Universidade Federal do Rio Grande do Norte
Programa de Pós-Graduação em Engenharia Elétrica e de Computação

SUPERVISÓRIO INTELIGENTE DE PROCESSOS NA INDÚSTRIA DO PETRÓLEO E GÁS: APLICAÇÃO EM UMA COLUNA DE DESTILAÇÃO SIMULADA INTEGRADA A INSTRUMENTAÇÃO REAL

Thiago Medeiros Barros

Natal - 2012

Universidade Federal do Rio Grande do Norte
Programa de Pós-Graduação em Engenharia Elétrica e de Computação

SUPERVISÓRIO INTELIGENTE DE PROCESSOS NA INDÚSTRIA DO PETRÓLEO E GÁS: APLICAÇÃO EM UMA COLUNA DE DESTILAÇÃO SIMULADA INTEGRADA A INSTRUMENTAÇÃO REAL

Trabalho de Dissertação de Mestrado, submetido ao Programa de Pós-Graduação em Engenharia Elétrica e de Computação do Centro de Tecnologia da Universidade Federal do Rio Grande do Norte, como parte dos requisitos necessários para obtenção do grau de Mestrado.

Thiago Medeiros Barros

Resumo

A necessidade de desenvolver uma arquitetura que promovesse a monitoração de softwares de simulação através de supervisórios, agregando módulos inteligentes e equipamentos como CLP's de acordo com necessidade do problema, foi a força motriz desse trabalho. No presente estudo será desenvolvido um supervisório inteligente sobre uma simulação modelada no Unisim. Será utilizado como comunicação entre o supervisório e o software de simulação o *OLE Automation*, o qual, junto ao uso de banco de dados, promoverá uma arquitetura de fácil manutenção e escalável. Agregando valor ao trabalho, serão desenvolvidos módulos inteligentes para pré-processamento e extração de características dos dados, e inferência de variáveis. Esses módulos terão como base principal o *software* Encog.

Palavras chaves: supervisório inteligente,Encog, OLEAutomation, simulação, coluna de destilação, SCADA

Abstract

The need to develop an architecture that promotes the monitoring of simulation software through supervisory, adding modules and smart devices such as PLCs according to the need of the problem, was the focus of this work. This present study will develop an intelligent supervisory on a simulation modeled in Unisim. It was used as communication between the supervisor and simulation software the technology OLE Automation, which with the use of the database will promote an architecture for scalable and easy to maintain. Adding value to the work, will be developed intelligent modules for pre-processing to feature extraction data, and inference variables. These modules will be based on the package Encog.

Palavras chaves: intelligent supervisory, Encog, *OLE Automation*, simulation,distillation column, SCADA

Lista de Figuras

Conteúdo

1. Introdução

Os níveis de hierarquia em sistemas de automação podem ser definidos como (Lima, 2004): Processos Físicos, Sensores e Atuadores, Controle Regulatório, Alarme e Intertravamento, Supervisão, Gerência.

Na camada de supervisão é possível encontrar desde sistemas mais simples, apenas com interfaces homem-máquina (IHM) locais, até a ilhas de supervisão equipadas com computadores poderosos e com os sistemas SCADA. O termo SCADA (*Supervisory Control And Data Acquisition*) na automação é utilizado para denominar sistemas de supervisão, controle e aquisição de dados composto por um ou mais computadores monitorando e controlando um processo (Viana, 2008).

Sistemas de controle computadorizados que monitora, controla e realiza diagnósticos sobre processos de larga escala geram um número excessivo de variáveis, fazendo com que operadores encontrem dificuldades para efetivamente monitorar esses dados, analisar o estado atual, detectar e diagnosticar anomalias e realizar ações apropriadas de controle. A fim de assistir o operador, informações do processo devem ser analisadas e apresentadas de maneira que reflita as principais tendências dos dados ou eventos dos processos (Rengaswamy & Venkatasubramanian, 1992). Sistemas de suporte a decisão inteligente incorpora uma variedade de técnicas (que podem ser de IA ou não), a fim de realizar o monitoramento, controle e diagnósticos sobre o processo. As técnicas incluem: orientadas a dados, analítica, baseado em conhecimento (Uraikul, W. Chan, & Tontiwachwuthikul, 2007).

Uma abordagem robusta proposta é a combinação dessas três técnicas, a fim de que através da integração seja possível minimizar os pontos fracos e maximizar os pontos fortes, entretanto tal estratégia vem se mostrando complexa em sua implementação real. Em Uraikul, W. Chan, & Tontiwachwuthikul (2007) expõe quatro propostas de *framework* que tentam executar tal abordagem.

O processo monitorado e controlado pelo SCADA não necessariamente é um processo físico real, podendo ser também uma simulação de uma planta industrial.

Pode-se definir que simulação é a obtenção da resposta temporal das variáveis de interesse (variáveis dependentes) de um modelo, quando se excita suas variáveis de entrada com sinais desejados e se definem os valores das condições iniciais das variáveis dependentes (Garcia C. , 1997). Simulação também pode ser definido como o processo de projetar o modelo de um sistema real e

conduzir experimentos com este modelo com o propósito de entender o comportamento do sistema e/ou avaliar várias estratégias de operação do sistema (Shannon, 1998).

A simulação pode ser definida como sendo "o desenvolvimento de um modelo lógico que reproduz a realidade, a fim de avaliar o comportamento e o desempenho de sistemas sob as mais variadas condições". Lembra ainda que um modelo cuidadosamente concebido pode eliminar grande parte da complexidade inicialmente imaginada, deixando apenas características que o analista considere como importantes (Pegden, 1990).

Uma coluna de destilação consiste em um equipamento que permite a separação dos componentes de uma mistura de líquidos miscíveis e é baseada na diferença de temperatura de ebulição de seus componentes individuais (Henley & Seader, 1981). Entre as vantagens da utilização deste processo encontra-se a flexibilidade de operação em relação à pressão, temperatura, volumes e principalmente à grande variedade de aplicações. É importante falar que na maioria das indústrias de transformação, 80% do custo operacional energético se encontra na operação de controle e outras vezes a coluna é o equipamento que impede o aumento da produção (Werle, 2007).

Os primeiros trabalhos, com metodologias propostas para a solução de sistemas de separação modelados prato a prato, surgiram na década de 30. Porém só a partir da década de 50, com o advento do computador digital, é que foram realizados investimentos sólidos no desenvolvimento de novos algoritmos e simuladores. Apesar deste investimento, apenas modelos simplificados eram utilizados devido à baixa capacidade de processamento. A partir de 70, os primeiros simuladores comerciais começaram a ser introduzidos na indústria e o desenvolvimento de modelos rigorosos não parou mais (Staudt, 2007).

Softwares de simulação como *Unisim* e o *Hysys* já obtiveram na academia e na indústria resultados comprovados de sua eficiência na simulação em colunas de destilação (Soos & Smejkal, 2001).

O objetivo deste trabalho é o desenvolvimento de um sistema supervisório inteligente, a fim de monitorar e prover um sistema de suporte à decisão para plantas simuladas em um software de simulação do processo de destilação de uma coluna. Além disso, tem o intuito de construir uma arquitetura capaz de agregar com facilidade novos módulos, tal como inferência de variáveis utilizando diversas técnicas inteligentes. A Figura 1 mostra a arquitetura do supervisório proposto.

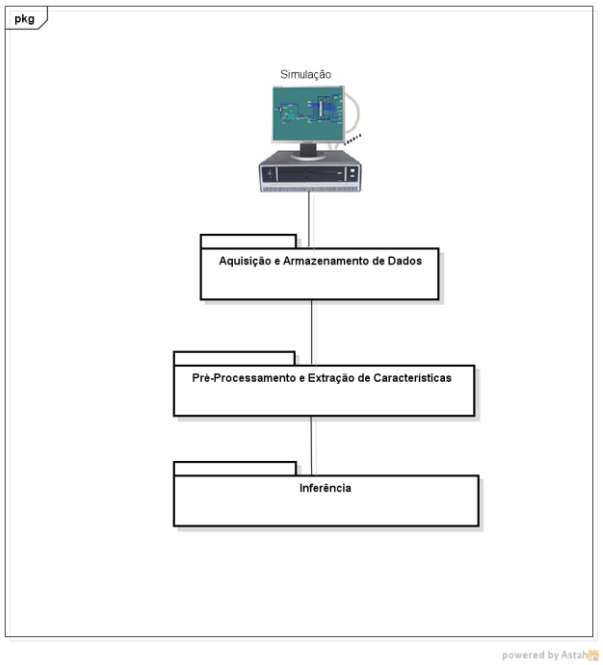


Figura 1: Arquitetura do Supervisório Desenvolvido

2. Objetivos

2.1 Objetivo Geral

- Desenvolver um sistema supervisório inteligente para monitoração de um processo de destilação na indústria do Petróleo simulado e de apoio à decisão.

2.2 Objetivos Específicos

- Desenvolver a comunicação entre o software de simulação e o sistema supervisório através do *OLE Automation*.
- Definir arquitetura de software que seja de fácil manutenção e escalável.
- Desenvolver módulos inteligentes para pré-processamento e extração de características dos dados
- Desenvolver módulos inteligentes de inferência de variáveis.

3. Revisão Bibliográfica

3.1 Sistemas de Automação Industrial

O crescente avanço tecnológico nas mais diversas áreas do conhecimento humano tem se mostrado, nos últimos anos, surpreendente. A utilização de automação nas indústrias tem sido cada vez maior, proporcionando um aumento na qualidade e quantidade da produção e, ao mesmo tempo, oferecendo preços atrativos. Ou seja, a utilização da automação aumenta a eficiência, tornando as empresas competitivas no mercado. Portanto, trata-se de um caminho de uma única mão. Para se fazer frente à concorrência procura-se aumentar a produtividade (razão entre o volume produzido e os recursos empregados), reduzir custos de produção e aumentar a qualidade dos produtos oferecidos. Ao mesmo tempo, para atender às exigências de diversidade do mercado consumidor e a gradativa redução da vida útil dos produtos, procura-se ampliar a flexibilidade na utilização dos sistemas produtivos (Maitelli, 2003).

A automação industrial pode ser definida como um conjunto de técnicas destinadas a tornar automáticos vários processos na indústria, substituindo o trabalho muscular e mental do homem por equipamentos diversos. O conceito de automação varia com o ambiente e experiência da pessoa envolvida. São exemplos de automação (Maitelli, 2003):

- Para uma dona de casa, a máquina de lavar roupa ou lavar louça.
- Para um empregado da indústria automobilística, pode ser um robô.
- Para uma pessoa comum, pode ser a capacidade de tirar dinheiro do caixa eletrônico.

A justificativa básica para automatizar um processo industrial, ou uma parte dele, pode se basear nas vantagens apontadas abaixo (Maitelli, 2003):

1. Qualidade: ao se produzir em uma faixa de tolerância à falhas estreitas através de um controle de qualidade eficiente, de uma compensação automática de deficiências do processo, ou do uso de tecnologias mais sofisticadas, obtém-se uma qualidade maior na produção;
2. Produtividade: obtida a partir do uso mais eficiente da matéria prima, energia, equipamentos e instalações através, por exemplo, da produção de refugo zero, como conseqüência de uma supervisão da qualidade, na qual o mínimo de matéria prima é desperdiçado;

3. Flexibilidade: consiste na capacidade de admitir com facilidade e rapidez, alterações nos parâmetros do processo de fabricação, em função de inovações freqüentes no produto, do atendimento a especificidades do cliente, ou da produção de pequenos lotes;

4. Viabilidade Técnica: permite que o sistema execute operações impossíveis de realizar por métodos convencionais.

3.2 Níveis de Hierarquia em Sistemas de Automação

A quantidade de níveis hierárquicos dentro de um sistema depende fundamentalmente do tamanho do processo e das necessidades relacionadas à planta industrial (Lima, 2004).

Um sistema de automação industrial genérico pode ser caracterizado em cinco níveis hierárquicos, conforme a Figura 2, sendo que em algumas plantas não há a presença de todos (Junior, 2007).

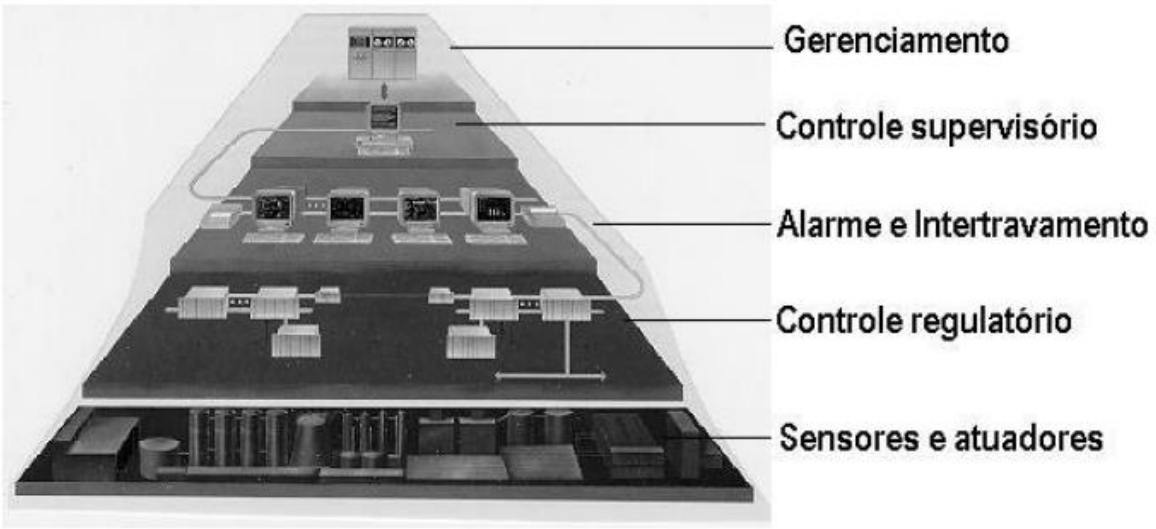


Figura 2: Pirâmide hierárquica dos sistemas de automação

- Processos Físicos: Consiste na base da pirâmide, estando presente em todos os sistemas de automação. Aqui estão inclusos todos os processos a serem automatizados. Neste nível encontramos o chão-de-fábrica: tanques, bombas, caldeiras, robôs, esteiras, motores, entre outros.

- Sensores e Atuadores: Camada que contém os dispositivos encarregados de manipular o processo produtivo, tomando as medidas necessárias a partir das informações enviadas pelo nível superior. Por ser a camada mais próxima do processo controlado, também possui a responsabilidade de fazer a aquisição dos dados.

- Controle Regulatório: Os sensores e atuadores do nível inferior estão diretamente conectados aos dispositivos desta camada. Os controladores de malha, CLP's, Sistemas Digitais de Controle Distribuído (SDCD), são exemplos de dispositivos que implementam o controle regulatório.

- Alarme e Intertravamento: Nível responsável pela segurança do processo como um todo. No momento em que os circuitos de segurança detectam falhas no sistema, os equipamentos são levados a um estado seguro (intertravamento), no qual a produção é preservada. Logo em seguida, alarmes são acionados para que os engenheiros de processo e automação possam tomar ações corretivas.

- Supervisão: Camada responsável pela emissão de relatórios de operação. Sua configuração varia desde sistemas mais simples, apenas com interfaces homem-máquina (IHM) locais, até a ilhas de supervisão equipadas com computadores poderosos e com os sistemas SCADA.

- Gerência: Consiste no nível mais elevado da hierarquia, sendo realizadas análises de dados por um corpo administrativo, o qual toma decisões de caráter econômico e de marketing.

3.3 Sistemas Supervisórios

O termo SCADA (*Supervisory Control And Data Acquisition*) na automação é utilizado para denominar sistemas de supervisão, controle e aquisição de dados composto por um ou mais computadores monitorando e/ou controlando um processo (Viana, 2008).

O processo pode ser industrial, infra-estrutura ou facilidade conforme descrito a seguir (Viana, 2008):

- Processos industriais incluem manufatura, geração de energia, refino de petróleo e muitos outros. Podem ser executados de forma contínua ou batelada. Os sinais tratados podem ser tanto analógicos quanto digitais;

- Processos de infra-estrutura podem ser públicos ou privados, e incluem tratamento e distribuição de água, coleta e tratamento de esgoto, linhas de óleo e gás, transmissão e distribuição de energia elétrica, e grandes sistemas de comunicação;

- Processos de facilidade ocorrem em instalações públicas e privadas, incluindo edifícios, aeroportos, navios, plataformas *offshore* e estações espaciais. Esses sistemas monitoram e controlam HVAC (*Heating, Ventilation and Air Conditioning*) e consumo de energia.

O objetivo principal dos sistemas SCADA é propiciar uma interface de alto nível do operador com o processo informando-o "em tempo real" de todos os eventos de importância da planta.

Historicamente temos que os softwares SCADA (*Supervisory Control And Data Acquisition*) surgiram em diversos tamanhos, diversos sistemas operacionais e com diversas funcionalidades englobadas. Na área de instrumentação, fez-se necessário uma adequação dos instrumentos para torná-los mais inteligentes. Logo, o padrão para transmissão de sinais analógicos de 4-20mA cedeu espaço para a transmissão digital de dados. Exemplos de alterações que comprovam integralmente o avanço associado à automação industrial (Maitelli, 2003).

Uma nova abordagem sobre sistemas supervisores são os sistemas supervisores inteligentes. A grande virtude de se utilizar supervisórios inteligentes é a possibilidade de realizar processamento simbólico e numérico de forma simultânea, valendo-se das vantagens de ambos (Nascimento JR. & Yoneyama, 2000).

Já há no mercado ferramentas disponíveis que utilizam o conceito de supervisórios inteligentes, utilizando técnicas como lógica nebulosa, mineração de dados e outros, a fim de tentar prever comportamentos do sistema ou descobrir novos padrões, com o intuito de atuar na área de apoio à decisão.

3.4 Supervisórios Inteligentes

Como dito anteriormente, Sistemas de controle computadorizados que monitora, controla e realiza diagnósticos sobre processos de larga escala geram um número excessivo de variáveis, fazendo com que operadores encontrem dificuldades para efetivamente monitorar esses dados, analisar o estado atual, detectar e diagnosticar anomalias e realizar ações apropriadas de controle.

Para isso, surgem os sistemas inteligentes de suporte à decisão, o qual incorpora uma variedade de técnicas (que podem ser de IA ou não), a fim de realizar o monitoramento, controle e diagnósticos sobre o processo.

É importante notar que Sistemas Inteligentes que são construídos integrando as diferentes abordagens podem melhorar a performance da planta, reduzir custos de operação, facilitar o trabalho do operador em tarefas intensivas, e permitir ao operador estar mais informado e realizar melhores decisões para tornar a planta mais segura. As abordagens mais comuns são: Orientadas a Dados, Analítica e Baseadas em Conhecimento (Chiang, Russell, & Braatz, 2001).

Orientadas a Dados

Detecções e diagnósticos precoces e precisos de falhas em processos industriais podem minimizar o tempo de inatividade, aumentar a segurança em plantas de processo e reduzir os custos de produção. As técnicas de monitoramento de processo, que tem se mostrado eficiente na prática, são baseados em modelos construídos quase que inteiramente sobre os dados do processo (Chiang, Russel, & Braatz, 2001). As abordagens mais populares são: análise de principais componentes (PCA em inglês), análise discriminante de Fisher, análise de mínimos quadrados parciais (PLS em inglês), e análise de variáveis canônicas. Dentre elas, PCA e PLS vem sendo mais adotada para extração de características sobre o histórico dos dados de processo (Yoon & MacGregor, 2004).

Analítica

A abordagem analítica geralmente envolve detalhados modelos matemáticos que utilizam alguma medição de entrada u e uma saída y, e geram características como resíduos r, estimação de parâmetros p, e estimação de estados x. Então, baseados nesses valores, detecção de falhas e diagnósticos pode ser realizados por comparação dos valores observados com aqueles em condições normais de funcionamento, quer diretamente, quer após algumas transformações. Métodos analíticos podem ser categorizados dentro de dois métodos comuns de estimação de parâmetros e baseado em observação (Garcia & Frank, 1996). No método de estimação de parâmetro, um resíduo é definido como a diferença entre o valor nominal e os parâmetros do modelo estimado, e os desvios nos parâmetros do modelo servirão de base para a detecção e isolamento de falhas (Isermann, 1994). No método baseado em observador, a saída do sistema é reconstruída usando o valor medido ou um subconjunto das medições com ajuda dos observadores. A diferença entre as medidas e as saídas estimadas é usada como vetor de resíduos (Ding & Guo, 1996). Quando há modelos matemáticos precisos disponíveis, os modelos analíticos podem melhorar os sistemas de monitoramento do processo, principalmente quando são acopladas as abordagens orientadas a dados ou conhecimento.

Baseadas em Conhecimento

Abordagens baseadas no conhecimento, aplicado em sistemas de raciocínio automatizado, envolvem informações incertas, contraditórias e não quantificáveis (Luo, Zhang, & Jennings, 2002). As tecnologias de inteligência artificial que são associadas com abordagens baseadas no conhecimento e adotadas para monitoramente, controle, e diagnósticos nos processos industriais incluem sistemas especialistas, lógica fuzzy, aprendizado de máquina e reconhecimento de padrões.

Integração das várias técnicas

A experiência mostra que não há uma única técnica em sistemas inteligentes que contemplem a extração de todo o conhecimento necessário para todas as atividades que serão desenvolvidas. Uma abordagem robusta proposta é a combinação dessas três técnicas, a fim de que através da integração seja possível minimizar os pontos fracos e maximizar os pontos fortes. Entretanto tal estratégia vem se mostrando complexa em sua implementação real. Em (Uraikul, W. Chan, & Tontiwachwuthikul, 2007) expõe quatro propostas de *framework* que tentam executar tal abordagem.

