entre o supervisório e o computador denominado "assetview" de acordo com a Fig. 5. A comunicação ocorre da forma cliente-servidor onde o computador denominado "assetview" contém os servidores OPC da Samr (DfiOleServer.0 e DF65Server.1). No

cliente temos o supervisorio feito em matlab 7 denominado ControlOPC.

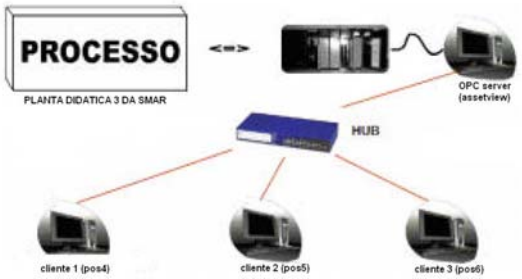


Fig. 5. Arquitetura Cliente Servidor OPC

Resultados

Os resultados obtidos para o sistema de controle de temperatura da agua no tanque de mistura da PD3 da Smar utilizado neste trabalho são mostrados a Continuação.

Apois de treinada a rede neural o resulado de foi que em estado estavel tem, erro maximo = 1.8469% , erro minimo = -1.7032 % como mostra-se na Fig. 6

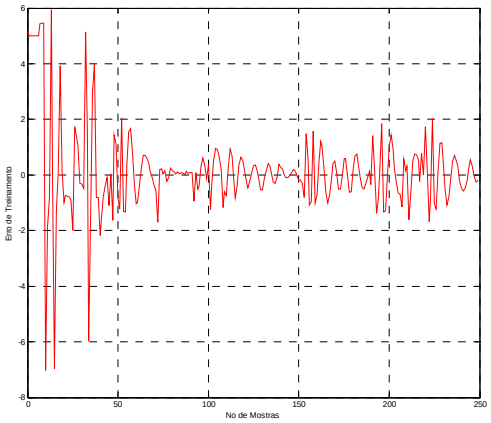


Fig. 6. Grafico do Erro de Treinamento

O controlador Neural desempeñose de forma similar a controlador PID com as condições de set point do tanque de aquecimento de 38ºC y o set point do tanque de mistura de 32ºC como mostrase na Fig. 7.

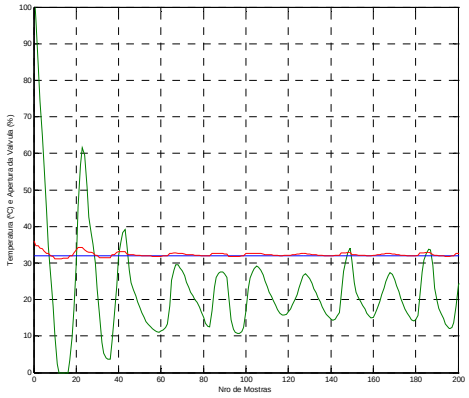


Fig. 7. Resposta do Sistema ao Controlador neural

O Control faz que o Sistema tenha um erro em estado estavel maximo de 0.8895 °C e um erro minimo -0.2178°C

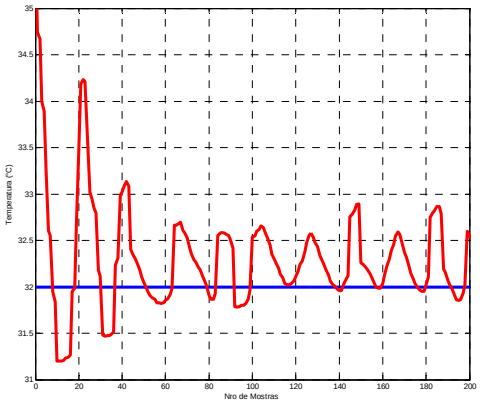


Fig. 7. Grafico em estado estavel

Conclusão

O esquema Neural apresentado neste trabalho baseou-se em (Gabriel, 1996). Os resultados em tempo real obtidos com o sistema de controle neural de Temperatua apresentaram um bom desempenho, similar a o controladoe PID isto porque o controlador neural foi treinada tendo como referencia o Controlador PID

REFERÊNCIAS BIBLIOGRÁFICAS

SMAR (2004). Manual de operação Planta Didática III. 127 p.

SMAR (2001). Equipamentos de Campo série 302 Foundation. Manual de instalação, operação e manutenção. 42 p.

OPC Foundation (1998). OPC Overview, disponível em http://www.opcfoundation.org/ (10 mar.2005)

MATLAB (2006). OPC Toolbox for use with MATLAB® and Simulink®. User´s Guide. v.2. Natick: The Mathworks Inc, 373 p.

Smar Equipamentos Industriais Ltda, Janeiro de2004, "Manual de Instruções dos Blocos Funcionais

Smar Equipamentos Industriais Ltda, Janeiro 2004, "The Fieldbus Reference Book"

Fieldbus FOUNDATIONä, June, 1997, "FOUNDATION™ Specification - Communication Profile"

Controle de Temperatura com Rede Neural na Planta Didática da Smar

Tópicos Avançados em Sistemas Mecatrônicos I

Edgar Amaya Simeón

Conteúdo

- 1. Objetivos
- 2. Indrodução
- 3. Sistema de Controle neural
- 4. Supervisorio
- 5. Resultados esperados
- 6. Referências;

1. Objetivos

- Mostrar os conceitos para a implementação de redes neurais.
- Desenvolver um algortim de treinamento da Rede Neural usando o metodo BackPropagation do Erro usando os dados obtidos do controlador PID
- Fazer o controle da temperatura do tanque de mistura utilizando a rede neural treinada.
- Desenhar um ambiente gráfico para visualização das grandezas obtidas através de OPC da planta Smar e os parâmetros do controlador neural usando o GUI de Maltab 7.

2. Introdução

- No tanque de mistura, a água quente proveniente do tanque de aquecimento é misturada com água fria para que esta se aqueça. A finalidade deste controle é manter a temperatura da água no tanque de mistura respondendo às variações da demanda da água atraves das três valvulas, a ação da válvula de água fria quando a temperatura for diferente da solicitada.

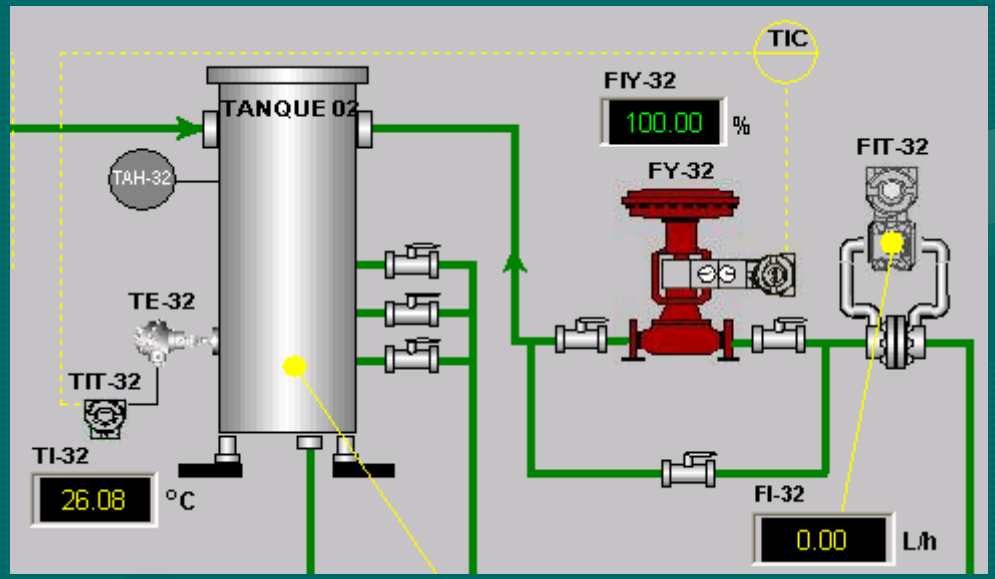


Fig. 1 Tanque 2 ou tanque de mistura

2. Introdução

2.1 TAGs que serão usados no Controle Neural

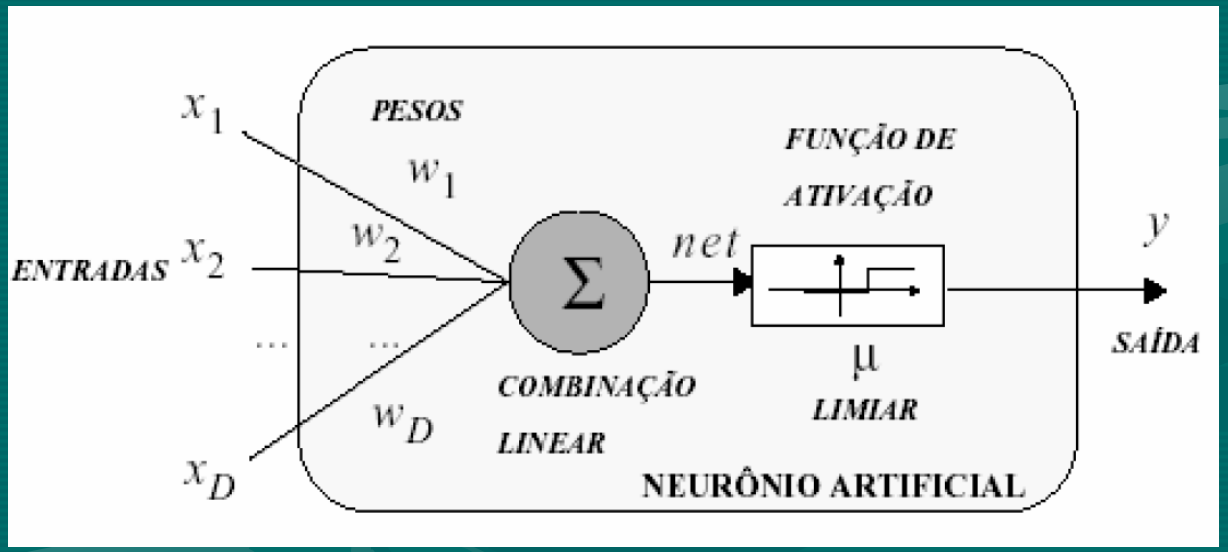
Os TAG dos dados precisados para fazer o cntrole neural obtuvieron com os seguintes TAG
Do servidor OPC Smar.DfiOleServer.0

TAGs de Entrada Fieldbus		
ID	**TAG OPC**	**Descrição**
TIT-32	TIT-32_AI1.OUT.VALUE	Temp. tanque de mistura
FY-32	FY-32_AO1.OUT.VALUE	Válvula na Entrada de água fria

3. Sistema de Controle neural

2.1 Rede Neural Artificial

No modelo do neurônio artificial, cada entrada recebe um estimulo, que é ponderado pelos pesos sinápticos. Todas as entradas são somadas, gerando
uma resposta, a qual é posteriormente modulada por uma função matemática (função de ativação)



Modelo de um neurônio de McCulloch e Pitts

A operação realizada pelo neurônio artificial sobre um vetor de entradas [*x0 ... xn*]*T* pode ser expressa pela equação:

$$y_s = f\left(\sum_{k=0}^{n} x_k w_k\right)$$

3. Sistema de Controle neural

A arquitetura neural utilizada está disposta da seguinte maneira: 3 neurônios na primeira camada, 20 neurônios na escondida e 1 na ultima camada. A função de ativação da camada escondida é tangente sigmoidal e da camada de saida Logaritmo Sigmoidal.

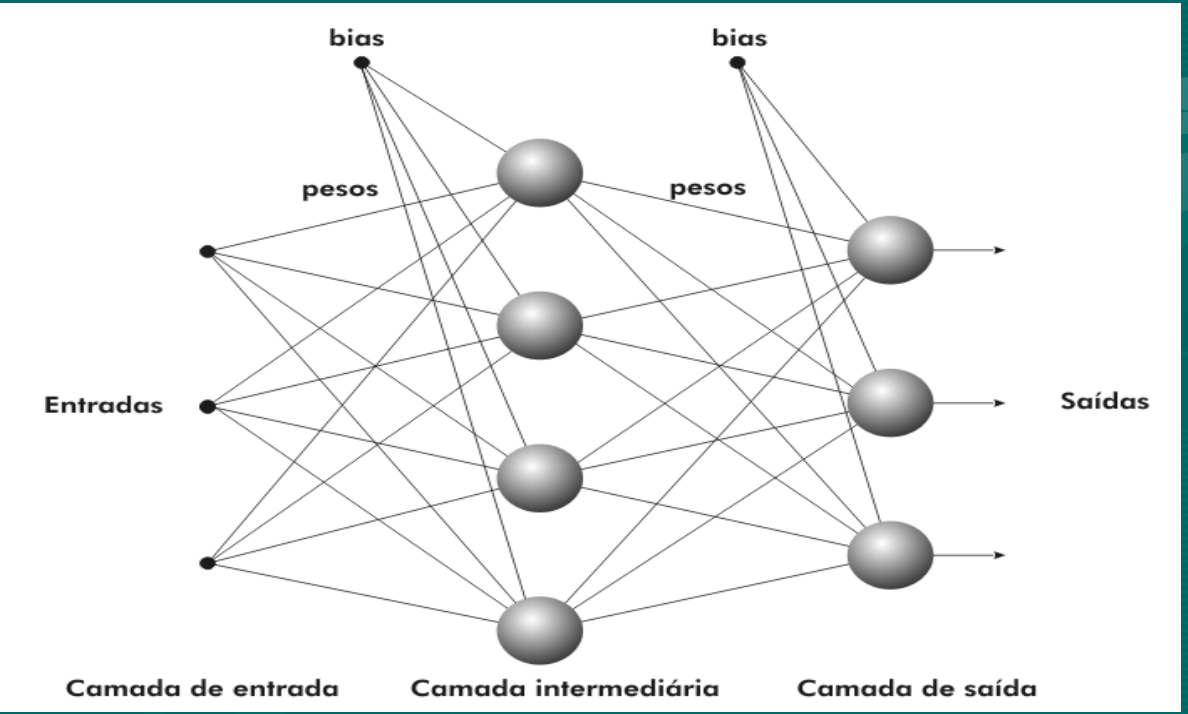
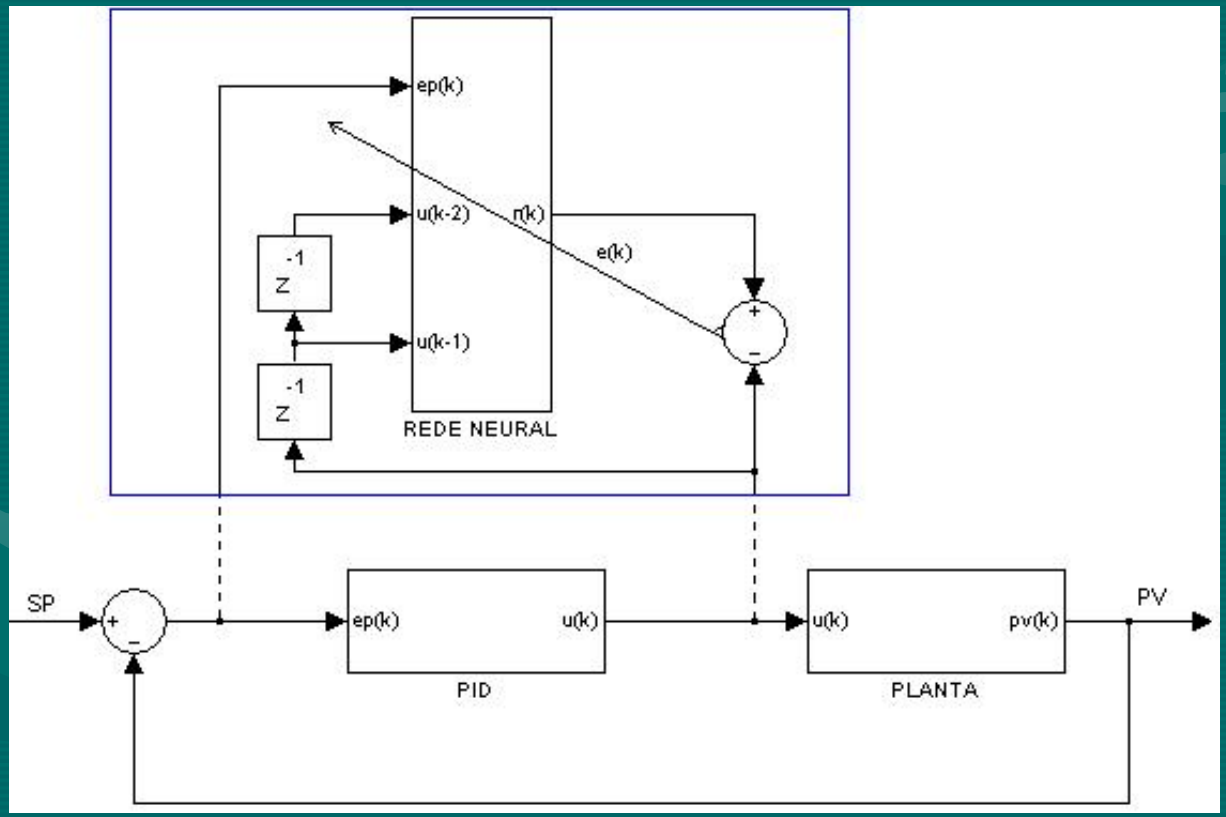


Fig. 2. Estrutura de um perceptron de múltiplas camadas.

3. Sistema de Controle neural

Treinamento do Controlador Neural

Para o treinamento do controlador neural utilizo-se os valores obtidos do controlador PID segun a seguinte figura

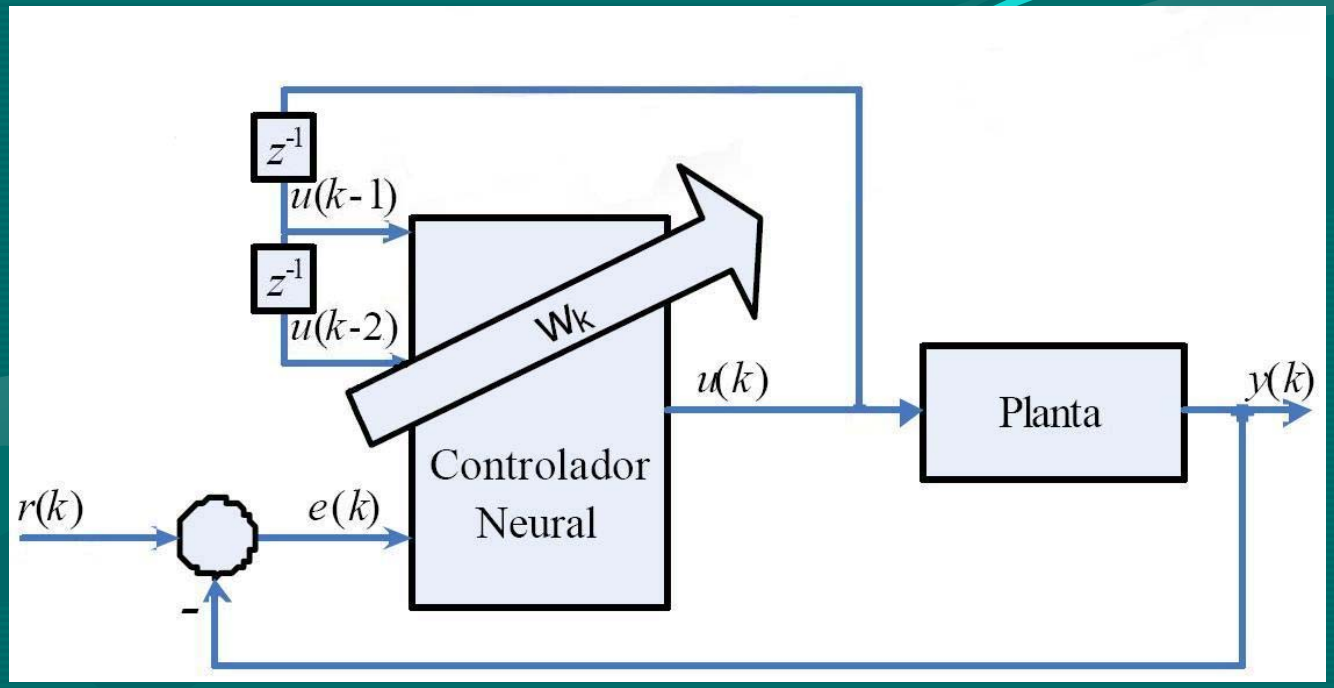


Modelo de Treinamento do Controlador Neural

3. Sistema de Controle neural

Neuro-controlador

Após o treinamento, espera-se que o Neuro-Controlador esteja pronto para gerar os sinais de controle que serão colocados a entrada da planta para fazer controlar. A estrutura do neuro-controlador deve calcular o sinal de controle dentro do período de amostragem do processo, que também inclui a leitura do sensor e a
escrita do sinal de controle.

