APLICABILIDADE FOSS: PROTOCOLO OPC, JAVA E SCILAB NA EDUCAÇÃO EM SISTEMAS DE CONTROLES INDUSTRIAS

Dionathan Barroso – dionathan.barroso@engenharia.ufjf.br
Jéssica Quintino – jessica.quintino@engenharia.ufjf.br
Alexander Silva – alexander.silva@engenharia.ufjf.br
Accacio Ferreira dos Santos Neto – accacio_ferreira@yahoo.com.br
Francisco Gomes – chico.gomes@ufjf.edu.br
Universidade Federal de Juiz de Fora, Departamento de Energia
Campus universitário, Bairro Martelos
CEP: 36036-330 – Juiz de Fora - MG

Resumo: *O objetivo deste trabalho é apresentar as ferramentas "Free and Open Source Software - FOSS" SCILAB e linguagem de programação JAVA, para aplicação na educação de Controle de Processos Industriais e Automação, viabilizadas através do protocolo de comunicação digital aberto "OLE for Process Control - OPC". O trabalho procura evidenciar suas características e possíveis posturas para utilização desse protocolo, verificando sua aplicabilidade e eficácia. De forma sucinta, o trabalho apresenta uma introdução e, em sequência, um tópico sobre o protocolo OPC, descrevendo definições e características. Como tópicos subseqüentes apresenta-se uma metodologia para educação e treinamento em controle de processos industriais, baseada em três módulos laboratoriais, e os resultados de simulações do controle de nível de um reservatório, por meio do controlador PID, direcionados aos objetivos estabelecidos. Como último tópico serão mostradas a configurações necessárias para se trabalhar com protocolo OPC na linguagem JAVA, e as conclusões sobre o desenvolvimento efetuado, bem como para os trabalhos futuros.*

Palavras-chave: *Educação em Controle, Controlador Lógico Programável - CLP, Scilab, JAVA, Protocolo OPC.*

1. INTRODUÇÃO

O desenvolvimento da tecnologia e seu constante aprimoramento permitem que, atualmente, tenha-se uma indústria diversificada que propicia a criação de novos dispositivos para as mais variadas aplicações. Neste cenário, as indústrias com enfoque em automação desenvolvem estratégias para a manipulação de dados obtidos através de seus dispositivos utilizando protocolos de comunicação entre os distintos equipamentos e softwares industriais.

Como conseqüência deste fato, constata-se o surgimento de distintos protocolos de comunicação, situação essa que culmina na dificuldade da comunicação entre dispositivos de fabricantes diferentes, como sensores, motores e computadores.

Percebe-se, ainda, o aumento da complexidade dos sistemas devido às necessárias conexões entre os elementos do mesmo sistema, situação essa onde a comunicação é fator importantíssimo.

Eis que surge, neste cenário, o protocolo de comunicação OPC, desenvolvido para a indústria de automação e suas necessidades de unificação dos protocolos, que especifica um conjunto de regras escritas e procedimentos de modo a possibilitar que múltiplos programas, ou aplicações, possam se comunicar entre si (OPC FOUNDATION, 2012).

Esta possibilidade abre perspectiva para uma ampla aplicação de controle aos mais variados tipos de problemas, fato esse que, associado à potencialidade do SCILAB e à versatilidade da linguagem de programação JAVA permitem a utilização de técnicas diversas, o que torna grande a aplicabilidade para as indústrias e potencializa soluções para a educação em engenharia de controle.

Face essas possibilidades, a primeira proposta foi a concepção e montagem de dois módulos laboratoriais para educação em engenharia, de baixo custo, porém tecnicamente consistentes; posteriormente após aquisição dos conhecimento necessários à utilização do protocolo OPC, foi verificação sua eficácia e confiabilidade para utilização em uma planta didática com componentes de chão de fábrica, e principalmente o uso de FOSS ("Free and Open Source Software").

Embora a característica mais conhecida associada com FOSS seja o seu baixo custo, praticamente nula para os usuários, outras características, não tão conhecidas, e também importantes, devem ser levadas em consideração e podem explicar seu poder de crescimento e utilização, em todo mundo, e sua aceitação nos meios acadêmicos e industriais. Elas incluem a confiabilidade, segurança e estabilidade, padrões abertos e independência dos fornecedores, redução da dependência de importação e o reforço da capacidade tecnológica local (OPEN SOURCE INITIATIVE, 2009).