Na comparação dos quatro *frameworks* IRTW, INTEMOR, OOSE e DKIT em termos das suas abordagens, uma série de semelhanças e diferenças podem ser observadas. Os quatro sistemas combinam os três métodos de solução, mas a prioridade que atribuem às abordagens são diferentes. INTEMOR, DKIT IRTW dá maior prioridade para o sistema baseado em conhecimento, considerando que a abordagem de solução OOSS atribuí prioridade à solução orientada a dados. DKIT e IRTW adota o mesmo esquema de atribuição de prioridade: abordagem baseadas em conhecimento, abordagens analíticas e, em seguida, orientada a dados; enquanto INTEMOR enfileiras a primeira abordagem, em seguida, as abordagens orientadas a dados e analítica. A classificação dos métodos de solução em OOSS é orientada a dados, analítico, a abordagem baseada em conhecimento. Embora os quatro sistemas integrados possui um grupo amplo de tarefas envolvidas em sistemas de processo, a ênfase também é diferente. INTEMOR se concentra principalmente na condição de sistema de monitoramento, diagnóstico de falhas e suporte de manutenção, enquanto a ênfase OOSS está no diagnóstico de falhas. Ambos DKIT e IRTW enfrentam o processo de acompanhamento, supervisão e diagnóstico das operações. Os quatro *frameworks* adotam as três abordagens, mas cada um tem seu próprio método de integração. INTEMOR usa um meta-sistema para coordenar as diferentes abordagens e integrar os mecanismos de representação de diferentes conhecimentos, já o DKIT inclui um *Coordenador* para gerenciar as diferentes abordagens e de resolução de conflitos. O meta-sistema inclui um banco de dados, base de regra e motor de inferência que pode ser usado para gerenciar e coordenar o raciocínio simbólico e cálculo numérico no sistema durante as várias fases de coleta e processamento de informações. O planejador de DKIT por outro lado, analisa as

recomendações geradas pelas diferentes abordagens, avalia os seus pontos fortes e fracos e produz um resultado final após a aplicação de algoritmos de resolução de conflitos baseado em um esquema de votação ponderada, derivado da teoria de probabilidade. O quadro OOSS inclui três camadas com as funcionalidades específicas de (1) geração de determinação e erro, extração de características (2) e reconhecimento de padrões, e (3) a criação de modelos, enquanto IRTW inclui uma hierarquia de modelos de dados, que descrevem diferentes tipos de controladores , atuadores, sensores, restrições lógicas, modelos de processos, falhas e processos em vários níveis de abstração.

Em termos de pontos fortes e fracos dos *frameworks*, a força principal é que todos eles promovem a integração das três abordagens de solução. O INTEMOR não só integra as abordagens, mas também esquemas de conhecimento, diferentes tarefas de representação e controle do processo. OOSS define claramente o fluxo de informações entre as operações de controle de processo, enquanto DKIT tem uma estrutura, que garante a solução dos problemas serem compartilhadas entre os diferentes módulos, por meio de uma estrutura de dados global. IRWT por outro lado, oferece uma estrutura genérica integrada para combinar as diferentes funcionalidades em operações de processo.

Os *frameworks* também sofrem de algumas fraquezas. O INTEMOR foi concebido para coordenar as abordagens de solução diversa, de representação de conhecimento e tarefas de controle de processo, se o domínio do problema abordado é complexo e envolve uma ampla base de conhecimento que pode crescer em tamanho, então o motor de inferência de meta-nível precisa lidar com um grande e, possivelmente, um número crescente de soluções plausíveis. Escalabilidade se tornaria um problema. Em contrapartida, OOSS visa integrar as funções de geração de erros, extração de características, reconhecimento de padrões e criação de modelo, e coordena os três métodos de solução através de uma estrutura de três camadas. No entanto, embora a estrutura em três camadas forneça um bom suporte em termos de fluxo de informações entre as operações de controle de processo, não está claro como a coordenação das diferentes tarefas envolvidas no controle do processo pode ser alcançada. Em comparação, o quadro DKIT definiu os mecanismos para coordenar as tarefas de controle de processos de controle, diagnóstico e supervisão. Enquanto a estrutura no DKIT pode apoiar adequadamente a organização dos módulos diversos, não se pode especificar os procedimentos computacionais envolvidos. O IRTW pode integrar as tarefas de controle e supervisão, diagnóstico, e monitoramento de processos e da estrutura de dados hierárquica, a qual oferece uma estrutura genérica integrada para combinar as diferentes funcionalidades. No entanto, a sua fraqueza é que a estrutura hierárquica de dados não pode

acomodar qualquer alteração no fluxo de informações. Em outras palavras, não é flexível e adaptável.

3.5 Inferência de Variáveis

Uma má estratégia de controle pode resultar em longos períodos de funcionamento com produtos mais puros do que o necessário e, consequentemente, um consumo maior de energia, ou produtos que não atendem as especificações definidas. Na literatura também é mostrado que um controle efetivo das colunas de destilação pode melhorar a qualidade do produto final, diminuir o consumo de energia e aumentar a capacidade de produção e a segurança do processo. No entanto, para realizar controle ou supervisão em plantas desse tipo faz-se necessário obter medições *online* de variáveis que muitas vezes são compostos químicos que estão diretamente relacionados com a qualidade dos produtos finais e, por esse motivo, são de grande interesse econômico (Rebouças, 2009).

Mensurar ou estimar variáveis desse tipo passa a ser uma tarefa difícil a partir do momento em que existem limitações tecnológicas ou devido às técnicas de medição utilizadas. Também podemos encontrar na literatura que os instrumentos que realizam a medição direta dessas composições são normalmente caros, de difícil manutenção e introduzem atrasos de medição significativos, o que impossibilita a criação de estratégias de monitoramento e controle de qualidade (Rebouças, 2009).

A falta ou os altos custos desses analisadores acabam incentivando o desenvolvimento de sensores virtuais, implementados através de softwares (*soft-sensors*), para medição indireta das variáveis primárias do processo. Uma possível maneira de se fazer essa medição é inferindo os valores das composições a partir das variáveis secundárias do processo. De maneira geral, os sistemas de inferência são construídos para realizar estimativas das variáveis primárias a partir das variáveis secundárias de fácil mensuração, tais como: temperaturas, pressões, níveis e vazões (Rebouças, 2009).

Existem três tipos de abordagem para a construção desses softwares: a modelagem fenomenológica, os métodos de regressão estatística ou ainda a modelagem a partir de alguma técnica de inteligência artificial. A escolha de qual técnica utilizar varia de acordo com a aplicação. No entanto, como a maioria dos processos químicos são geralmente complexos de se modelar, devido as suas inerentes características de não-linearidade, os desenvolvedores optam, normalmente, por utilizar uma das duas últimas abordagens (Rebouças, 2009).

3.6 Mineração de Dados

Mineração de Dados é um tópico na área de Aprendizado de Máquina que envolve técnicas para descobrir e descrever padrões dentro dos dados. Ela é utilizada como ferramenta para auxiliar no entendimento dos dados e realizar predições dos mesmos (Witten, Frank, & Hall, 2011). Na literatura encontram-se diversas técnicas, as quais podem ser utilizadas juntas ou separadas dependendo do problema. Algumas delas com bastante destaque são: NaiveBayes, Árvores de Decisão, KNN, Máquina de Vetores de Suporte, Redes Neurais e Agrupamentos.

Uma **Rede Neural** é um processador maciçamente paralelamente distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso (Haykin, 2007). As redes neurais artificiais forem desenvolvidas com o intuito de se assemelhar as estruturas biológicas encontradas nos seres vivos, devido a capacidade de armazenar conhecimento que esses apresentam. Esse aprendizado se dá através das conexões, ou pesos sinápticos, que existe entre os neurônios.

No caso das RNA's o neurônio é formado por uma estrutura somadora e uma função de ativação e as ligações são os pesos sinápticos.

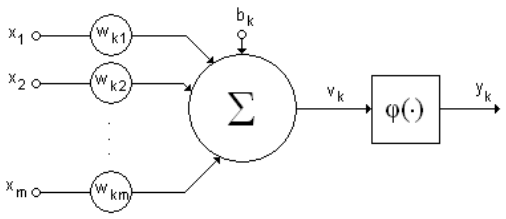


Figura 3 Neurônio Artificial

As redes neurais são utilizadas nos mais diversos problemas devido a características como: aprendizado de modelos não lineares, como exemplo o uso em classificação de padrões, filtragem de sinais, análise de imagens e controle/identificação de sistemas dinâmicos; sua capacidade de generalização e adaptabilidade; tolerância à falhas e a facilidade para realizar mapeamento de relação entrada-saída. Na literatura há inúmeras aplicações com sucesso do uso de redes neurais na área de inferência, como vistos em Bawazeer & Zilouchian (1997), Bo, Li, Zhang, Sun, & Wang (2003) e Fortuna, Luigi, Giannone, Graziani, & Xibilia, (2007).

Diversas são os tipos de redes neurais existentes para os mais diversos fins, tais como: perceptron com uma camadas, perceptron de múltiplas camadas, redes de funções de base radial, redes de Kohonen, redes ARTs, redes de Hopfield, entre outros.

A rede neural mais famosa e utilizada é a *MultiLayerPerceptron* (MLP), o qual utiliza vários neurônios maciçamente conectados e organizados em camadas. A quantidade de neurônios, tal como a quantidade de camadas depende diretamente do problema. Entretanto, há estudos que mostram que uma MLP de três camadas (entrada, oculta e saída) é capaz de mapear qualquer função, seja linear ou não-linear (Hornik, Stinchcombe, & White, 1989). Além dessa característica, na literatura é dito que para modelagem de processos dinâmicos uma única camada oculta apresenta um melhor desempenho (Silva, 2010).

O treinamento da rede neural MLP, que se resume ao ajuste dos pesos sinápticos em função do erro entre a resposta da rede e a resposta esperada, é caso de bastante estudo, sendo formulados diversos algoritmos de treinamento. O mais famoso deles é o *backpropagation*, que é um algoritmo que ajustas os pesos sinápticos baseado na informação do erro da sua camada de saída, retropropagando este erro para as outras camadas e ajustando cada peso sináptico.

A equação abaixo expressa matematicamente a saída i da rede.

$$\delta_i(t) = g_i[\emptyset, \theta] = F_i\left[\sum_{j=1}^{n_h} \rho_{i,j} f_j\left(\sum_{l=1}^{n_\emptyset} w_{j,l}\emptyset_l + w_{j,0}\right) + \rho_{i,0}\right]$$

O vetor θ contem os pesos sinápticos ($w_{j,l}$) e *biases*($\rho_{i,j}$). Os neurônios da camada escondida e o número de entrada das redes são, respectivamente, n_h e $n_\emptyset$. Enquanto que F_i e f_j são as funções de ativação dos neurônios das camadas de saída e oculta, respectivamente.

As redes neurais é uma das diversas técnicas para a descoberta do conhecimento. De forma breve, abaixo é descrito o processo normal para a descoberta de conhecimento.

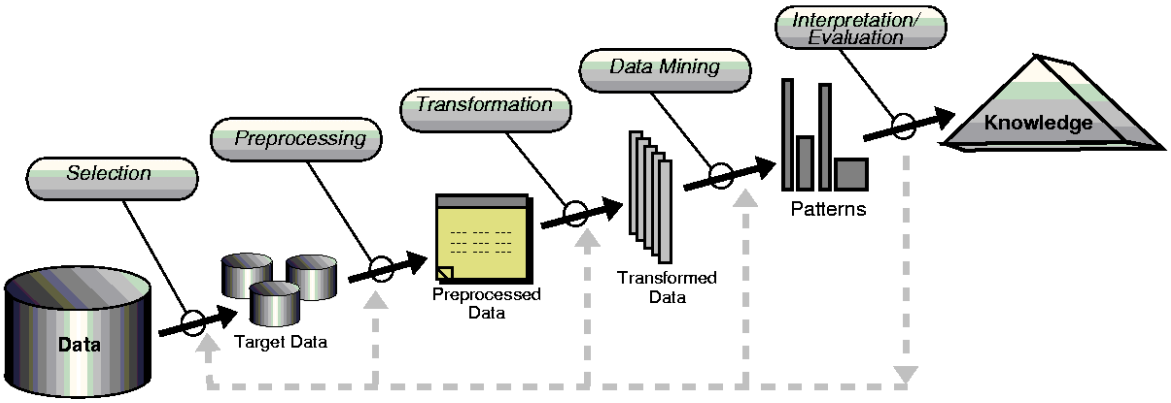


Figura 4: Fluxograma de Aquisição de conhecimento

- Pré-processamento:
  - Limpeza dos dados, o qual informações consideradas desnecessárias são removidas
  - Assegurar formatos consistente aos dados
  - Processo mínimo:
    - Redução de dimensionalidade
    - Combinação de atributos
- Transformação:
  - Transformar dados em formatos compatíveis
  - Depende da técnica de mineração de dados utilizada
  - Processo mínimo:
    - Escolher e executar o algoritmo de acordo com a tarefa fim
- Mineração de Dados:
  - Extração de padrões de comportamentos dos dados
  - Processo mínimo:
    - Interpretação dos resultados com possível retorno aos passos anteriores
    - Consolidar o conhecimento em documento
- Interpretação e Avaliação:
  - Uma vez identificado os padrões, estes são interpretados e selecionados, a fim de especificar quais darão suporte a tomadas de decisões humanas

3.7 Identificação do modelo através de Redes Neurais

Realizar inferência e identificação de sistemas podem ser vistos como o mesmo problema (Linhares, 2009), uma vez que depois de realizado o treinamento, a rede neuronal deve ser capaz de representar de forma satisfatória a dinâmica entre as variáveis primárias e secundárias do processo.

As estruturas utilizadas para identificação de modelos não-lineares são generalizações de modelos lineares. Pode-se afirmar que as estruturas são definidas a partir do seu vetor de regressão, o qual é caracterizado como um conjunto de variáveis utilizadas para estimar a saída do sistema (Lucena, 2005).

Estruturas FIR (Finite Impulse Response), ARX (AutoRegressive eXternal input), ARMAX (AutoRegressive Moving Average eXternal input), OE (Output Error) e SSIF (State Space Innovations Form) são algumas das estruturas lineares mais conhecidas. Se o vetor de regressão for selecionado para modelos ARX, a estrutura do modelo será chamada NNARX (Neural Network ARX) (Rebouças, 2009).

No trabalho foi utilizado o modelo NNARX descrito em (Norgaard, Magnus, Ravn, Poulsen, & Hansen, 2000) representado na Figura 5.

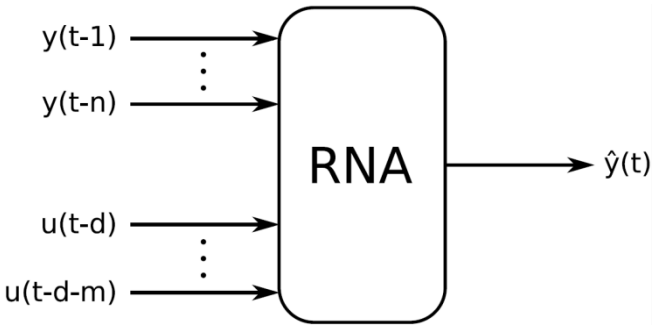


Figura 5: Estrutura NNARX

A expressão matemática para o modelo não-linear pode ser definida como:

$$\hat{y}(t) = f(y(t-1), \dots, y(t-n), u(t-d), \dots, u(t-d-m))$$

Na expressão temo $\hat{y}$ representando a saída estimada, d o atraso de transporte, n a ordem da saída, m a ordem da entrada da planta, $f(.)$ uma função não linear mapeada pela rede neural, y a saída da planta e u a entrada da planta.

Se tratando de uma simulação, podemos desprezar a atraso d do sistema, já a ordem do sistema geralmente é definida a partir de um estudo empírico no qual se utiliza a ordem de saída e entrada igual $(m = n)$.

3.8 Seleção do Modelo

Diversos modelos geralmente são gerados no processo para inferência de variável, uma vez que há vários parâmetros que devem ser selecionados para testes, combinando-os de forma combinatória. Para avaliar o modelo que melhor se adapta ao problema, deve-se selecionar índices de avaliação. Para o trabalho, o índice utilizado foi o erro médio quadrático (EMQ) definido por:

$$EMQ = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2}$$

Nessa equação temos N como número de amostras de validação, y_i o valor real e $\hat{y}_i$ valor estimado.

3.9 Análise de Componentes Principais

A técnica de análise de componentes principais (ACP ou PCA em inglês) foi iniciada por Pearson (1901) e desenvolvida por Hotelling (1933) é, segundo Jolliffe (2002), a mais antiga e conhecida técnica de análise de dados multivariáveis. Utilizada de forma explosiva após o advento dos computadores, pode ser encontrada nas mais diversas aplicações: compressão de dados, processamento de imagens, reconhecimento de padrões, entre outras.

O objetivo do PCA é realizar a redução da dimensão dos dados, principalmente quando constituídos de um grande número de variáveis inter-relacionadas, tentando manter de forma ótima a variação do conjunto de dados originais (Warne, Prasad, Siddique, & Maguire, 2004).

A redução é realizada a partir do mapeamento de um sistema composto por p variáveis em k combinações lineares ($k < p$) não correlacionadas, chamadas de componentes principais, geralmente obtidas a partir da matriz de covariância ou de correlação. O sistema de variabilidade do vetor aleatório composto das p variáveis originais pode ser aproximado pelo sistema de variabilidade do vetor aleatório que contém as k componentes principais. A qualidade dessa aproximação pode ser medida a partir da proporção da variância total das variáveis mantidas no sistema (Mingoti, 2005).

3.10 Coluna de Destilação na Indústria do Petróleo

Como já visto na introdução, uma coluna de destilação consiste em um equipamento que permite a separação dos componentes de uma mistura de líquidos miscíveis e é baseada na diferença de temperatura de ebulição de seus componentes individuais (Henley & Seader, 1981).

A coluna de destilação apresenta um comportamento não-linear, associado ao acoplamento das variáveis (Marlin, 1995), restrições na operação, constantes de tempo elevadas, com respostas lentas e a presença de atraso em virtude de sua geometria em estágios (Henley & Seader, 1981), gerando transientes elevados quando o processo é perturbado.

Com relação a não linearidade, (Ansari & Tadé, 2000) este comportamento é característico do processo de destilação uma vez que o ponto de operação no qual uma coluna é perturbada determinada seu estado estacionário final. O que se observa é que os ganhos do processo dependem deste ponto de operação e da qualidade do produto desejado (Shinskey, 1984). A não linearidade existe também devido ao acoplamento entre as variáveis, o equilíbrio líquido-vapor, característico dos equipamentos e outros (Marangoni, 2005).

Em relação ao acoplamento das variáveis, vale destacar a escolha das variáveis que resultam em pouco grau de interação é uma importante decisão de projeto, logo, inúmeros estudos como técnicas diferentes de desacoplamento (Bettoni, Bravi, & Chianese, 2000) na melhor seleção de estruturas de controle, no uso do domínio da análise no domínio da frequência entre outros podem ser considerados um vasto campo de estudo.

Em relação às restrições, podemos enfatizar que o ponto de operação ótimo de uma coluna de destilação deve respeitar algumas restrições, como as limitações hidráulicas, na separação, e na transferência de calor (Abou-Jeyab, Gupta, Gervais, Branchi, & Woo, 2001). As restrições estão relacionadas às fases vapor e líquido e ao *holdup* dos pratos.

A constante no tempo elevada pode ser atribuída ao transporte de fluidos, à troca térmica e à composição de misturas. Ainda, observam-se tempos mortos associados ao controle de temperatura. Embora o atraso e a constante de tempo serem pequenos para cada prato (na ordem de segundos), a dinâmica da coluna toda é lenta e dependendo das características das unidades pode ser da ordem de horas. A separação é realizada em estágios e quando o processo é perturbado são exercidas mudanças em cada prato conferindo um novo ponto de equilíbrio. Esse comportamento é arrastado por toda a coluna pelas fases líquida e vapor. E o resultado final consiste em um tempo de resposta muito elevado (Shinskey, 1996).

3.11 Processo de Destilação

A destilação é uma operação unitária que visa separar os componentes de uma fase líquida através de sua vaporização parcial. Essa separação dos constituintes está baseada nas diferenças de volatilidade, a qual os vapores produzidos são normalmente mais ricos nos componentes mais

voláteis do que o líquido, o que possibilita a separação de frações enriquecidas nos componentes desejados. É importante enfatizar que a destilação é o primeiro processo do refino, único que tem como entrada o petróleo e todos do qual os processos da refinaria dependem, direta ou indiretamente, de alguma saída da Destilação (Almeida, 2008).

O líquido e o vapor contêm, em geral, os mesmos componentes, mas em quantidades relativas diferentes, logo, a destilação não pretende obter produtos puros e diferentes entre si, gerando misturas ainda complexas de hidrocarbonetos e contaminantes, diferenciadas por suas faixas de ebulição (Almeida, 2008).

O processo de destilação ocorrem nas chamadas Torres de Destilação de Petróleo, as quais podem ser classificadas como:

- Torre de pré-fracionamento.
- Torre de destilação atmosférica.
- Torre de retificação ou torre retificadora.
- Torre de destilação a vácuo.
- Torre debutanizadora de nafta.
- Torre de fracionamento de nafta.

Os vários componentes do petróleo bruto têm tamanhos, pesos e temperaturas de ebulição diferentes. Por isso, o primeiro passo é separar esses componentes.

Devido à diferença de suas temperaturas de ebulição, eles podem ser facilmente separados por um processo chamado de destilação fracionada (Almeida, 2008).

1- Aquecer a mistura de duas ou mais substâncias (líquidos) de diferentes pontos de ebulição a alta temperatura. O aquecimento costuma ser feito com vapor de alta pressão.

2- A mistura entra em ebulição formando vapor (gases). A maior parte das substâncias passa para a fase de vapor.