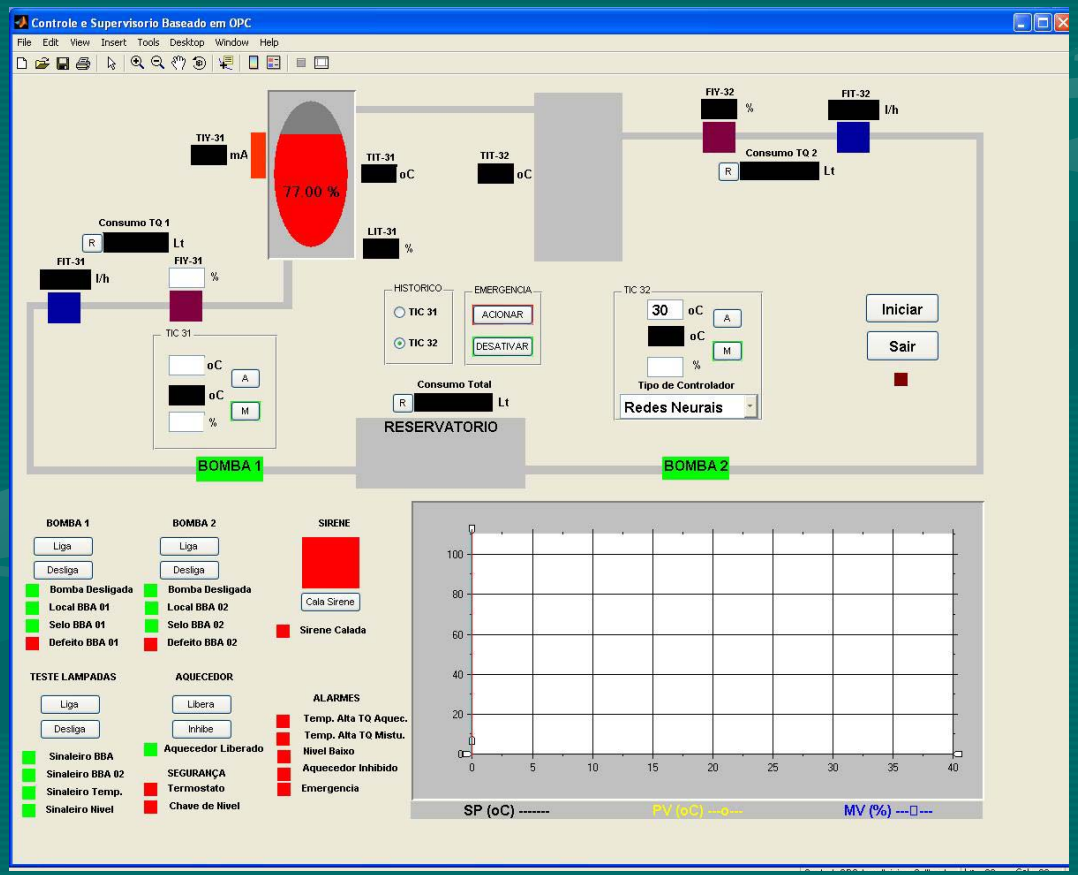


Controlador Neural.

4. Supervisorio

Sistema Supervisório

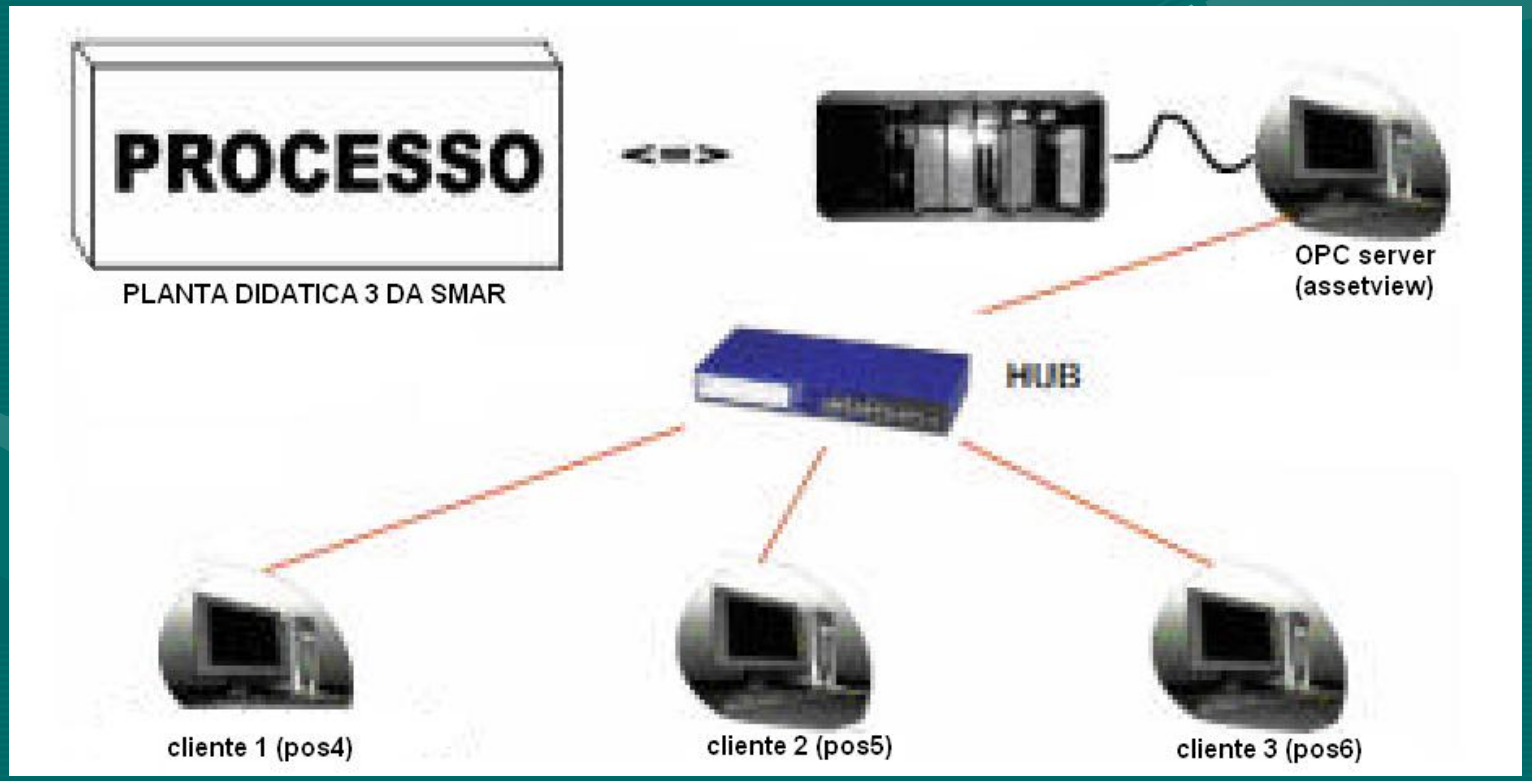
Tuvo-se a necessidade de centralizar as informações com o objetivo de se ter acesso ao máximo de informações no menor tempo possível para
promover a interface homem/máquina. O sistema supervisório utilizado foi feito no Matlab7 usando o GUI e dois active x para a vizualição das grandezas do proceso



4. Supervisorio

Comunicação de Dados

O protocolo de comunicação OPC(*Ole for Process Control*) foi utilizado para a comunicação entre o supervisório e o computador denominado "assetview" de acordo com a Fig. 5. A comunicação ocorre da forma cliente-servidor onde o computador denominado "assetview" contém os servidores OPC (Smar.DfiOleServer.0 e Smar.DF65Server.1). No cliente temos o supervisorio feito em matlab 7 denominado ControlOPC.



5. Resultados esperados

Treinamento

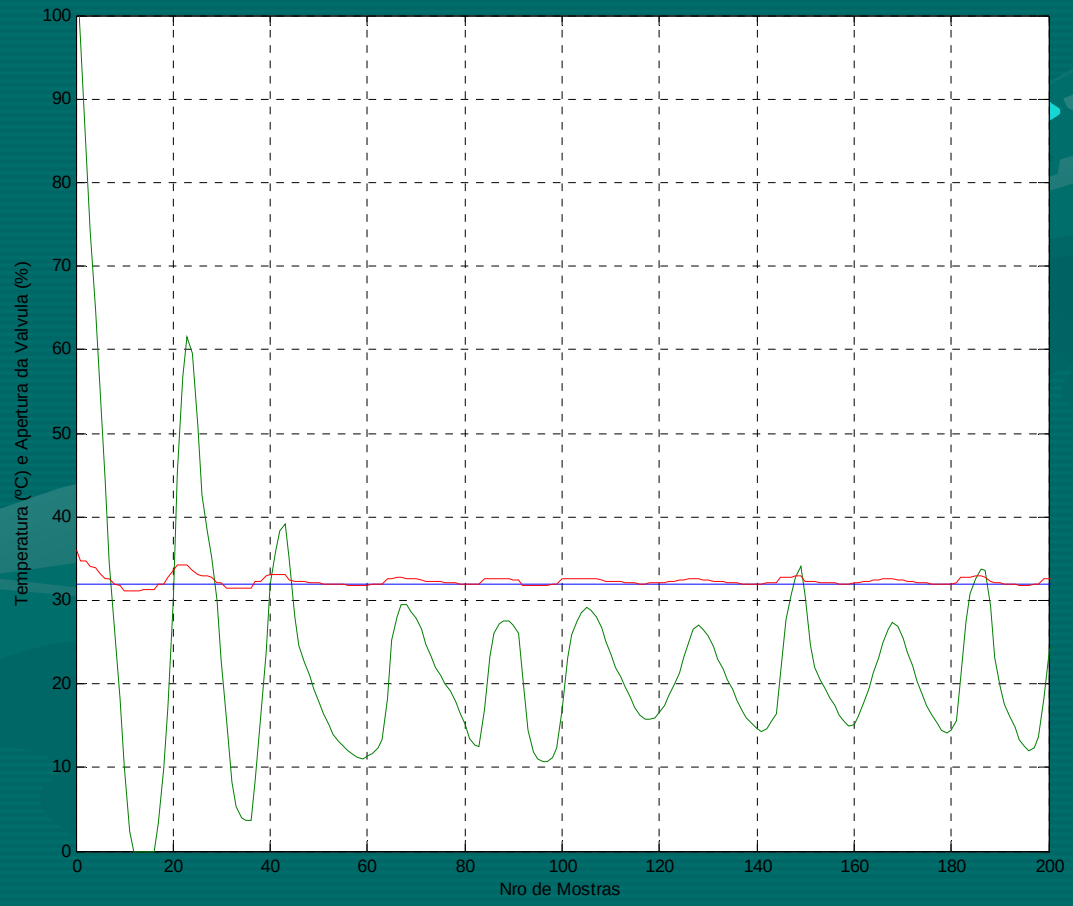
Os resultados obtidos para o sistema de controle de temperatura da agua no tanque de mistura da PD3 da Smar utilizado neste trabalho são mostrados a continuação.

Apois de treinada a rede neural o resulado de foi que em estado estavel tem, erro maximo = 1.8469% , erro minimo = -1.7032 % como mostra-se na figura

5. Resultados esperados

Controlador Neural

O controlador Neural desempeñose de forma similar a controlador PID com as condições de set point do tanque de aquecimento de 38ºC y o set point do tanque de mistura de 32ºC observandose



5. Resultados esperados

Erro em estado estavel.

O Control faz que o Sistema tenha um erro em estado estavel maximo de 0.8895 °C e um erro minimo -0.2178°C

6. Referências

- MATLAB (2006). **OPC Toolbox for use with MATLAB® and Simulink®.** User´s Guide. v.2. Natick: The Mathworks Inc, 373 p.
- JANG, J.S.R. GULLEY, N. (1999). **MATLAB® Neural Network toolbox**. User´s Guide. v.1. Natick: The Mathworks Inc, 235 p.
- SMAR (2004). Manual de operação Planta Didática III. 127 p.
- SMAR (2001). **Equipamentos de Campo série 302 Foundation.** Manual de instalação, operação e manutenção. 42 p.
- SMAR (2005). **Manual de instruções dos blocos funcionais Fieldbus Foundation**. 334 p.

SISTEMA ESPECIALISTA, BASEADO EM REGRAS DE PRODUÇÃO, PARA AVALIAÇÃO DE SAÚDE DOS EQUIPAMENTOS DA PLANTA DIDÁTICA III DA SAMAR

Rosimarci Pacheco Tonaco[1] , Alberto José Alvares[1]

1- Grupo de Automação e Controle - GRACO, Departamento de Engenharia Mecânica, Faculdade de Tecnologia, UnB – Universidade de Brasília. Brasília, DF.

1. Introdução

O presente trabalho tem o objetivo de apresentar um Sistema Especialista que realiza a avaliação de saúde dos equipamentos da Planta Didática III da Smar. O objetivo do sistema é monitorar os equipamentos, que estão presentes na planta, com o intuito de avaliar a saúde dos mesmos.

Sistemas Especialistas são sistemas que solucionam problemas que são resolvíveis apenas por pessoas especialistas (que acumularam conhecimento exigido). O conhecimento que será adquirido pelo sistema é a fonte de raciocínio do mesmo. Para representar esse conhecimento a literatura oferece várias linguagens de representação do conhecimento. O presente trabalho utiliza as regras de produção como linguagem de representação do conhecimento, esta é de fácil manipulação e entendimento. Essa linguagem foi escolhida, visto que o Jess (*Java Expert System Shell*) foi utilizado no desenvolvimento do mesmo, e este trabalha com regras de produção para representar o conhecimento que será manipulado.

O Jess é um biblioteca de funções, que pode ser embutida no Java, desenvolvida para a implementação de sistemas especialistas baseados em regras. Um sistema baseado em regras possui uma base de regras, na qual o conhecimento é armazenado e a cada problema novo (ou entrada) utiliza a base de regras para resolver a nova situação.

O Jess permite o desenvolvimento de sistemas especialistas utilizando programação declarativa (ou embutindo suas funções no Java). Esta é a forma mais próxima da linguagem natural para a implementação de sistemas, e é utilizada para solucionar problemas que envolvem controle, diagnóstico, predição, classificação e reconhecimento de padrões.

A arquitetura de um sistema baseado em regras típico possui os seguintes módulos de trabalho, esta pode ser visualizada na figura 1:

- máquina de inferência;
- base de regras, e;
- memória de trabalho.

A máquina de inferência, internamente, consiste de:

- teste padrão;
- agenda, e;
- máquina de execução.

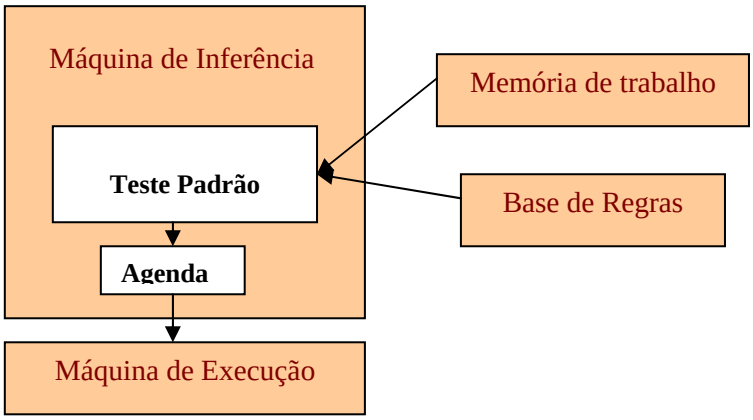


Figura 1: Arquitetura de um sistema baseado em regras.

A figura 1 mostra a arquitetura de um sistema especialista, bem como sua forma de trabalhar. A máquina de inferência é responsável por receber da memória de trabalho, as regras da base de regras e as entradas. Em seguida, a máquina de inferência, realiza o teste das entradas de acordo com as regras fornecidas e manda o resultado para a agenda que enviará o mesmo para a máquina de execução, que realizará o raciocínio para resolver o problema em questão.

2. A Planta Didática III da Smar

O objetivo da Planta Didática SMAR é demonstrar didaticamente a operação das diversas malhas de controle utilizando equipamentos e ferramentas de configuração, em software, desenvolvidos para aplicação em controle industrial. A Planta Didática SMAR é monitorada e operada de uma estação, constituída de um microcomputador do tipo PC e um software de supervisão, que efetua a aquisição de dados dos equipamentos e o apresenta por meio de animações de telas. Permite também atuar nos registros modificando valores internos dos equipamentos e nos modos operacionais das malhas de controle.

Nesta planta, o valor dos equipamentos pode estar armazenado em um banco de dados, ou podem ser acessados diretamente, via OPC. Os equipamentos de fabricação Smar, podem ser acessados tanto via OPC como via banco de dados, os equipamentos Rockwell, no entanto, só podem ser acessados via OPC.

O objetivo do sistema especialista é monitorar os equipamentos da planta, verificando a saúde de cada um deles com o intuito de alertar para possíveis falhas/faltas que poderão acontecer. Outro propósito é realizar a manutenção preditiva, que irá sugerir paradas de acordo com os problemas apresentados nos equipamentos. A figura 2 apresenta a planta didática III.

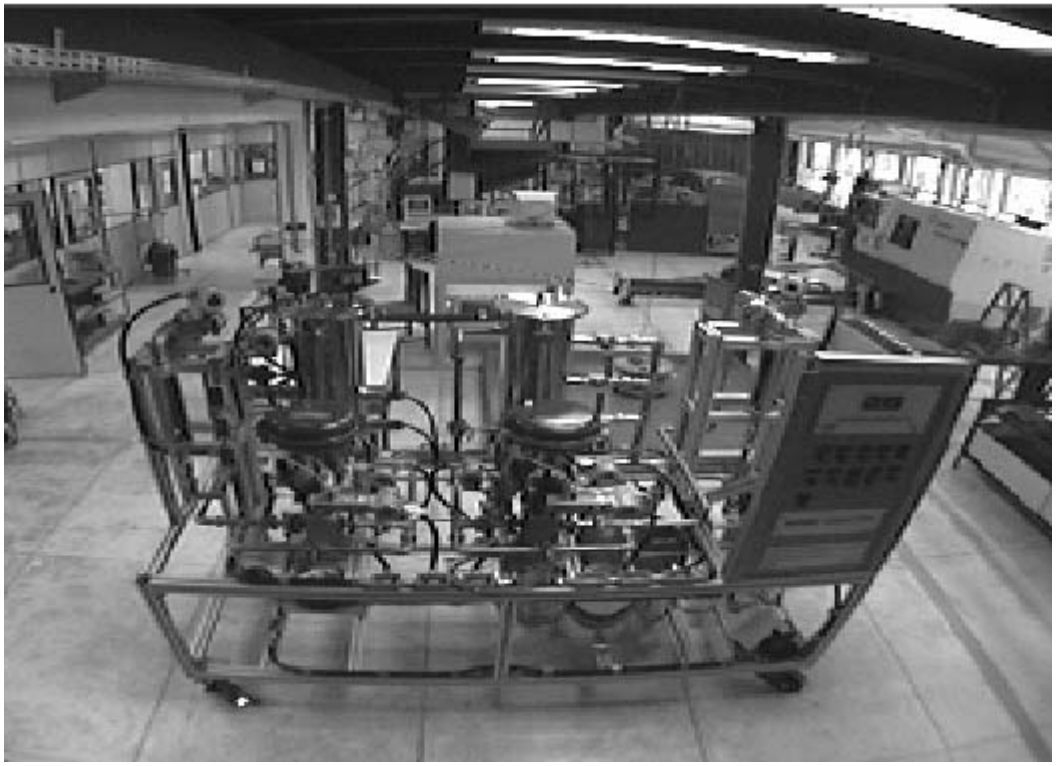


Figura 2: Planta Didática III da Smar.

A próxima seção apresenta uma descrição do sistema desenvolvido para ser aplicado na Planta Didática III.