O trabalho está estruturado como segue: a seção 2 apresenta o protocolo OPC, a seção 3 sua implementação com as linguagens FOSS e a seção 4 apresenta a metodologia utilizada no trabalho. A seção 5 apresenta a utilização do OPC com Java e as conclusões, na seção 6, encerram o trabalho.

2. PROTOCOLO DE COMUNICAÇÃO OPC

O OPC é um protocolo de comunicação desenvolvido diante da necessidade da indústria e automação de estabelecer parâmetros unificados para uma comunicação única entre dispositivos, reduzindo os custos de integração, bem como o ciclo total para os softwares na indústria de automação e controle, de modo geral. Esse desenvolvimento propiciou a padronização da comunicação de dados com um conjunto de especificações e regras de escrita e procedimentos (OPC FOUNDATION, 2012).

A arquitetura OPC pode ser classificada em duas hierarquias, os “clientes” OPC e os “servidores” OPC. Os clientes OPC são aplicações (softwares) que se conectam com um ou mais servidores para interagir com os itens disponibilizados, efetuando a leitura das informações de entrada e escrita nos itens de saída, quando necessário. Os Servidores (fonte de dados) são softwares disponibilizados pelo fabricante de um CLP, ou demais equipamentos de campo, sendo os computadores os responsáveis pelo reconhecimento de *drivers* e compartilhamento dos dados, fornecidos pelo servidor OPC, que reconhecem os dados, provenientes da rede de comunicação dos equipamentos da planta industrial, e os “traduzem” para o padrão OPC. (SANTOS NETO, 2012). Essa comunicação é válida somente para OPC-DA (Data Access), uma vez que existem diferentes tecnologias OPC.

3. IMPLEMENTAÇÃO DO OPC E SOFTWARES LIVRES

3.1. Motivação

Por se tratar de um protocolo de comunicação de padrão industrial aberto para transmissão de dados em tempo real, o OPC está sendo adotado pela maioria dos fabricantes de produtos, tornando-se um padrão de mercado na comunicação com o chão de fábrica. Nesse cenário torna-se cada vez mais favorável o desenvolvimento de sistemas que utilizem a comunicação de chão de fábrica, proporcionando melhor desempenho e otimização dos processos, o aprendizado dos discentes nas disciplinas de controle de processos, capacitando-os para o mercado de trabalho.

Na busca por soluções que proporcionassem o aprofundamento na aquisição de conhecimento sobre o protocolo OPC, de forma a garantir transferência deste aprendizado aos demais discentes e pesquisadores, bem como para o treinamento de profissionais na utilização, estudo de técnicas, e pesquisa na área de Controle de Processos, começou-se o desenvolvimento do presente trabalho. Foram então desenvolvidos, nesta diretriz, um módulo laboratorial que contem sensores, atuadores, e componentes utilizados no chão de fábrica. Nos tópicos a seguir estes módulos serão apresentados de forma sucinta e objetiva; posteriormente, será discutido o módulo laboratorial de maior robustez, destaque na apresentação deste artigo.

3.2. Objetivos

Deseja-se desenvolver um sistema todo por FOSS utilizando a tecnologia OPC, composto por softwares clientes para o supervisório de controle, aquisição e envio de dados, obtenção das funções de transferência, que possam ser utilizados no auxílio à pesquisa, amparo para a disciplina de controle de processos e treinamento em geral. A interface de interação homem-máquina, responsável pelo gerenciamento do processo, foi desenvolvida em SCILAB. O software utilizado para a implementação de estratégias de controle de alto nível, permitindo ao usuário definir subsistemas de controle e a aplicação de técnicas de controle avançadas, foi desenvolvido em linguagem de programação JAVA.

4. METODOLOGIA

Os dois primeiros módulos são caracterizados por sua simplicidade, fácil construção materiais de fácil aquisição. A técnica de controle selecionada para ser aplicada junto aos dados fornecidos pelo servidor OPC foi o controlador PID, por motivo de simplicidade e robustez perante variados processos.

Normalmente, as implementações utilizando controladores PID envolvem técnicas analógicas. Quando os processos envolvidos utilizam computadores digitais não mais esta-se tratando de uma aplicação analógica, pois o processo de amostragem deve, então, ser acrescido à lei de controle (ASTROM et al., 1995).