3- O vapor entra no fundo de uma coluna longa (coluna de destilação fracionada) cheia de bandejas ou placas.

a. Elas possuem muitos orifícios ou proteções para bolhas a fim de permitir a descida do líquido

b. As placas aumentam o tempo de contato entre o vapor e os líquidos na coluna

c. Elas ajudam a coletar os líquidos que se formam nos diferentes pontos da coluna

d. Há uma diferença de temperatura pela coluna (mais quente embaixo, mais frio em cima)

4- O vapor sobe pela coluna.

5- Conforme o vapor sobe pelas placas da coluna, ele esfria.

6- Quando uma substância na forma de vapor atinge uma altura em que a temperatura da coluna é igual ao ponto de ebulição da substância, ela condensa e forma um líquido. A substância com o menor ponto de ebulição irá se condensar no ponto mais alto da coluna. Já as substâncias com pontos de ebulição maiores condensarão em partes inferiores da coluna.

7- As placas recolhem as diferentes frações líquidas.

8- As frações líquidas recolhidas podem:

a. Passar por condensadores, onde serão resfriadas ainda mais, e depois ir para tanques de armazenamento;

b. Ir para outras áreas para passar por outros processos químicos, térmicos ou catalíticos.

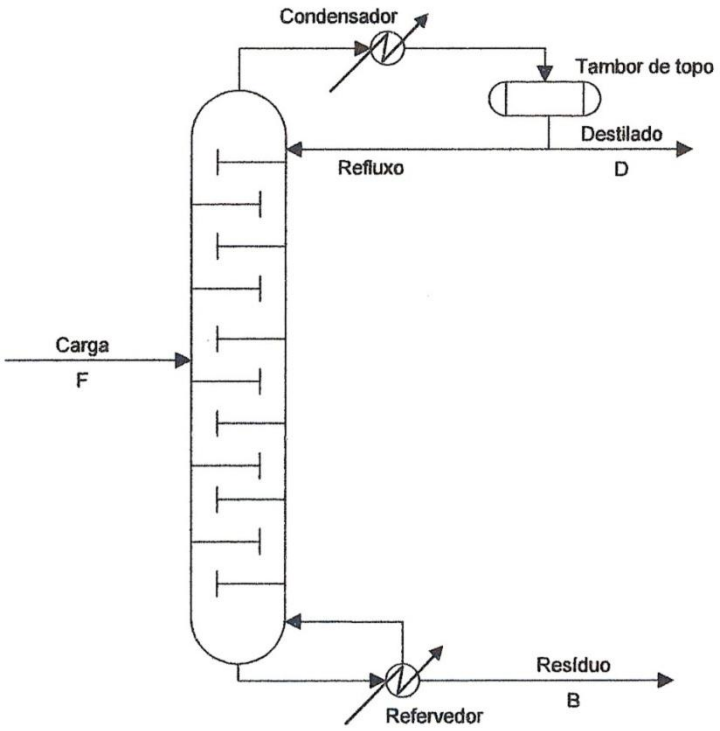


Figura 6: Coluna de Destilação

Pode-se classificar os processos contidas na destilação da seguinte forma (Almeida, 2008):

- Físicos:
    - Destilação do petróleo;
    - Desasfaltação;
    - Desaromatização;
    - Desparafinação;
    - Desoleificação;
    - Extração de Aromáticos;
    - Adsorção
- Químicos:
    - Craqueamento;

- Hidrocraqueamento catalítico;
- Reformação;
- Alquilação Catalítica;
- Viscorredução;
- Coqueamento retardado.

- Tratamento:
  - Dessalgação;
  - Cáustico;
  - Cáustico Regenerativo;
  - Tratamento Bender;
  - Tratamento DEA/MEA;
  - Hidrotratamento.

- Auxiliares:
  - Geração de Hidrogênio;
  - Tratamento de água;
  - Geração de vapor e energia;
  - Tratamento de Efluentes;
  - Recuperação de enxofre

3.12 Simulação

Uma simulação pode ser definida como sendo o desenvolvimento de um modelo lógico que reproduz a realidade, a fim de avaliar o comportamento e o desempenho de sistemas sob as mais variadas condições (Loureiro, 1995). O uso de modelos de simulação é necessário quando se pretende apreender alguma coisa a respeito de alguns sistemas reais, e não se pode observar ou fazer experimentos diretamente, seja por ainda não existirem ou pela dificuldade em lidar com os mesmos em situação real. Lembra ainda que um modelo cuidadosamente concebido pode eliminar

grande parte da complexidade inicialmente imaginada, deixando apenas características que o analista considere como importantes (Pegden, 1990).

Historicamente, devido ao aumento da competitividade, as indústrias têm sido levadas cada vez mais a promoverem caros processos de automação fabril. Isto implica geralmente no reexame e na redefinição das políticas e dos procedimentos existentes. Infelizmente, até mesmo o mais cuidadoso planejamento não é capaz de evitar a ocorrência de falhas, tais como: erros na determinação das capacidades de máquinas; insuficiência de espaços nos buffers; bloqueios de transportes, como por exemplo pontes rolantes, entre outros. Neste contexto, a simulação ganha importância como ferramenta de análise que possibilita o estudo a priori, evitando a ocorrência destes tipos de erros (Loureiro, 1995).

Pode-se definir que simulação é a obtenção da resposta temporal das variáveis de interesse (variáveis dependentes) de um modelo, quando se excita suas variáveis de entrada com sinais desejados e se definem os valores das condições iniciais das variáveis dependentes (Garcia C. , 1997). Simulação também pode ser definida como o processo de projetar o modelo de um sistema real e conduzir experimentos com este modelo com o propósito de entender o comportamento do sistema e/ou avaliar várias estratégias de operação do sistema (Shannon, 1998).

É importante enfatizar também que simulação é a ferramenta mais poderosa disponível para tomadas de decisão (*decision-makers*) responsáveis pelo projeto e operação de processos e sistemas complexos (Shannon, 1998). Com ela é possível o estudo, análise e avaliação de situações que de outra forma não seria possível. No mundo cada vez mais competitivo, simulação torna-se uma metodologia de solução de problema indispensável para engenheiros, projetistas e gerentes.

Devido ao fato de ser facilmente compreendido, um modelo de simulação é freqüentemente mais simples de ser justificado do que alguns modelos analíticos. Além disto, um modelo de simulação costuma ter mais credibilidade, uma vez que pode ser comparado com o sistema real, ou porque requer pouca simplificação, capturando mais as características reais do sistema.

A maioria dos modelos de simulação são chamados de modelos de entrada e saída, isto é, eles produzem uma saída dada as condições das entradas. Os modelos de simulação não podem gerar por si mesmo uma solução ótima como é o caso dos sistemas analíticos. Eles servem apenas como ferramenta de análise do comportamento do sistema sob uma determinada condição.

Alguns dos benefícios do uso da simulação são listados abaixo (Pegden, 1990):

a) Novas políticas, procedimentos operacionais, regras de decisão, estruturas organizacionais, fluxos de informação, etc., podem ser explorados sem que provocar distúrbios nos processos em uso;

b) Novos projetos de layout, sistemas de transporte, máquinas e equipamentos, softwares, podem ser testados antes de sua implantação, avaliando assim a necessidade de compra ou modificação;

c) Hipóteses sobre como e por que certos fenômenos ocorrem podem ser testados;

d) O fator tempo pode ser controlado, isto é, pode ser expandido ou comprimido, permitindo aumentar ou diminuir a velocidade a fim de se estudar um fenômeno;

e) Permite a análise de quais variáveis são significativas para o desempenho do sistema e como estas variáveis se interagem;

f) Gargalos podem ser identificados;

g) Um trabalho de simulação pode ser comprovadamente importante para o entendimento de como o sistema realmente funciona;

h) Novas situações, onde há pouca informação ou conhecimento a respeito, podem ser manipuladas a fim de se prever eventos futuros, isto é, a simulação é uma poderosa ferramenta para responder questões do tipo "o que acontecerá se".

A despeito das várias vantagens da utilização da simulação, deve-se ressalvar algumas restrições ou dificuldades na implantação de um modelo de simulação. As principais são (Pegden, 1990):

a) Necessidade de treinamento, uma vez que a qualidade da análise depende da qualidade do modelo, e portanto da habilidade do modelador;

b) Algumas vezes os resultados da simulação podem ser de difícil interpretação. Isto é devido ao fato da simulação tentar capturar a aleatoriedade de um sistema real, levando a dificuldade de identificação se um evento ocorreu devido a aleatoriedade ou a interações de elementos do sistema;

c) Análises feitas através do uso de simuladores podem ser demoradas e caras, podendo até mesmo inviabilizar seu uso.

Os primeiros trabalhos com metodologia proposta para a solução de sistemas de separação modelados prato a prato, surgiram na década de 30. Porém só a partir da década de 50, com o advento do computador digital, foram realizados investimentos sólidos no desenvolvimento de novos algoritmos e simuladores. Apesar deste investimento, apenas modelos simplificados era utilizados devido à baixa capacidade de processamento. A partir de 70, os primeiros simuladores comerciais começaram a ser introduzidos na indústria e o desenvolvimento de modelos rigorosos não parou mais (Staudt, 2007).

Resultados positivos de comparações da experiência real com dados simulados - HYSYS e ASPEN (Soos & Smejkal, 2001) e *batch distillation* entre softwares comerciais BATCHFRAC, HYSYS e CHEMCAD (Jiménez, Basualdo, Toselli, & Rosa, 2000) - comprovam a eficiência de simuladores de coluna de destilação.

Uma simulação pode ser dividida em duas categorias: estática e dinâmica. A grande diferença entre as mesmas é devido à influência da variável tempo, no qual, no caso da dinâmica, o processo pode ser modificado ao longo do tempo, já na estática não é levado em consideração à dinâmica do processo. No trabalho será utilizado uma coluna de destilação dinâmica debutanizadora implementada no UNISIM, a qual é descrita na seção de Implementação do trabalho.

3.13 Interprocess Comunication (IPC)

Nos dias atuais a necessidade de que *softwares* implementados em diferentes linguagens de programação compartilhem dados entre si, tal que, cada um cumprindo seu papel específico, gerem um único sistema com um propósito maior é um fato constante. Essa comunicação entre processos se dá o nome de IPC (Interprocess Communication), o qual denota um conjunto de chamadas de sistema que permite um processo do modo usuário possa:

- Sincronizar-se com outros processos
- Enviar mensagens para outros processos ou receber mensagens deles
- Compartilhar pedaços da memória com outros processos

Podemos qualificar os tipos de IPC como:

- Pipes e FIFO's
- Mensagens
    - System V IPC (Uso de Semáforos)
    - POSIX Messages
- Memória Compartilhada

O sistema operacional Windows© provê mecanismos para facilitar a comunicação e compartilhamento de dados entre aplicações. Tipicamente, aplicações utilizam IPC em uma arquitetura estilo cliente/servidor. O cliente é uma aplicação ou um processo que requisita um serviço de outra aplicação ou processo. O servidor é uma aplicação ou processo que responde a requisição do cliente. Algumas aplicações podem atuar tanto quanto cliente, como servidor (Microsoft, 2010).

Os mecanismos suportados pelo Windows são:

- Clipboard
- COM
- Data Copy
- DDE
- File Mapping
- Mailslots
- Pipes
- RPC
- Windows Sockets

Neste trabalho será enfatizada a tecnologia COM.

O OLE promove serviços que torna fácil para aplicações chamar sobre outras a edição de seus dados. Por exemplo, um processador de texto que utiliza OLE poderia embutir um gráfico de uma planilha do Excel o qual teria seus dados atualizados online à medida que os mesmos foram alterados na planilha.

O alicerce do OLE é o *Component Object Model (*COM). A aplicação que utiliza COM pode comunicar com uma grande variedade de outros componentes. Os componentes interagem como objetos e clientes.

OLE suporta *compound documents* e permite uma aplicação embutir ou *linkar* dados de outra. Objetos COM prover acesso para um objeto de dados através de uma ou mais conjuntos de funções relacionadas, conhecidas como interfaces.

Ole Automation

A comunicação entre o supervisório desenvolvido neste trabalho e o software de simulação do processo de destilação foi realizada a partir do OLE *Automation* (MICROSOFT, 2009). O software de simulação incorpora uma avançada arquitetura com tecnologia OLE para prover um *framework*

baseado em componentes que pode facilitar a customização, atualização e manutenção das mudanças de requisitos do usuário. Através dele pode-se esconder a complexidade da simulação pela construção de um outro programa o qual permite apenas acessar os parâmetros desejados da simulação (OLI SYSTEMS, 2007).

A ferramenta OLE (MICROSOFT, 2009) torna possível que uma aplicação manipule objetos implementados em outra aplicação, ou expor objetos que por sua vez podem ser manipulados. *Automation* é um padrão baseado sobre *Microsoft's Component Object Model* (COM). Não é necessário entender todos os detalhes do *Automation* ou COM para utilizar as funcionalidades que elas proporcionam.

Servidores do *Automation* e clientes implementam COM interfaces que são sempre derviadas do *IDispatch* e retornam um conjunto específico de tipos de data denominado de tipos *Automation*. Você pode automatizar qualquer objeto que implemente uma interface *Automation*, fornecendo métodos e propriedades que possa ser acessados por outras aplicações. *Automation* é disponível para ambos objetos OLE e COM. O objeto *Automation* pode ser local ou remoto (em outra máquina acessado através da rede); sendo assim, existe duas categorias de *Automation* (MICROSOFT, 2009):

- *Automation* (local)
- *Remote Automation* (sobre a rede, usando COM distribuídos, ou DCOM).

Automation desenvolveu-se do *Object Linking and Embedding* (OLE). O OLE permitia ao usuário recuperar um objeto específico como uma planilha do Excel e colocá-lo dentro de outro objeto, como um documento de texto. Ao mudar os valores no *spreadsheet* será modificado automaticamente dentro do documento de texto. Esse foi uma característica poderosa sem adicionar complexidade no código escrito. Simplesmente um problema de cortar e colar (OLI SYSTEMS, 2007).

Usando um *Automation* cliente como o Visual Basic, o usuário pode escrever código para acessar estes objetos e interagir como OLI Pro. O usuário final não necessita ver o código fonte do OLI Pro ou mesmo entender o que foi preciso para expor os objetos. Tudo o que é preciso é o conhecimento dos objetos que são disponíveis (OLI SYSTEMS, 2007).

Automation trabalha dentro do estilo cliente/servidor. Um servidor é que providencia um serviço que pode ser usado por clientes se eles sabem o protocolo correto. OLI Pro é uma aplicação *Automation Server*. Por exemplo, você pode usar *Automation* para acessar algum valor de

temperatura do *UniSim* dentro do OLI Pro para calcular propriedades da temperatura como a coeficientes de fugacidade, K-values, entropia e entalpia (OLI SYSTEMS, 2007).

OLI Pro pode ser executado de qualquer ferramenta que suporte *Automation*. Logo, linguagens de programação que suportam *Automation* através de bibliotecas como o C++ podem acessar o OLI Pro.

Existem 340 objetos *Automation* disponíveis dentro do OLI Pro. Estes objetos contém juntos 5.000 propriedades e métodos. Uma das tarefas mais difíceis no uso do *Automation* é determinar quais objetos são disponíveis e como recuperar as propriedades de interesse (OLI SYSTEMS, 2007).

De acordo com a Microsoft (MICROSOFT, 2009), uma dificuldade em criar os métodos do *Automation* é fornecer um mecanismo uniforme e seguro para repassar dados entre servidores e clientes *Automation*. *Automation* usa o tipo *VARIANT* para passar os dados. O tipo *VARIANT* é uma *tagged union*. Ele tem um membro data para o valor (isto é um anônimo C++ *union*) e um membro *data* indicando o tipo de informação guardada dentro da *union*. O tipo *VARIANT* suporta um número de padrões de tipos: 2- e 4-bytes inteiro, 4- e 8-bytes ponto flutuante, *strings*, e valores booleanos. Além disso, ele suporta o *HRESULT* (códigos de erros OLE), *CURRENCY* (um tipo ponto-fixo numérico), e *DATE* (data e tempo) tipos, tão bom quanto ponteiros para *IUnknown* e interfaces *IDispatch*.

O tipo VARIANT é encapsulado dentro da classe *COleVariant*. O suporte às classes *CURRENCY* e *DATE* são encapsuladas dentro das classes *COleCurrency* e *COleDateTime* respectivamente.

Abaixo exemplos de como se comunicar com o programa *Unisim* através do MATLAB, C++ e VB Script.

MATLAB

```
%instancia o software de simulação
hy = feval('actxserver', 'Hysys.Application');

% abre uma planta
sc = invoke(hy.SimulationCases, 'Open',
'E:\tcc\debutdyn_planta\debutdyn_ss.hsc');

% recupera algum controlador
fic100 = hy.Flowsheet.Flowsheets.Item('COL1').Operation.Item('FIC-
100');
```

C++ Builder

```
Variant HysysCOM;
HysysCOM = Variant::CreateObject("UniSimDesign.Application");
HysysCOM.OlePropertyGet("SimulationCases").
                    OleProcedure("Open","E:\tcc\debutdyn_planta\debutdyn.usc");
HysysCOM.OlePropertyGet("ActiveDocument").
                                OlePropertyGet("Flowsheet").
                                OlePropertyGet("Flowsheets").
                                OlePropertyGet("Item", "COL1").
                                OlePropertyGet("Operations").
                                OlePropertyGet("Item", "FIC-100");
```

VBscript

```
Set hyApp = CreateObject("UniSimDesign.Application")
hyApp.Open("E:\tcc\debutdyn_planta\debutdyn.usc")
fic100 = hyApp .ActiveDocument.Flowsheet.Flowsheets.Item("COL1").Operations.Item("FIC-100")
```

O *Ole Automation* frente ao DDE – também tecnologia da Microsoft para comunicação entre processos do Windows - é de recomendação da própria Microsoft (MICROSOFT, 2009), a qual fala que as bibliotecas OLE e DDEML (Dynamic Data Exchange Management Library):

a) Oferecem toda a funcionalidade de mensagens DDE.
b) O uso de bibliotecas reduzirá o tempo de desenvolvimento do aplicativo. Uma exceção a essa regra pode ocorrer quando um aplicativo requer apenas um subconjunto muito limitado da biblioteca OLE ou DDEML capacidade.
c) As bibliotecas irão incorporar refinamentos futuros. Eles provavelmente incluem melhor desempenho por meio do uso de um novo mecanismo IPC (comunicação entre processos) em vez de usar mensagens DDE.

Outra tecnologia bem aceita na indústria e na academia para comunicação de processos e equipamentos voltados para automação industrial é o OPC (*OLE for Process Control*). Entretanto, não foram encontrados servidores e clientes OPC do software de simulação utilizado.

3.14 Programação Orientada a Objetos

O conceito de programação orientada a objetos tem suas raízes na SIMULA 67, mas não foi amplamente desenvolvido até que a evolução da Smalltalk resultasse na produção de sua versão 80. Uma linguagem com esta característica deve oferecer três recursos chaves: tipos de dados abstratos, herança e um tipo particular de vinculação dinâmica (Sebesta, 2003).

Os tipos de dados abstratos em linguagens orientadas a objetos usualmente são chamados de classes. As instâncias das classes são chamadas de objetos. Uma classe definida pela herança de outra é uma classe derivada ou subclasse. Uma classe da qual a nova é derivada é chamada sua classe-pai ou superclasse. Os subprogramas que definem as operações em objetos de uma classe são chamados de métodos. As chamadas aos métodos têm o nome de mensagens. A coleção inteira de métodos de um objeto é chamada de protocolo de mensagem ou de interface de mensagem do objeto. No caso da herança, a subclasse herda todas variáveis e métodos da superclasse. Tal técnica auxilia no problema da reutilização de tipos abstratos e na organização do programa (Sebesta, 2003).

Outro conceito importante é o polimorfismo, o qual se caracteriza pela vinculação dinâmica de mensagens a definições de métodos. Ou seja, métodos herdados, por exemplo, podem ser sobrescritos pelas subclasses. Além disso, dependendo dos argumentos advindos na chamada do método, o mesmo pode ser vinculado dinamicamente para fluxos de sequência completamente diferentes. Como na herança, esse conceito auxilia no problema da reutilização do software e organização, evitando, por exemplo, o uso de switch na programação (Sebesta, 2003).

3.15 UML

A UML (*Unified Modeling Language*) é uma linguagem-padrão para a elaboração da estrutura de projetos de software. Ela poderá ser empregada para a visualização, a especificação, a construção e a documentação de artefatos que façam uso de sistemas complexos de software (Booch, Rumbaugh, & Jacobson, 2006).

A UML é um padrão relativamente aberto, controlado pelo OMG (*Object Management Group*), um consócio aberto de empresas. O OMG foi formado para estabelecer padrões que suportassem interoperabilidade, especificamente a de sistemas orientados a objetos (Kobryn, Booch, Jacobson, & Rumbaugh, 2006).

A UML nasceu da unificação das muitas linguagens gráficas de modelagem orientadas a objetos que floresceram no final dos anos oitenta, início dos noventa.

A UML é apenas uma linguagem, sendo somente uma parte de um método para desenvolvimento de software. A UML é independente do processo de desenvolvimento, entretanto especialmente utilizada em processos de desenvolvimento orientado a casos de usos, centrado na arquitetura, iterativo e incremental.

Podemos utilizar a UML para visualizar, especificar, construir e documentar o sistema. Para isso, a mesma utiliza três tipos de blocos de construção (Booch, Rumbaugh, & Jacobson, 2006):

- Itens
- Relacionamentos
- Diagramas

Itens são as abstrações identificadas como cidadões de primeira classe em um modelo; os relacionamentos reúnem esses itens; os diagramas agrupam coleções interessantes de itens.

Caracterização dos Itens:

- Estruturais
- Comportamentais
- Agrupamentos
- Anotacionais

Os estruturais são as partes mais estáticas do modelo, representando conceitos ou objetos físicos; os comportamentais representam comportamentos no tempo e no espaço; os agrupamentos são as partes organizacionais dos modelos; os anotacionais são partes explicativas do modelo UML .