3. Descrição do Sistema

Abaixo serão listadas as etapas de desenvolvimento do sistema:

- 1: Gerar base de regras que conterá informações sobre as condições de operação dos equipamentos, indicando condições normais de operação, condições de risco, etc;
- 2: Adquirir do Assetview os TAGs referentes aos equipamentos da Smar. Tais dados estão armazenados em SQL, para realizar essa aquisição será necessário criar uma comunicação JDBC, do Java, que fará as consultas necessárias ao banco de dados;
- 3: Desenvolver o módulo de raciocínio do sistema especialista para fornecer conhecimento útil sobre a saúde dos equipamentos. Esse módulo avaliará as informações (adquiridas via OPC e/ou SQL) considerando o conhecimento armazenado na base de regras. O módulo de raciocínio usará as funções do Jess, visto que esse é capaz de implementar o processo de inferência de um sistema especialista.

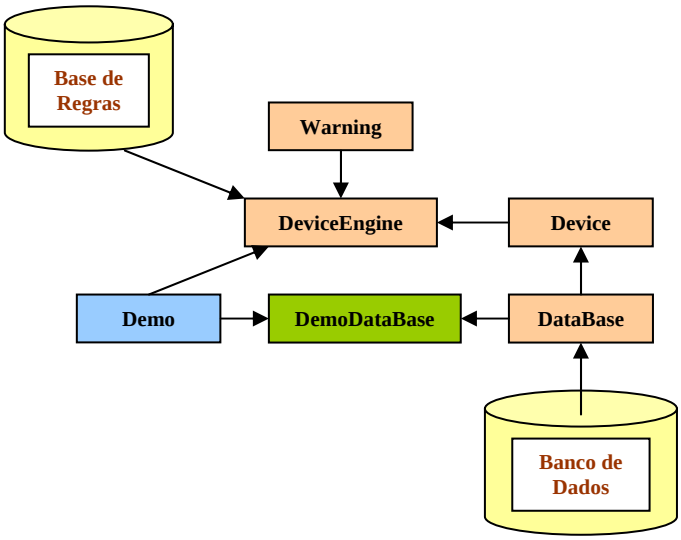


Figura 3: Arquitetura do Sistema Especialista.

A figura 3 apresenta graficamente o detalhamento do sistema. Em seguida será apresentado uma descrição textual das classes que foram desenvolvidas.

A base de regras, que será utilizada no sistema, deve ser gerada na sintaxe aceita pelo Jess, essa sintaxe é semelhante ao CLIPS (linguagem declarativa utilizada para o desenvolvimento de sistemas inteligentes). Assim, o Jess será capaz de carregar as regras e realizar o processo de inferência que fornecerá a solução mais provável. A base de regras é um arquivo do CLIPS .clp. Um exemplo de regras, feito em Jess, pode ser observado na figura 4.

```
import gov.sandia.jess.example.pricing.model.*)
(deftemplate Device    (declare (from-class Device)))
(deftemplate Warning   (declare (from-class Warning)))

;; Now define the rules themselves. Each rule matches a set
;; of conditions and actions.

(defrule quality-device-Good
    "Verifica a qualidade do device para fazer a avaliação de
saúde."
    (Device {quality > 0})
    =>
    (add (new Warning "Device operando na forma aceitável!")))

(defrule quality-device-Bad
    "Verifica a qualidade do device para fazer a avaliação de
saúde."
    (Device {quality < 1})
     =>
    (add (new Warning "A qualidade do device não está boa!")))
```

Figura 4: Regras de produção feitas para o Jess.

Para avaliar a saúde dos equipamentos (Smar) será necessário recuperar do banco de dados (AssetView/SQL) informações sobre as condições dos mesmos. A classe *Database* fará essa comunicação/recuperação com o banco de dados.

Uma vez que o processo tenha sido executado com sucesso, a classe *DeviceEngine* deverá requisitar da classe *DataBase* essas informações. Tais informações servirão de entrada para a classe *DeviceEngine*, que fará a avaliação da saúde dos equipamentos.

Para executar essa avaliação a classe *DeviceEngine* consultará a base de regras. A base de regras contém fatos a respeito das condições de trabalho dos equipamentos, indicando situações de risco e/ou condições normais (no presente trabalho foi utilizado apenas o TAG *"quality"*, que indica a qualidade do device). A classe *DeviceEngine* executará um processo conhecido como *"fire the rules"*, que consiste em avaliar as entradas de acordo com os fatos contidos na base de regras. Executado esse processo, a classe *DeviceEngine* terá condições de fornecer a saúde do equipamento que estará sendo analisado.

Para fornecer as condições de saúde dos equipamentos, a classe *DeviceEngine* precisará executar cinco passos:

1. criar uma instância da classe *"Rete"* que é a classe principal do Jess. Esta permite criar uma máquina de inferência para realizar o processo de raciocínio;
2. reiniciar a "*engine*"; isso significa limpar a memória de trabalho para que seja executado um novo processo de inferência;
3. carregar o arquivo de regras (qualidade.clp);
4. executar as regras;
5. extrair os resultados.

A figura abaixo mostra um trecho de código em Java que implementa as etapas citadas anteriormente utilizando as funções do Jess:

```
public DeviceEngine(Database aDatabase) throws JessException {
        // Create a Jess rule engine
        engine = new Rete();
        engine.reset();

        // Load the pricing rules
        engine.batch("qualidade.clp");

        // Load the catalog data into working memory
         database = aDatabase;
        // Mark end of catalog data for later
        marker = engine.mark();
    }
...
// Fire the rules that apply to this order
        engine.run();

        // Return the list of offers created by the rules
        return engine.getObjects(new Filter.ByClass(Warning.class));
```

Figura 6: Código Java para implementar a máquina de inferência do Jess.

O resultado será um objeto na memória de trabalho, que corresponderá a saúde do equipamento. Um exemplo de resultado do processamento pode ser visto na figura 7.

4. Caso de Teste

Para testar o sistema foi desenvolvido um caso de teste que fornece como entrada três situações nas quais são apresentados os TAGs referentes ao equipamento. O objetivo foi testar se as regras geradas (figura 4) são capazes de avaliar a saúde do equipamento. Foram fornecidos como entrada os seguintes dados:

a. Descrição do TAG;
b. Identificador;
c. Valor atual;
d. Qualidade.

Os dados referentes a descrição, identificação e valor atual não são testados nas regras. O único TAG que interessa é o TAG referente a qualidade do equipamento. No caso de teste esse TAG pode receber um de dois valores possíveis, 1 para qualidade boa/aceitável, e 0 para qualidade ruim. O resultado do processamento é apresentado na figura 7.

```
Items for device 123:
       Descrição: Medidor de vazão
       Identificador: Fit31
       Valor Atual: 1700
       Qualidade: 0
Warnings for device 123:
   A qualidade do device não está boa!
   java.util.HashMap$KeyIterator@10bc49d

Items for device 567:
       Descrição: Medidor de temperatura
       Identificador: Tit31
       Valor Atual: 25.0
       Qualidade: 1
Warnings for device 567:
   Device operando na forma aceitável!
   java.util.HashMap$KeyIterator@ce5b1c

Items for device 666:
       Descrição: Medidor de Nível
       Identificador: Lit31
       Valor Atual: 0.9
       Qualidade: 1
Warnings for device 666:
   Device operando na forma aceitável!
   java.util.HashMap$KeyIterator@1e8a1f6
```

Figura 7: Resultado do processamento do Sistema Especialista.

O sistema fornece a descrição do TAG, identificador, valor atual, qualidade, e uma mensagem indicando a qualidade do TAG após a valiação.

Esse caso de teste serviu para avaliar como o sistema executa todas as funções que foram herdadas do Jess pelo Java. É um exemplo da utilização da biblioteca de funções do Jess embutidas no Java. O caso de teste permitiu verificar a facilidade de implementar sistemas especialistas usando o Jess, visto que toda a parte de inferência já vem pronta no pacote.

5. Conclusões e Trabalhos Futuros

O presente trabalho apresentou um exemplo de sistema especialista desenvolvido a partir das funções do Jess. Vale ressaltar que a linguagem utilizada para o desenvolvimento do sistema foi o Java. O Jess foi embutido no Java para que suas funções pudessem ser utilizadas.

O desenvolvimento do sistema permitiu verificar a facilidade de implementar sistemas especialistas, pois o Jess oferece todas as classes e funções necessárias para o mecanismo de inferência. O desenvolvedor deve gerar a base de regras na linguagem entendida pelo Jess e este executará o raciocínio.

Foi desenvolvido um caso de teste para avaliar o desempenho do sistema gerado, e se as saídas estavam de acordo com as entradas. Os resultados mostram que a resposta gerada foi correta e o desempenho do sistema é bom para o problema em questão.

Uma proposta para trabalhos futuros é aumentar a base de regras para aumentar a capacidade de raciocínio e desenvolver o módulo de comunicação JNI.

6. Referências

DEITEL, H. M.; DEITEL, P.J. **Java: como programar.** 6ª ed., Editora Pearson Prentice Hall, São Paulo, 2006.

FRIEDMAN-HILL, E. **Jess in Action: Rule-Based Systems in Java.** 1ª ed. Editora Manning. Greenwich, CT 2003.

GIARRATANO, J. **CLIPS: User's Guide.** Versão 6.10, 1998.

SMAR (2004). **Manual de operação Planta Didática III.** 127 p.

SMAR (2001). **Equipamentos de Campo série 302 Foundation.** Manual de instalação, operação e manutenção. 42 p.

Sistema Especialista Java/ Jess

Tópicos Avançados em Sistemas Mecatrônicos I

Rosimarci Pacheco Tonaco

Conteúdo

- Introdução;
- Planta Didática III da Smar;
- Descrição do Sistema;
- Caso de Teste;
- Resultados;
- Conclusões e Trabalhos Futuros;
- Referências;

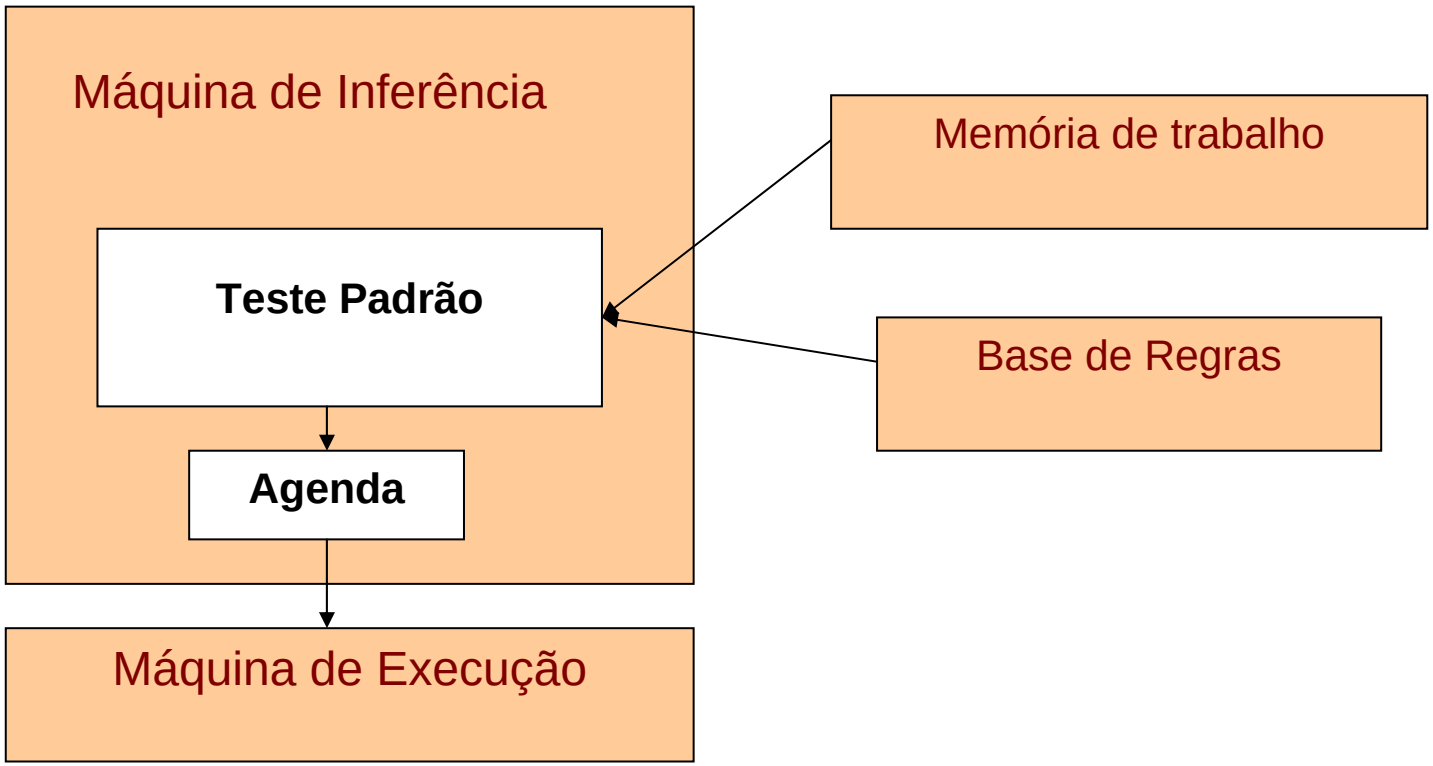
Introdução

- O presente trabalho tem o objetivo de apresentar um Sistema Especialista que realiza a avaliação de saúde dos equipamentos da Planta Didática III da Smar.
- Sistemas Especialistas são sistemas que solucionam problemas que são resolvíveis apenas por pessoas especialistas (que acumularam conhecimento exigido).
- O Jess (*Java Expert System Shell*) foi utilizado no desenvolvimento do mesmo, e este trabalha com regras de produção para representar o conhecimento que será manipulado.

Introdução

- O Jess permite o desenvolvimento de sistemas especialistas utilizando programação declarativa (ou embutindo suas funções no Java).
- A arquitetura de um sistema típico baseado em regras possui os seguintes módulos de trabalho:
  - máquina de inferência;
  - base de regras, e;
  - memória de trabalho.
- A máquina de inferência, internamente, consiste de:
  - teste padrão;
  - agenda, e;
  - máquina de execução.

Introdução



Arquitetura de um sistema baseado em regras.

Planta Didática III da Smar

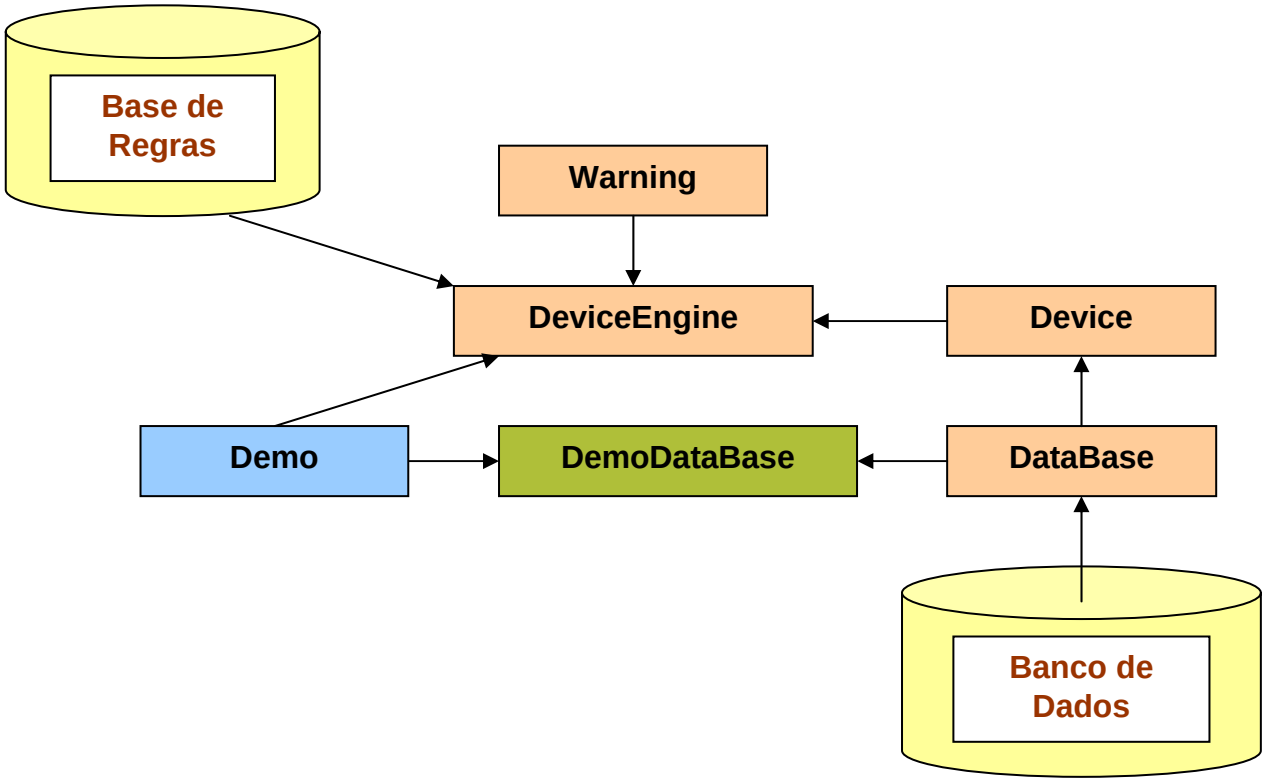
- **O objetivo do sistema especialista é monitorar os equipamentos da planta, verificando a saúde de cada um deles com o intuito de alertar para possíveis falhas/faltas que poderão acontecer. Outro propósito é realizar a manutenção preditiva, que irá sugerir paradas de acordo com os problemas apresentados nos equipamentos.**

Descrição do Sistema

Etapas de desenvolvimento do sistema:

- 1: Gerar base de regras que conterá informações sobre as condições de operação dos equipamentos, indicando condições normais de operação, e condições de risco;
- 2: Adquirir do Assetview os TAGs referentes aos equipamentos da Smar. Tais dados estão armazenados em SQL, para realizar essa aquisição será necessário criar uma comunicação JDBC, do Java, que fará as consultas necessárias ao banco de dados;
- 3: Desenvolver o módulo de raciocínio do sistema especialista para fornecer conhecimento útil sobre a saúde dos equipamentos. Esse módulo avaliará as informações (adquiridas via OPC e/ou SQL) considerando o conhecimento armazenado na base de regras. O módulo de raciocínio usará as funções do Jess, visto que esse é capaz de implementar o processo de inferência de um sistema especialista.

Descrição do Sistema



Arquitetura do Sistema Especialista.

Descrição do Sistema

```
import gov.sandia.jess.example.pricing.model.*)
(deftemplate Device    (declare (from-class Device)))
(deftemplate Warning   (declare (from-class Warning)))

;; Now define the rules themselves. Each rule matches a set
;; of conditions and actions.

(defrule quality-device-Good
    "Verifica a qualidade do device para fazer a avaliação de saúde."
    (Device {quality > 0})
    =>
    (add (new Warning "Device operando na forma aceitável!")))

(defrule quality-device-Bad
    "Verifica a qualidade do device para fazer a avaliação de saúde."
    (Device {quality < 1})
    =>
    (add (new Warning "A qualidade do device não está boa!")))
```

Regras de produção feitas em Jess.

Descrição do Sistema

Para fornecer as condições de saúde dos equipamentos, a classe *DeviceEngine* precisará executar cinco passos:

- criar uma instância da classe *"Rete"* que é classe principal do Jess. Esta permite cria uma máquina de inferência para realizar o processo de raciocínio;
- reiniciar a "*engine*"; isso significa limpar a memória de trabalho para que seja executado um novo processo de inferência;
- carregar o arquivo de regras (qualidade.clp);
- executar as regras;
- extrair os resultados.

Descrição do Sistema

```
public DeviceEngine(Database aDatabase) throws JessException {
        // Create a Jess rule engine
        engine = new Rete();
        engine.reset();

        // Load the pricing rules
        engine.batch("qualidade.clp");

        // Load the catalog data into working memory
         database = aDatabase;
        // Mark end of catalog data for later
        marker = engine.mark();
    }
...
// Fire the rules that apply to this order
        engine.run();

        // Return the list of offers created by the rules
        return engine.getObjects(new Filter.ByClass(Warning.class));
```

- Código Java para implementar a máquina de inferência do Jess.

Caso de Teste

- Para testar o sistema foi desenvolvido um caso de teste que fornece como entrada três situações nas quais são apresentadas os TAGs referentes ao equipamento. O objetivo foi testar se as regras geradas são capazes de avaliar a saúde do equipamento. Foram fornecidos como entrada os seguintes dados:
  - Descrição do TAG;
  - Identificador;
  - Valor atual;
  - Qualidade.