De acordo com o descrito anteriormente, a aplicação deste trabalho também se insere no escopo de um processo de obtenção de dados digital; nessa situação, o servidor OPC foi configurado para adquirir os informações do módulo de nível com intervalo de amostragem de 100 ms. Diante destas conclusões a lei de controle não mais

será a do PID analógico e sim a lei descrita abaixo obtida por meio da aproximação da regra "backward rectangular" (ASTROM et al., 1995) como:

$$u(k) = K_p(e(k) + K_i i(k) + K_d e'(k)) \quad (1)$$

onde *u* é a variável de controle, *e* o valor do erro, relativo à referência, o índice *k* a representação da discretização do processo e a integral do erro; *K*p o ganho proporcional; *K*i o ganho integral que é a relação do tempo de amostragem divido pelo tempo integral ou *Ti; Kd* o ganho derivativo que é a relação do tempo derivativo, ou Td, dividido pelo tempo de discretização.

A utilização do controlador PID impõe a necessidade de definição dos parâmetros do controle. Devido à grande simplicidade e aplicabilidade do método de Ziegler e Nichols, ele foi escolhido para a sintonia dos controladores (ZIEGLER et al., 1942). Na Tabela 1 a seguir tem-se os valores de sintonia dos controladores PID utilizando a técnica proposta por Ziegler e Nichols.

Tabela 1 – Sintonia dos controladores PID segundo Ziegler e Nichols

Controlador	K_p	T_i	T_d
P	$\tau/(k\theta)$	---	---
PI	$0,9\tau/(k\theta)$	$3,33\theta$	---
PID	$1,2\tau/(k\theta)$	2	$0,5\theta$

onde:

- τ é a constante de tempo;
- θ é o tempo morto;
- *k* é o ganho do processo, ou relação entre a resposta estacionária (*yss*) e o degrau aplicado.

A metodologia empregada consiste em utilizar os sistemas a seguir, expondo-os a determinadas condições a fim de analisar tanto o funcionamento do protocolo OPC, quantos os controles P, PI e PID para o controle de nível, temperatura, vazão e pressão.

4.1. Módulo de Nível de baixo custo

Este módulo é caracterizado por sua simplicidade, fácil construção e facilidade de aquisição dos materiais, sendo portátil e de baixo custo. (SANTOS NETO *et a*l., 2012). O módulo de nível utilizado compõe-se dos seguintes elementos: 1 sensor de nível; 2 eletrobombas; 2 reservatórios. Estes elementos estão expostos da maneira apresentada na Figura 1. Consegue-se observar o sensor de nível acoplado ao reservatório através da mangueira; pode-se ainda observar os reservatórios suspensos pelo suporte e as bombas logo abaixo. Neste sistema proposto e nos demais, entre o módulo de nível e o computador utiliza-se um dispositivo denominado CLP.

Os CLP's, ou os Controladores Lógicos Programáveis, foram desenvolvidos no final dos anos 60, com a finalidade de substituir painéis de relés em controles baseados em lógicas combinacional / sequenciais, em linhas de montagem nas indústrias de manufatura, principalmente automobilística, sendo progressivamente adotados pelas indústrias de processos (ATOS AUTOMAÇÃO INDUSTRIAL, 2006).

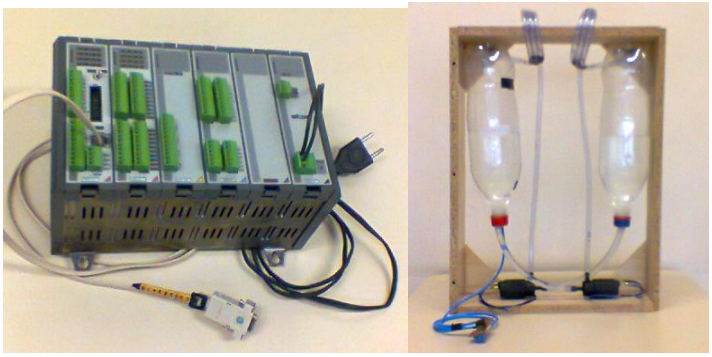

Figura 1 – CLP Atos e Módulo de Nível.

De forma simplificada pode-se descrever que o cerne do sistema é o protocolo de comunicação OPC, que permite que o CLP envie e receba valores de tensões ou correntes de origem analógica ou digital, pois o equipamento permite o uso do servidor OPC. Como o CLP da Atos utilizado não fornece uma corrente necessária para alimentar a bomba foi utilizado um circuito auxiliar, formado por um transistor, um resistor e uma fonte de alimentação, de forma que o CLP controla o transistor como uma fonte de corrente que, consequentemente, controle a bomba ou motor de esguicho.