Os Diagramas na UML são a representação gráfica de um conjunto de elementos. Eles auxiliam na visualização de um sistema sob várias perspectivas. A UML inclui 13 diagramas:

- Diagrama de classes
- Diagrama de objetos
- Diagrama de componentes
- Diagrama de estruturas compostas

- Diagrama de casos de uso
- Diagrama de sequências
- Diagrama de comunicações
- Diagrama de gráficos de estados
- Diagrama de atividades
- Diagrama de implementação
- Diagrama de pacote
- Diagrama de temporização
- Diagrama de visão geral da interação

No projeto desenvolvido foram utilizados o diagrama de classes, de sequência e de implementação, e portanto, apenas esses serão detalhados.

Diagrama de Classes

O diagrama de classes exibe um conjunto de classes, interfaces e colaborações, bem como seus relacionamentos. Esses diagramas expressa a visão estática da estrutura do sistema.

Diagrama de Sequências

Diagramas de Sequências são utilizados para fazer a modelagem dos aspectos dinâmicos do sistema. Na maior parte, isso envolve a modelagem de instâncias concretas ou prototípicas de classes, interfaces, componentes e nós, juntamente com as mensagens que são trocadas entre eles.

Diagrama de Implementação

Diagramas de implementação mostram a configuração de nós de um processamento em tempo de execução e os artefatos que neles existem. Este diagrama tem as mesmas propriedades comuns a todos os outros, o que diferencia é o seu conteúdo em particular.

3.16 Banco de Dados

De acordo com o livro Sistemas de Banco de Dados (Elmasri & Navathe, 2005), os bancos de dados e a sua tecnologia estão provocando um grande impacto no crescimento do uso de computadores. É viável afirmar que eles representam um papel crítico em quase todas as áreas em que os computadores são utilizados, incluindo negócios, comércio eletrônico, engenharia, medicina, direito, educação e as ciências da informação, para citar apenas algumas delas.

Um banco de dados é uma coleção de dados relacionados. Os dados são fatos que podem ser gravados e que possuem um significado implícito. Por exemplo, considere nomes, números telefônicos e endereços de pessoas que você conhece. Esses dados podem ter sido escritos em uma agenda de telefones ou armazenados em um computador, por meio de programas como o Microsoft Access ou Excel. Essas informações são uma coleção de dados com um significado implícito, consequentemente, um banco de dados (Elmasri & Navathe, 2005).

Um Sistema Gerenciador de Banco de Dados (SGBD) é uma coleção de programas que permite aos usuários criar e manter um banco de dados. O SGBD é, portanto, um sistema de software de propósito geral que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados entre vários usuários e aplicações. A definição de um banco de dados implica especificar os tipos de dados, as estruturas e as restrições para os dados a serem armazenados em um banco de dados (Elmasri & Navathe, 2005).

A construção de um banco de dados é o processo de armazenar os dados em alguma mídia apropriada controlada pelo SGBD. A manipulação inclui algumas funções, como pesquisas em banco de dados para recuperar um dado específico, atualização do banco para refletir as mudanças no minimundo e gerar os relatórios dos dados. O compartilhamento permite aos múltiplos usuários e programas acessar, de forma concorrente, o banco de dados (Elmasri & Navathe, 2005).

Outras funções importantes do SGBD são a proteção e a manutenção do banco de dados por longos períodos. A proteção inclui a proteção do sistema contra o mau funcionamento ou falhas (*crashes*) no hardware ou software, e segurança contra acessos não autorizados ou maliciosos. Um banco de dados típico pode ter um ciclo de vida de muitos anos, então, os SGBD devem ser capazes de manter um sistema de banco de dados que permita a evolução dos requisitos que se alteram ao longo do tempo (Elmasri & Navathe, 2005).

As vantagens do uso do SGBD podem ser listadas como (Elmasri & Navathe, 2005):

- Controle de Redundância
- Restrição de Acesso Não Autorizado
- Garantia de Armazenamento de Estruturas para o Processamento Eficiente de Consultas
- Garantia de Backup e Restauração
- Múltiplas Interfaces para os Usuários
- Representação de Relacionamentos Complexos entre os Dados
- Restrições de Integridade

- Possibilidade de Inferências e Ações Usando as Regras
- Implicações Adicionais do Uso da Abordagem de um Banco de Dados
    - Potencial para Garantir Padrões.
    - Redução no Tempo de Desenvolvimento de Aplicações.
    - Flexibilidade
    - Disponibilidade para Atualizar as Informações
    - Economias de Escala

O modelo relacional representa o banco de dados como uma coleção de relações. Informalmente, cada relação se parece com uma tabela de valores ou, em alguma extensão, com um arquivo de registros 'plano' (Elmasri & Navathe, 2005).

Quando uma relação é pensada como uma tabela de valores, cada linha na tabela representa uma coleção de valores de dados relacionados. No modelo relacional, cada linha na tabela representa um fato que corresponde a uma entidade ou relacionamento do mundo real. O nome da tabela e os nomes das colunas são usados para ajudar na interpretação do significado dos valores em cada linha (Elmasri & Navathe, 2005).

Na terminologia do modelo relacional formal, uma linha é chamada tupla, um cabeçalho de coluna é conhecido como atributo, e a tabela é chamada relação. O tipo de dado que descreve os tipos de valores que podem aparecer em cada coluna é representado pelo domínio de valores possíveis (Elmasri & Navathe, 2005).

A linguagem SQL pode ser considerada uma das maiores razões para o sucesso dos bancos de dados relacionais no mundo comercial. Como se tornou padrão para os bancos relacionais, os usuários têm pouca preocupação ao migrar suas aplicações de banco de dados, originadas por outros tipos de sistemas de banco de dados — por exemplo, em rede/e hierárquico —, para o sistema relacional (Elmasri & Navathe, 2005).

A SQL é uma linguagem de banco de dados abrangente: ela possui comandos para definição de dados, consultas e atualizações. Assim, ela tem ambas as DDL (*data definition language*) e DML (*data manipulation language*). Além disso, tem funcionalidades para a definição de visões (*views*) no banco de dados, a fim de especificar a segurança e as autorizações para as definições de restrições de integridade e de controles de transação. Ela também possui regras para embutir os comandos SQL em linguagens de programação genérica como Java, COBOL ou C/C++ (Elmasri & Navathe, 2005).

Como exemplos da DDL temos:

- *ALTER TABLE*: Modifica a estrutura de uma tabela através da adição de novas colunas ou renomeação de colunas já existentes e também o seu próprio nome;
- *CREATE TABLE*: Cria uma nova tabela no banco de dados corrente. Os nomes das tabelas devem ser únicos;
- *DROP TABLE*: Apaga uma determinada tabela do banco de dados corrente.

Como exemplos de DML temos:

- *SELECT*: Consulta ao banco de dados, com o objetivo de retornar tabelas com campos e registros especificados na consulta.
- *INSERT*: Insere novos registros na tabela especificada.
- *DELETE*: Apaga registros de uma tabela especificada;
- *UPDATE*: Altera os dados existentes nos registros de uma tabela especificada.

4. Implementação do Trabalho

O principal objetivo deste trabalho é o desenvolvimento de um supervisório inteligente capaz de atuar e armazenar dados de uma simulação de um processo de destilação, os quais serão utilizados por módulos inteligentes para inferência de variáveis. Em suma, o supervisório desenvolvido irá realizar a comunicação com a simulação modelada no Unisim através do *Ole Automation*, utilizará um banco de dados para armazenar as variáveis de processo e através de módulos inteligentes realizará inferências e pré-processamento dos dados.

Como requisitos não funcionais do sistema foi definido que o mesmo deve ser flexível, escalável e de fácil manutenção.

O supervisório será implementado na linguagem de programação JAVA. Essa escolha foi influenciado devido as características de ser uma linguagem que implementa o paradigma da Programação Orientada a Objetos, ter uma rica biblioteca, ser um projeto Open Source, ser independente de sistema operacional e suportar o *Ole Automation*.

De forma a ficar mais claro, o supervisório será dividido nos seguintes módulos:

- Módulo de Aquisição e Armazenamento de Dados
- Módulo de Perturbação
- Módulo de Pré-processamento
- Módulo de Inferência de Variáveis

Cada módulo individual pode ser visto como um *plugin* do supervisório (Figura 7), tornando, este, um cliente dos serviços dos *plugins*. Na arquitetura proposta, é possível realizar a expansão das funcionalidades do supervisório através do desenvolvimento de novos módulos (*plugins*), independente da tecnologia para o seu desenvolvimento, com exceção do requisito que o mesmo ofereça uma interface *OleAutomation*. Por exemplo, caso haja alguma nova técnica para processamento de sinal desenvolvida na plataforma do *Matlab*, a mesma pode ser utilizada pelo Supervisório, uma vez que o *Matlab* oferece uma interface *OleAutomation* para comunicação, logo o supervisório se comporta como um cliente que solicita e envia dados ao *plugin* desenvolvido no *Matlab* e, este como um servidor de serviços, receber os dados, processa e responde a partir do mesmo canal de comunicação estabelecido. Portanto, as funcionalidades desenvolvidas neste trabalho podem ser expandidas de acordo com a necessidade do usuário.

Figura 7: Visão do Supervisório a partir dos Plugins

De forma mais detalhada a Figura 8 demonstra claramente como se dá a comunicação entre o Supervisório e algum novo *plugin* desenvolvido. Primeiro o Supervisório solicita a instância do *OleAutomation* a requisição ou envio de dados ao novo *plugin*. O *OleAutomation* então toma a responsabilidade de intermediar o canal de troca de dados entre o cliente (Supervisório) e o servidor (*Plugin*).

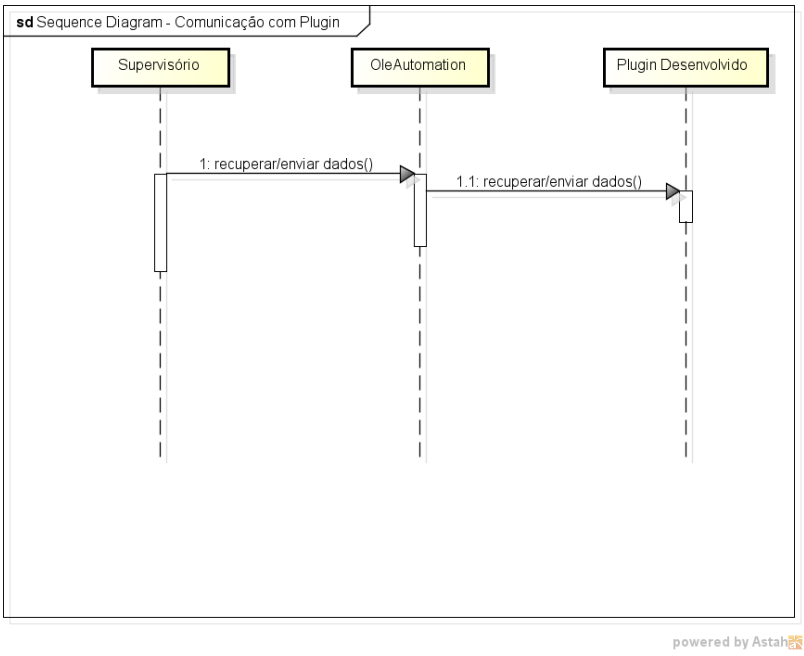


Figura 8: Comunicação com Plugin

Abaixo será descrito os objetivos e funcionalidades de cada módulo:

Módulo de Aquisição e Armazenamento de Dados

Para a comunicação entre o Supervisório e a coluna de destilação simulada foi utilizado o *Ole Automation*. Como já foi referenciado anteriormente, ele é capaz de realizar a comunicação de forma simples e eficiente entre o software de simulação e o supervisório. Entretanto, como visto no manual, a grande dificuldade é encontrar o caminho que deve ser realizado pelo software de simulação para encontrar o objeto desejado. Uma vez encontrado o caminho na hierarquia de objetos fornecidos pelo software de simulação (340 no total de acordo com o manual), seria o momento de referenciá-lo no supervisório. Para isso, cada objeto ou propriedade que fosse acessada do software de simulação pelo supervisório, deveria possuir em linha de código correspondente no *script* responsável pela comunicação.

Diante desse contexto, surge o problema da manutenção do sistema, uma vez que, caso a planta mude totalmente ou apenas evolua de forma simples - por exemplo a visualização de novas propriedades pelo supervisório - as linhas de código as quais fazem referência aos componentes e propriedades devem ser alteradas - acrescentando, caso necessite visualizar uma nova propriedade; removendo, caso não queira mais que um componente seja visualizado; editando, caso o caminho para o acesso à propriedade seja alterado. Pensando nisso, foi idealizada uma nova estratégia, para que de forma dinâmica os caminhos fossem construídos e que os mesmos não fossem referenciados em linhas de código, mas sim armazenados em algum lugar adequado, a fim de facilitar a manutenção à medida que a planta simulada evolua.

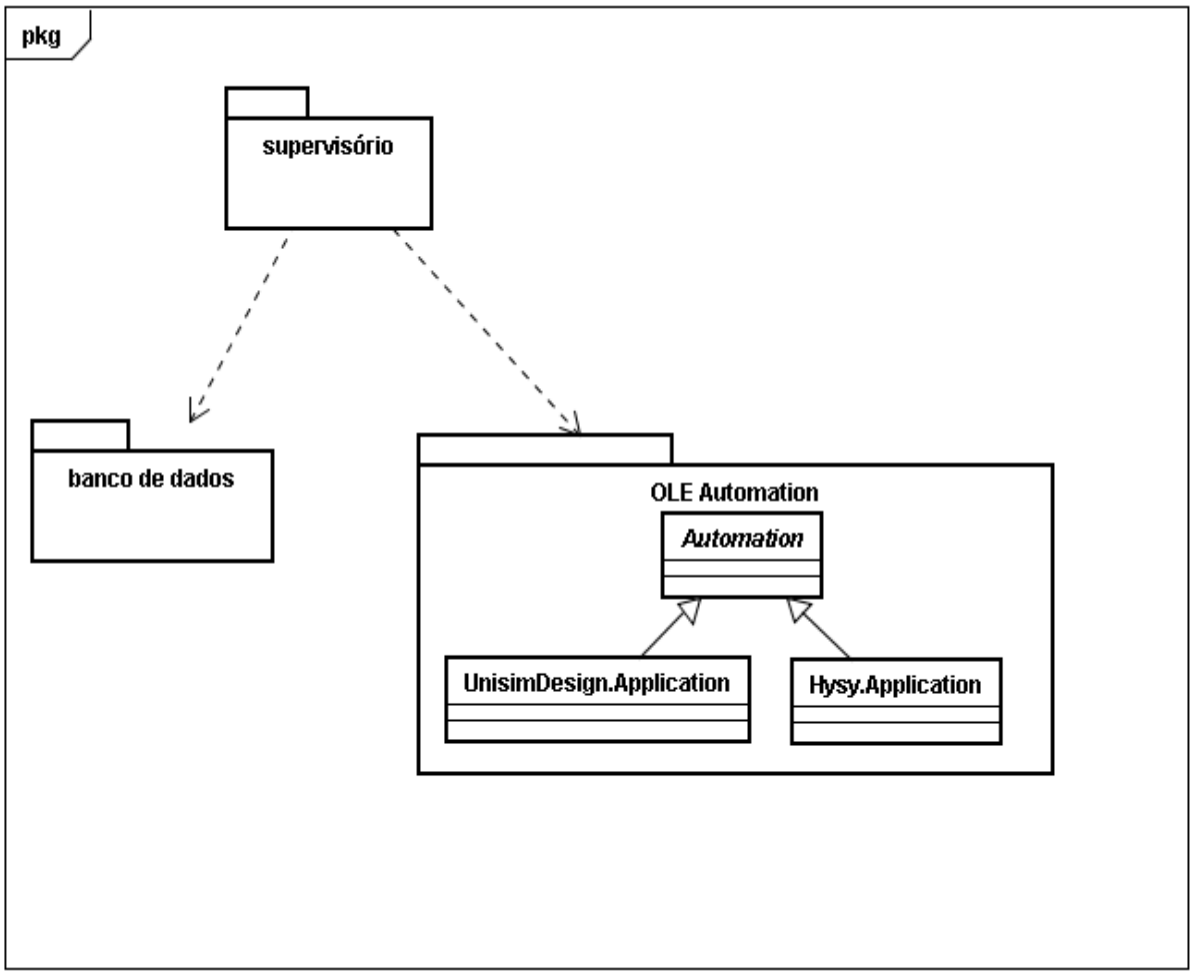


Figura 9: Diagrama de classes para comunicação entre supervisório, software de simulação e banco de dados

A solução consiste na utilização do conceito de *Tags* dentro dos *Databooks*. Os *Databooks* são uma espécie de planilha de dados que podem referenciar qualquer propriedade dentro da simulação através do uso de *Tags,* proporcionando o acesso externo dos seus valores. Estas *Tags* na verdade consiste em um identificador único para uma variável de processo monitorada. Essas *Tags* então são armazenadas no banco de dados, a fim de identificar as variáveis desejadas no software de simulação. Portanto, para a adição, edição ou remoção de alguma variável monitorada, apenas alterando o banco de dados, o supervisório estará atualizado e apto para continuar a monitoração do processo simulado.

De forma mais detalhada na Figura 10 é apresentado o diagrama de sequência que recupera todas as variáveis cadastradas no *DataBook*. A operação consiste na requisição dos dados pelo usuário ao formulário principal do programa (*FrmMain*), logo após é solicitado à uma instância do *ConnSim* as variáveis que estão cadastradas no *DataBook*. O *ConnSim* é uma classe que apenas encapsula os métodos do *ActiveXComponent*, sendo este o objeto *OleAutomation* que de fato irá

realizar a interface entre o supervisório e a simulação. O *ActiveXComponent* primeiramente irá recuperar todas as *Tags* cadastradas no *DataBook* e só assim irá requisitar ao software de simulação os valores individuais de cada uma. Para recuperar o valor de apenas de uma única variável, o usuário fornece o valor da *Tag* cadastrada no *DataBook*, como visto no diagrama de sequência da Figura 11.

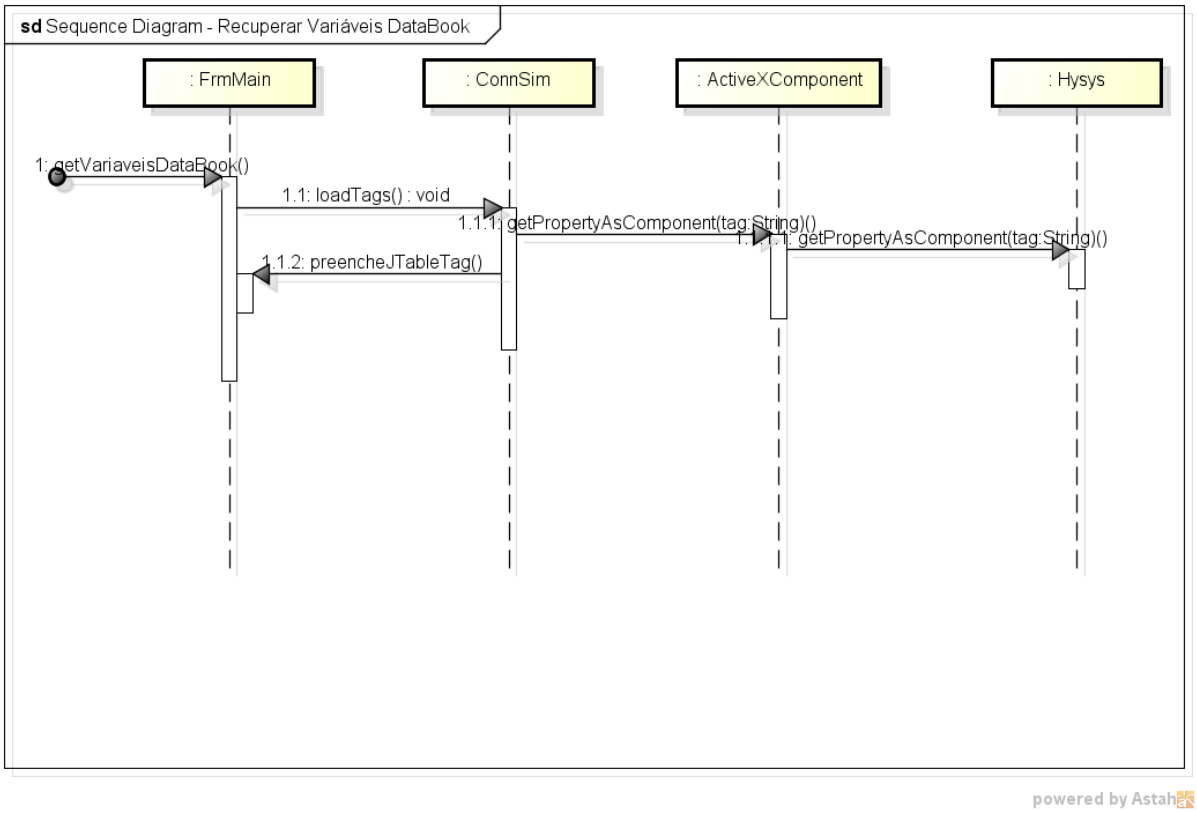


Figura 10: Diagrama de Sequência para recuperar dados do Databook

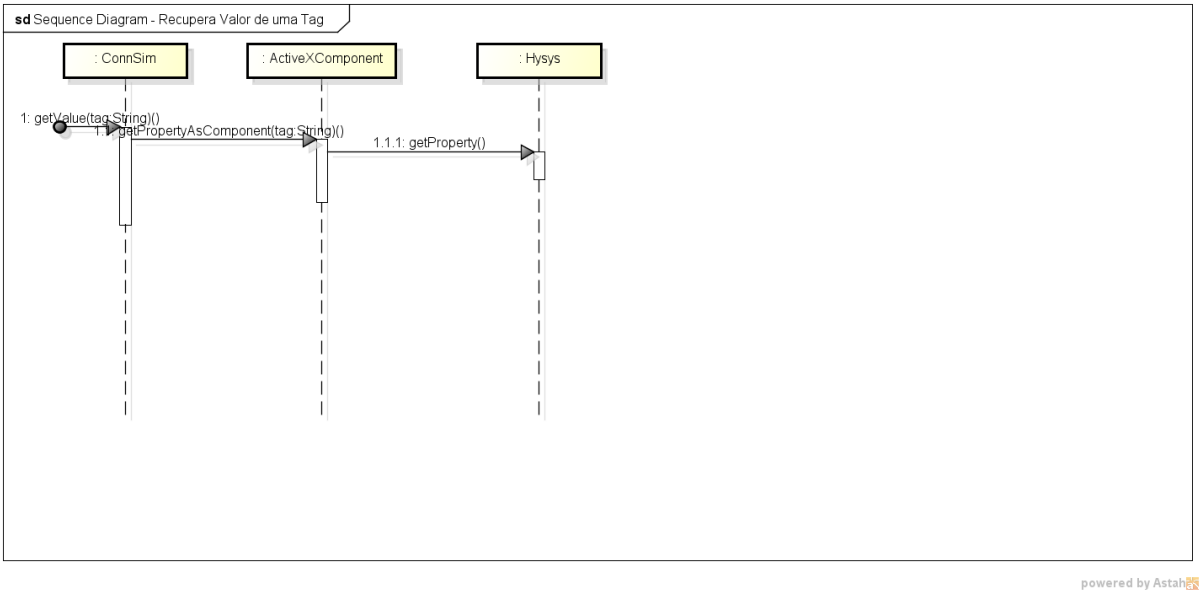


Figura 11: Recuperar Valor de uma Tag específica

Como dito anteriormente, para acessar qualquer variável ou, dependendo da permissão, modificar o seu valor, a mesma deve está cadastrada no *DataBook* da simulação atual conectada ao Supervisório. Para adicionar, remover e editar a permissão da variável basta fazer os seguintes passos no software de simulação.