Caso de Teste

- Como a comunicação com o banco de dados não foi possível, desenvolveu-se uma classe teste chamada DemoDataBase que simula a comunicação com o banco de dados. Essa classe é responsável por enviar as entradas para a classe que executa o processo de raciocínio.

Caso de Teste

```
Items for device 123:
          Descrição: Medidor de vazão
          Identificador: Fit31
          Valor Atual: 1700
          Qualidade: 0
Warnings for device 123:
   A qualidade do device não está boa!
   java.util.HashMap$KeyIterator@10bc49d

Items for device 567:
          Descrição: Medidor de temperatura
          Identificador: Tit31
          Valor Atual: 25.0
          Qualidade: 1
Warnings for device 567:
   Device operando na forma aceitável!
   java.util.HashMap$KeyIterator@ce5b1c

Items for device 666:
          Descrição: Medidor de Nível
          Identificador: Lit31
          Valor Atual: 0.9
          Qualidade: 1
Warnings for device 666:
   Device operando na forma aceitável!
   java.util.HashMap$KeyIterator@1e8a1f6
```

Resultado do processamento do Sistema Especialista.

Resultados

- O caso de teste para avaliar o desempenho do sistema gerado, mostrou se as saídas estavam de acordo com as entradas.

- Os resultados mostraram que a resposta gerada foi correta e o desempenho do sistema é bom para o problema em questão.

Conclusões e Trabalhos Futuros

- O desenvolvimento do sistema permitiu implementar um sistema especialista, pois o Jess oferece todas as classes e funções necessárias para o mecanismo de inferência.
- Uma proposta para trabalho futuro é aumentar a base de regras para aumentar a capacidade de raciocínio e desenvolver o módulo de comunicação JNI.

Referências

- DEITEL, H. M.; DEITEL, P.J. **Java: como programar.** 6ª ed., Editora Pearson Prentice Hall, São Paulo, 2006.
- FRIEDMAN-HILL, E. **Jess in Action: Rule-Based Systems in Java.** 1ª ed. Editora Manning. Greenwich, CT 2003.
- GIARRATANO, J. **CLIPS: User's Guide.** Versão 6.10, 1998.
- SMAR (2004). **Manual de operação Planta Didática III.** 127 p.
- SMAR (2001). **Equipamentos de Campo série 302 Foundation.** Manual de instalação, operação e manutenção. 42 p.

CONTROLADOR FUZZY EM PLANTA SMAR PD-3

Victor Rafael R Celestino

Universidade de Brasília (UnB)
Departamento de Engenharia Mecânica
Grupo de Automação e Controle (GRACO)
Brasília, DF, Brasil
Emails: cvictor@uol.com.br

Abstract — In this work, a fuzzy controller was implemented in MATLAB® 7, acting on a Smar PD-3 didactic plant by means of OPC technology. The controller implementation was performed using MATLAB® Fuzzy Logic Toolbox, which provides an intuitive and very friendly interface. The controller obtained satisfactory results for several combinations of input variables values, although the project parameters of the linearized model were not obtained simultaneously.

Keywords — Control systems, Fuzzy logic, Fieldbus Foundation.

Resumo — Neste trabalho foi implementado um controlador “Fuzzy” em MATLAB® 7, atuando na planta didática PD3 da Smar pelo uso da tecnologia OPC. A implementação do controlador foi realizada utilizando a Fuzzy Logic Toolbox do MATLAB®, que proporciona uma interface intuitiva e bastante amigável. O controlador obteve resultados satisfatórios para diversas combinações de valores das variáveis de entrada, apesar dos parâmetros de projeto do modelo linearizado não terem sido obtidos simultaneamente.

1. Introdução

Este trabalho foi desenvolvido como parte da disciplina 169536 – Tópicos em Controle e Automação, oferecida pelo Prof. Dr. Alberto J. Álvares (http://AlvaresTech.com). O trabalho foi realizado no GRACO (Grupo de Automação e Controle – www.graco.unb.br).

Foi utilizada a planta didática PD3 da Smar (www.smar.com.br), com o System 302, empregando protocolo Foundation Fieldbus (www.fieldbus.org). A planta PD3 permite estudar uma malha de controle, utilizando os mesmos equipamentos e ferramentas de configuração do System 302, utilizados em automação industrial.

O objetivo deste trabalho foi implementar um controlador “Fuzzy” em MATLAB® 7, atuando na planta PD3 pelo uso da tecnologia OPC (OLE – Object Linking and Embedding – for Process Control). Para implementação do controlador e sua comunicação com a planta foram utilizadas as “toolboxes” Fuzzy Logic Toolbox e OPC Toolbox do MATLAB®.

Este artigo está assim organizado. A Planta didática PD3 está descrita na Seção 2, com o respectivo modelamento linearizado adotado. O controlador fuzzy é apresentado na Seção 3. Os ensaios e resultados obtidos são discutidos na Seção 4, e a conclusão do trabalho é apresentada na Seção 5.

2. Planta Didática SMAR PD-3

O objetivo da Planta SMAR é demonstrar didaticamente a operação das diversas malhas de controle utilizando os mesmos equipamentos e ferramentas de configuração, em software, desenvolvidos para aplicação em controle industrial. Em um arranjo compacto, esta planta torna acessível aos instrutores e aprendizes todos os componentes desta malha, não sendo apenas uma estrutura para ser observada, mas também para ser manipulada (SMAR, 2004).

Na implementação destas malhas estão contidas as mesmas características e situações encontradas pelos profissionais de instrumentação com os recursos da alta tecnologia disponível no mercado. Além das fornecidas, outras malhas podem ser geradas a partir da estrutura física montada sem a necessidade de alterá-las mecanicamente, apenas modificando a configuração dos dispositivos.

A planta utiliza a tecnologia Foundation Fieldbus, barramento industrial de comunicação responsável por conectar os dispositivos de campo tais como sensores, atuadores, indicadores e controladores, que permitem o controle das malhas de temperatura, vazão e nível existentes, auxiliando no aprendizado de Instrumentação Industrial e Sistemas de Controle (DUARTE et al, 2006).

A Planta Didática SMAR PD-3, apresentada na figura 1, é monitorada e operada de uma estação, constituída de um microcomputador do tipo PC e um software de supervisão, que efetua a aquisição de dados dos equipamentos e o apresenta por meio de animações de telas. Permite também atuar nos registros modificando valores internos dos equipamentos e nos modos operacionais das malhas de controle.

Figura 1 – Planta Didática SMAR PD-3

A Planta Didática apresenta duas malhas de vazão: malha 1 e a malha 2. A malha de vazão 1 controla a vazão de água bombeada pela bomba nº 1 do tanque principal ao tanque nº 1 (tanque de água quente). Este por sua vez quando está cheio retorna a água para o tanque principal. O controle de vazão se dá por meio de um posicionador Filedbus FY302 de uma válvula tipo globo (FY-31). A medição de temperatura se dá por meio de um transmissor de temperatura Filedbus TT302 (TIT-31).

A malha de vazão 2 controla a vazão de água bombeada pela bomba nº 2 do tanque principal ao tanque nº 2 (tanque de água fria). A medição de vazão se dá por meio de um transmissor de pressão diferencial Fieldbus LD302D (FIT-32). O controle de vazão se dá por meio de um posicionador Filedbus FY302 de uma válvula tipo globo (FY-32). A medição de temperatura se dá por meio de um transmissor de temperatura Filedbus TT302 (TIT-32).

A partir de experimentação na planta, observou-se que ambas as malhas de controle tem comportamento semelhante, dado pela curva do gráfico da figura 2.

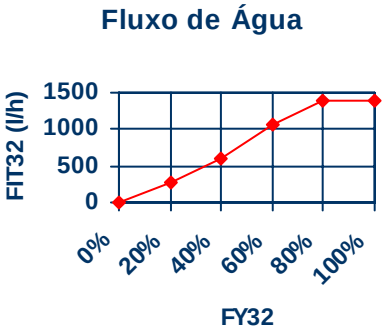


Figura 2 – Vazão na Planta PD-3

3. Controlador Fuzzy

O controlador nebuloso aplicado neste trabalho é composto de um conjunto de regras de inferência do tipo Se <premissa> E <premissa> Então <resultado>, que definem ações de controle em função das diversas faixas de valores que as variáveis de entrada da planta podem assumir.

Utilizando Fuzzy Logic Toolbox (JANG e GULLEy, 1999), foi implementado o controlador com uso do editor FIS, uma ferramenta do MATLAB®. O controlador foi construído no modelo Mamdani e para os cálculos usa a composição max-min, operador min para representar o conectivo E, operador max para o conectivo OU, além do operador min na implicação e max na agregação. Na defuzzificação é usado o método do centróide.

O controlador utiliza três variáveis de entrada, que são extraídas da saída da planta: TIT31 (temperatura do tanque 1), TIT32 (temperatura do tanque 2) e TIT32SP (“set-point” para a temperatura do tanque 2). A saída é o valor de controle da vazão da malha 2 (FY32).

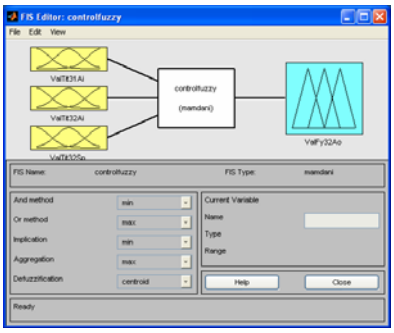


Figura 3 – Controlador Fuzzy

As variáveis de entrada assumem os valores: fria (F1), morna (M1, M2, M3), quente (Q1, Q2, Q3), e muito quente (MQ1, MQ2, MQ3). As funções de pertinência têm a forma apresentada na figura 4.

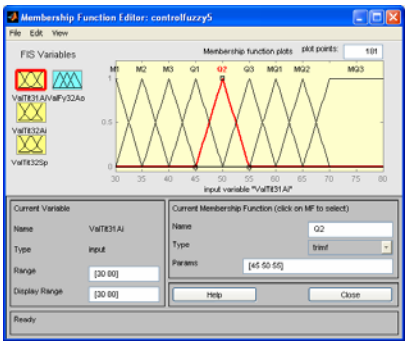


Figura 4 – Funções de Pertinência da Entrada

A variável de saída assume os valores fechada, 1/10, 1/8, 1/6, 1/5, 1/4, 3/10, 1/3, 3/8, 2/5, 1/2, 5/8, 2/3, 3/4, 5/6 e aberta. A sua função de pertinência foi discretizada com base no modelamento linearizado da planta e é apresentada na figura 5.

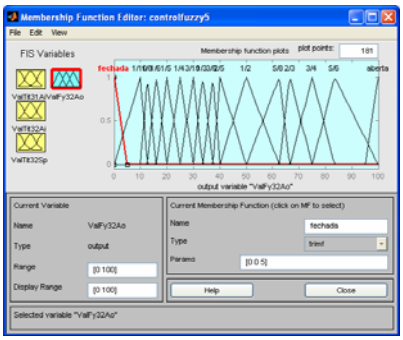


Figura 5 – Função de Pertinência da Saída

4. Resultados Obtidos

O Controle Fuzzy foi testado na planta SMAR PD-3, para diversas combinações de valores das variáveis de entrada, obtendo uma resposta satisfatória, apesar dos parâmetros de projeto do modelo linearizado não terem sido obtidos simultaneamente (FY31=50%) e (Tfria=25 graus C). Para utilizar a vazão de 50% na malha de água quente, é necessário que o retorno do tanque 1 seja para o reservatório (tanque principal), o que aumenta a temperatura da água fria.

Observou-se que com o fluxo de entrada do tanque de água quente maior do que 20 a 30%, e com água de entrada de fonte externa (sem recirculação no reservatório), não é possível obter TIT31>40 graus C.

5. Conclusão

Neste trabalho foi implementado um controlador “Fuzzy” em MATLAB® 7, atuando na planta PD3 pelo uso da tecnologia OPC.

A implementação do controlador foi realizada utilizando a Fuzzy Logic Toolbox do MATLAB®, que proporciona uma interface intuitiva e bastante amigável.

O controlador obteve resultados satisfatórios para diversas combinações de valores das variáveis de entrada, apesar dos parâmetros de projeto do modelo linearizado não terem sido obtidos simultaneamente.

Apesar do projeto implementado trabalhar com “set-points” numéricos, esses poderiam ser estabelecidos na forma de regras linguísticas, que resultariam no controle esperado.

Referências

MARTINS, W.F. GOMES, G.M.P. CUNHA, A.E.C. (2006). **Controlador Nebuloso Aplicado ao Sistema Plataforma-Esfera.** XVI Congresso Brasileiro de Automática, p. 1602-1607.

DUARTE, C.R.M. FIGUEIREDO, L.C. CORRÊA, M.V. (2006). **Utilização do MATLAB® no ensino da tecnologia OPC aplicada a controle de processos.** XVI Congresso Brasileiro de Automática, p. 1429-1434.

MATLAB (2006). **OPC Toolbox for use with MATLAB® and Simulink®.** User´s Guide. v.2. Natick: The Mathworks Inc, 373 p.

SMAR (2005). **Manual de instruções dos blocos funcionais Fieldbus Foundation**. 334 p.

SMAR (2004). Manual de operação Planta Didática III. 127 p.

JANG, J.S.R. GULLEY, N. (1999). MATLAB® fuzzy logic toolbox. User´s Guide. v.1. Natick: The Mathworks Inc, 235 p.

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA MECÂNICA

Controle “Fuzzy” em Planta Didática Smar PD-3 - Planejamento do Trabalho

169536 – Tópicos em Controle e Automação

Victor Rafael R. Celestino

Conteúdo

- Introdução
- Objetivos do Trabalho
- Atividades desenvolvidas
- Modelagem do Sistema
- Resultados Obtidos
- Referências

Introdução

- Este trabalho foi desenvolvido como parte da disciplina 169536 – Tópicos em Controle e Automação, oferecida pelo Prof. Dr. Alberto J. Álvares (http://AlvaresTech.com).
- O trabalho foi realizado no GRACO (Grupo de Automação e Controle – www.graco.unb.br).
- Foi utilizada a planta didática PD3 da Smar (www.smar.com.br), com o System 302, empregando protocolo Foundation Fieldbus (www.fieldbus.org).

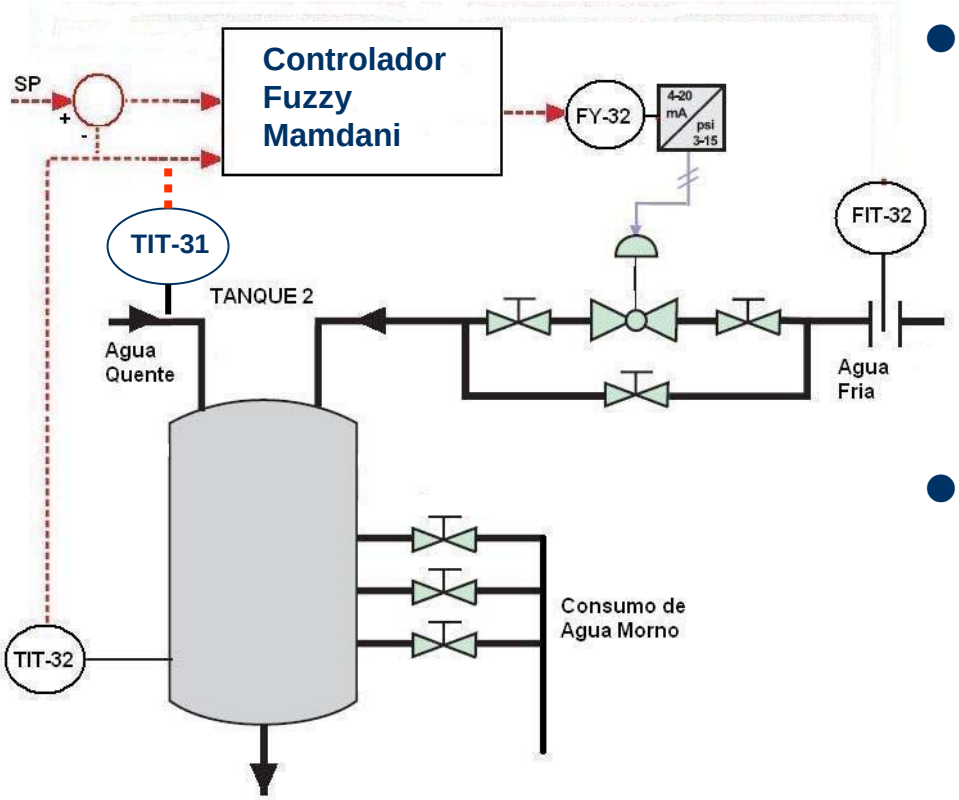
Objetivos do Trabalho

- Estudar uma malha de controle na planta PD3, utilizando os mesmos equipamentos e ferramentas de configuração do System 302, utilizados em automação industrial.
- Implementar um controlador “Fuzzy” em MATLAB® 7, atuando na planta PD3 pelo uso da tecnologia OPC (OLE – *Object Linking and Embedding – for Process Control*).

Atividades desenvolvidas

- Modelagem do Sistema
- Comunicação com MATLAB® com OPC toolbox.
- Projeto e implementação do controlador fuzzy no MATLAB®.
- Teste do controlador fuzzy na Planta PD3.

Modelagem do Sistema



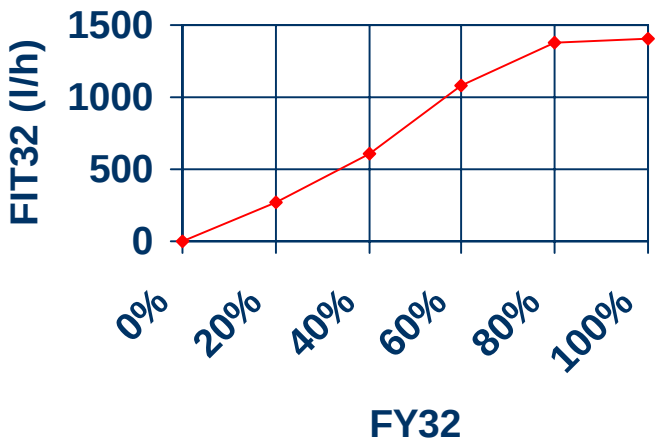
- O controlador tem como entradas:
  - TIT-31: temperatura da água quente
  - TIT-32: temperatura do tanque 2
- E como saída:
  - FY-32: posição da válvula de entrada de água fria

Modelagem do Sistema

- Equação geral do sistema físico:

$$\rho V c_V \frac{dT}{dt} = w c_V (T_0 - T) = c_V \{q_{quente}(TIT31 - TIT32_{SP}) - q_{fria}(TIT32_{SP} - T_{fria})\}$$

Fluxo de Água



$$q_{quente} = 840 l/h (FY31 = 50\%)$$

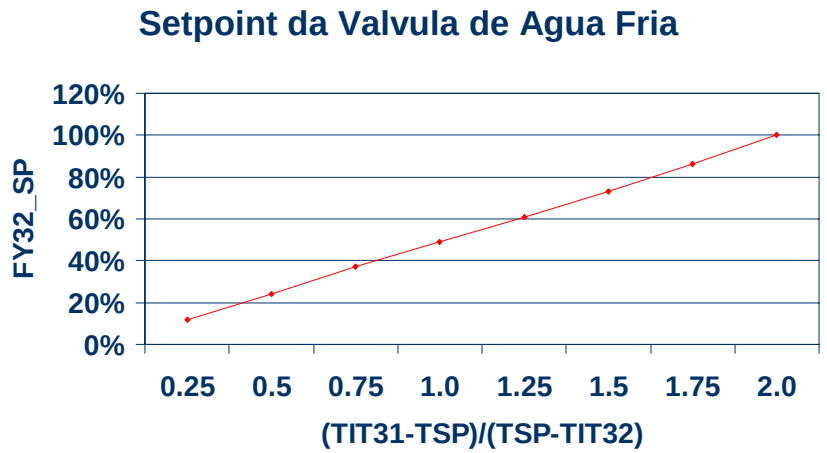
- No equilíbrio dT/dt = 0 :

$$q_{fria} = 840 * \frac{(TIT31 - TIT32_{SP})}{(TIT32_{SP} - T_{fria})}$$

Modelagem do Sistema

- Obtenção do SP de FY-32:

$$q_{fria} = 840 * \frac{(TIT31 - TIT32_{SP})}{(TIT32_{SP} - T_{fria})} \Rightarrow FY32_{SP} = 0{,}4897 * \frac{(TIT31 - TIT32_{SP})}{(TIT32_{SP} - T_{fria})}$$

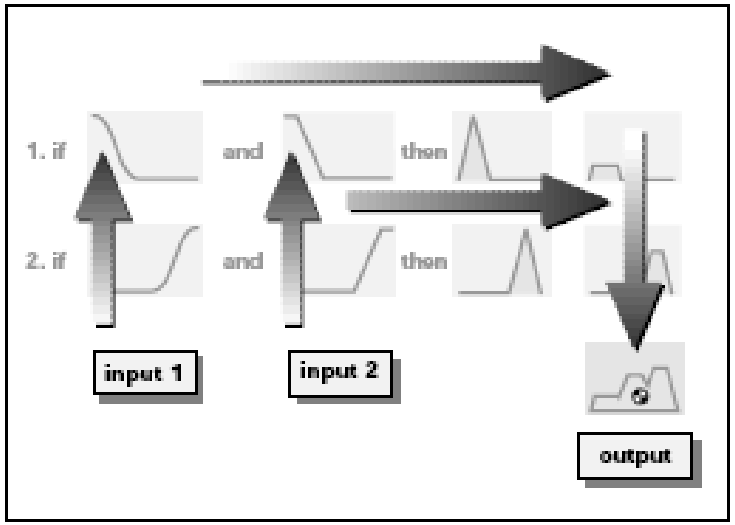


MATLAB ® - OPC ToolBox

- Foi criado um OPC-DA (*Data Access Client Objects*), adicionando grupo e itens, para comunicar-se com o servidor OPC do System 302.
- Os itens são os nomes das variáveis, que se deseja obter dados (*read*) ou enviar dados (*write*).
- A comunicação foi feita com os servidores:
  - Smar.DfiOleServer.0; e
  - Smar.DF65.Server.1.