4.2. Módulo de Nível e temperatura

O Módulo Laboratorial Tanques Duplos foi desenvolvido para análise e estudo da dinâmica de variáveis associadas ao controle de processos industriais (CAMPOS, 2007), sendo selecionadas as malhas de nível e temperatura, em configurações SISO e TITO (Figura 2), com características integradoras e atraso de transporte (WADE, 2004).

Figura 2. Módulo de Nível e Temperatura

O sistema é composto por dois reservatórios em acrílico com capacidade de 6 litros, sensores de nível e temperatura, duas eletrobombas automotivas, sensores de nível e temperatura, resistor de aquecimento, agitador para homogeneizar a temperatura da água no tanque, e um CLP Atos descrito no item anterior.

Este módulo possibilita a realização de práticas laboratoriais com uma visão diferenciada. Uma primeira observação é que o módulo possibilita a análise dinâmica e o controle, em configuração SISO - *"Single-input and Single-ouput"* - e/ou MIMO - *"Multiple-Input and Multiple-Output"*, de malhas de nível de fluidos, integrativas, e de

temperatura, com grande atraso de transporte, através da utilização de um sistema supervisório; os módulos didáticos, geralmente, não apresentam a opção MIMO.

4.3. Planta Laboratorial

O Módulo Laboratorial ou Planta Didática de Processos Contínuos (Figura 3) consiste de um sistema de tanques duplo, com aquecimento e resfriamento, que possibilita o controle das quatro variáveis de processo - vazão, nível, temperatura e pressão - em circuito único, configurado segundo topologias distintas. O sistema permite alteração das variáveis de controle e controladas, para cada malha distinta, em concepção por realimentação ("feedback") ou por antecipação ("feedforward"). Opera baseada na padronização de sinais de 4 a 20 mA, adicionalmente ao protocolo digital Profibus PA. (GOMES, 2008). Sua característica diferencial é a utilização de sensores e atuadores que replicam uma ambiência industrial, com seus problemas e não linearidades, mas também permitindo seu ajuste e calibração.

Buscando reproduzir o uso da arquitetura do protocolo OPC com o sistema da planta didática, foram utilizados dois clientes, configurados no cliente SCILAB, um sistema supervisório que possibilita a configuração e operação do módulo tanto em malha aberta como fechada para a identificação dos modelos e sintonia de controladores. Ao iniciar o ambiente, é possível escolher entre identificar as malhas ou controlar as malhas; ao selecionar a identificação (Figura 4) é possível fazer por meio do Teste da Resposta ao Degrau ou pelo Método dos Mínimos Quadrados recursivo.

Figura 3. Planta Didática de Controle real.

Após a identificação o sistema supervisório gera gráfico e os valores pertinentes de cada método e junto do Método dos Mínimos Quadrados é possível obter a qualificação do modelo estimado por meio de técnicas de validação de modelos como o Coeficiente de Correlação Múltipla da "Equação 2" (COELHO & COELHO, 2004).

$$R^2 = 1 - \frac{\sum_{k=1}^{n}[y(k)-\hat{y}(k)]^2}{\sum_{k=1}^{n}[y(k)-\bar{y}]^2} \quad (2)$$

Onde y(k) é a saída real, $\hat{y}(k)$ é a saída estimada e $\bar{y}$ é a medida das N amostras da experiência. Para valores de R^2 igual a 1 significa que houve uma exata adequação do modelo para os dados medidos do processo.

Já para a sintonia das malhas, é possível utilizar o controle P, controle PI e controle PID. O sistema possibilita a alteração dos dados da sintonia em tempo real, como a referência, os ganhos dos controladores e perturbações. Adicionalmente às características específicas de cada malha, o projeto do sistema prevê complexidades adicionais, comuns em ambientes industriais.