Abra a aba *Databook*

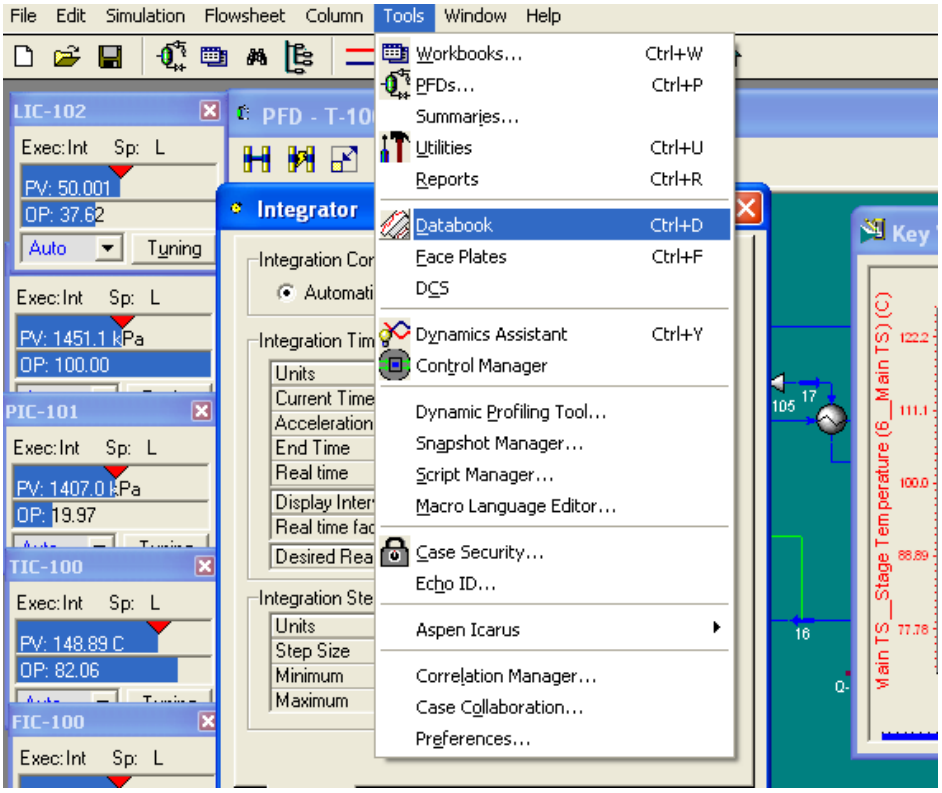


Figura 12: Acessando Databook

Adicione um novo *Data Tables*

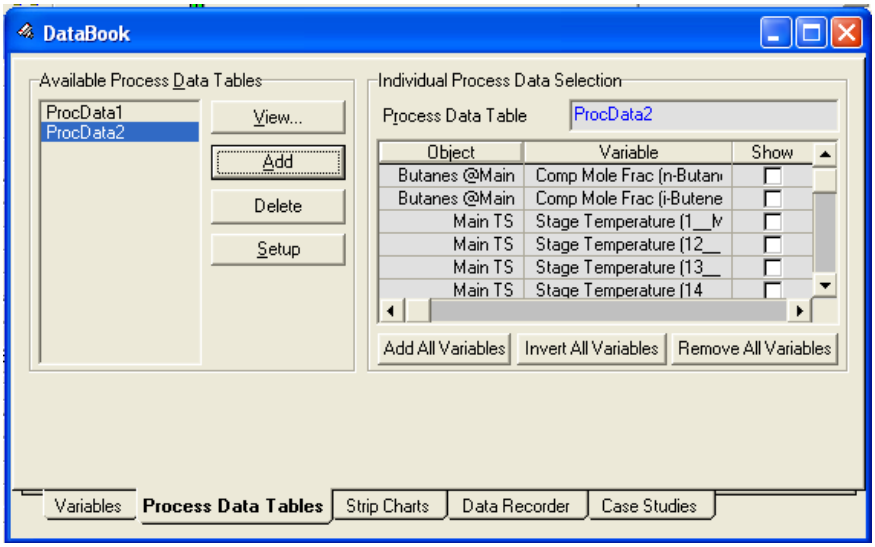


Figura 13: Adicionando um novo Data Table

Habilite a aba *Variables*. Clique no *Insert* e selecione a variável desejada.

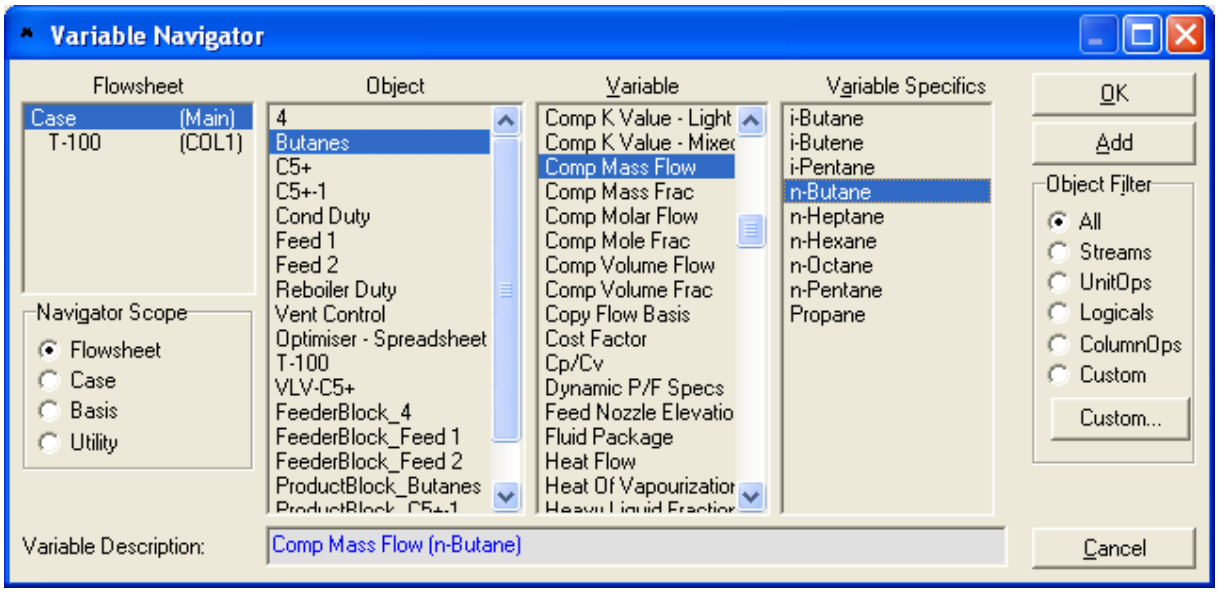
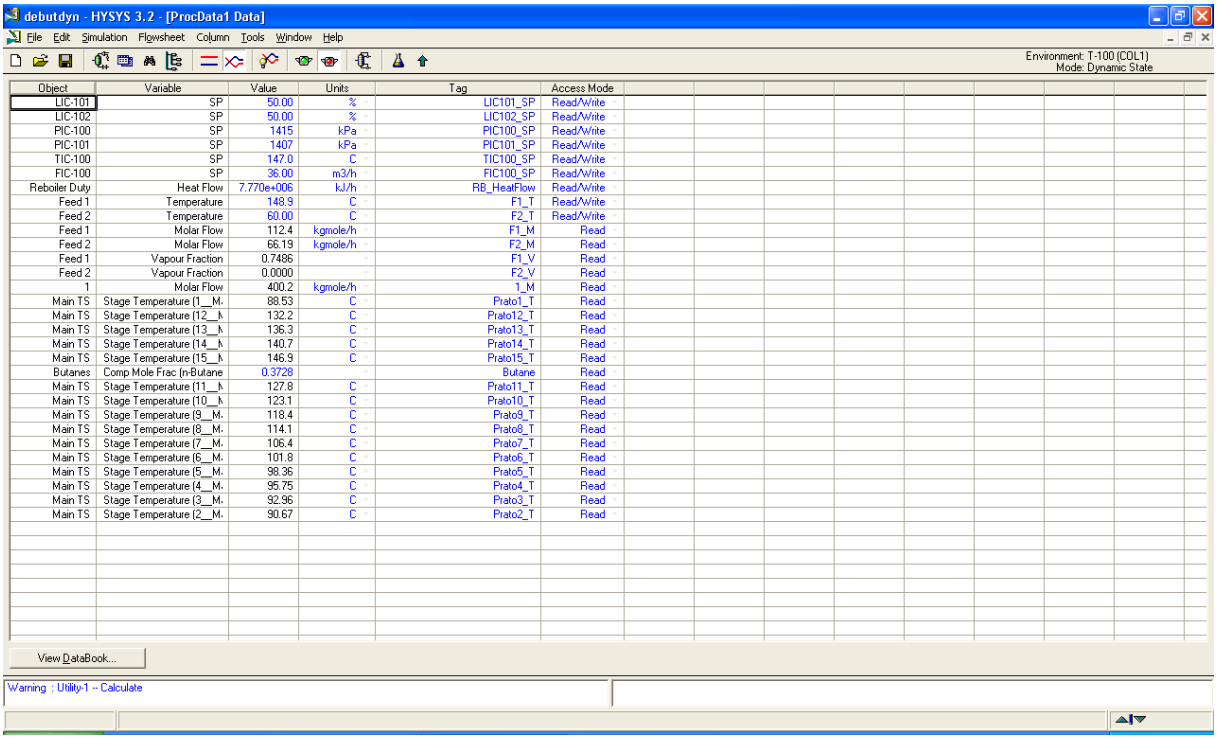


Figura 14: Selecionando variável

Para acessar os valores das variáveis do *Databook*, o caminho utilizado terá como modelo a seguinte linha de código:

ActiveDocument.DataTables.Item("NOME_PLANILHA").VarDefinitions.Item("NOME_TAG").Variable.Value.

Os valores “NOME_PLANINHA” e “NOME_TAG” devem ser substituídos pelo nome da planilha criada para aquela simulação e o nome da *tag* da variável monitorada, por exemplo: ProcData1 e LIC101_SP respectivamente.



Object	Variable	Value	Units	Tag	Access Mode
LIC-101	SP	50.00	%	LIC101_SP	Read/Write
LIC-102	SP	50.00	%	LIC102_SP	Read/Write
PIC-100	SP	1415	kPa	PIC100_SP	Read/Write
PIC-101	SP	1407	kPa	PIC101_SP	Read/Write
TIC-100	SP	147.0	C	TIC100_SP	Read/Write
FIC-100	SP	36.00	m3/h	FIC100_SP	Read/Write
Reboiler Duty	Heat Flow	7.770e+006	kJ/h	RB_HeatFlow	Read/Write
Feed 1	Temperature	148.9	C	F1_T	Read/Write
Feed 2	Temperature	60.00	C	F2_T	Read/Write
Feed 1	Molar Flow	112.4	kgmole/h	F1_M	Read
Feed 2	Molar Flow	66.19	kgmole/h	F2_M	Read
Feed 1	Vapour Fraction	0.7486		F1_V	Read
Feed 2	Vapour Fraction	0.0000		F2_V	Read
1	Molar Flow	400.2	kgmole/h	1_M	Read
Main TS	Stage Temperature (1__M.	88.53	C	Prato1_T	Read
Main TS	Stage Temperature (12__M	132.2	C	Prato12_T	Read
Main TS	Stage Temperature (13__M	136.3	C	Prato13_T	Read
Main TS	Stage Temperature (14__M	140.7	C	Prato14_T	Read
Main TS	Stage Temperature (15__M	146.9	C	Prato15_T	Read
Butanes	Comp Mole Frac (n-Butane	0.3728		Butane	Read
Main TS	Stage Temperature (11__M	127.8	C	Prato11_T	Read
Main TS	Stage Temperature (10__M	123.1	C	Prato10_T	Read
Main TS	Stage Temperature (9__M.	118.4	C	Prato9_T	Read
Main TS	Stage Temperature (8__M.	114.1	C	Prato8_T	Read
Main TS	Stage Temperature (7__M.	106.4	C	Prato7_T	Read
Main TS	Stage Temperature (6__M.	101.8	C	Prato6_T	Read
Main TS	Stage Temperature (5__M.	98.36	C	Prato5_T	Read
Main TS	Stage Temperature (4__M.	95.75	C	Prato4_T	Read
Main TS	Stage Temperature (3__M.	92.96	C	Prato3_T	Read
Main TS	Stage Temperature (2__M.	90.67	C	Prato2_T	Read

Figura 15: Configuração do Databook

Uma vez desenvolvido o módulo de comunicação com a planta simulada, o supervisório armazena de forma eficiente os dados requisitados à simulação através do uso de um SGBD. Esse módulo também será responsável em fornecer os dados em qualquer ponto do sistema, tal como o de pré-processamento, inferência, monitoramento, controle entre outros.

Na Figura 16 é apresentada a modelagem Relacional do sistema responsável por armazenar os dados e as configurações de cada simulação monitorada pelo supervisório.

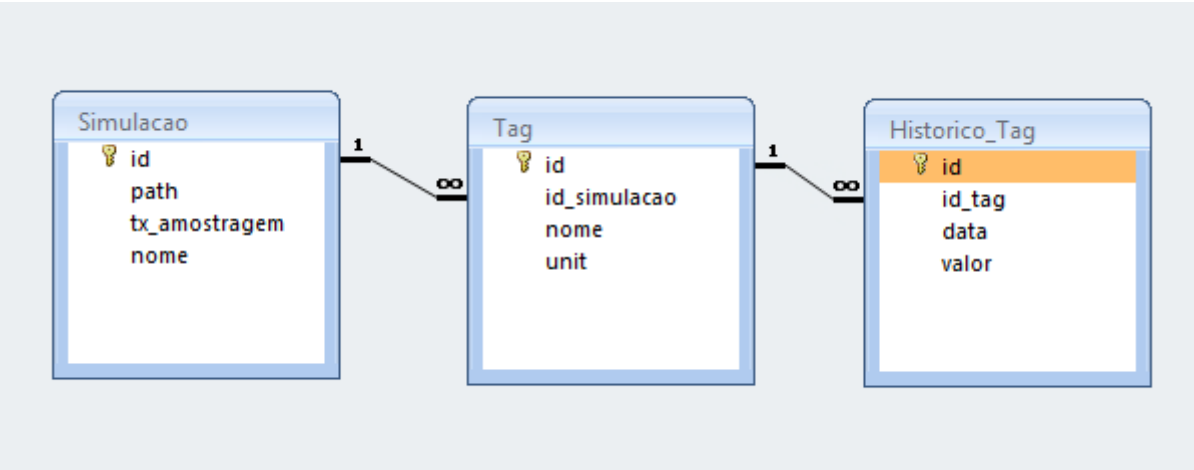


Figura 16: Modelo Relacional

Na tabela **Simulacao** temos como campos:

- id: identificador único da simulação
- path: caminho físico da simulação
- tx_amostragem: taxa de amostragem a qual foi registrada os dados da simulação
- nome: nome simbólico para a simulação

Na tabela **Tag** temos como campos:

- id: identificador único da Tag
- id_simulacao: identificador da simulação a qual esta Tag está associada
- nome: nome da Tag dentro do DataBook
- unit: unidade física da variável registrada

Na tabela **Historico_Tag** temos como campos:

- id: identificador único no histórico
- id_tag: identificador da Tag armazenada
- valor: valor da variável em um estado de tempo
- data: inteiro que representa a data em que o valor foi salvo no banco de dados

O modelo Relacional acima descreve como os dados estão armazenados dentro do Banco de Dados. A primeira Tabela a ser destacada é a *Simulacao*. Nela se encontra as principais configurações da simulação como o todo, sendo elas o caminho físico do arquivo da modelagem no computador e a taxa de amostragem que a mesma foi realizada para aquisição de dados. A segunda tabela representa as *Tags*. Para cada variável de processo que será armazenado no banco de dados há no *DataBook* uma *Tag* associada a mesma. Um grupo de *Tags* constituí uma simulação, permitindo ao usuário selecionar quais variáveis de processo serão monitoradas. A terceira Tabela - *Historico_Tag* - representa o valor de uma dada *Tag* para uma dada Simulação em um instante de tempo. Ou seja, é o histórico da variável de processo de uma determinada simulação salva ao longo do tempo. É importante ressaltar que o valor salvo no banco de dados para definir o momento que foi registrado a variável é um inteiro que representa a quantidade de milissegundos transcorridos desde 01/01/1970 (Oracle, 2003). Para cada aquisição, todas as variáveis que foram salvas terão o mesmo

inteiro para representar o momento de aquisição. Isso é de fundamental importância para realizar as junções dos dados para analise, a fim de garantir a sincronização dos dados.

Todo o módulo de aquisição de dados foi implementado a partir do uso de *threads* (Oracle, 2003), tornando o mesmo independente do fluxo do programa principal. Como vantagem dessa tecnologia, o usuário poderá arbitrar o momento que os dados serão armazenados pelo supervisório, sem bloquear o programa principal de outras atividades. As aquisições dos dados pré-selecionados serão realizadas de forma cíclica em uma taxa de amostragem de 1 segundo. Entretanto é importante enfatizar que 1 segundo no tempo real não representa o mesmo tempo dentro da simulação, uma vez que esta permite o aumento da velocidade do processo, podendo então realizar várias horas de simulação em um tempo equivalente a apenas minutos do tempo real. Essa proporção entre quanto 1 segundo no tempo real equivale na simulação depende de cada caso, como será visto no caso de estudo.