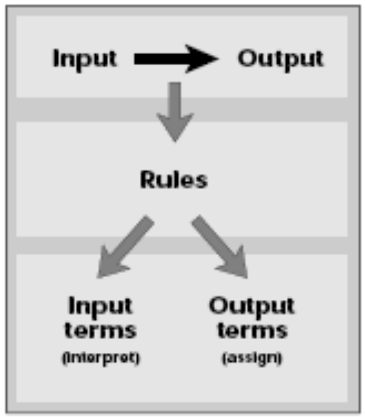
Servidor 'Smar.DfiOleServer.0'	
TAG	**Item**
TIT-31	TIT31_AI1.OUT.VALUE
TIT-32	TIT32_AI1.OUT.VALUE
FY-32	FY31_AI1.OUT.VALUE

Controle “Fuzzy”: Modelo de Mamdani



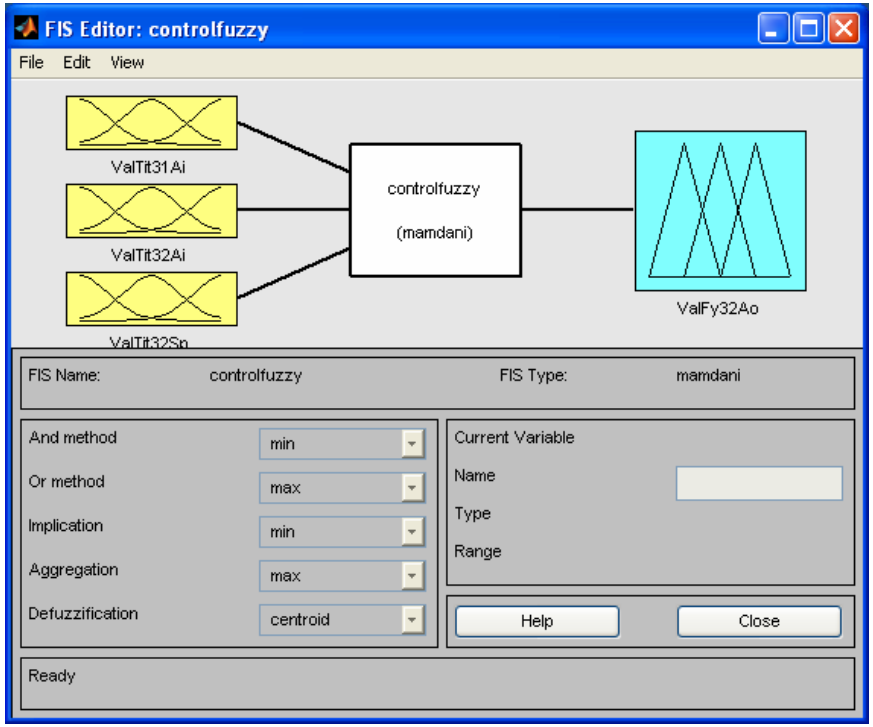
- De forma similar a Martins et ali (2006), o controlador usa o modelo de Mamdani.
- Para os cálculos, utiliza a composição *max-min*.
- Operador *min* para o conectivo E, nas regras.
- operador *max* para a agregação da saída.
- Na defuzzificação, é usado o método do centróide.

Controle “Fuzzy” no MATLAB®



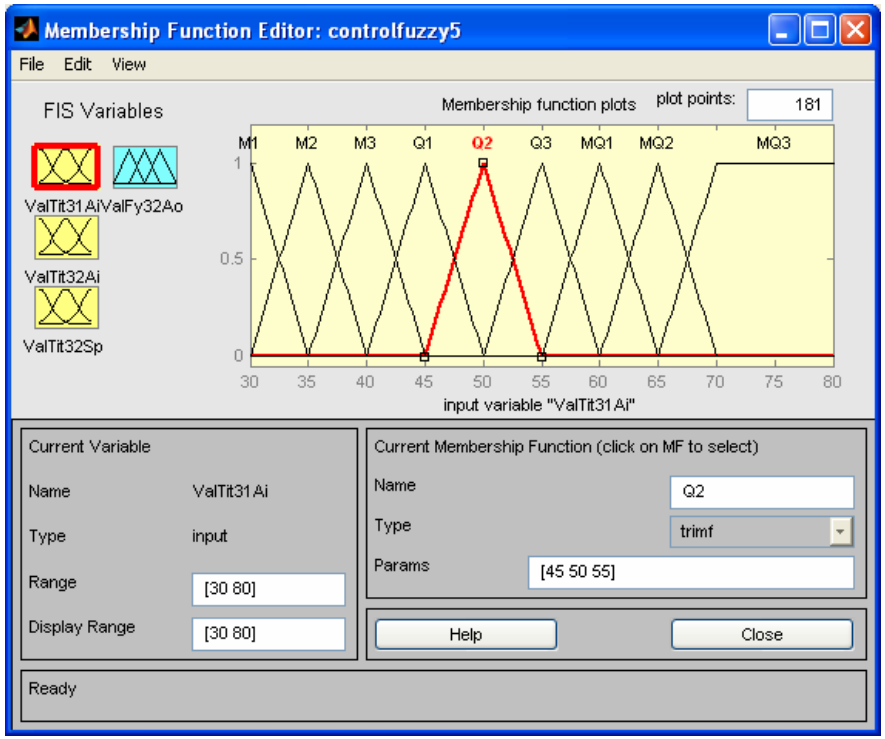
- O controlador Fuzzy foi projetado no Toolbox do MATLAB.
- FIS Editor: gerencia a construção do controle fuzzy.
- Membership function editor: auxilia a construção das funções de pertinência.
- Rule Editor: auxilia a construção das regras de inferência fuzzy.

Controle “Fuzzy” no MATLAB®



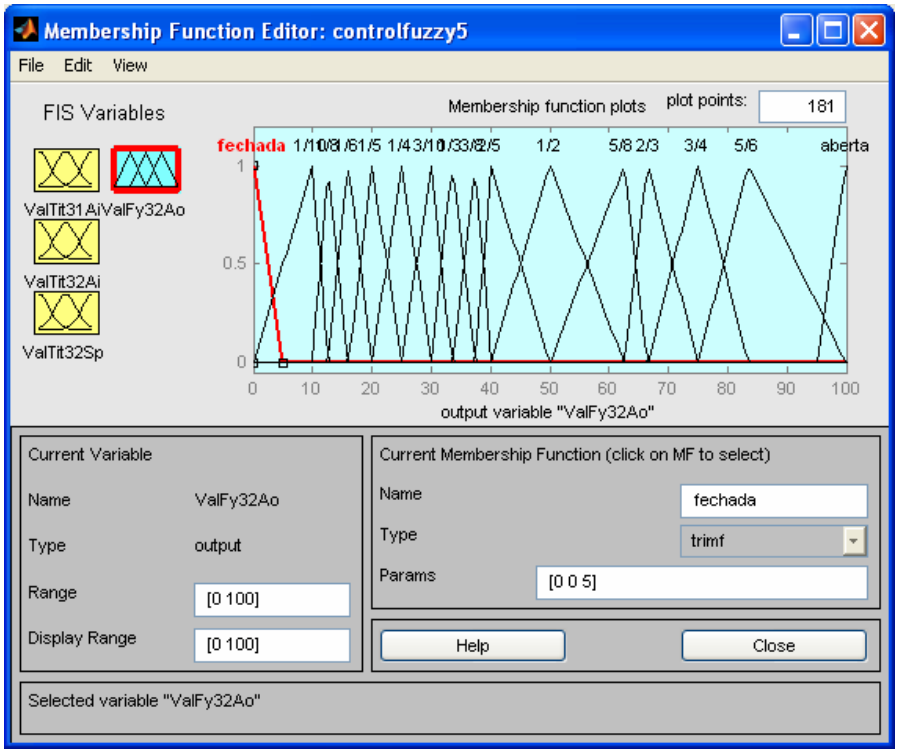
- O primeiro passo no FIS Editor foi estabelecer as variáveis de entrada e saída.
- Variáveis de entrada são:
  - ValTit31Ai,
  - ValTit32Ai, e
  - ValTit32Sp.
- A Variável de saída é:
  - ValFy32Ao.

Controle “Fuzzy” no MATLAB®



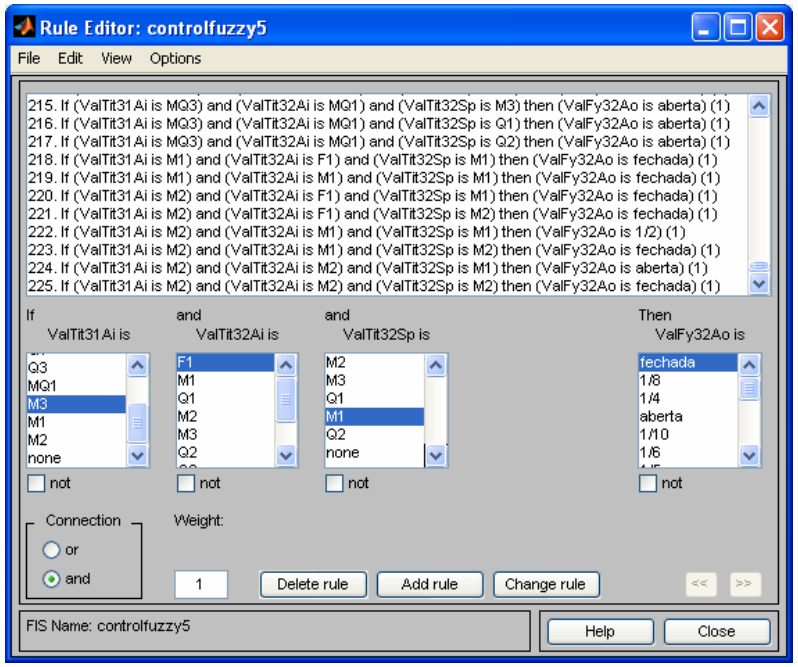
- O segundo passo no Membership function editor foi estabelecer as funções de pertinência das variáveis de entrada.
- As variáveis de entrada (ValTit31Ai, ValTit32Ai e ValTit32Sp) tem valores: fria (F1), morna (M1, M2, M3), quente (Q1, Q2, Q3), e muito quente (MQ1, MQ2, MQ3).

Controle “Fuzzy” no MATLAB®



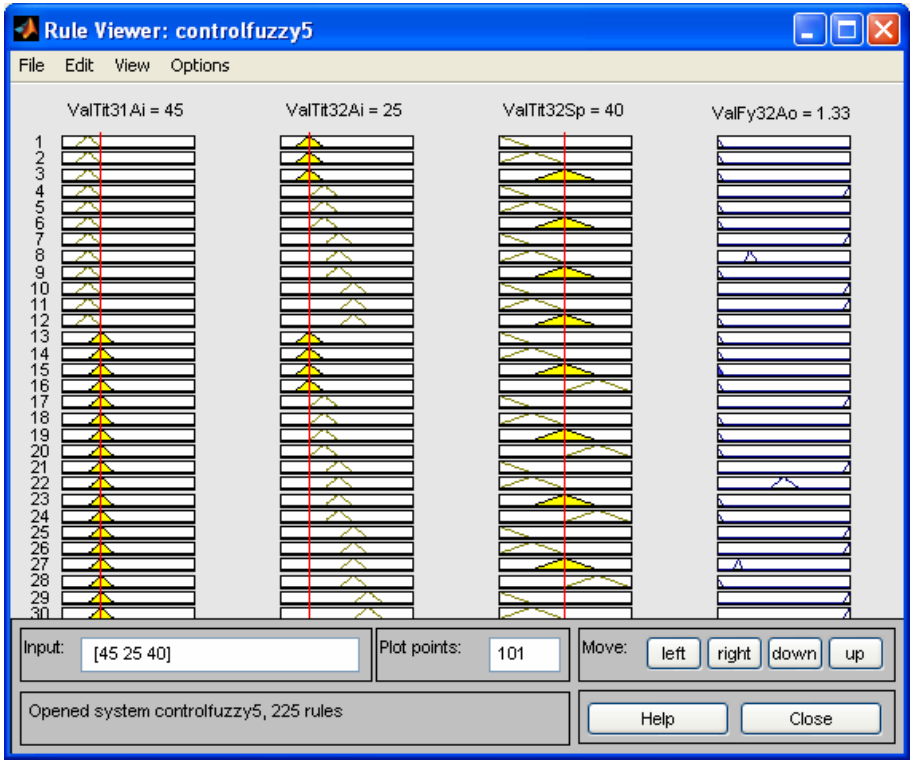
- E as funções de pertinência das variáveis de saída.
- A variável de saída é (ValFy32Ao) tem valores fechada, 1/10, 1/8, 1/6, 1/5, 1/4, 3/10, 1/3, 3/8, 2/5, 1/2, 5/8, 2/3, 3/4, 5/6 e aberta.

Controle “Fuzzy” no MATLAB®



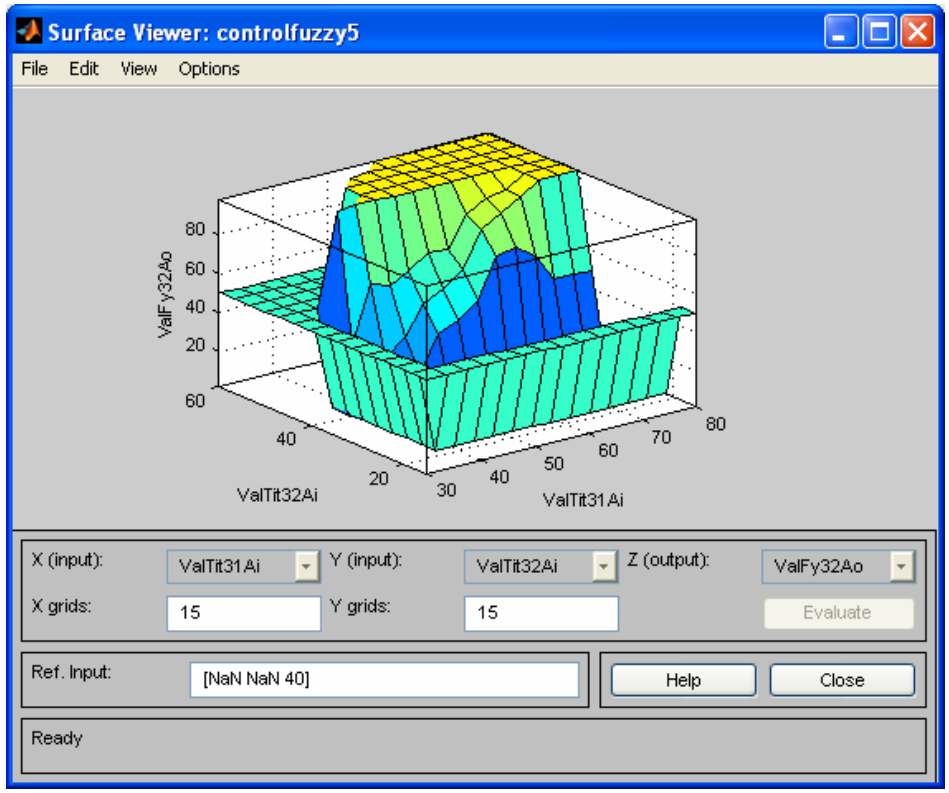
- O terceiro passo no Rule editor foi estabelecer as regras de inferência fuzzy.
- Foram estabelecidas um total de 225 regras, do tipo “SE” ValTit31Ai “AND” ValTit32Ai “AND” ValTit32Sp “ENTÃO” ValFy32Ao.

Controle “Fuzzy” no MATLAB®



- As regras de inferência fuzzy podem ser visualizadas no Rule Viewer, simulando diferentes valores das entradas.
- O exemplo mostra o resultado para TIT31=45, TIT32=25 e TIT32SP=40, que é ValFy32Ao=1.33%.

Controle “Fuzzy” no MATLAB®



- As regras de inferência fuzzy podem ser visualizadas no Surface Viewer, simulando diferentes valores das entradas.
- Notar que a região de interesse é para TIT32<60 e 30<TIT31<80.

Implementação do Controlador

- Alterações de linhas do arquivo ControlerOPC.m

```
%------------------- Controle com Logica Fuzzy --------------
case 2,
% Chamada a função do algoritmo de Logica Fuzzy
victor = readfis('controlfuzzy2');
ValFy32Ao = evalfis([ValTit31Ai, ValTit32Ai, ValTit32Sp], victor);
```

Resultados Obtidos

- Os Set-Points poderiam ser estabelecidos na forma de regras linguísticas, que resultariam no controle esperado. No projeto implementado, esses são estabelecidos em valores numéricos.
- O Controle Fuzzy foi testado na planta SMAR PD-3, para diversas combinações de valores das variáveis de entrada, obtendo uma resposta satisfatória, apesar dos parâmetros de projeto não terem sido obtidos (FY31=50%) e (Tfria=25 graus C).
- Observou-se que com o fluxo de entrada do tanque de água quente maior do que 20 a 30%, e com água de entrada de fonte externa (sem recirculação no reservatório), não é possível obter TIT31>40 graus C.

Referências

- MARTINS, W.F. GOMES, G.M.P. CUNHA, A.E.C. (2006). **Controlador Nebuloso Aplicado ao Sistema Plataforma-Esfera.** XVI Congresso Brasileiro de Automática, p. 1602-1607.
- DUARTE, C.R.M. FIGUEIREDO, L.C. CORRÊA, M.V. (2006). **Utilização do MATLAB® no ensino da tecnologia OPC aplicada a controle de processos.** XVI Congresso Brasileiro de Automática, p. 1429-1434.

Referências

- MATLAB (2006). **OPC Toolbox for use with MATLAB® and Simulink®.** User´s Guide. v.2. Natick: The Mathworks Inc, 373 p.
- SMAR (2005). **Manual de instruções dos blocos funcionais Fieldbus Foundation**. 334 p.
- SMAR (2004). Manual de operação Planta Didática III. 127 p.
- JANG, J.S.R. GULLEY, N. (1999). **MATLAB® fuzzy logic toolbox**. User´s Guide. v.1. Natick: The Mathworks Inc, 235 p.

CONTROLADOR PID EM PLANTA SMAR PD-3

Carlos Frederico Maciel Silveira

Universidade de Brasília (UnB)
Departamento de Engenharia Mecânica
Grupo de Automação e Controle (GRACO)
Brasília, DF, Brasil
Emails: fredericofurlan@yahoo.com.br

Abstract — In this work, was implemented a PID controller (using MATLAB® 7) for the purpose of remote controlling a Smar PD-3 didactic plant by means of OPC technology. The purpose of the PID was to control the temperature level on the mixture tank of the plant. The controller obtained satisfactory results for the tests that had been carried out, with good performances on the settling time (3%) and rising time (90%).

Keywords — Control systems, PID, Fieldbus Foundation.