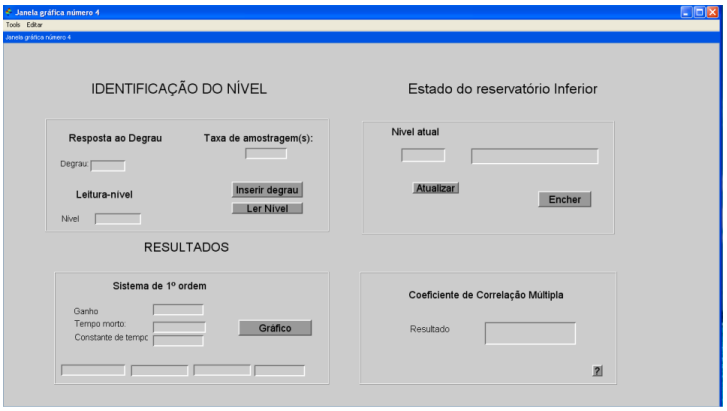

Figura 4 – Tela de identificação nível

Dentre estas vale citar: Atrasos de transporte variáveis; Modelagem não linear; Controle não linear; Alteração de variáveis de controle para uma mesma malha através da potência dissipada na resistência de aquecimento do tanque inferior, através da vazão do fluido, mantendo-se constante a potência da resistência de aquecimento, através da taxa de resfriamento do radiador, para vazão e potência constantes, e também para qualquer das configurações citadas, é possível alteração nas variáveis durante o processo.

O Sistema Supervisório desenvolvido em linguagem de programação JAVA, (Figura 5) disponibiliza interface gráfica e publicação das telas gráficas dinâmicas, em tempo real. O supervisório conta ainda com Tutoriais sobre protocolo OPC, características de configuração, metodologias e técnicas de controle. Para se fazer o uso desta linguagem destacam-se no próximo tópico as configurações necessárias.

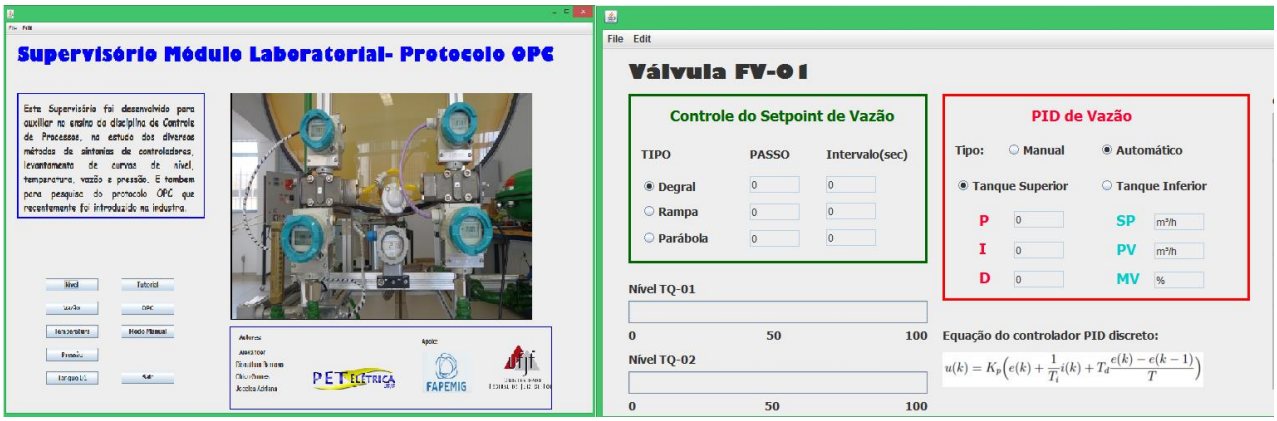

Figura 5 – Supervisório JAVA "Cliente OPC" e tela para Controlador para Vazão

Obtido os parâmetros com o supervisório em SCILAB, o supervisório desenvolvido em linguagem de JAVA "cliente OPC" pode simular características especificas de cada malha, como no exemplo figura 5.

Este módulo encontra-se em aprimoramento e seu supervisório JAVA, ainda em desenvolvimento, permitirá aos alunos da Universidade Federal de Juiz de Fora (UFJF), práticas laboratoriais de forma a fazê-los irem ao encontro das tecnologias e protocolos de comunicação recentemente difundidos na indústria.

5. BIBLIOTECAS OPC PARA JAVA

Para estabelecer a conexão entre o software servidor e o software cliente desenvolvido em linguagem de programação JAVA faz-se necessário o uso de uma biblioteca acoplada a essa linguagem. Dentre as disponíveis no cenário atual, podemos destacar três compatíveis a tecnologia OPC Data Access, são elas: JEasyOPC, JOpcClient, JOPC-Bridge.

Dentre estas três, a primeira foi a escolhida por ser uma biblioteca de código aberto, requisito deste trabalho; as demais são bibliotecas proprietárias, comercializadas; destaca-se, portanto, a utilização da JEasyOPC, suas características, vantagens e desvantagens, configurações e complementações necessárias para seu uso.