Para exemplificar o funcionamento geral do módulo de aquisição primeiro deve-se configurar a simulação que será realizada (Figura 18), a qual deve preencher o título da simulação, taxa de amostragem, local físico no computador do arquivo de simulação e as variáveis que serão monitoradas. Após selecionar qual simulação será executada (Figura 19) e iniciar a execução da mesma, os dados serão salvos no banco de dados como descrito pelo diagrama de sequência da Figura 17. A sistemática para inserir os dados no banco de dados segue a sequencia de passos:

- Requisição do Usuário para salvar as variáveis selecionadas ao programa principal
- Instanciação da *Thread* que irá realizar a comunicação com o Supervisório e recuperar as *Tags* selecionadas
- Configuração do Banco de Dados para inserção dos dados
- Requisição de forma cíclica ao *OleAutomation* dos dados previamente selecionados pelo usuário
- Após cada requisição os dados serão salvos dentro de um vetor com o momento da requisição (Oracle, 2003)
- Após o fim da simulação, os dados contidos dentro do vetor serão salvos no banco de dados

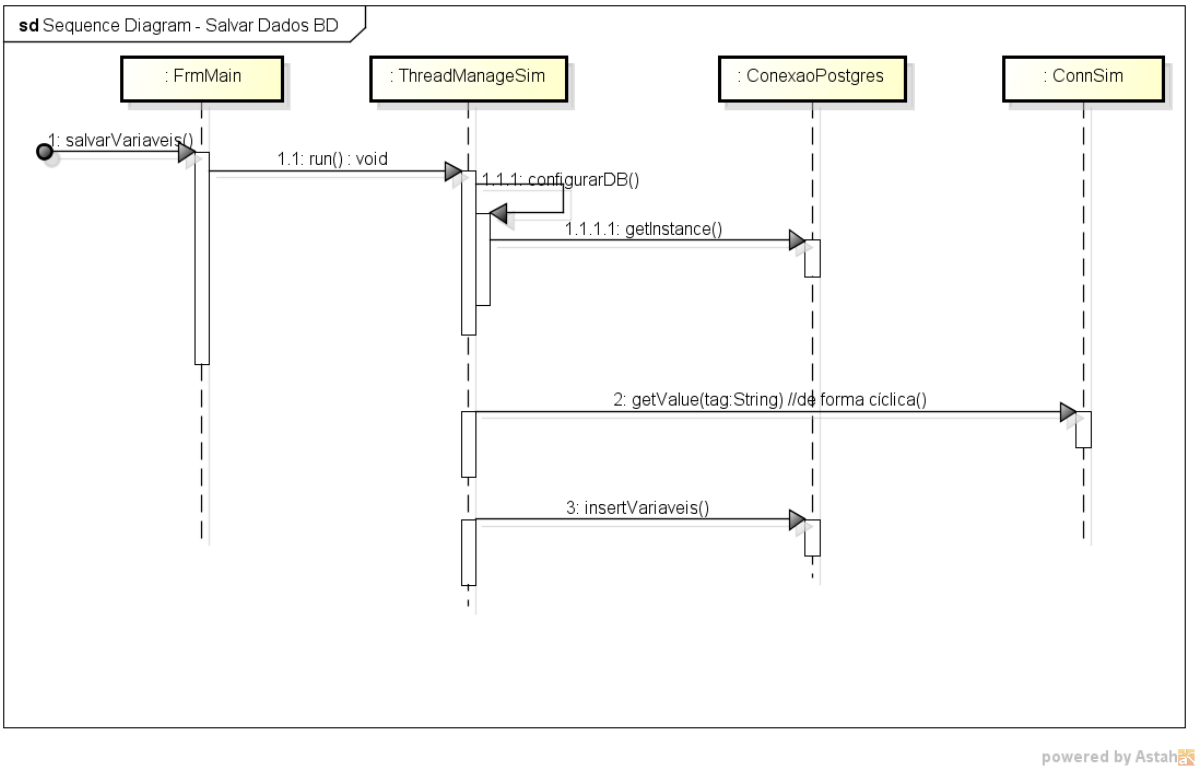


Figura 17: Inserir dados de simulação no Banco de Dados

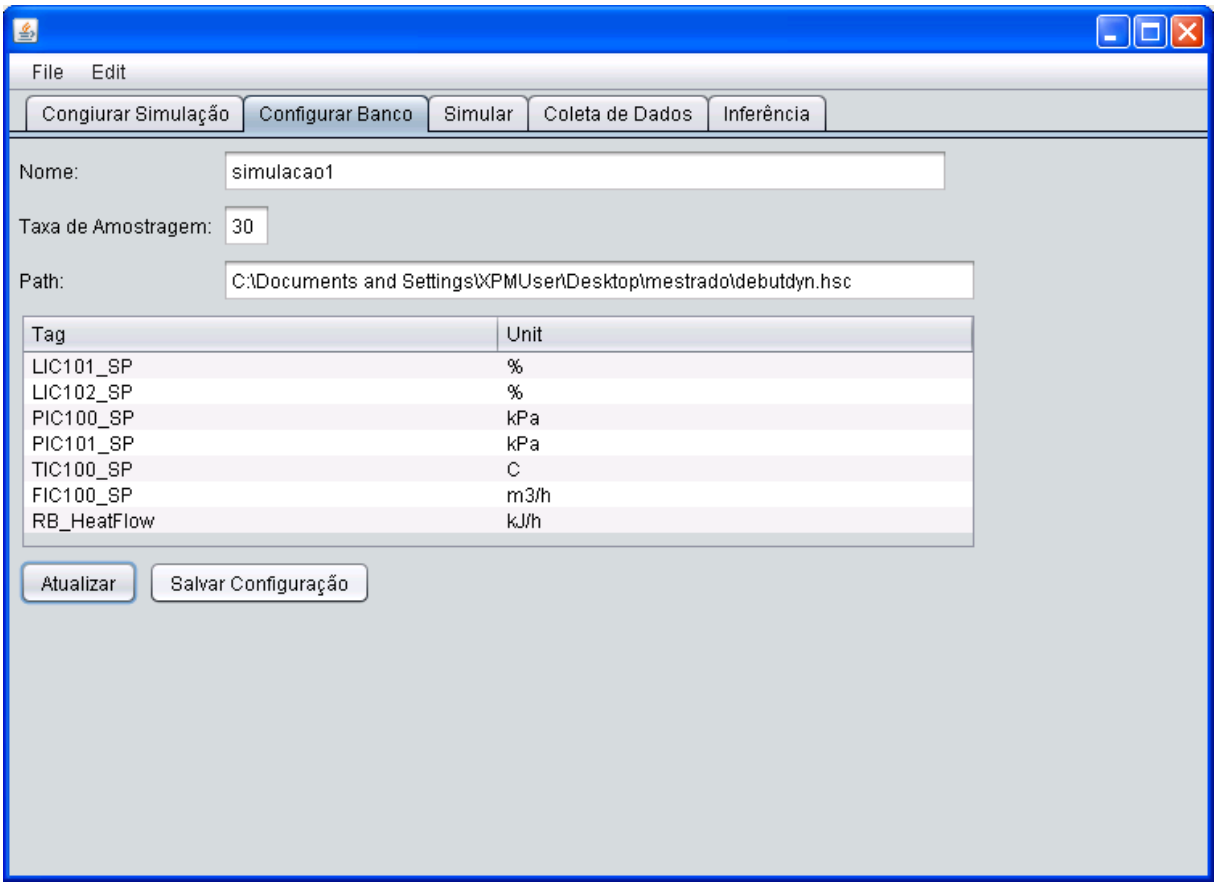


Figura 18: Configuração da Simulação

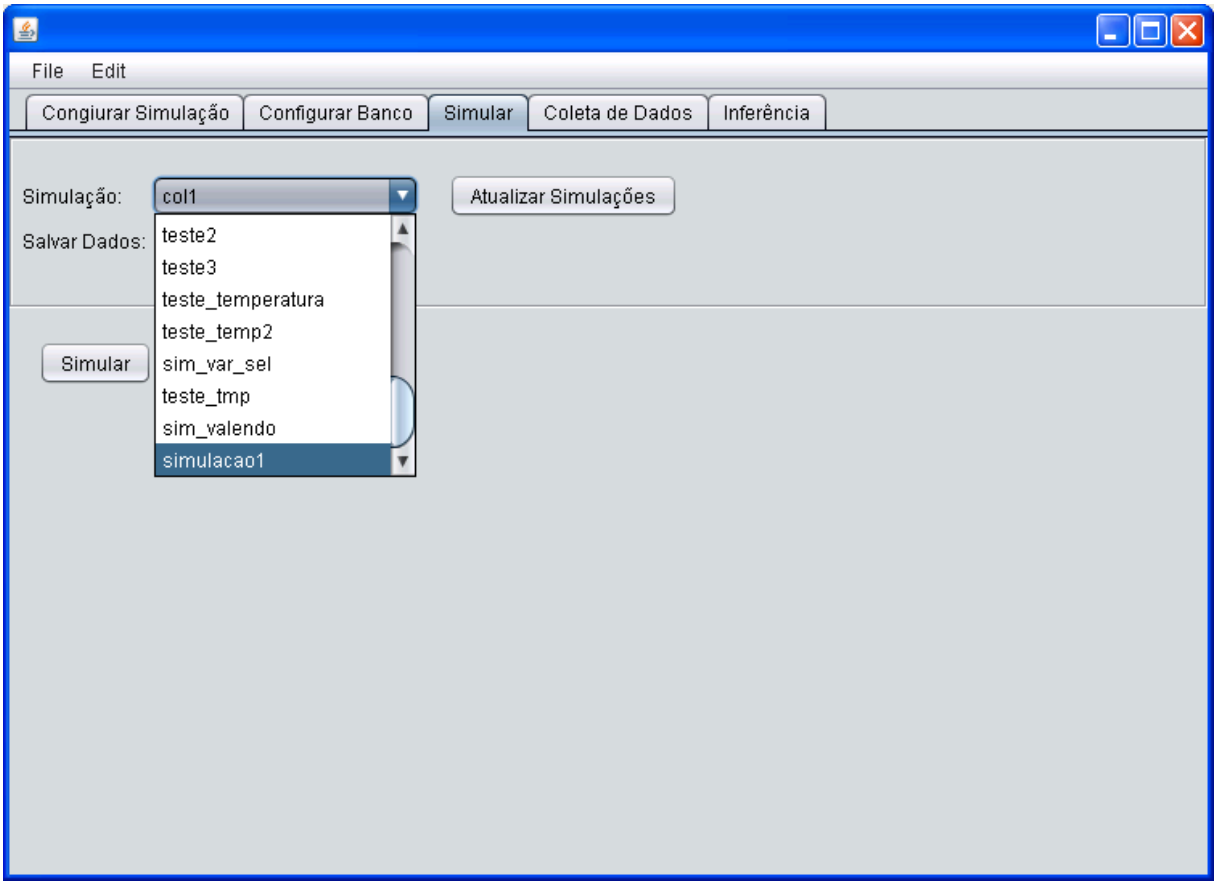


Figura 19: Seleção da Simulação

Módulo de Perturbação

A fim de modelar a dinâmica da planta através de técnicas inteligentes é necessária uma base de dados representativa, na qual registre as variáveis do processo sobre o espaço de estados que a mesma pode se encontrar para os mais diversos pontos de operação. Baseado em Zanata (2005) a metodologia adotada para perturbação da Planta se deu através da utilização de degraus de duração e amplitude randômicos conhecidos como *Pseudo Randon Signal* (PRS) nos valores de 5 a 15 unidades de tempo para duração da perturbação e de +/- 20% dos valores de *Set Points* dos controladores em estado estacionário utilizados na planta. É importante destacar que o tempo de duração do sinal PRS é contado em quantidade de unidades de tempos do mundo real, ou seja, 1 segundo no tempo real deve ser multiplicado pela velocidade que simulação executa o processo. A fim de ter o valor real de quanto tempo durou o sinal modificado na simulação, por exemplo, se a taxa for de 30 para 1, 1 segundo no tempo real representa 30 segundos na simulação.

Para realizar a perturbação da planta, inicialmente deve ser selecionadas as variáveis desejadas como visto na Figura 20. Após a seleção deve ser acionado o botão de perturbação.

Internamente o supervisório executará o procedimento demonstrado no diagrama de sequencias da Figura 21.

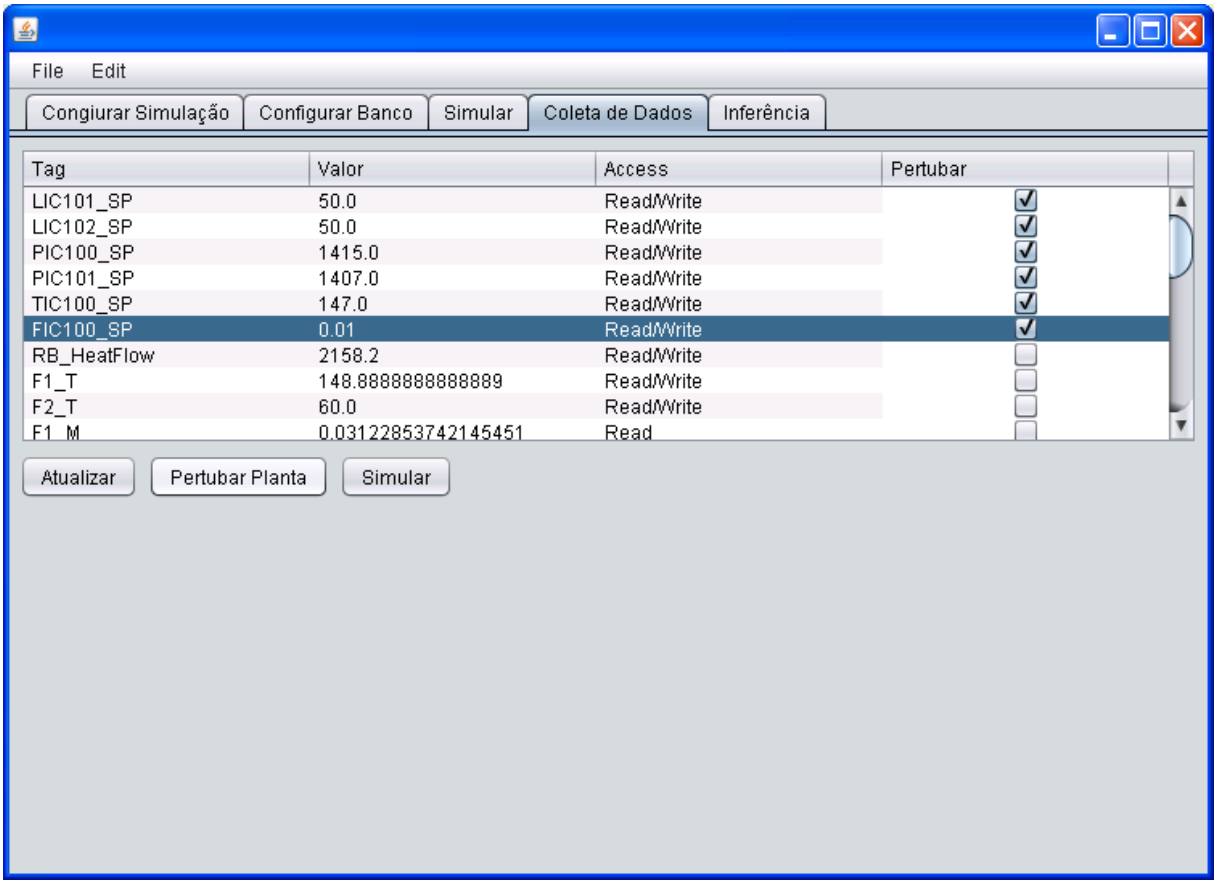


Figura 20: Configuração das variáveis a serem perturbadas

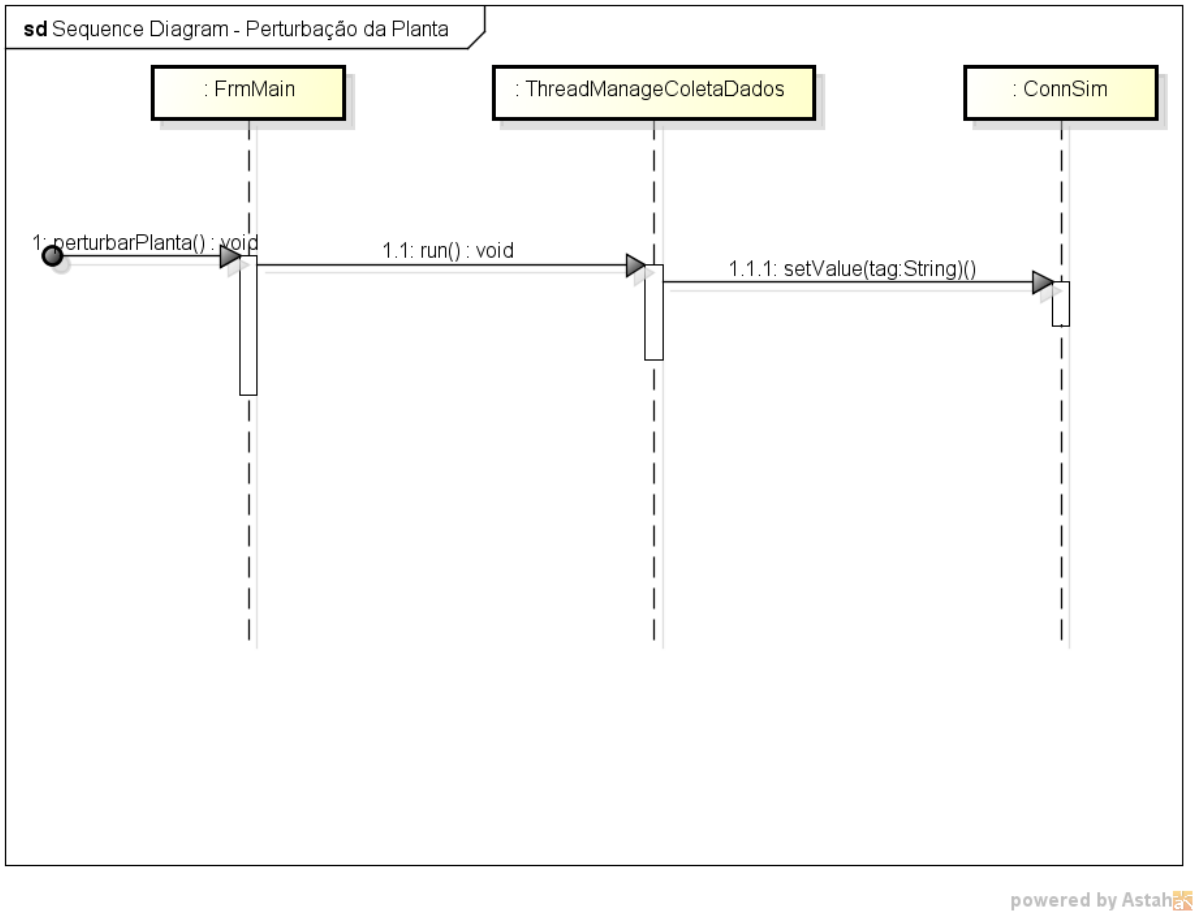


Figura 21: Diagrama de Sequência para Perturbar a planta simulada

O procedimento da perturbação consiste na sequência de passos: recuperar os valores médios das variáveis desejadas em estado estacionário; gerar dois valores randômicos: o primeiro para perturbar a amplitude das variáveis e o segundo para saber a quantidade de unidades de tempo que será forçado tal situação de perturbação. À medida que o sistema for perturbado, o módulo de aquisição de dados pode em concomitância ser habilitado, a fim de registrar no banco de dados a dinâmica das variáveis de processos selecionadas previamente no módulo de aquisição de dados.

O módulo de perturbação, tal como no módulo de aquisição de dados, foi implementado dentro de uma *thread* secundária independente ao resto do programa, o qual tornar arbitrário ao usuário o momento que melhor lhe convier para perturbação da planta.

Módulo de Inferência de Variáveis

Para o modelo de inferência foi utilizado o Pacote *Encog* para o uso de Redes Neurais Artificiais. Esse pacote disponibiliza várias arquiteturas e técnicas de treinamentos e possibilidade ao usuário alterar os pesos internos da rede (Heaton Research , 2008). Como validação do pacote foi utilizado

para comparação o ToolBox de Redes Neurais do Matlab. O pacote *Encog* demonstrou performance semelhante, logo, por se tratar de uma ferramenta gratuita, foi selecionada ao invés ToolBox do Matlab. Nesse mesmo módulo foi desenvolvido funções para importações de pesos para a rede. Tal conjunto de funções se deu no intuito de aumentar a maleabilidade do sistema, uma vez que estando os dados disponíveis no banco de dados, é possível utilizar outros pacotes para treinamento de redes neurais ou técnicas inteligentes, tal como *Weka* (University of Waikato, 2011). Uma vez treinada, os pesos poderão ser incorporados ao módulo de inferência através das funções de importação. Apenas é necessário que seja seguido o padrão da ordem dos pesos. O padrão que os arquivos de importações devem seguir é:

- **Pesos da camada de entrada:** deve ter tantas linhas quantos forem os neurônios da camada oculta e em cada linha devem ser separados por vírgulas os valores dos pesos tantos quanto for o tamanho da entrada da RNA. Como exemplo, se temos uma rede com a configuração 15:30:1, temos então no primeiro arquivo para importar os dados da camada de entrada 30 linhas, sendo cada uma contendo 15 valores dos pesos separados por vírgulas.
- **Pesos da camada oculta:** deve possuir uma única linha e terá os valores dos pesos separados por vírgulas. Como exemplo, uma rede com configuração 15:30:1 haverá uma única linha com 30 valores separados por vírgulas.
- **Bias da camada oculta:** devem possuir tantas linhas quanto forem os pesos da camada escondida e cada linha deve possuir um único valor. Como exemplo, uma rede com configuração 15:30:1 haverá 30 linhas cada uma possuindo um único valor
- **Bias da camada de saída:** deve possuir um único valor. Como exemplo, uma rede com configuração 15:30:1 haverá um único valor.

Módulo de Pré-processamento

O módulo de pré-processamento é responsável no supervisório por duas funcionalidades: transformar os dados em relação ao tempo - conhecidos também como atrasadores ou regressores - tanto na entrada como na saída, e realizar a redução de variáveis através de técnicas como PCA. A necessidade da redução de dados se dá pelo fato de que o treinamento para a inferência de uma variável não-linear, geralmente, apresenta a necessidade de um número relativamente grande de variáveis de entradas (principalmente devido ao número de regressores do conjunto de entradas e saídas). Para isso, a fim de diminuir o conjunto de variáveis de entradas, foi utilizado a técnica de Análise de Componentes Principais.

Estudo de Caso

4.1 Coluna Selecionada

Como planta didática para o desenvolvimento do supervisório, foi selecionada a planta *debutdyn.*

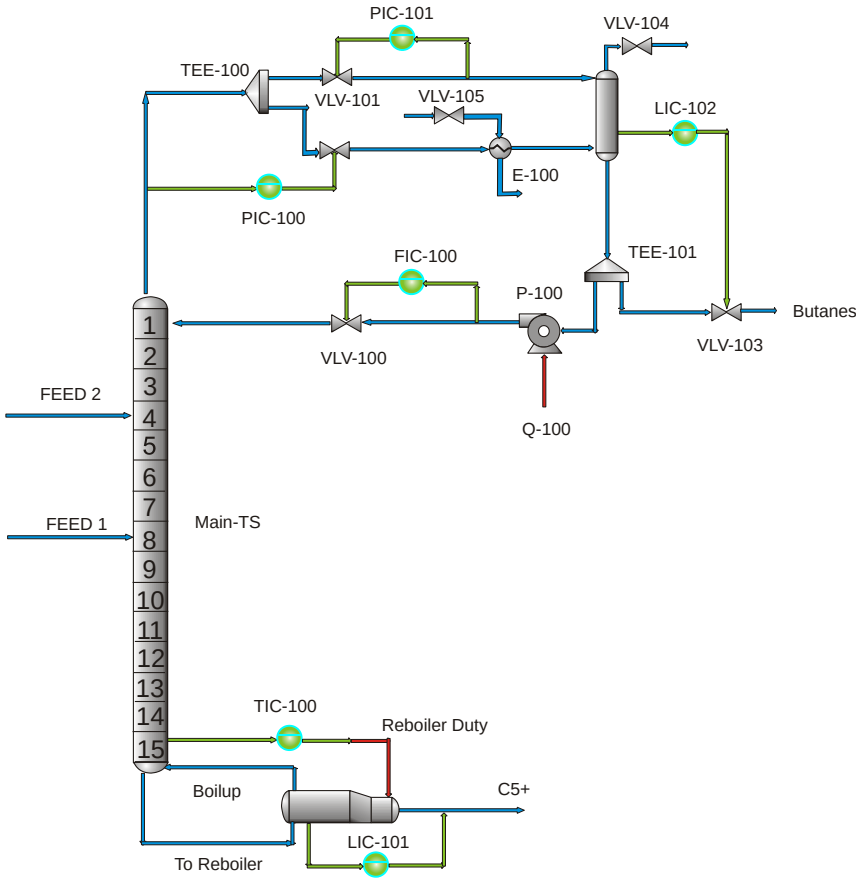


Figura 22: Coluna Debutdy

A planta *debutdyn* é um modelo geral para uma coluna de destilação usando mais detalhes do que apenas o módulo do condensador.