Resumo — Neste trabalho, foi implementado um controlador PID (usando MATLAB® 7) com a finalidade de controlar remotamente a planta didática PD3 da Smar pelo uso da tecnologia OPC. O função do controlador PID era a de controlar o nível de temperatura no tanque de mistura da planta. O controlador obteve resultados satisfatórios para os testes que foram realizados, com boas performances no tempo de assentamento (3%) e no tempo de subida (90%).

1. Introdução

Neste trabalho foi realizado o controle do nível de temperatura do tanque de mistura da planta didática PD-3 da Smar, agindo-se através de um controlador PID remoto implementado em MATLAB® 7. O acesso à planta se dava através da tecnologia OPC (OLE – Object Linking and Embedding – for Process Control).

A planta conta com uma série de malhas de controle compostas por dispositivos industriais com função de sensoriamento, atuação e controle. Estes dispositivos funcionam, todos eles, dentro do padrão Foundation Fieldbus. A planta conta ainda com a suíte de programas Smar System 302.

Na seção 2 a seguir, explica-se o funcionamento básico da planta. Na seção 3, explica-se a técnica de controle PID e sua implementação como controlador remoto da planta. Na seção 4, tem-se os testes realizados para o controlador em questão, e seus resultados. Na seção 5, tem-se as conclusões finais.

2. Planta Didática SMAR PD-3

A idéia principal por trás da planta PD-3 da Smar é demonstrar didáticamente a operação sobre suas as diversas malhas de controle, usando os mesmos equipamentos e ferramentas de configuração ultilizados no dia a dia das aplicações de automação industrial.

Cada malha de controle da planta é configurada por *software*, de forma a se conseguir alterar completamente seu funcionamento sem nenhuma alteração mecânica necessária para isso. As situações propostas pelas malhas são sínteses das situações encaradas nas grandes aplicações de automação industrial.

A tecnologia Foundation Fieldbus, usada pela planta, funciona dentro do paradigma de controle distribuído. Neste tipo de controle, não mais um conjunto de computadores industriais centrais são os únicos responsáveis pelo controle da planta. Os próprios dispositivos de campo são dotados de “inteligência”, executando eles mesmos, parte dos algoritmos de controle e supervisão necessitados pelo processamento global da planta. Este paradigma faz já algum tempo, é uma forte tendência nas aplicaçãos de automação industrial.

A planta PD-3 é normalmente monitorada e operada localmente através de uma estação, contituída de um *PC* com o conjunto de *softwares* de supervisão, configuração e controle necessários ao funcionamento da planta. Tarefas como a configuração das malhas de controle, captura dos dados dos dispositvos podem ser realizadas por este.

Figura 1 – Planta Didática SMAR PD-3

A panta PD-3 disponível, veio pré-configurada com duas malhas. A primeira, chamada de tic31, realiza o controle de temperatura do tanque 1 (o mais a esquerda na figura 1) usando dispositivos Foundation Fieldbus. Água com temperatura ambiente oriunda de um reservatório através da ação de uma bomba chega ao tanque 1 depois de passar por uma vávula proporcional do tipo globo (dispositivo FY-31). Dentro do tanque a água é aquecida por uma resistência (dispositivo TY-31) até que se alcance o valor de *set point* colocado para esta temperatura. O próprio TY-31 realiza o algoritmo de controle. O volume de água em questão é atirado em seguida no tanque 2, o "tanque de mistura". A segunda malha pré-configurada na planta é a malha de controle de temperatura do tanque de mistura. Água com temperatura ambiente oriunda de um reservatório através da ação de uma bomba chega ao tanque 2 depois de passar por uma vávula proporcional do tipo globo (dispositivo FY-32). Esta "mistura-se" com o fluxo de água quente vindo do tanque 1. O controle da temperatura é feito então, através da abertura e fechamento de FY-32, permitindo que mais ou menos água "fria" se misture com a água quente. O próprio dispositivo FY-32 é o responsável pelo algoritmo de controle. O dispositivo responsável pela leitura da temperatura é o TIT-32. A figura 2 traz uma representação da malha 2.

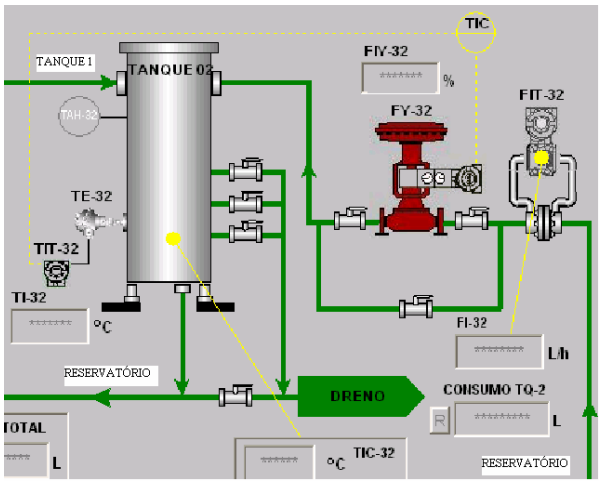


Figura 2 – Representação da malha 2.

A proposta do trabalho é então retirar o controle da temperatura do tanque de mistura "das mãos" do FY-32, e delegá-lo a um agente remoto. Este agente remoto será um PID implementado em MATLAB® 7, rodando em uma estação remota, e acessando os dispositivos da planta via OPC. O PID em questão terá como entrada a temperatura do tanque de mistura, representado como a *tag* TIT32_AI1.OUT.VALUE, pelo servidor de OPC da planta. O valor de *set point* será inidicado remotamente pelo usuário. O valor da ação de controle será escrito na *tag* OPC FY32_AI1.OUT.VALUE, a qual representa o valor da atuação da válvula FY-32.

3. Implementação do controlador PID

A utilidade dos controladores PID está na sua aplicabilidade geral à maioria dos sistemas de controle. Em particular, quando o modelo matemático da planta não é conhecido e, portanto. Embora em muitas aplicações eles não possam proporcionar um controle ótimo.

O controle PID (Proporcional, Integrativo e Derivativo) tem seu funcionamento baseado na seguinte idéia:

- São dadas as entradas *set point* (valor de desejado para a variável de sistema), e valor real da variável de sistema;
- A diferença entre o valor real e o *setpoint* constitui no *erro*;
- Este erro é dado como entrada no controlador PID o qual processará uma ação de controle para o sistema de forma a tentar sempre igualar o erro a zero. Obtendo-se assim o valor desejado depois de um certo tempo de atuação do PID;

O processamento do erro se dá de acordo com o esquema apresentado na figura 3. O termo e(t) reprensenta o erro em função do tempo. O termo *Error*, representa o erro. O diagrama *Process*, representa o processo a ser controlado. E por fim, o termo *Output*, representa o valor real da variável a ser controlada:

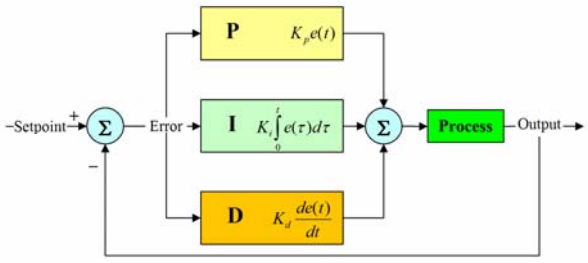


Figura 3 – Esquema básico de um controlador PID.

A ação de controle sobre sobre o processo, é resultado do somatório da ação individual de três termos. O primeiro, o mais acima na figura 3, tem como ação individual a multiplicação do erro por uma constante K_p. O segundo, tem como ação idividual a multiplicação da integral do erro (desde o instante que o PID foi ativado) por uma constante K_i. O terceiro, tem como a ação individual a multiplicação da derivada do erro por uma constante K_d.

Considerando-se K_i como K_p/T_r, K_d como K_p*T_d, e ainda, u(t) como a ação de controle, tem-se a seguinte representação do controlador PID no domínio da freqüência:

$$\frac{U(s)}{E(s)} = K_p\left(1 + \frac{1}{T_i s} + T_d s\right) \quad \text{(eq . 1)}$$

Aplicando a transformada Z, chega-se a:

$$\frac{U(z)}{E(z)} = \left(K_p + \frac{K_p * T * z}{T_i * (z-1)} + \frac{T_d * K_p * (z-1)}{T * z}\right)$$

(eq . 2)

Dado que a mutiplicação por z^{-1} representa o atraso de um intervalo de amostragem (T), tem-se a seguinte equação a diferenças para o controlador PID sob sua forma digital:

$$u(k) = \left(K_p + \frac{K_p}{T_i} * T + \frac{T_d * K_p}{T}\right) * e(k) +$$

$$(T_d * K_p) * T * e(k-1) + \frac{K_p}{T_i} u(k-1)$$

(eq. 3)

A equação acima, usada de maneira recursiva, serviu de núcleo para o algoritmo do controle PID remoto proposto no tema do trabalho. O algoritmo então foi implementado na linguagem MATLAB® 7 num cliente OPC remoto. Este acessava a planta via os servidores de OPC residentes no PC monitor da planta. No mesmo cliente remoto, em conjunto com o PID, foi implementado um amplo supervisório de controle da planta, para fins de auxílio às atividades do PID. Tinha funções como: ligar e desligar bombas, etc. Este foi implementado também em MATLAB® 7 e também fazia uso da tecnologia OPC.

Segue-se então com a fase da sintonia do PID. Foi usado o segundo método de Ziegler-Nichols, onde faz-se um ajuste experimental dos parâmetros do PID. O método funciona da seguinte maneira:

- Coloca-se uma função degrau no *set point* e "liga-se" o PID, com o termo T_r igual a infinito e o termo T_i igual a zero;
- Ajusta-se o ganho K_p até atingir-se a condição de estabilidade marginal. Denomina-se este ganho, de K_{pc}, ou ganho crítico. O período da senóide resultante como saída marginalmente estável é indicado por P_{pc};
- Coloca-se o valor final de K_p como 0.6 K_{pc}, o valor de T_i, como 0.5 P_{pc}, e por último, o valor de T_d como 0.125 P_{pc};

Usando-se o algoritmo acima e aplicando-se um pequeno ajuste empírico em cima dos valores encontrados teve-se: algoritmo acima teve-se: $K_p = 3.7$; $T_i = 9$; $T_d = 1$. Para todos os testes acima e para todos os resultados que se seguem, usou-se T = 2 segundos.

4. Resultados Obtidos

Foi definido um teste para análise do funcionamento do PID sintonizado. O tanque 1 era aquecido usando sua malha de controle automático com o *set point* de 38ºC. Desligava-se a bomba que jogava a água fria para tanque de mistura, afim de o tanque de mistura, ou tanque 2, também se estabilizasse em 38ºC. Quando chegava-se a tal estabilização, ligava-se a bomba referente a entrada fria do tanque de mistura e ligava-se o controle PID remoto feito para temperatura deste, com o *set point* de 32ºC. Colheu-se os dados, e em seguida, estes foram analisados.

As duas figuras a seguir, sintetizam o comportamento do PID frente ao teste. ValFy32Ao simboliza a ação de controle enviada à válvula TY-32. ValFy32Sp simboliza o *set point*. ValTit32Ai simboliza o valor real medido de temperatura.

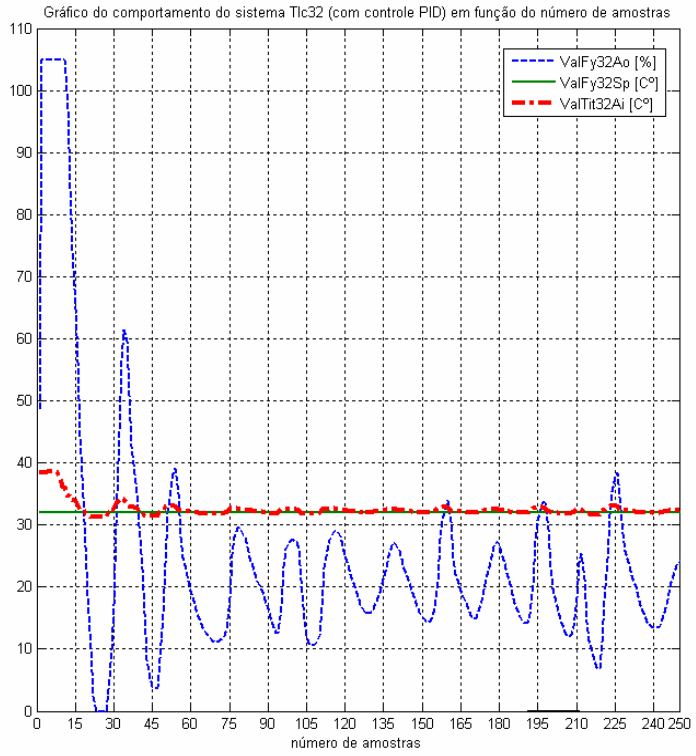


Figura 4 – Resultado do controle PID remoto sobre a planta.

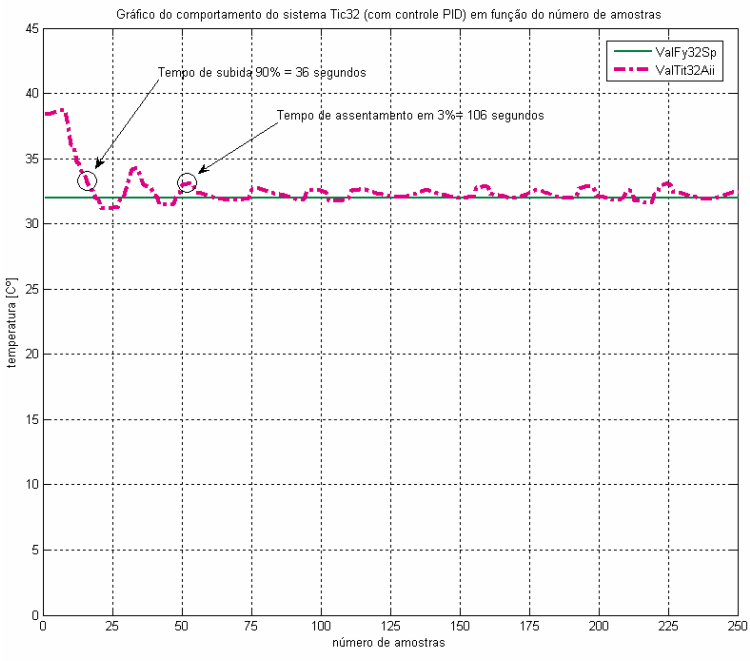


Figura 5 – Detalhes de tempo de subida e tempo de assentamento.

Observa-se que o controle PID obteve um ótimo resultado, com tempo de assentamento (3%) em aproximadamente um minuto e meio, tempo de subida (90%) em cerca de meio minuto, e sobressinal muito baixo. Observa-se ainda, pela figura 4 que a sensibidade da planta a ação de controle foi otimizada com os parâmetros PID escolhidos, pois a ação de contole age de maneira gradual, sem saltos repentinos.

5. Conclusão

Neste trabalho foi implementado um controlador PID em MATLAB® 7, atuando na planta PD3 pelo uso da tecnologia OPC.

O controlador obteve resultados satisfatórios para um controlador PID, mostrando ótimas caraterísticas de tempo de subida, tempo de assentamento e sobressinal.

Quanto à parte do projeto relacionada à tecnologia OPC, esta mostrou-se uma maneira confiável de obter e enviar dados numa arquitetura cliente servidor.

Referências

DORF, R.C., BISHOP, R.H. (1998). **Sistemas de Controle Moderno.** Ed. LTC, Rio de Janeiro, 669 p.

OGATA, K. (2004). **Engenharia de controle moderno.** Ed. LTC, Rio de Janeiro, 710 p.

MATLAB (2006). **OPC Toolbox for use with MATLAB® and Simulink®.** User´s Guide. v.2. Natick: The Mathworks Inc, 373 p.

SMAR (2005). **Manual de instruções dos blocos funcionais Fieldbus Foundation**. 334 p.

SMAR (2004). Manual de operação Planta Didática III. 127 p.

ANEXO 1

Paper SUBMETIDO AO CAIP2007

Sistema de Manutenção Baseada Em Condição Para Usina Hidrelétrica de Balbina

Alberto José Álvares[1], Edgar Amaya Simeón[1], Rosimarci Pacheco Tonaco[1]
(1) Universidade de Brasília, Faculdade de Tecnologia, Departamento de Engenharia Mecânica, Curso de Sistemas Mecatrônicos. Campus Universitário Darcy Ribeiro, 70910-900 – Asa Norte – Brasília – DF, Brasil. (alvares@AlvaresTech.com, eamaya@unb.br, rosimarci@hotmail.com)

RESUMO

A metodologia foi desenvolvida no contexto do projeto Modernização da Área de Automação de Processos das Usinas Hidrelétricas de Balbina e Samuel, onde o objetivo é o desenvolvimento de um sistema inteligente de manutenção preditiva da usina que será utilizado em Balbina. A metodologia SIMPREBAL é baseada em conceitos de manutenção centrada em confiabilidade, sendo utilizada para analisar os modos e efeitos de falhas das Unidades Geradoras Hidráulicas de Balbina a partir das grandezas monitoradas pelo sistema de supervisão e controle da Usina, focando a sua análise no sistema da Turbina. A metodologia proposta é genérica, podendo ser utilizada também no sistema do Gerador. O objetivo é conceber uma metodologia para coleta e análise de dados monitorados nas unidades geradoras da usina de Balbina e a implementação de um sistema computacional com vistas à produção de diagnósticos de estados de funcionamento e de dados que auxiliem a tomada de decisão quanto às ações operacionais e de manutenção das máquinas, visando o aumento da disponibilidade dos equipamentos.

INTRODUÇÃO

Conforme definido por Giacomet (2001, p.27), “manutenção é toda ação realizada em um equipamento, conjunto de peças, componentes, dispositivos, circuito ou estrutura que se esteja controlando, mantendo ou restaurando, a fim de que o mesmo permaneça em operação ou retorne a função requerida”, ou seja, o conjunto de condições de funcionamento para o qual o equipamento foi projetado, fabricado ou instalado.

Independente da definição que se utilize de manutenção percebe-se que as definições utilizam a expressão “manter”, “restabelecer”, “conservar”, “restaurar” ou “preservar” a função requerida do ativo físico de um sistema.

Para realizar o processo de manutenção existem alguns tipos de ferramentas de manutenção. Dentre elas pode-se citar:

1. Manutenção Centrada em Confiabilidade (MCC): visa racionalizar e sistematizar a determinação das tarefas adequadas a serem adotadas no plano de manutenção, bem como garantir a confiabilidade e a segurança operacional dos equipamentos e instalações com menor custo.
2. Manutenção Preditiva: manutenção baseada em condição (contínua e periódica), a partir das grandezas monitoradas por um sistema de supervisão e controle, complementada por informações de inspeções periódicas, para realizar o diagnóstico e o prognóstico de falhas.
3. Manutenção Proativa: a literatura relata dois tipos de prognóstico de máquinas. O primeiro e mais usual é chamado de Vida Útil Remanescente (vida residual) e refere-se à observação ao longo do tempo antes que uma falha ocorra, tendo como referência a idade da máquina, a sua condição e o perfil de operação passado. É similar ao diagnóstico utilizando abordagens estatísticas, Inteligência Artificial (IA) e abordagens baseadas em modelo. A segunda técnica de prognóstico incorpora políticas de manutenção, sendo chamada neste caso de manutenção baseada em monitoração (CBM). A principal idéia do prognóstico incorpora políticas de manutenção para otimização de acordo com certos critérios como risco, custo, confiabilidade e disponibilidade.

Modelo de Referência Usado para Manutenção Baseada em Condição: CIM-OSA

Este modelo será utilizado como referência para o desenvolvimento do sistema de manutenção inteligente baseado em condição, bem como a arquitetura baseada em OSA-CBM (*Open System Architecture for Condition Based Maintenance*) descrita no site http://www.osacbm.org.

A arquitetura OSA-CBM consiste em sete camadas (fig. 1). A noção de uma arquitetura estendida em camadas usada aqui é consistente com o conceito usado em Buschman (1996). Uma camada é vista como uma coleção de tarefas semelhantes ou funções em níveis diferentes de abstração.

Figura 1 – Arquitetura OSA-CIM e suas 7 camadas.

As camadas hierárquicas representam uma transição lógica ou um fluxo da saída dos sensores para a camada de tomada de decisão, através das camadas intermediárias. A camada de apresentação é uma exceção dentro da arquitetura, pois permite comunicação ponto-a-ponto entre esta camada e qualquer outra.