5.1. JEasyOPC

A biblioteca OPC foi criada para facilitar a conexão entre softwares em Java e um servidor OPC. Baseada e abstraída de outra biblioteca do gênero, JEasyOPC, essa biblioteca visa tornar a conexão entre servidores OPC e um software em JAVA algo prático, haja vista que os métodos atuais são bem complexos e demandam instalação e configurações complicadas. Esta biblioteca pode ser facilmente encontrada na internet para download, estando disponível a qualquer pessoa que dela necessite para sua aplicação em JAVA, utilizando o protocolo OPC. Porém, para seu uso, é necessário fazer configurações e implementações adicionais no código, para que se torne funcional.

Esse processo de implementação da biblioteca foi um dos pontos de alto desafio, pois, ao deparar-se com um código incompleto - e que em primeiro momento não se obtinha sucesso na sua execução junto ao código do software cliente -, ou seja, em termos práticos, "o programa não compilava" devido a problemas de referência e importações de classes da biblioteca. A segunda frustação foi ao se tentar obter respostas e soluções para contornar e solucionar o problema, pesquisando na web em fóruns, artigos científicos, e com pesquisadores de outras universidades, pois essas fontes não retornaram com sucesso as respostas para as dúvidas e passos que são de extrema necessidade para solucionar o problema; nas diversas publicações o máximo que se pode encontrar são passagens diretas de um ponto para outro na configuração, o que impossibilita o leitor prosseguir na implementação. Nos fóruns a resposta típica é "o problema foi resolvido", mas sem a explicação de como foi feito.

Percebeu-se a necessidade de buscar um conhecimento mais aprofundado na linguagem JAVA para resolver esta situação, e após algumas semanas, a tão esperada comunicação do sistema estava realizada. Por tais motivos, no tópico a seguir são descritas as configurações e complementações necessárias, de modo a propiciar ao leitor um passo-a-passo de como fazer o uso da biblioteca.

5.2. Configurando e Implementando a biblioteca JEasyOPC.

Por ser inviável a descrição de todo o código necessário na configuração da biblioteca, devido ao limite de texto no artigo, e para a maior comodidade do usuário, foi disponibilizado no blog Energia Inteligente (http://energiainteligenteufjf.com/) do Programa de educação Tutorial da Engenharia Elétrica da UFJF, o pacote com toda documentação para download; abaixo são descritas as melhorias feitas e como o usuário deve fazer seu uso. Após fazer o download tais procedimentos técnicos são necessários para a instalação da biblioteca:

OPC.jar: Pacote que contém a biblioteca, onde estão todas as classes necessárias ao funcionamento da biblioteca. Deve ser importado para o projeto;

JCustomOpc.dll: DLL escrita em Delphi, responsável pela comunicação entre a biblioteca OPC e o servidor OPC. Deve ser copiada para a pasta do projeto;

Data.zip: Pacote ZIP contendo os arquivos de propriedades da DLL JCustomOpc e da biblioteca JEasyOPC. Deve ser copiado para a mesma pasta onde estiver o arquivo JCustomOpc.dll e deve ser importado para o projeto;

Para fazer as importações no NetBeans isso pode ser feito clicando com o botão direito em Libraries->Add Jar/Folder e selecionando os dois arquivos. Quando o projeto for compilado, será criada uma pasta chamada lib no mesmo local onde está o arquivo .jar do projeto principal. Por fim, copie para esta pasta os Data.zip e JCustomOpc.dll.

5.3. Métodos

OPC(Construtor): responsável por definir as propriedades de um objeto da classe OPC e não é invocado diretamente, sendo chamado na criação do objeto. Recebe como parâmetros três strings, o host do servidor, o nome do servidor e um nome para ser identificado no servidor;

Conectar: responsável por tentar fazer conexão com o servidor. Ele não recebe nenhum parâmetro e retorna um tipo booleano que indica o sucesso ou falha da conexão. Deve ser invocado depois que todos os grupos, itens e funções automáticas estiverem definidas, pois, para fazer a conexão, ele envia ao servidor as informações sobre todos os grupos e itens requeridos, e em caso de sucesso, cria um objeto da classe privada “Alteração”, estendido classe Thread, que é responsável por receber dados do servidor de forma assíncrona e utilizar esses dados em funções automáticas definidas;