O controle de pressão da coluna é obtido principalmente manipulando o excesso de vapor de em torno do condensador. O PIC-101 controla o excesso da pressão acumulada liberando-o de forma controlada. A pressão do prato de topo é controlado pelo PIC-100, que essencialmente mantém uma constante queda de pressão entre o prato de topo e o do acumulador.

A vazão de refluxo está sobre o controle de fluxo, com o nível do acumulador sendo controlado pela vazão do produto. Há o controle de temperatura no fundo da coluna, e o controle de nível para o vaso.

Itens adicionais que poderiam ser modelados são válvulas de alívio - provavelmente no topo da coluna, e um para o acumulador. O produto de fundo poderia ser colocado no controle de análise, cascateado para o controle de temperatura.

Dados Selecionados

Toda e qualquer variável de processo dentro da simulação pode ser salva no supervisório, entretanto tal facilidade não é encontrada ao se tratar de processos reais, principalmente quando se trata de composições químicas. Um dos objetivos do trabalho é através da utilização de variáveis de fácil aquisição em um processo real (pressão, vazão e temperaturas) inferir o valor de composições químicas especialmente do produto de saída, uma vez que são esses valores que devem ser mantidos dentro de especificações corretas, ou seja, diretamente associado à lucratividade da planta.

É importante salientar que em identificação de sistemas há três tipos de variáveis de entradas (variáveis que aferem a saída), todas elas independentes entre si, que são:

- Variáveis Manipuladas
- Perturbações medidas
- Perturbações não-medidas

Além desses parâmetros, há variáveis internas que influenciam na composição dos produtos, portanto, é comum que variáveis como temperaturas dos pratos sejam utilizadas como entradas no sistema de inferência (Kano, Miyazaki, Hasebe, & Hashimoto, 2000 e Fileti, Pedrosa, & Pereira, 1999).

A partir da literatura algumas variáveis já surgem como candidatas naturais para servirem de base na utilização de algoritmos inteligentes para inferência da composição química, uma vez que estão diretamente ligadas a qualidade final do produto, são elas: a temperatura dos pratos, o fluxo molar da saída do topo e o fluxo de calor do Reboiler (Zanata, 2005).

Como critério das temperaturas de pratos selecionadas para inferência da composição foi utilizado a correlação entre os valores de cada temperatura com a composição do n-butane a partir de uma simulação utilizando o módulo de Perturbação com 321 amostras.

A partir da Figura 23 foram selecionados as temperaturas dos pratos 1, 2, 3, 4 e 15.

Para completar o conjunto das variáveis utilizadas foram selecionadas os *SetPoints* de todos controladores da planta, uma vez que são essas onde o sistema atua na perturbação da planta modificando a dinâmica da planta, sendo elas: LIC101, LIC102, PIC100, PIC101, TIC100, FIC100.

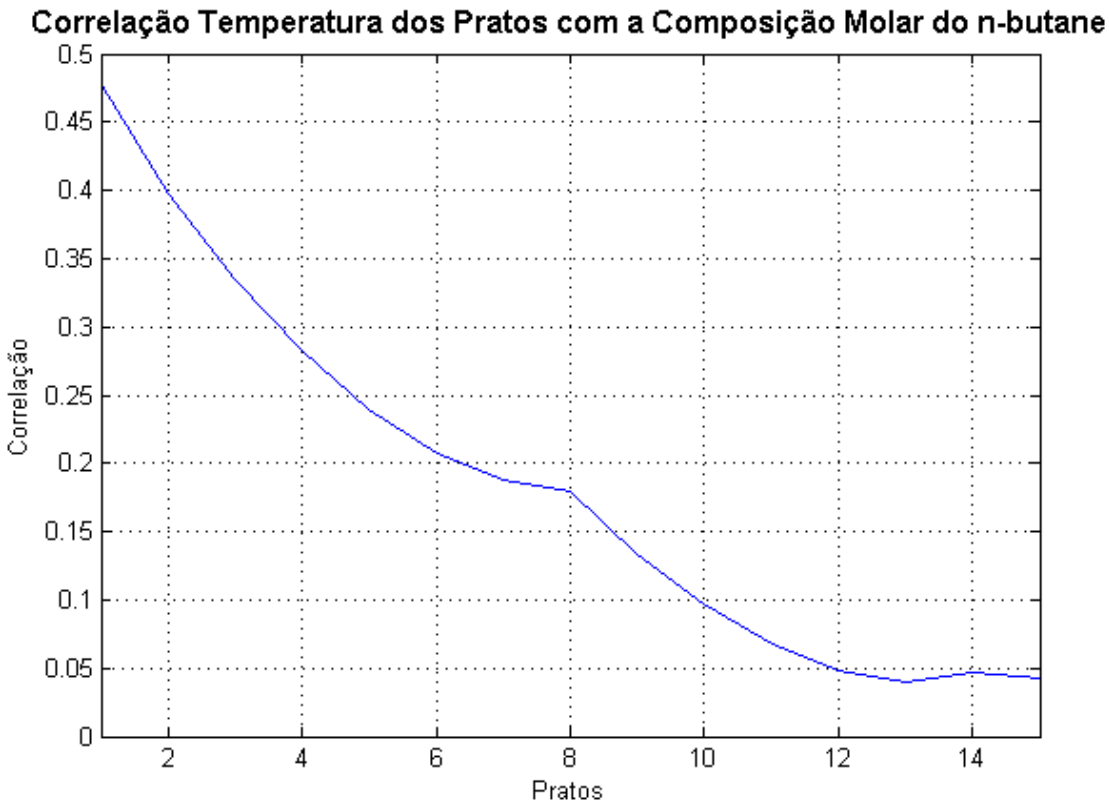


Figura 23: Gráfico exibindo a correlação entre temperaturas dos pratos e o n-butane

Inferência de Variáveis

A metodologia utilizada para selecionar a melhor rede para a inferência do *n-butane* através das variáveis selecionadas da seção anterior foi através da experimentação de várias configurações de redes até encontrar com o melhor desempenho do erro de validação. Inicialmente foram gerados 1532 registros os quais representam um total de 932,95 minutos a um fator de tempo real desejado de 30 para 1 – ou seja, a cada 1 minuto no tempo real era equivalente a 30 minutos na simulação - das 14 variáveis: *setpoints* dos controladores LIC101, LIC102, PIC100, PIC101, TIC100, FIC100; Temperaturas dos pratos 1, 2, 3, 4, 15; fluxo molar da saída do topo e o fluxo de calor do Reboiler e a composição do n-butane, sendo as 13 primeiras o conjunto de entrada e o *n-butane* o conjunto para saída desejada. Os dados foram divididos em dois conjuntos: treinamento consistindo dos pontos entre 1 a 800 e 1200 a 1400 em um total de 1000 pontos e para validação foram utilizados os pontos de 900 a 1200 e de 1400 ao fim gerando um total de 532 pontos. Todos os pontos foram normalizados de -1 a 1.

Para configuração da rede ideal para a inferência do *n-butane* foram variados alguns parâmetros: ordem do sistema (2, 3, 4 e 5) - conhecida também como atrasos - na entrada e saída da rede, quantidade de neurônios da camada oculta (10, 15 e 20) e porcentagem da informação utilizada pelo PCA (90%, 95%, 98%). Foram geradas 50 redes neurais para cada quantidade de neurônios, a fim de se ter um valor estatístico confiável para o erro, gerando um total de 150 redes para cada configuração de atraso e porcentagem de informação utilizada no PCA. Logo, um total de 1800 redes neurais distintas foram testadas. Como critério de avaliação de seleção da rede neural foi levado em consideração o menor erro de validação.

Na Figura 24 nota-se como a rede neural é sensível a ordem do sistema - definido pela quantidade de atrasos na entrada e saída, por exemplo, um sistema de ordem 2 possui dois atrasos na entrada e dois atrasos na saída. Para cada número de atrasos foi selecionado a rede com melhor desempenho variando a quantidade de neurônios da camada oculta. É importante observar o caso de ordem do sistema igual a zero - sem atrasos na entrada e na saída - com um resultado de forma insatisfatória e, à medida que é aumentado a quantidade de atrasos o sistema, a RNA se torna mais estável, gerando resultados satisfatórios até um momento de saturação. Acredita-se que isso se deve ao fato da variável ser bastante não-linear e ser um sistema dinâmico, logo, a rede neural apenas com a entrada atual não consegue realizar a predição para a inferência da composição.

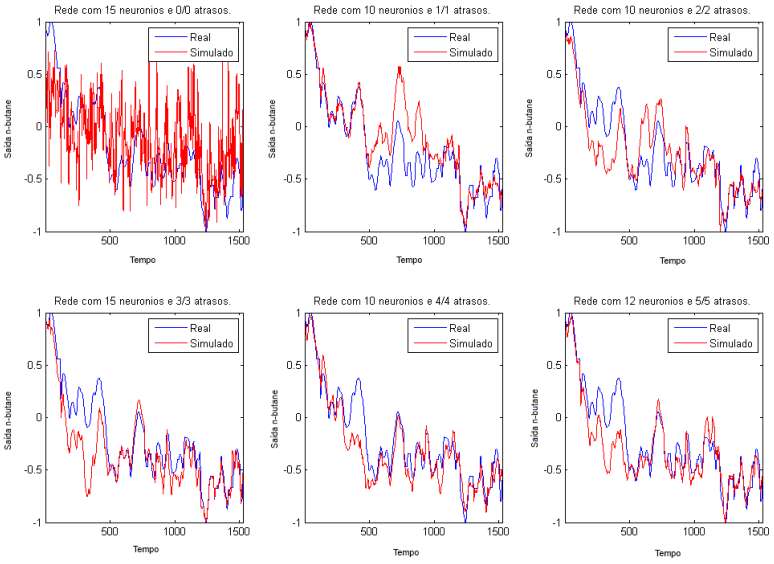


Figura 24: Saída das RNAs variando os atrasos com a melhor quantidade de Neurônios da camada oculta.

Na Figura 25 é possível notar o processo de *Overfiting* no treinamento das redes, uma vez que o erro de validação aumenta de forma direta com o número de neurônios, ou seja, a rede neural está decorando alguns padrões, não conseguindo extrapolar e generalizar para os demais dados.

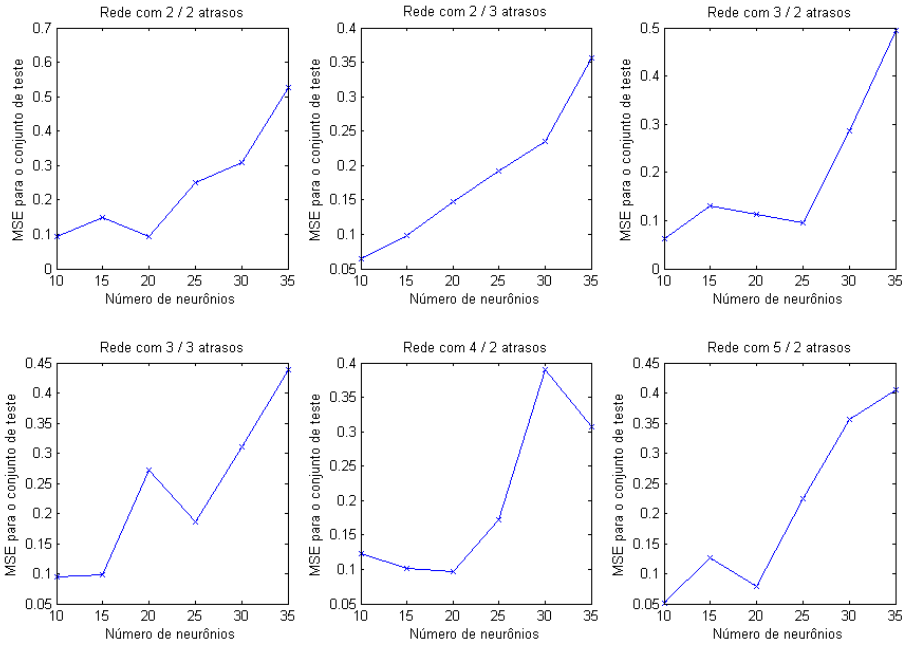


Figura 25: Erro por quantidade de Neurônios

Na **Erro! Auto-referência de indicador não válida.** há a compilação dos erros de validação para cada configuração. Nela está contida apenas o erro do melhor teste – para cada par dos parâmetros (atrasos, % de informação do PCA) foi variado o número de neurônios da camada oculta, sendo cada valor executado 50 vezes. A melhor rede de fato foi utilizando todas as variáveis, sem a redução imposta pelo PCA. Entretanto a utilização do PCA pode aumentar um pouco o erro, entretanto ainda possuí um resultado satisfatório.

Tabela 1: Tabela dos Erros de Validação

%PCA/Atrasos(entrada/saída)	0/0	1/1	2/2	3/3	4/4	5/5
100%	0.1572	0.0107	0.0115	0.0140	0.0125	0.0100
90%	X	X	0.0457	0.0366	0.0128	0.0208
95%	X	X	0.0122	0.0146	0.0253	0.0355
98%	x	X	0.0200	0.0136	0.0310	0.0253

5. Discussão e Conclusão

No mundo atual há uma enorme tendência no tocante do desenvolvimento de modelagens sobre processos industriais, principalmente ao se tratar da industrial do petróleo, uma vez que a mesma necessita atender critérios de segurança em testes de controles, prever comportamentos, redução de custos entre outros. A partir de tal fato, a necessidade de desenvolver supervisórios sobre esses novos ambientes, tal que se torne transparente ao operador - se o mesmo está trabalhando sobre uma planta simulada ou real - torna-se uma realidade. Desenvolver então uma arquitetura que torne possível tal transparência e que permita de forma fácil acrescentar novos módulos ao supervisórios - tal como estratégias de controles, técnicas inteligentes de inferências de variáveis – utilizando as melhores tecnologias para cada fim, sem se preocupar em qual tipo de linguagem ou ambiente os mesmos estão desenvolvidos, foi o foco do trabalho. Com isso em mente, têm-se como resultado o desenvolvimento de um Supervisório Inteligente capaz de armazenar e monitorar dados de uma coluna de destilação simulada, além de realizar inferência da composição química da saída de forma satisfatória.

De forma mais detalhada, as tecnologias utilizadas no desenvolvimento do supervisório podem ser listadas como: o linguagem de programação JAVA para o desenvolvimento do supervisório, o pacote Encog para implementação das Redes Neurais Artificiais, o *OleAutomation* para realizar troca de dados com a simulação modelada no Unisim, e o Banco de Dados Postgres para armazenamento das informações advindas da simulação. A arquitetura promovida pela comunicação *OleAutomation* aliado ao armazenamento de dados em um Banco de Dados, sendo todos ligados e centralizados pelo supervisório desenvolvido, permite ao mesmo a flexibilidade da agregação de futuros módulos em qualquer tecnologia, desde que a mesma dê suporte a comunicação por *OleAutomation*. Portanto, o supervisório desenvolvido apresenta uma arquitetura consistente, escalável e de fácil manutenção para suportar os requisitos dos novos ambientes de desenvolvimento caracterizado pelo uso de simulações na indústria do petróleo, além de cumprir com sucesso a monitoração, atuação e armazenamento das variáveis de processos da simulação e, além disso, realizar através de técnicas inteligentes a inferência da composição química de saída da planta a partir de variáveis secundárias como temperatura, vazão e pressão. É importante ressaltar que a medição de uma variável química em plantas reais é um processo demorado e caro, logo, a utilização de variáveis de fácil medição em uma planta real (temperaturas, vazão e pressão), para a realização da inferência de uma composição química é, sem dúvida, uma característica essencial nos supervisórios inteligentes, a qual foi contemplada pelo supervisório desenvolvido.

A partir do desenvolvimento de tal arquitetura para o desenvolvimento do supervisório, têm-se como trabalhos futuro o desenvolvimento de novos módulos, a fim de conceber um supervisório mais robusto e completo para auxílio do suporte à decisão. A princípio esses módulos são: módulo para controle utilizando técnicas inteligentes (Fuzzy), módulo de detecção de tendências não-supervisionado, módulo para transmissão de dados via Fieldbus. Além do desenvolvimento de novos módulos, há o intuito de desenvolvimento de uma proposta de metodologia para agregação das diversas técnicas inteligentes, a fim de realizar detecção automática de anomalias e correção em tempo real. Para isso, a agregação de módulos de técnicas não-supervisionadas para detecção de diversos pontos de operação, além do controle inteligente, a fim de levar a planta ao ponto de operação desejado, será o foco principal nos futuros projetos.

6. Referências Bibliográficas

Abou-Jeyab, R. A., Gupta, Y. P., Gervais, J. R., Branchi, P. A., & Woo, S. S. (2001). Constrained multivariable control of a distillation column using a simplified model predictive control algorithm.

Almeida, A. (2008). *REFINO.* Acesso em 26 de 11 de 2009, disponível em fadepe: http://www.fadepe.com.br/restrito/conteudo_pos/potro2_AULA-REFINO.ppt

Ansari, R. M., & Tadé, M. O. (2000). Nonlinear Model-Based Process Control: Application in Petroleum Refining. London: Springer.

Bawazeer, K., & Zilouchian, A. (1997). Prediction of products quality parameters of a crude. *International Conference* , 157-162.

Bettoni, A., Bravi, M., & Chianese, A. (2000). Inferential control of a sidestream column.

Bo, C. M., Li, J., Zhang, S., Sun, C. Y., & Wang, Y. R. (2003). The application of neural network soft sensor technology to an advanced control system of distillation operation. *Proceedings of the Internacional Joint Conference on Neural Networks , II*, 1054-1058.

Booch, G., Rumbaugh, J., & Jacobson, I. (2006). *UML Guia do Usuário.* CAMPUS.

Chiang, H., Russel, L., & Braatz, D. (2001). *Fault Detection and Diagnosis in Industrial System.* London: Springer.

Chiang, L. H., Russell, E., & Braatz, R. (2001). *Fault Detection and Diagnosis in Industrial Systems.* Springer.

Ding, X., & Guo, L. (1996). Observer-based fault detection optimized in the frequency domain. *Proceedings of the 13th IFAC World Congress* , pp. 157-162.

ELIPSE SOFTWARE. (2009). *Elipse Software - E3 and Elipse SCADA products home - software HMI SCADA - automação industrial*. Acesso em 26 de 11 de 2009, disponível em http://www.elipse.com.br/inicial.aspx?idioma=1

ELIPSE SOFTWARE. (8 de 10 de 2009). Manual de Referência de Scripts do E3.

ELIPSE SOFTWARE. (08 de 10 de 2009). Tutorial do E3.

Elmasri, R., & Navathe, S. (2005). *Sistemas de Banco de Dados.* PEARSON.

FAIRBANKS, M. (Agosto, 2002). Petrobrás: Investimentos ajustam refino para usar mais óleo nacional, Química e Derivados.

Fileti, A., Pedrosa, L., & Pereira, J. (1999). A self tuning controller for multicomponent batch distillation with soft sensor inference based on a neural networks. *Computers & Chemical Engineering Supplment , 23*, pp. S261-S264.

Fortuna, Luigi, Giannone, P., Graziani, S., & Xibilia, M. G. (2007). Virtual instruments based on stacked neural networks to improve product quality monitoring in a refinary. *IEEE Transactions instrumentation and Measurement* , 95-101.

Garcia, A., & Frank, M. (1996). On the relationship between observer and parameter identification based approaches to fault detection . *Proceedings of the 13th IFAC World Congress, vol. N. Piscataway.* , pp. 25-29.

Garcia, C. (1997). *Modelagem e Simulação.* edusp.

Haykin, S. (2007). *Redes Neurais - Princípios e Prática.* BOOKMAN.

Heaton Research . (2008). Acesso em 24 de 07 de 2012, disponível em Encog Machine Learning Framework: http://www.heatonresearch.com/encog

Henley, E. J., & Seader, J. D. (1981). Equilibrium-Stage Separation Operation in Chemical Engineering. Canada.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks* , 359-366.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal componentes . *Journal of Psychology* , 417-441.

Isermann, R. (1994). Integration of fault detection and diagnosis methods. *Fault Detection, Supervision, and Safety for Technical Processes* .

Jiménez, L., Basualdo, M., Toselli, L., & Rosa, M. (2000). Dynamic modeling of batch distillation: comparison between commercial software. *European Symposium on Computer Aided Process Engineering* , pp. 1153-1158.

Jolliffe, I. (2002). *Principal Component Analysis* (2º ed.). Springer.