A seguir apresentamos um exemplo das sete camadas estudando o caso do mancal guia do sistema da turbina e como estas se ajustam ao sistema SIMPREBAL em desenvolvimento:

ESTUDO DE CASO: MANCAL GUIA DO SISTEMA DA TURBINA

Esta seção apresenta uma breve explicação sobre os mancais do grupo turbina-gerador, bem como, um estudo de caso utilizando como exemplo o mancal guia da turbina. O mancal guia do sistema da turbina é responsável por suportar o empuxo radial do eixo da turbina evitando que a vibração causada pela água possa deslocá-la, acarretando problemas ao sistema.

O arranjo dos mancais do grupo turbina-gerador (ver figura 2) é o seguinte:
- O mancal de guia da turbina é colocado próximo à extremidade inferior do eixo, na tampa superior e compatível com o arranjo e o acesso à vedação do eixo.

- O mancal de escora é suportado por uma estrutura cônica apoiada na tampa superior.
- O mancal de guia do gerador será locado imediatamente acima do rotor do gerador, no centro da aranha superior do gerador, cujo peso deve ser suportado pela carcaça do estator, mas as forças radiais do mancal devem ser suportadas pela estrutura de concreto.

Com este arranjo, o peso das partes girantes do grupo turbina-gerador mais o empuxo hidráulico vertical serão transmitidos pelo mancal de escora, através da estrutura cônica, à tampa superior e por esta, através do pré-distribuidor.

O conjunto do Mancal Guia da turbina é constituído de:

- Mancal de casquilho bi-partido em aço carbono com deposição de metal patente;
- Sistema de troca de calor em forma de serpentinas.

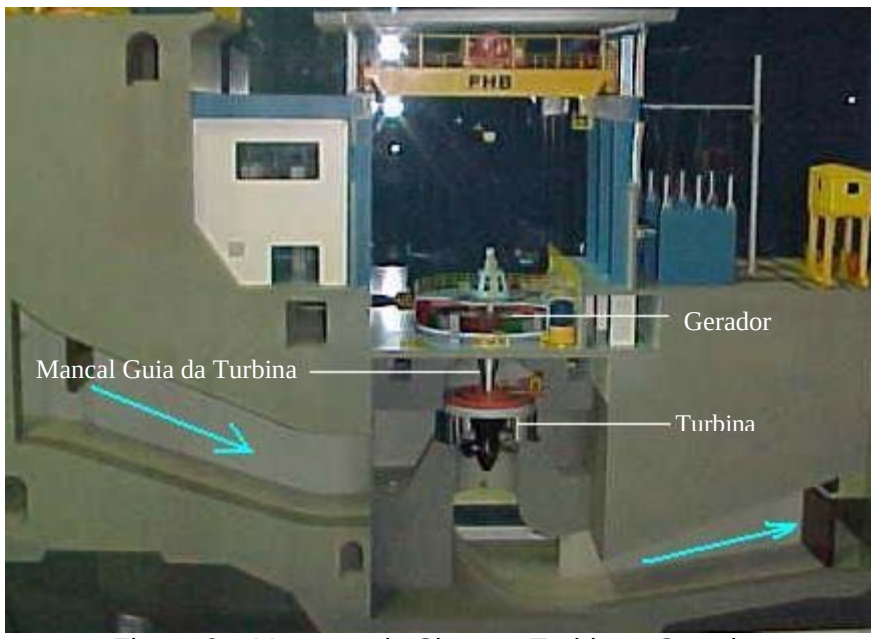


Figura 2 – Maquete do Sistema Turbina - Gerador

A figura 3 apresenta a Arquitetura OSA-CIM e suas 7 camadas adaptadas para o problema em questão.

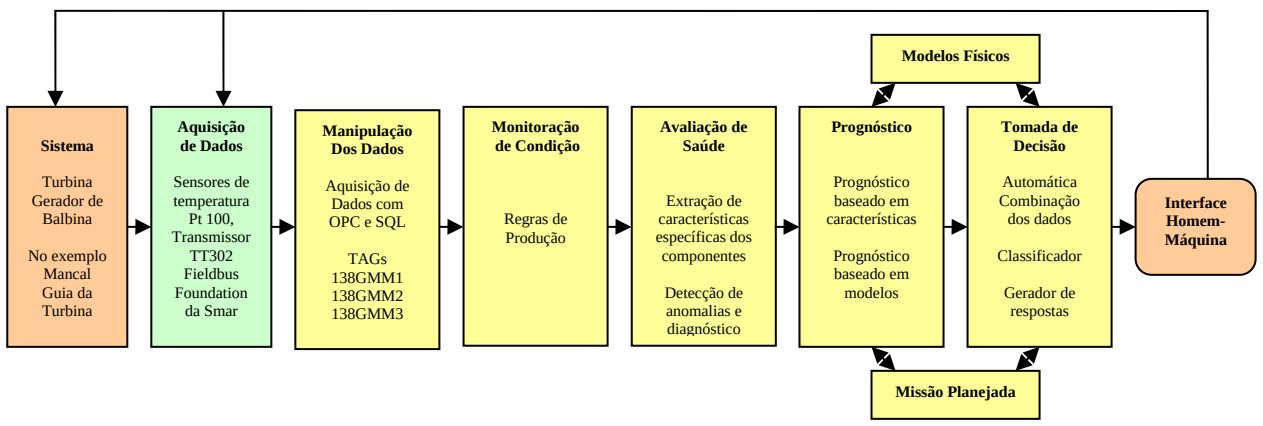


Figura 3 - Adaptação das 7 camadas OSA-CIM para o Sistema Inteligentes.

O módulo sensor é responsável por adquirir do sistema os dados necessários para o processamento da informação relativo ao mancal guia da turbina no exemplo analisado. Tais dados são disponibilizados pelos TAGs de cada equipamento. Uma vez adquirido os dados necessários, o módulo responsável pela manipulação dos mesmos executará a tarefa de pré-processamento (conversão de valores contínuos para valores discretos, por exemplo), extração da características importantes para o processo de raciocínio e posterior geração na base de dados com base nas informações conseguidas nas etapas anteriores. O módulo responsável por monitorar as condições do sistema da turbina, visa alertar para possíveis variações que possam causar danos ao processo. Essa monitoração é feita usando regras de produção. Um exemplo de regras de produção pode ser visto na figura 5.

Falha	Causa provável	Solução
Vibração crescente	Funcionamento com cargas abaixo do nominal.	Evitar que a unidade geradora funcione por tempo demasiado com cargas baixas.
Bombas hidráulicas não funcionam.	Polaridade invertida. (Nem todas as bombas admitem rotação em ambos os sentidos)	Inverter a polaridade do motor elétrico.
Palhetas diretrizes não vedam devidamente.	Algum detrito esta impedindo o fechamento total das palhetas.	Abrir e fechar as palhetas por controle manual para facilitar a passagem dos detritos pelas palhetas.

Figura 4 – Possíveis dificuldades de funcionamento.

A avaliação da saúde do equipamento ocorre através da extração das características de cada equipamento e posterior detecção de anomalias e diagnósticos dos mesmos (ver figura 4). Para tanto, deve-se criar um conjunto de regras (ver figura 5) de produção que seja capaz de expressar e avaliar a condição atual do equipamento, evitando desta forma uma possível falha/falta. Em seguida, um prognóstico é sugerido a partir das informações dos módulos anteriores. Este pode ser baseado em modelos pré-estabelecidos ou feito considerando as características encontradas na avaliação de saúde. Uma vez sugerido o prognóstico, pode-se iniciar a etapa de tomada de decisão. Nesse módulo o sistema inteligente apresenta algumas sugestões para a solução do problema, cabe ao usuário tomar a decisão mais aplicada. Para permitir a camunicação entre homem e máquina, uma interface intuitiva será elabora. Isto visa facilitar o entendimento do processo que estará sendo executado pelo sistema inteligente.

Se Vibração = crescente
Então Funcionamento = baixo

Se Bomba_Hidráulica = não_funciona
Então Polaridade = invertida

Se Palhetas_diretrizes = não_vedam
Então Fechamento_Total = impedido

Figura 5 – Regras de produção para monitoramento de controle.

Uma vez definida as funções do sistema utilizando as sete camadas do modelo OSA-CIM, pode-se definir os módulos das camadas para o sistema de resfriamento do mancal guia da turbina, o componente tem três sensores de temperatura TT302 seus tag são: 138GMM1, 138GMM2, 138GMM3.

Aplicando as sete camadas para o Sistema de resfriamento.

1. **Módulo de sensor:** A aquisição de dados, no presente trabalho corresponde a obtenção dos TAGs referentes a cada equipamento, é feita via OPC e/ou SQL para TAGs da

instrumentação Fieldbus/Smar e Rockwell. No caso de estudo os sensores de temperatura usados são dispositivos termos resistentes Pt100 e o transmissor é TT302 Fieldbus Foundation da Smar, System 302. Tendo-se três sensores e transmissores deste tipo.

2. **Processamento de sinal:** O sistema deve acessar os dados da Usina de Balbina a partir dos Bancos de Dados SQL utilizados pelo Sistema de Monitoramento de Balbina, ou diretamente à instrumentação Fieldbus e Rockwell por meio dos servidores OPC, sendo os que disponibilizaram as informações on-line dos equipamentos. Para obtenção das variáveis on-line via OPC pode-se utilizar o Matlab usando o OPC toolbox, desenvolvendo um programa para aquisição das TAGs necessários pra avaliação dos equipamentos, e gerar o código Matlab para C/C++ e fazer uma interface C/C++ com Java através de JNI. Outra opção é usar OPC-JNI sem usar o Matlab (ver Figura 6). No caso de estudo tem-se os TAGs 138GMM1,138GMM2, 138GMM3. Os valores das TAGs são amostrados em formato numérico ao software de avaliação. Suas TAGs de alarme ALM / HI (80°C) e TRIP / HIHI (85°C) são 138GMM1A , 138GMM1TA, 138GMM2A , 138GMM2TA, 138GMM3A e 138GMM3TA respectivamente.

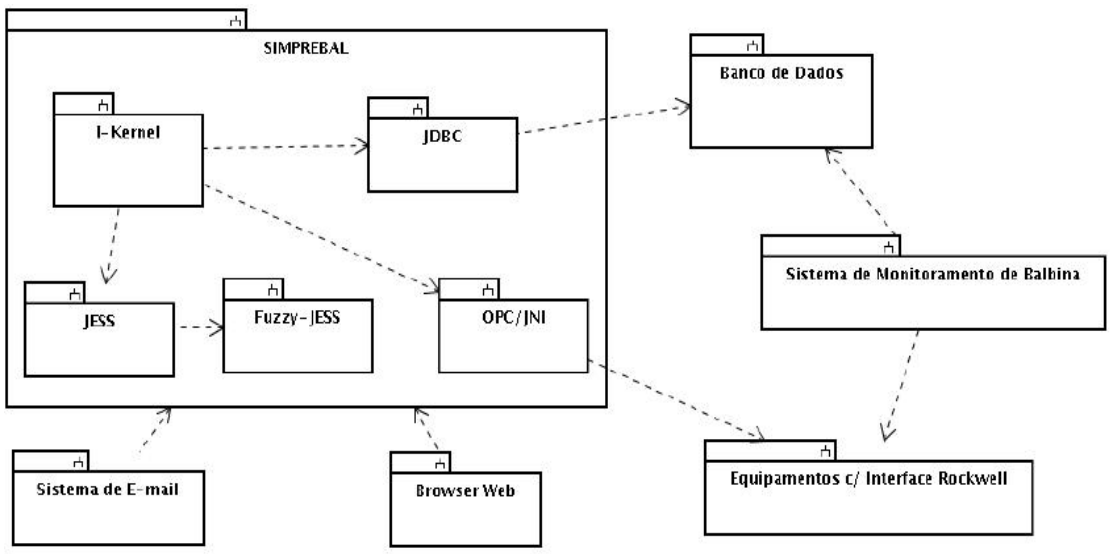


Figura 6 – Visão Geral do Projeto SIMPREBAL, demonstrando o I-Kernel e sua interação com outros módulos

3. **Monitoração de Condição:** Esta camada determina a condição do sistema atual, subsistema, ou indicadores de condição de componente baseando-se em algoritmos, sistemas inteligentes e na saída dos módulos de sensores e processamento de sinal. É possível fazer uso de histórico de condição local e fornecer parâmetros para o modelo. Duas dimensões devem ser consideradas: primeira, o conversor FieldBus e o sistema Ass*etview* da Smar fornecem análises relativa à condição da instrumentação *FieldBus*; segunda dimensão, o sistema SIMPREBAL fornece a condição do sistema monitorado. No caso de estudo:
   Se 138GMM1A > 80°C, então A temperatura do óleo estará alta.
   Se 138GMM1TA > 85°C, então A temperatura do óleo estará muito alta.

4. **Avaliação de saúde:** Na primeira dimensão o conversor *FieldBus* e o sistema *Assetview* da Smar fornecem a avaliação da saúde da instrumentação FieldBus. Na segunda dimensão o sistema SIMPREBAL fornece a avaliação da saúde do sistema monitorado. A saída desta camada é um índice de saúde do equipamento monitorado

5. **Prognósticos**: Na tarefa de prognóstico, considera-se a avaliação da saúde do sistema, subsistemas, componentes, o escalonamento empregado (predição de uso) e a capacidade de raciocínio do modelo para predizer o estado de saúde do equipamento. O sistema SIMPREBAL irá utilizar abordagem baseada de Sistemas Inteligentes baseado em Sistemas

Especialistas, Lógica Fuzzy e redes neurais, que serão desenvolvidos usando as ferramentas Jess, FuzzyJess, MatLab e OPC/JNI.

6. **Tomada de decisão:** Integra informações necessárias para a tomada de decisão, considerando informação sobre a saúde dos equipamentos e para predizer a saúde de um sistema e/ou subsistema. O sistema SIMPREBAL irá realizar a tomada de decisão baseando-se na sua base de conhecimento gerada a partir da árvore de faltas/falhas, da árvore de sintomas, e pelas informações de inspeções realizadas pelo sistema de Manutenção Preditiva Total (TPM).

7. **Apresentação:** Suporta a apresentação de informação para o controle das entradas dos usuários de sistemas (por exemplo, manutenção e operação). Saídas incluem qualquer informação produzida pelas camadas mais baixas e as entradas incluem qualquer informação requerida pelas camadas mais baixas. Fornece também a interface homem/máquina, que será desenvolvida para GUI (Interface Gráfica com o Usuário) baseada em browser (Netscape, Mozilla, IExplore, entre outros) usando html, XML, javascript e applets (Java). A forma de interagir do usuário com o sistema mostra-se na figura 7.

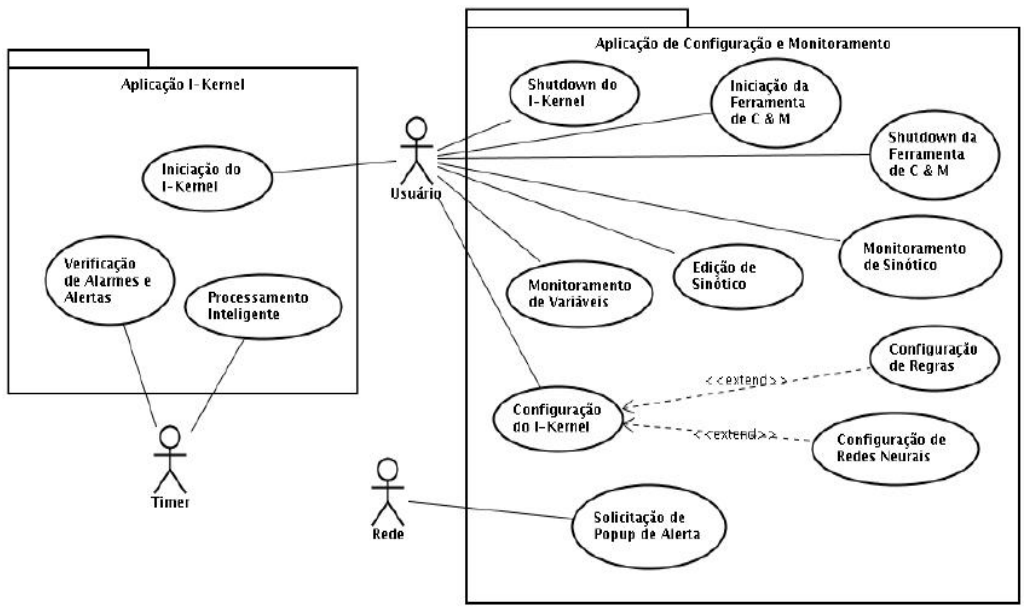


Figura 7 – Casos de Uso do Sistema

CONCLUSÃO

Com a proposta apresentada no presente artigo, espera-se realizar a tarefa de manuntenção antes que o equipamento apresente problema. Esse processo de manutenção é conhecido como manutenção preditiva e é baseada em confiabilidade. O sistema SIMPREBAL será desenvolvido usando técnicas de Inteligência Artificial (IA), o que facilitará o processo de tomada de decisão, visto que um dos objetivos do sistema é avaliar e monitorar o equipamento para oferecer prognósticos confiáveis. Permitindo dessa maneira que o operador possa tomar decisões de manutenção com base em informações geradas pelo sistema.

A metodologia envolve conceitos de diferentes áres do conhecimento que serão processados via técnicas de IA, como regras de produção e Lógica Fuzzy. O Sistema Inteligente SIMPREBAL auxiliará na tomada de decisão sugerindo prognósticos, para que a manutenção do equipamento possa ocorrer antes que a falha no mesmo ocorra. Isto implica em ganho de tempo e maior confiabilidade do sistema da usina. Vale ressaltar que a metodologia aqui proposta ainda está em fase de elaboração, resultados concretos serão obtidos posteriormente.

REFERÊNCIAS

TANAKA, K. WANG, H.O. (2001). Fuzzy control systems design and analysis. A linear matrix inequality approach. New York: John Wiley & Sons, 305 p.
SMAR (2001). Equipamentos de Campo série 302 Foundation. Manual de instalação, operação e manutenção. 42 p.
JANG, J.S.R. GULLEY, N. (1999). MATLAB fuzzy logic toolbox. User´s Guide. v.1. Natick: The Mathworks Inc, 235 p.
MATLAB (2006). OPC Toolbox for use with MATLAB and Simulink. User´s Guide. v.2. Natick: The Mathworks Inc, 373 p.
JANG, J.S.R. GULLEY, N. (1999). MATLAB Neural Network toolbox. User´s Guide. v.1. Natick: The Mathworks Inc, 235 p.
SMAR (2001). Equipamentos de Campo série 302 Foundation. Manual de instalação, operação e manutenção. 42 p.
SMAR (2005). Manual de instruções dos blocos funcionais Fieldbus Foundation. 334 p.
ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS, “NBR 6023: Informação e Documentação - Referências - Elaboração”, ABNT, Rio de Janeiro, 2000.
ELETRONORTE – CENTRAIS ELÉTRICAS DO NORTE DO BRASIL, [on line] Disponível na internet < http://www.eln.gov.br >, Acesso em: 08/05/2006.
FILHO, G. B., “Dicionário de Termos de Manutenção e Confiabilidade”, 2ª edição, Rio de Janeiro: Ciência Moderna, 2000.
FLEMING, P. V.; FRANÇA, S. R.R. O., “Considerações Sobre a Implementação Conjunta de TPM e MCC na Indústria de Processos”, ABRAMAN – 12° Congresso Brasileiro de Manutenção: TT044, São Paulo, 1997.
SIQUEIRA, I. P., “Manutenção Centrada na Confiabilidade – Manual de Implementação”, Qualitymark, Rio de Janeiro, 2005.
MIL-P-24534, “Planned Maintenance System: Development of Maintenance Requirement Cards, Maintenance Index Pages, and Associated Documentation”, Naval Sea Systems Command, U.S.
MOUBRAY, J., “RCM II – Reliability Centered Maintenance”, 2ª edição, New York: Industrial Press Inc, 1997.
WYREBSKI, J., “Manutenção Produtiva Total – Um Modelo Adaptado”, Dissertação (Mestrado em Engenharia de Produção) – Programa de Pós-Graduação em Engenharia de Produção, UFSC, 1998.

AGRADECIMENTOS

Agradecemos a Eletronorte, Manaus Energia e ao Eng. Antonio Araújo da Eletronorte.