Desconectar: responsável por tentar desconectar do servidor. Não recebe nenhum parâmetro e retorna um booleano indicando o sucesso ou falha do processo de desconexão. Ao desconectar todos os objetos que controlam as funções automáticas são pausados, de forma que podem ser executados rapidamente no caso de uma reconexão;

setConexao: responsável por alterar as informações do servidor antes de realizar a conexão. Recebe como parâmetros três strings, o host do servidor, o nome do servidor e um nome para ser identificado no servidor;

setTempo: esse método é responsável por alterar o tempo que o software irá esperar entre uma verificação de servidor e outra, onde são buscadas as diferenças ocorridas no servidor nesse intervalo de tempo. Recebe como parâmetro um inteiro indicando o tempo em milisegundos;

Conectado: retorna um booleano indicando o estado atual de conexão com o servidor;

AdicionarGrupo: esse método é responsável por registrar um grupo no software e deve ser chamado antes do método Conectar e do método AdicionarItem. Recebe como parâmetros as informações de um grupo no servidor OPC, que são uma string informando o nome do grupo, um booleano indicando se o grupo está ativo, um inteiro indicando a taxa de atualização do grupo e um flot indicando a porcentagem de banda morta do grupo. O método retorna um inteiro que indica o código único de identificação do grupo para o sistema;

AdicionarItem: método responsável por registrar um item no software e deve ser chamado antes do método Conectar. Recebe como parâmetros as informações de um item registrado no servidor OPC, que são um inteiro contendo o código único de identificação do grupo, uma string informando o nome do item, um booleano indicando se o item está ativo ou não e uma string contendo o Acess Path do item. O método retorna um inteiro que indica o código único de identificação do item para o sistema ou o valor -1, indicando que houve falha na adição do item;

ler: método responsável por fazer uma leitura manual de um item no servidor OPC e deve ser chamado depois do método Conectar. Recebe como parâmetros dois inteiros contendo o número único de identificação de um grupo e de um item respectivamente. Retorna uma instância da classe Variant contendo o valor atual do um item no servidor e também pode retornar o valor null em caso de falha na leitura;

escrever: responsável por fazer uma alteração do valor de um item no servidor OPC e deve ser chamado depois do método Conectar. Recebe como parâmetros dois inteiros contendo o número único de identificação de um grupo e de um item respectivamente e também uma instância da classe Variant, essa por sua vez contendo o novo valor a ser atribuído ao item no servidor. Retorna um booleano indicando a falha ou sucesso do processo de alteração do valor do item;

addFuncao: registra uma função à ser executada sempre que houver alteração em um item no servidor e deve ser chamado antes do método Conectar. Recebe como parâmetros dois inteiros contendo o código de identificação único de um grupo e de um item respectivamente e também uma instância da classe pública Funcao, classe essa que está definida dentro da classe OPC. Retorna um inteiro contendo o número de identificação único da função para o sistema;

rmFuncao: responsável por remover o registro de uma função previamente registrada pelo método addFuncao. Recebe como parâmetro um inteiro contendo o código único de identificação da função;

Funcao: classe abstrata que define uma ação a ser executada quando o valor de um item sofre alguma alteração no servidor OPC. A classe contem um método abstrato chamado Acao, que recebe como parâmetro uma instância da classe Variant que conterá o novo valor do item no servidor.

5.4. Exemplo de código

```
OPC opc;
{
    opc = new OPC("127.0.0.1", "Atos.OPCConnect.1", "OPC1");
        int grupo = opc.AdicionarGrupo("Leitura", true, 500, 0);
        int item = opc.AdicionarItem(grupo, "Bucket Brigade.Int1", true, "");
        int funcao = opc.addFuncao(item, grupo, new OPC.Funcao);
    {
                @Override
```

```
                public void Acao(Variant v) {
                        label.setText(v.toString());
                }
}
    opc.Conectar();
}
{
    opc.Desconectar();
}
```

6. CONCLUSÃO

Os resultados obtidos até o presente com o projeto deverão conduzir ao desenvolvimento completo de sistema de controle baseado em redes que permita uma boa adaptação a qualquer meio industrial, onde exista algum servidor OPC conectado. Trata-se de uma realidade que, iniciada em 1996 - ano de lançamento do OPC - vem crescendo vertiginosamente no ambiente industrial, mas, infelizmente, ainda é abordada de forma totalmente incipiente na educação em controle de processos, criando um possível descompasso entre a formação dos engenheiros e a realidade que encontrarão no mercado de trabalho, após sua formatura. O principal benefício de seu aprendizado e utilização consiste na redução dos custos e tempo de desenvolvimento de interfaces e drivers de comunicação, com consequente redução de custo de integração de sistemas. Esta potencialidade facilita sobremaneira que sejam escolhidos, dentro de diversas opções, os tipos de controle que melhor se adaptarão ao processo.