Junior, J. R. (2007). Sistemas Integrados de Monitoramento de Instrumentos em Rede Foundation Fieldbus para Melhoria dos Processos de Medição e Controle na Indústria do Petróleo. *Trabalho de Conclusão de Curso* . Universidade Federal do Rio Grande do Norte.

Kano, M., Miyazaki, K., Hasebe, S., & Hashimoto, I. (2000). Inferencial constrol system of distillation compositions using dynamic partial least squares resgression. *Journal of Process Control , 10*, pp. 157-166.

Kobryn, C., Booch, G., Jacobson, I., & Rumbaugh, J. (2006). *UML Essencial.* Bookman.

Lima, F. S. (2004). Estratégia de escalonamento de controladores pid baseado em regras fuzzy para redes industriais foundation fieldbus usando blocos padrões. *Dissertação de Mestrado* . Universidade Federal do Rio Grande do Norte.

Linhares, L. L. (2009). *Sistema de Inferência baseado em redes neurais para controle de plantas de processamento de gás natual.* Dissertação de Mestrado, Universidade Federal do Rio Grande do Norte.

Loureiro, F. M. (1995). Dissertação de Mestrado: Desenvolvimento de um Gerador de "Scheduling" para uma Indústria de Produção sob Encomenda: Uma Abordagem Baseada no Uso de Controladores Difusos e Algoritmos Genéticos. Universidade Federal de Santa Catarina.

Lucena, P. B. (2005). *Análise de um controlador baseado no jacobiano estimado da planta através de uma rede neural.* Dissertação de Mestrado, Universidade Federal do Rio Grande do Norte.

Luo, X., Zhang, C., & Jennings, R. (2002). A hybrid model for sharing information between fuzzy, uncertain and default reasoning models in multi-agent systems. . *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* , pp. 401-450.

Maitelli, A. L. (2003). Controladores lógicos programáveis. *Notas de aula da disciplina "Controladores Lógicos Programáveis".*

Marangoni, C. (2005). Implementação de uma Estratégia de Controle com Ação Distribuída em uma Coluna de Destilação. Brasil: Universidade Federal de Santa Catarina.

Marlin, T. (1995). Process Control: designing processes and control systems for dynamic performance. Singapora: Mc-Graw Hill.

Microsoft. (2010). Acesso em 27 de junho de 2011, disponível em Site da Rede de Desenvolvimento da Microsoft: http://msdn.microsoft.com/en-us/library/aa365574(v=vs.85).aspx

MICROSOFT. (2009). *MFC Library Reference Automation*. Acesso em 26 de 11 de 2009, disponível em Automation: http://msdn.microsoft.com/en-us/library/dt80be78.aspx

MICROSOFT. (2009). *MFC Library Reference Automation*. Acesso em 26 de 11 de 2009, disponível em Automation: http://msdn.microsoft.com/en-us/library/dt80be78.aspx

MICROSOFT. (s.d.). *ODBC--Visão geral de Open Database Connectivity*. Acesso em 26 de 11 de 2009, disponível em support.microsoft.com: http://support.microsoft.com/kb/110093

MICROSOFT. (2009). *Usar mensagens DDE ou OLE e bibliotecas DDEML*. Acesso em 26 de 11 de 2009, disponível em support.microsoft: http://support.microsoft.com/kb/74802

MICROSOFT. (2009). *VBScript*. Acesso em 26 de 11 de 2009, disponível em msdn.microsoft: http://msdn.microsoft.com/en-us/library/t0aew7h6%28VS.85%29.aspx

Mingoti, S. A. (2005). *Análise de dados através de métodos de estatísticas multivariada: uma abordagem aplicada.* UFMG.

Nakamura, E. (s.d.). Acesso em 27 de junho de 2011, disponível em Site do Laboratório de Sistemas Distribuídos - Unicamp: http://www.lsd.ic.unicamp.br/mc514/sites/default/files/Comunica%C3%A7%C3%A3o%20entre%20processos.pdf

Nascimento JR., C. L., & Yoneyama, T. (2000). *Artificial em Controle e Automação.*

Norgaard, Magnus, Ravn, O., Poulsen, N. K., & Hansen, L. K. (2000). *Neural Networks for Modelling and Control of Dynamics Systems.* Springer-Verlag.

OLI SYSTEMS. (2007). Customization Guide. *Manual* .

Oracle. (2003). *System (Java 2 Platform SE v1.4.2)*. Acesso em 12 de 07 de 2012, disponível em http://docs.oracle.com/javase/1.4.2/docs/api/java/lang/System.html#currentTimeMillis()

Oracle. (2003). *Thread (Java 2 Platform SE v1.4.2)*. Acesso em 12 de 07 de 2012, disponível em http://docs.oracle.com/javase/1.4.2/docs/api/java/lang/Thread.html

Pearson, K. (1901). On lies and planes of closest fit to systems of points in space . *Philosophical Magazine* , 559-572.

Pegden, C. (1990). Introduction to Simulation Using SIMAN. New Jersey: McGraw-Hill.

Rebouças, D. L. (Junho de 2009). Sistema de Inferência Neural e Processamento Estatístico Multivariável Aplicado a Indústria do Petróleo. *Trabalho de Conclusão de Curso* . Natal, RN, Brasil: Universidade Federal do Rio Grande do Norte.

Rengaswamy, R., & Venkatasubramanian, V. (1992). An Integrated Framework for Process Monitoring, Diagnosis, and Control using Knowledge-based Systems and Neural Networks. *IFAC* , 49-54.

Sebesta, R. (2003). CONCEITOS DE LINGUAGEM DE PROGRAMAÇAO. Bookman.

Shannon, R. E. (1998). INTRODUCTION TO THE ART AND SCIENCE OF SIMULATION. Industrial Engineering, Texas A&M University.

Shinskey. (1996). Process Control Systems.

Shinskey, F. G. (1984). Distillation Control, 2º ed. Massachusetts: Mc-Graw-Hill.

Silva, I. N. (2010). *Redes Neurais artificiais para engenharia e ciências aplicadas: Curso Prático.* São Paulo: Artileber.

Soos, M., & Smejkal, Q. (2001). Comparison of computer simulation of reactive distillation using ASPEN PLUS and HYSYS software. *Chemical Engineering and Processing* , pp. 413–418.

Staudt, P. B. (2007). Dissertação de Mestrado: Modelagem e Simulação de Colunas de Destilação. Porto Alegre, Rio Grande do Sul, Brasil.

University of Waikato. (2011). *Weka 3: Data Mining Software in Java*. Acesso em 24 de 07 de 2012, disponível em http://www.cs.waikato.ac.nz/ml/weka/

Uraikul, V., W. Chan, C., & Tontiwachwuthikul, P. (2007). Artificial intelligence for monitoring and supervisory. *Engineering Applications of Artificial Intelligence* , 115-131.

Viana, D. W. (2008). *SISTEMA SCADA SUPERVISÓRIO.* RJ: IFF.

Warne, K., Prasad, G., Siddique, N. H., & Maguire, L. P. (2004). Development of a hybrid pca-anfis measurement system for monitoring product quality in the coating industry. *IEEE Internacional Conference on Systems, Man and Cybernetics* .

Werle, L. O. (2007). Minimização dos Transientes através do Aquecimento Distribuído em uma Coluna de Destilação. *Dissertação. Mestrado em Engenharia Química* . Universidade Federal de Santa Catarina.

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining Pratical Machine Learning Tools and Techniques.* Elsevier.

Yoon, S., & MacGregor, F. (2004). Principle-component analysis of multiscale data for process monitoring and fault diagnosis. *AIChE Journal* , pp. 2891-2903.

Zanata, D. R. (2005). DESENVOLVIMENTO DE SENSOR VIRTUAL EMPREGANDO REDES NEURAIS PARA MEDIÇÃO DA COMPOSIÇÃO EM UMA COLUNA DE DESTILAÇÃO. *Dissertação de Mestrado* . São Paulo: Escola Politécnica da Universidade de São Paulo.

7. Anexo

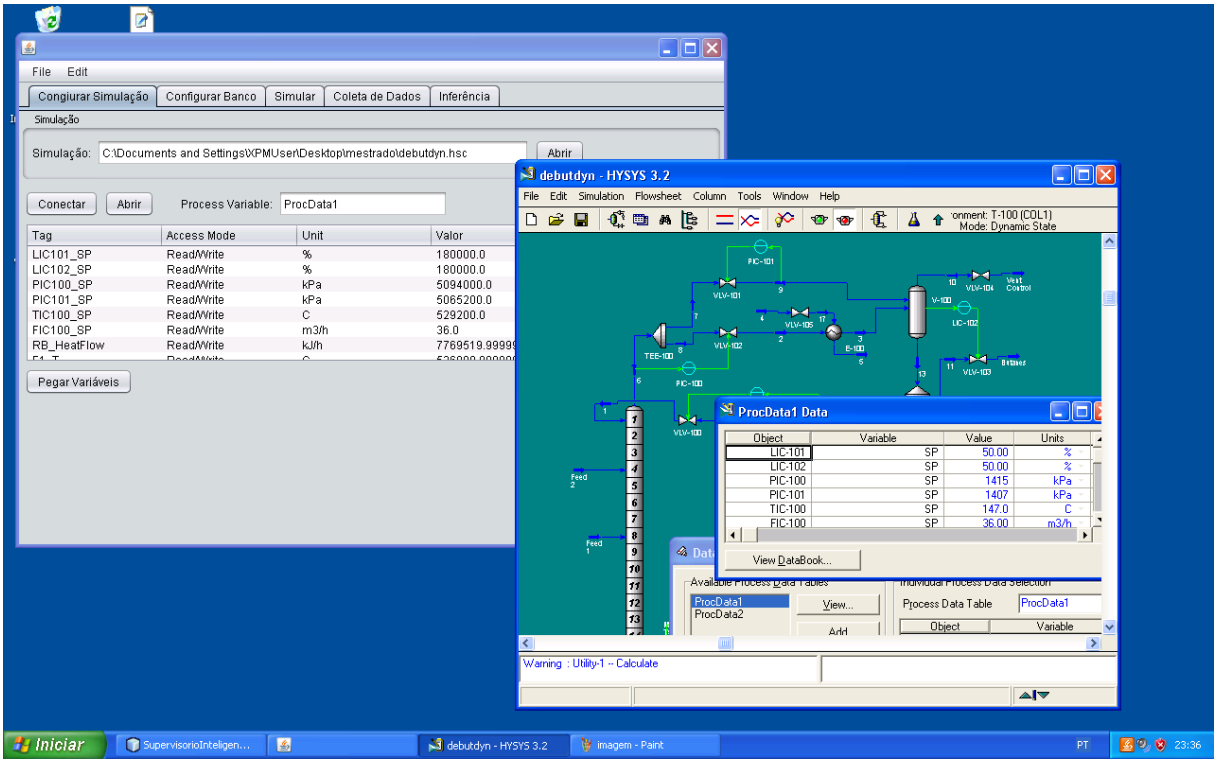


Figura 26: Supervisório em atuação

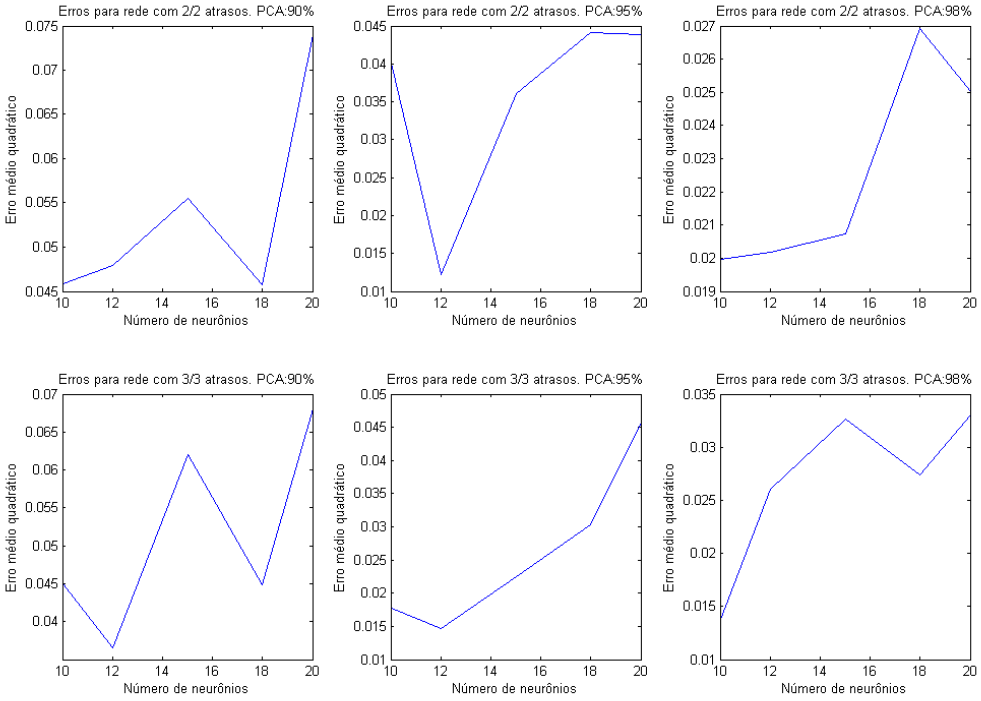


Figura 27: Erros de Diversas configurações de RNAs considerando o PCA

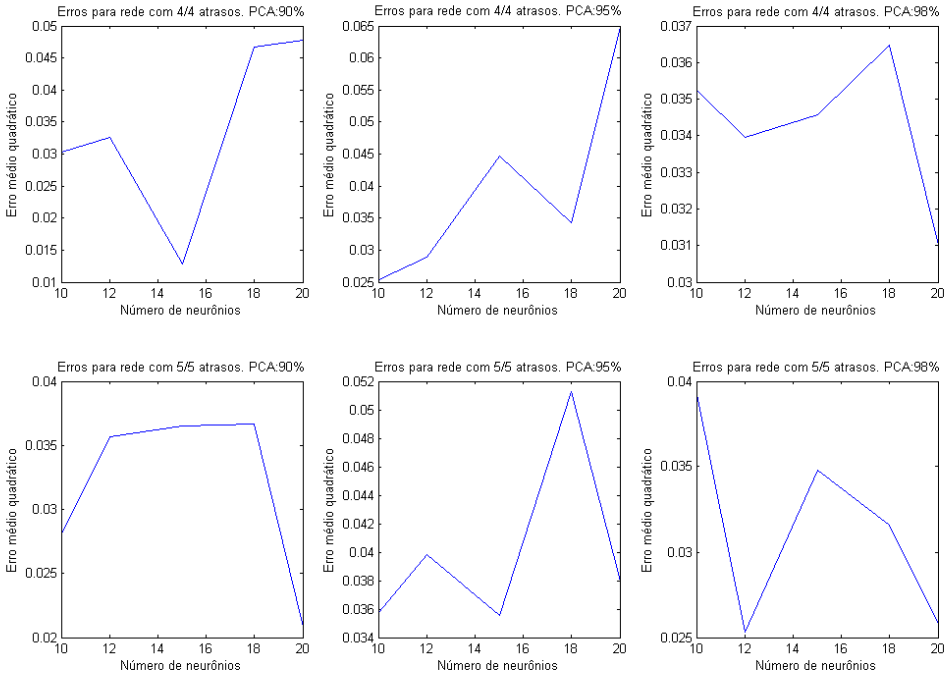


Figura 28: : Erros de Diversas configurações de RNAs considerando o PCA

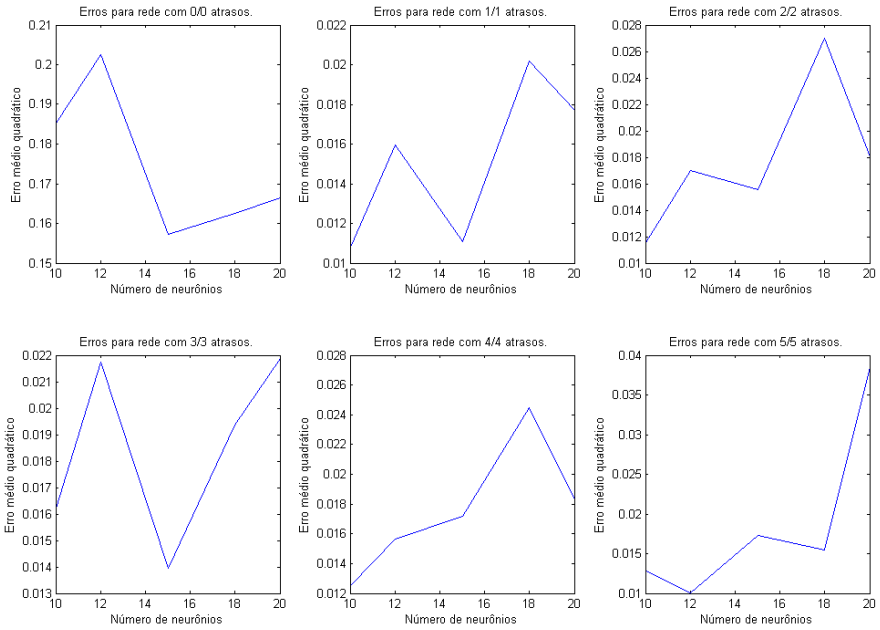


Figura 29: : Erros de Diversas configurações de RNAs sem considerar o PCA

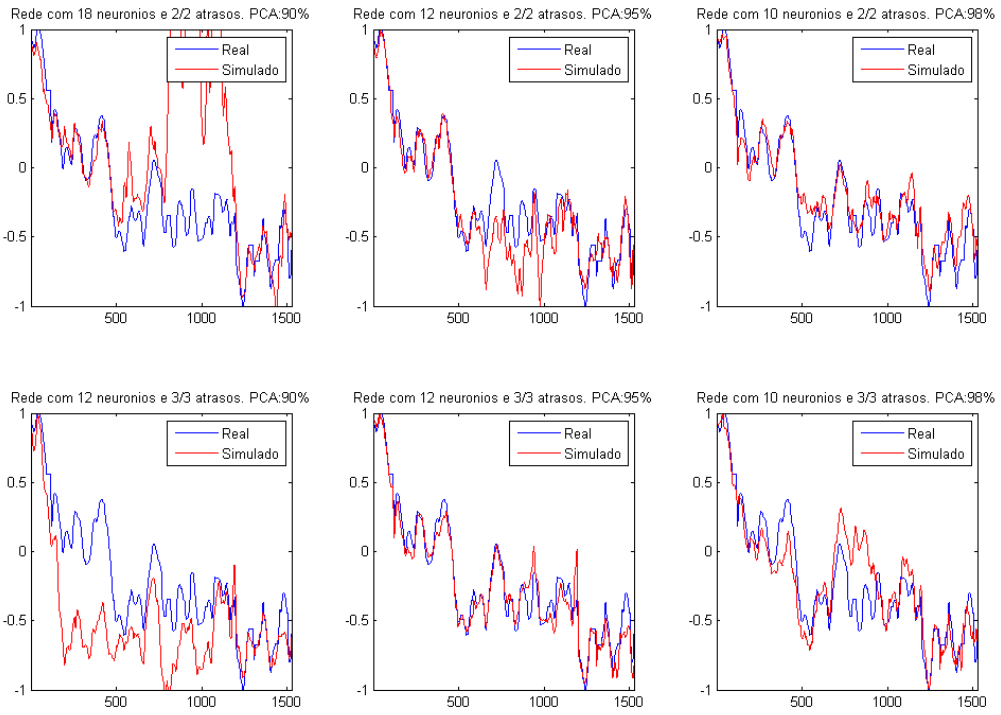


Figura 30: Saída das RNAs com diversas configurações considerando o PCA

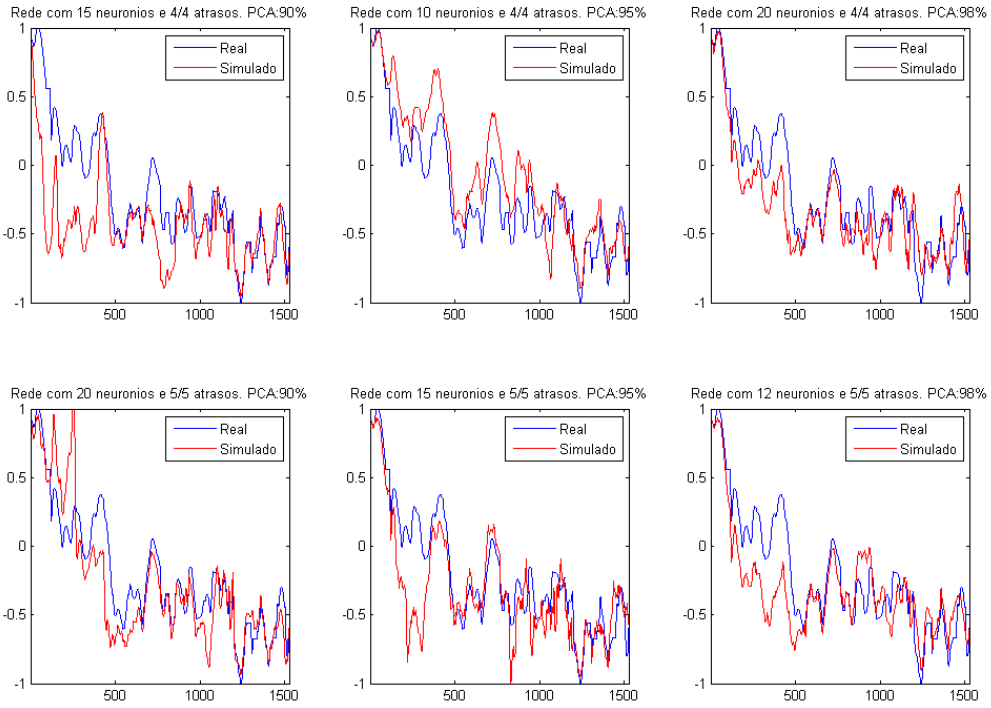


Figura 31: Saída das RNAs com diferentes configurações considerando o PCA