Análise FMEA Para Aplicação da Metodologia de Manutenção Centrada em Confiabilidade: Estudo de Caso em Turbinas Hidráulicas

Alberto José Álvares[1], Rodrigo de Queiroz Sousa[1], Luciana Pereira Fernandes[1]
(1) Universidade de Brasília, Faculdade de Tecnologia, Departamento de Engenharia Mecânica, Curso de Sistemas Mecatrônicos. Campus Universitário Darcy Ribeiro, 70910-900 – Asa Norte – Brasília – DF, Brasil. (alvares@AlvaresTech.com, rodrigoqsouza@gmail.com, luciana.lpf@gmail.com)

RESUMO

O desenvolvimento das chamadas novas tecnologias tem promovido mudanças fundamentais na estrutura e nos processos de trabalho. A adoção mais intensa de sistemas automatizados e modernos equipamentos tem levado as áreas de manutenção a uma posição estratégica, em face da importância da disponibilidade operacional para o resultado global das empresas. As atividades de manutenção, neste contexto, devem ser realizadas de forma rápida e atender às expectativas de segurança, conservação de meio-ambiente, custo e qualidade. Para atender a esses requisitos foi desenvolvida uma metodologia de manutenção denominada Manutenção Centrada em Confiabilidade (MCC). Tal metodologia tem profundo embasamento no estudo das funções dos equipamentos e conseqüências das falhas.

O objetivo geral deste trabalho é avaliar o impacto da aplicação da MCC no sistema de geração energia elétrica. O FMEA (Failure Modes and Effects Analysis, Análise dos Efeitos e Modos de Falha) será a ferramenta que utilizada para fornecer o suporte no estudo das falhas do sistema de circulação do mancal de escora de uma turbina hidráulica do tipo Kaplan utilizada pela usina hidrelétrica (UHE) de Balbina – Amazonas ─ Brasil.

INTRODUÇÃO

A MCC (Manutenção Centrada em Confiabilidade) consiste em uma metodologia que visa determinar as tarefas adequadas a serem adotadas no plano de manutenção a fim de garantir a manutenibilidade das funções específicas de cada dispositivo, equipamento ou instalação.

Esta metodologia tem sido adotada amplamente no setor produtivo desde meados dos anos 70. Ela tem sido importante por atender às novas exigências de mercado, como qualidade e durabilidade dos bens de consumo, bem como determinar as tarefas que devem ser adotadas no plano de manutenção de modo a garantir a confiabilidade e a segurança operacional dos equipamentos e instalações ao menor custo.

Quando se usa a MCC para estabelecer um método de manutenção, é necessário ter em mente que os métodos de manutenção estabelecidos devem responder às seguintes perguntas:

- Quais as **funções** preservar?
- Quais as **falhas** funcionais?
- Quais os **modos** de falha?
- Quais os **efeitos** das falhas?
- Quais as **conseqüências** das falhas?
- Quais as **tarefas** aplicáveis e efetivas?
- Quais as **alternativas** restantes?

Para desenvolver um método de manutenção de acordo com as especificações da MCC foi utilizado como ferramenta de auxílio o FMEA. O FMEA é uma ferramenta que visa a identificação e análise

dos modos e causas de falhas nos níveis mais baixos do sistema empresarial e posterior estudo das conseqüências das falhas e adoção de medidas de manutenção. Com o FMEA é possível identificar com maior facilidade as ações que poderiam eliminar ou reduzir a chance de uma falha potencial ocorrer.

O FMEA pode fornecer, com maior facilidade e objetividade, respostas àquelas questões supracitadas referentes às falhas dos equipamentos. As etapas de implementação da MCC dependem das respostas a tais questões.

METODOLOGIA SIMPREBAL: MANUTENÇÃO BASEADA EM CONDIÇÃO ASSOCIADA A MCC

O contexto de aplicação do FMEA no atual artigo está associado ao projeto de modernização da área de automação de processos da usina hidrelétrica de Balbina. O objetivo deste projeto corresponde ao desenvolvimento da metodologia de um sistema de manutenção preditiva para a usina. Esta metodologia, denominada SIMPREBAL (Sistema Inteligente de Manutenção Preditiva de Balbina), é baseada em conceitos de manutenção centrada em confiabilidade (MCC), sendo utilizada para analisar as funções, os modos e efeitos de falhas das Unidades Geradoras Hidráulicas de Balbina a partir das grandezas monitoradas pelo sistema de supervisão e controle da usina. A análise está focada no sistema da turbina, entretanto a metodologia proposta é genérica, podendo ser utilizada também no sistema do gerador. O SIMPREBAL procura monitorar a condição atual do sistema e subsistemas utilizando instrumentação FieldBus fornecida pela empresa Smar. Com a monitoração das condições do sistema é possível desenvolver um sistema computacional com vistas à produção de diagnósticos de estados de funcionamento e de dados que auxiliem a tomada de decisão quanto a ações operacionais e de manutenção das máquinas visando o aumento da disponibilidade dos equipamentos. O sistema de manutenção preditiva da usina tem por base o banco de dados a ser gerado com os modos e efeitos de falhas adquiridos no FMEA.

ESTUDO DE CASO: SISTEMA DE CIRCULAÇÃO DO MANCAL ESCORA DA TURBINA

Dentre os critérios para a escolha do equipamento a ser desenvolvida a aplicação, optou-se pelo mancal de escora da turbina por ser o equipamento com maior número de falhas registradas nos históricos de manutenção da usina, representando aproximadamente 18% do número de falhas. Foram analisados os principais componentes mancal propriamente dito, bem como os componentes associados aos seus sistemas auxiliares (sistemas de injeção, circulação e resfriamento de óleo), resultando um total de 24 componentes.

O mancal de escora da UHE de Balbina é do tipo a patins com cuba fixa, apoiado sobre cone suporte e com sistemas auxiliares associados para dissipação de calor (sistemas de circulação de óleo e de resfriamento) e partida da unidade (sistema de injeção de óleo).

A função primordial do mancal é transferir os esforços axiais da turbina (empuxo hidráulico e peso próprio das peças girantes) ao concreto. No caso específico do mancal de escora de Balbina, os esforços não são simplesmente transmitidos, mas também equalizados por meio de um conjunto óleo-dinâmico (membranas e placas de assento). Diz-se, portanto, se tratar de um sistema auto-compensado.

O mancal é constituído de um conjunto de patins (segmentos) independentes, cada patim se apóia sobre uma membrana, fabricada em aço, com espessura fina, oferecendo uma grande flexibilidade. Estas membranas são fixadas sobre uma placa de assento, que transmite a carga ao suporte do mancal. A placa de assento possui canais que interligam as membranas. O conjunto formado por esses canais e pelas membranas forma uma câmara estanque, a qual é preenchida com óleo. A figura 1 apresenta a placa de assento com seu conjunto de membranas e a colocação de alguns patins sobre a mesma.

Fig. 1: Montagem dos patins sobre o conjunto de membranas do mancal escora.

Quando o mancal é solicitado, surge uma pressão no interior da membrana, a qual aumentará até se equilibrar com a carga vertical que é aplicada. Há uma distribuição quase perfeita da carga aplicada, entre os patins, logo, a pressão do óleo é a mesma sobre cada um dos patins, e o esforço necessário para deformar as membranas é desprezível, diante da carga suportada pelo mancal.

Os patins, e a parte inferior do anel de escora ficam imersos no óleo. A cuba de óleo permite a formação de um filme de óleo entre o colar de escora (rotativo) e os patins (fixos). Este filme permite a transferência de cargas consideráveis com pequenos dispêndios de energia. A espessura nominal do filme de óleo para UHE Balbina é 9,0 centésimos de milímetro.

Uma rótula é montada no interior de cada uma das membranas, com a finalidade de permitir o funcionamento do mancal em caso de falta de óleo no circuito interno das membranas. Portanto, a rótula é um dispositivo de segurança. A figura 2 mostra a representação de uma vista em corte do mancal de escora.

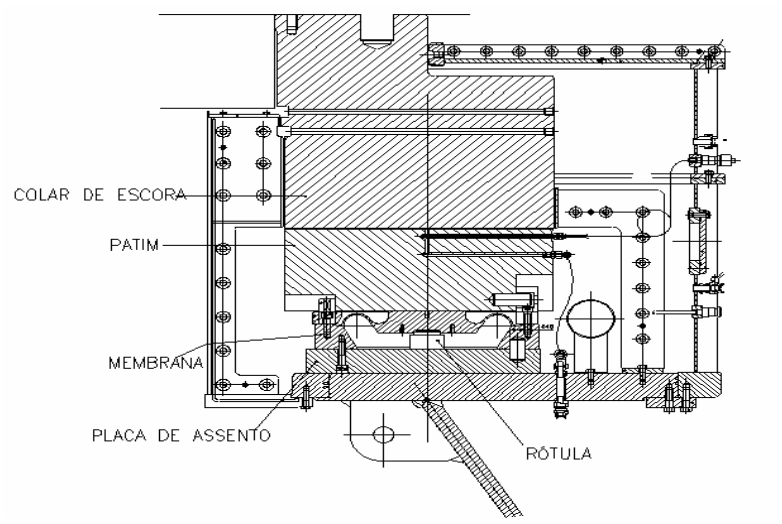


Fig. 2: Croqui do mancal de escora (vista em corte).

Delimitado o sistema, partiu-se para a análise de suas funções, modos e efeitos das falhas.Como o sistema do mancal de escora é complexo, optou-se por disponibilizar somente a planilha como o FMEA para o sistema de refrigeração do mancal de escora. Esta planilha pode ser visualizada na Tabela 1.

Tabela 1: FMEA do sistema de refrigeração do mancal de escora da turbina Kaplan.

FMEA DO MANCAL DE ESCORA DA UHE DE BALBINA - SISTEMAS AUXILIARES					FATORES PARA AVALIAÇÃO DO COMPONENTE						TAREFA PROPOSTA PARA MANUTENÇÃO
SISTEMA DE CIRCULAÇÃO					SEGURANÇA E MEIO AMBIENTE	CORTE DE CARGA	DETECÇÃO	SEVERIDADE	FREQÜÊNCIA DA FALHA	AVALIAÇÃO GERAL (NPR)	
FUNÇÃO	Dissipar o calor gerado no mancal de escora e lubrificar seus componentes										
COMPONENTES	FUNÇÃO DO COMPONENTE	FALHA FUNCIONAL	MODO DE FALHA	EFEITO DA FALHA (O que acontece quando falha)							
Motobomba Principal	Bombear óleo	Não obedece comando para partir	Quebra por desgaste excessivo ou engrupamento do mecanismo interno	Indicação de falha da bomba principal	1	3	1	7	2	168	- Ensaios funcionais - Inspeção Visual - Ensaio de medição de vibração - Lubrificar os rolamentos
				Partida da bomba auxiliar							
		Não obedece comando para parar		Operação desnecessária da bomba principal	4	1	1	6	1		
		Ruído anormal	Falta de óleo nos rolamento	Risco de quebra da bomba	2	1	3	5	2		
		Baixa pressão	Vazamento no selo mecânico	Vazamento de óleo	4	3	1	7	2		
				Risco iminente de indisponibilidade do sistema de circulação por pressão insuficiente							
Motobomba Auxiliar	Bombear óleo	Não obedece comando para partir	Quebra por desgaste excessivo ou engripamento do mecanismo interno	Indicação de falha da bomba auxiliar	1	5	1	7	1	280	- Ensaios funcionais - Inspeção Visual - Ensaio de medição de vibração - Lubrificar os rolamentos
				Parada emergencial da máquina							
		Não obedece comando para parar		Operação desnecessária da bomba auxiliar	4	1	1	6	1		
		Ruído anormal	Falta de óleo nos rolamento	Risco de quebra da bomba	2	1	3	5	1		

		Baixa pressão	Vazamento no selo mecânico	Indicação de falha da bomba principal Partida da bomba auxiliar	4	5	1	7	2		
Pressostato de partida e parada das motobombas	Indicar pressões baixas ou muito baixas, gerar alarmes, desligar e partir motobombas	Não partir as motobombas	Quebra do instrumento	Falta de reposição de óleo ao sistema	1	7	1	7	2	875	- Verificar ajuste - Inspeção Visual
				Risco de parada da máquina							
		Partir indevidamente as motobombas	Ajuste incorreto do contato	Indicação incorreta de pressão baixa	5	5	5	7	1		
				Distúrbio no funcionamento normal do sistema							
			Operação indevida	Risco de acidente pessoal							
		Não parar as motobombas	Quebra do instrumento	Sobrepressão	4	1	1	6	1		
				Vazamento de óleo							
		Parar indevidamente as motobombas	Ajuste incorreto do contato	Distúrbio no funcionamento normal do sistema	5	1	5	6	1		
			Operação indevida	Risco de acidente pessoal							
Manômetro das motobombas	Indicar localmente a pressão de saída das motobombas	Indicar valor incorreto de pressão	Descalibração do instrumento	Leitura de pressão incorreta	1	1	5	3	2	30	- Calibrar - Inspeção Visual
				Risco de ajuste incorreto de outro instrumentos							
		Não indicar medição de pressão	Quebra devido ao engripamento do mecanismo	Perda de indicação	1	1	1	3	1		
Painel Elétrico	Indicar alarmes, trips e condições de operação do sistema	Não indicar alarmes	Queima dos LED's	Perda de indicação	1	1	1	3	2	6	- Verificar posição, nível e alinhamento - Verificar fixação e estado geral dos componentes internos no painel
			Mau contato na fiação								

											- Verificar etiquetas de identificação dos cabos e terminais - Verificar lâmpadas de sinalização - Verificar aterramento geral
Tubulações e conexões	Conduzir óleo	Não conduzir o óleo corretamente	Vazamento por ruptura da tubulação	Risco de acidente pessoal	5	3	5	6	2	900	- Inspeção Visual dos estado geral das tubulações - Drenar e purificar o óleo
				Impossibilidade de reposição de óleo							
			Obstrução por deformação permanente	Risco de parada da máquina							
			Obstrução por contaminação do fluido	Distúrbio no funcionamento normal do sistema							
Válvulas de isolamento	Isolar acessórios de supervisão e controle	Não isolar os acessórios de supervisão e controle	Falta de aperto no fechamento da válvula	Impossibilidade de executar manutenção nos acessórios de supervisão e controle	4	1	7	7	1	245	-Teste operacional - Verficação de pressão de atuação - Inspeção Visual
		Isolar inadequamente os acessários de supervisão e controle	Engripamento da haste provocando vazamento interno	Risco de acidente pessoal	5	1	7	7	1		
Válvulas de alívio de pressão	Limitar a pressão em caso de obstrução do circuito	Não aliviar a pressão	Emperramento	Rompimento da tubulação e vedações	5	5	1	8	1	200	-Teste operacional - Verficação de pressão de atuação - Inspeção Visual
			Obstrução por quebra de mecanismo interno	Quebra da motobomba							
		Operar abaixo da pressão máxima	Ajuste indevido	Distúrbio no funcionamento normal do sistema	1	1	4	5	2		
			Desgaste na sede da válvula								

O FMEA foi obtido através da determinação das falhas e modos de falhas de cada um dos componentes que fazem parte do sistema de circulação da turbina. Para cada falha foi indicado o seu efeito no sistema da turbina. O índice severidade aplicado a cada modo de falha corresponde à gravidade de impacto que cada falha produz na hidrelétrica. O índice detecção é a probabilidade de uma dada falha ocorrer. Ocorrência é a quantidade de ocorrência de falha em um período de 3 anos. O índice Segurança e meio ambiente caracteriza em que intensidade a falha pode afetar ou a segurança das pessoas que trabalham no local ou o meio ambiente local. O corte de carga corresponde à parada da turbina. A Tabela 2 mostra o significado dos valores correspondentes a cada classificação da Tabela 1.

Tabela 2: Classificação dos índices utilizados no FMEA.

Severidade	
1	Muito Baixa
2-3	Baixa
4-5-6	Moderada
7-8	Alta
9	Muito Alta
10	Catastrófica
Ocorrência	
1	Menor ou igual a 1 em 10 anos
2	1 falha no período analisado
3	2 falhas
5	3 falhas
7	4 falhas
10	5 ou mais falhas
Detecção	
1	Probabilidade muito alta de detecção
2-3	Probabilidade alta de detecção
4-5-6	Probabilidade moderada de detecção
7-8	Probabilidade pequena
9	Probabilidade muito pequena
10	Probabilidade remota
Segurança e meio ambiente	
1	Não afeta a segurança e/ou o meio ambiente
2-3	Remota possibilidade de afetar a segurança e/ ou o meio ambiente
4-5-6	Possibilidade moderada de afetar a segurança e/ou o meio ambiente
7-8	Grande possibilidade de afetar a segurança e/ ou o meio ambiente
9-10	Afeta a segurança e/ ou o meio ambiente
Corte de carga	
1	Não provoca corte de carga
3	Risco remoto de provocar corte de carga
5	Risco moderado de provocar corte de carga
7	Provoca corte de carga de até 5% da carga da instalação
10	Provoca corte de carga maior ou igual a 5% da demanda máxima da instalação

RESULTADOS

A análise FMEA mostrou que, para as 8 funções principais do sistema de circulação de óleo do mancal de escora, foram identificadas 16 falhas funcionais, que indicaram 20 modos de falha. A

partir destes modos de falha foram propostas 4 tarefas corretivas – lubrificar rolamentos, calibrar instrumentos de medição, drenar tubulações e purificar o óleo – além de 11 tarefas preventivas, que incluem verificação das condições de operação e busca de falhas.

O FMEA indicou ainda que os componentes com maior número de prioridade de risco (NRP) são os pressostatos e as tubulações. Os pressostatos devido a sua importante função de controle automatizado do sistema, ligando e desligando as motobombas de acordo com a pressão medida e por ser um dispositivo de inspeção sujeito a falhas ocultas. E as tubulações devido à dificuldade em se detectar os locais vazamentos ou obstruções e à grande severidade de se ter uma má condução de óleo no sistema. Tal fato poderia provocar a não formação do filme de óleo entre os patins e o colar de escora, levando ao contato direto metal-metal e provocando graves avarias no mancal. Uma forma de se prevenir este tipo de falha seria a instalação de fluxostatos nos pontos críticos da tubulação.

Pode ser observada a potencialidade da aplicação da análise FMEA, no contexto de implementação da metodologia MCC, no sentido de otimizar o plano de manutenção, definindo as tarefas a serem contempladas nesse plano, sejam preventivas ou corretivas, e a oportunidade de realização de discussões técnicas com profundidade suficiente para uma reavaliação dos procedimentos de manutenção atualmente adotados e resgate do conhecimento dos profissionais envolvidos no processo de manutenção.

CONCLUSÕES

O presente artigo conseguiu expor a aplicabilidade da metodologia MCC a partir do desenvolvimento da análise FMEA. O FMEA se mostrou uma ferramenta importante de suporte ao desenvolvimento do sistema inteligente baseado em MCC - o SIMPREBAL. O sistema que visa a manutenção preditiva na usina hidrelétrica de Balbina fornece o devido enfoque à preservação das funções dos equipamentos.

O estudo de caso do sistema de circulação de óleo do mancal de escora revelou que existem dois componentes críticos no sistema, os pressostatos de partida e parada das motobombas e a tubulação, portanto, os procedimentos de manutenção preventiva devem dar destaque a estes componentes. Por fim, foram sugeridas tarefas aplicáveis para manutenção, para cada um dos componentes, baseadas nos modos de falha encontrados.

REFERÊNCIAS

ELETRONORTE – CENTRAIS ELÉTRICAS DO NORTE DO BRASIL S/A, “*Relatório Mensal de Operação*”, maio de 2002.

Moubray, J., *“RCM II – Reliability Centered Maintenance”*, 2ª edição, New York: Industrial Press Inc, 1997.

Siqueira, I. P., *“Manutenção Centrada na Confiabilidade – Manual de Implementação”*, Qualitymark, Rio de Janeiro, 2005.

Hammet, P., 2000, “Failure Modes and Effects Analysis, Michigan, USA, 9p.

Eti, M. C.; Ogaji, S. O. T.; Probert, S. D., *“Development and implementation of preventive maintenance practices in Nigerian industries”*, Applied Energy xxx (2006) xxx-xxx, [on line] disponível na internet < http://www.sciencedirect.com >

Filho, G. B., *“Dicionário de Termos de Manutenção e Confiabilidade”*, 2ª edição, Rio de Janeiro: Ciência Moderna, 2000.

Bigatto, B. V.; Moretti, D. C., *“Aplicação do FMEA: Estudo de caso em uma empresa de transporte de cargas”*, NORTEGUBISIAN

AGRADECIMENTOS

Agradecemos a Eletronorte, a Manaus Energia e ao Eng.° Antônio Araújo da Eletronorte pelo material cedido.