Diante dos resultados apresentados observa-se a aplicabilidade e a utilidade do sistema que mostraram-se satisfatórios, o que demonstra que a metodologia empregada desde os elementos físicos até a escolha do uso do protocolo de comunicação é válida.

No mais, pode-se dizer que o trabalho vem obtendo resultados que propiciam compreender o vasto campo de aplicação que o protocolo de comunicação OPC possui. Os módulos desenvolvidos, de baixo custo, fácil operação e tecnicamente consistentes e estruturados possibilitam utilização laboratorial mais intensa, com análise de situações distintas e diversificadas.

Um destaque importante é quanto aos usos de softwares livres - no caso do Scilab e da linguagem JAVA - que se adéquam perfeitamente ao protocolo OPC, por meio de toolbox e bibliotecas respectivas, sendo motivadores para continuação deste trabalho, em especial o JAVA, que possuiu um potencial muito grande para criação de ambientes interativos e interfaces gráficas.

A soma dos três módulos estudados resulta em um acúmulo de conhecimento gradativo que tem o potencial de propiciar, a qualquer profissional ou investigador, o aprendizado nas mais diversas técnicas de controle, sobre o protocolo OPC, dentre outros. Sua utilização nas atividades laboratoriais reforça o conhecimento técnicos dos alunos em áreas diversas, abrangendo não só os aspectos mais conceituais da teoria de controle, mas os aspectos práticos de conexão e configuração de sistemas, presentes no meio industrial e cujo conhecimento torna-se, cada vez mais, uma necessidade para os engenheiros.

7. AGRADECIMENTOS

Agradecemos ao Programa de Educação Tutorial (PET-MEC) e FAPEMIG.

8. REFERÊNCIAS BIBLIOGRÁFICAS

ATOS AUTOMAÇÃO INDUSTRIAL. Curso básico de Controladores Programáveis, Ref.5-0026.110, Manual - Fevereiro/2006. Disponível em: http://www.atos.com.br. [Acessado em: abril de 2013]

ASTROM, K.J. & Hagglund, T. Hagglund. 1995. *PID Controllers, Theory, Design and Tuning* (2nd)

CAMPOS, M. C. M. M.; Teixeira, H. C. G. Controle Típico de Equipamentos e Processos Industriais. ed. Edgard Blücher: São Paulo, 2007.

COELHO, A. A. R.; COELHO, L. S. Identificação de Sistemas Dinâmicos Lineares. ed.UFSC, 2004.

GOMES, F. J. & PINTO, D. P. Laboratórios Integrados para Controle de Processos e Análise da Eficiência Energética de Sistemas Industriais, COBENGE, 2008.

OPC FOUNDATION, What is OPC?, 2012. Disponível em:http://www.opcfoundation.org/Default.aspx/01_about/01_whatis.asp?MID=AboutOPC. [Acessado em: 21 de abril de 2013]

OPEN SOURCE INITIATIVE. Disponível em: <http://opensource.org/history> Acesso em: 12 dez. 2012

SANTOS NETO, A. F.; BARROSO, D. S. Tornando a Educação em Controle de Processos mais realista: a utilização do protocolo OPC, COBENGE, 2012.

ZIEGLER, J. G. & Nichols, N. B. 1942. Optimal settings for automatic controllers. Transactions of the ASME.

WADE, H. L. Basic and Advanced Regulatory Control: System Design and Application. ed. Instrumentation System, 2004.

FOSS APPLICABILITY: PROTOCOL OPC, JAVA AND SCILAB IN EDUCATION SYSTEMS CONTROLS INDUSTRIAS

Abstract: *The objective of this article is focused at presenting tools of Free Open Source Software (FOSS), SCILAB and JAVA programming language, for the study of many Process Control and Automation techniques through the communication protocol OLE for Process Control (OPC); highlighting their features and an assumed form of use of this protocol, checking its applicability and effectiveness. Briefly, the article will present an introduction and, in sequence, a topic about the OPC protocol, describing some definitions and characteristics. As subsequent topics we present a methodology for training based on three laboratory modules, the last is still under development, the results of simulations of the control level of a reservoir, through the PID controller, adjusted toward our goals. As a last topic will be shown in necessary settings to work with OPC protocol using JAVA, pertinent findings will be made over the entire development as well as future work.*

Key-words: *Education in Control, Programmable Logic Controller - PLC, Scilab, JAVA, OPC protocol.*