UNIVERSIDADE FEDERAL DE OURO PRETO
ESCOLA DE MINAS
CECAU - COLEGIADO DE ENGENHARIA DE
CONTROLE E AUTOMAÇÃO

EMANUEL BALDUINO DA CRUZ

ACESSO REMOTO DE UM SERVIDOR OPC

MONOGRAFIA DE GRADUAÇÃO EM
ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Ouro Preto, 2017

EMANUEL BALDUINO DA CRUZ

ACESSO REMOTO DE UM SERVIDOR OPC

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como parte dos requisitos para a obtenção do Grau de Engenheiro de Controle e Automação.

Orientadora: Prof Regiane de Sousa e Silva Ramalho

Ouro Preto
Escola de Minas - UFOP
Março/2017

Ficha SISBIN:

C957a Cruz, Emanuel B. da.
Acesso remoto de um servidor OPC [manuscrito] / Emanuel B. da Cruz. - 2017.

36f.: il.: color; linhas de códigos.

Orientador: Prof. Dr. Regiane de S. e Silva Ramalho.

Monografia (Graduação). Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia de Controle e Automação e Técnicas Fundamentais.

1. Cliente/servidor (Computadores) - OPC Server. 2. JavaScript (Ling. de programação de computador) - Utgard. 3. Controle remoto. I. Ramalho, Regiane de S. e Silva. II. Universidade Federal de Ouro Preto. III. Titulo.

CDU: 681.5

Catalogação: ficha@sisbin.ufop.br

Folha de assinaturas:

Monografia defendida e aprovada, em 22 de fevererio de 2017, pela comissão avaliadora constituída pelos professores:

Profa. M. Sc. Regiane de Sousa e Silva Ramalho - Orientadora

Prof. M. Sc. José Alberto Naves Cocota Júnior – Professor Convidado

Prof. Dr. Alan Kardek Rêgo Segundo – Professor Convidado

Dedico este trabalho aos amigos e familiares que me apoiaram incondicionalmente durante todo o meu trajeto.

AGRADECIMENTOS

Gostaria de estender agradecimentos a todos os envolvidos, diretamente ou indiretamente, no meu processo de graduação. Todos os técnicos e servidores que possibilitaram a universidade a prestar as melhores condições para seus alunos. Principalmente ao corpo docente por sempre acreditarem no meu crescimento e por me apresentarem um mundo novo de aprendizado e conhecimento.

Também agradeço a Fundação Gorceix e o laboratório LAP por darem possibilidades aos alunos de explorar e se desenvolver em outras áreas de interesse.

Agradeço ainda a empresa IHM que me possibilitou um convívio próximo com a área que escolhi atuar. A vivencia do dia a dia e o trabalho em equipe, para que o serviço seja feito e entregue no prazo estipulado, serviu como um grande aprendizado.

RESUMO

Este trabalho aborda uma aplicação utilizando controle à distância (acesso remoto). O trabalho consiste em criar uma ferramenta para acesso via internet de um servidor OPC que pode conter informações de diversos equipamentos de uma planta. Para isso, utilizou-se da plataforma Java para criar a interface web. A integração com o servidor OPC é feita por uma biblioteca Java (UTGARD) que lê e escreve informações em variáveis.

Palavras-chave: *Java, Servidor OPC, Acesso Remoto, Utgard*

ABSTRACT

This thesis concerns with an application which utilizes remote control (remote access). This work aims at creating a tool as to provide remote access via internet to an OPC server holding information of several equipment in a given plant. For this end, a Java platform was made use of to create a web interface. The OPC server to client integration is made by means of a Java library (UTGARD), which can read and write information into the server's variables.

Keywords: *Utgard, remote access, OPC Server, Java*

SUMÁRIO

LISTA DE FIGURAS

LISTA DE SIGLAS E ABREVIATURAS

OPC: OLE for Process Control

OLE: Object Linking and Embedding

COM: Component Object Model

DCOM: Distributed Component Object Model

EAD: Ensino à distância

WWW: World Wide Web

JSF: Java Server Faces

JNI: Java Native Interface

JVM: Java Virtual Machine

DLL: Dynamic-Link Library

HTML: Hyper Text Markup Language

HTTP: Hypertext Transfer Protocol

DEP: Data Execution Prevention

TCP: Transmission Control Protocol

CLSID: Class Identifier

GUID: Globally Unique Identifier

CATID: Category ID

1 INTRODUÇÃO

Com o passar do tempo, a tecnologia tem evoluído e novas técnicas são desenvolvidas em todas as áreas, em sua maioria visando uma melhoria na vida do ser humano. Dentre muitos benefícios buscados, a melhoria na acessibilidade de informações é uma delas. A possibilidade de trocar dados a distância sempre foi uma motivação. Conseguir acessar, agir e visualizar situações sem precisar estar fisicamente onde a fonte se encontra, seria de grande utilidade em muitas situações, sendo elas acadêmicas, industriais ou até mesmo no ambiente doméstico.

Nesse meio de compartilhamento de informações, a internet surge como a principal e mais eficiente opção. Junto com a evolução nas ferramentas de controle, aquisição e distribuição de dados via rede de computadores, tem-se desenvolvido melhores condições para o acompanhamento e o controle de experimentos remotos (MARCHEZAN, 2006). Com essa perspectiva, em grandes empresas e universidades, tem-se desenvolvido interesse nos weblabs/laboratórios controlados remotamente.

O ensino de Engenharia na área de Automação Industrial, especialmente Automação da Manufatura e Robótica, deve combinar necessariamente teoria e prática. Estudantes devem obter conhecimento e desenvolver habilidades que requerem o uso intensivo de laboratórios que consistem de equipamentos de grande porte como: máquinas-ferramenta com comando numérico, robôs industriais, robôs móveis, redes e protocolos de comunicação, células flexíveis de manufatura, entre outros. Sistemas são caros e nem sempre estão disponíveis nos laboratórios das universidades e escolas técnicas brasileiras. Experimentos interativos em plantas/sistemas reais motivam os estudantes e também desenvolvem uma abordagem de resolução de problemas reais de Engenharia. Desta forma, o ensino na área de Automação e de muitas outras áreas das Engenharias requer, obrigatoriamente, atividades de laboratório que deverão ser executadas/ministradas de forma presencial ou sob a óptica do Ensino a Distância (EAD) utilizando de mecanismos e metodologias que viabilizem a sua execução de forma remota, e para isto é fundamental a disponibilidade de laboratórios que permitam a realização de experimentos remotos.(ÁLVARES; FERREIRA, 2003).

Num ambiente industrial é notável um grande potencial de crescimento e diversificação de aplicações remotas. Entre as vantagens do uso de acesso remoto via internet em aplicações industriais merecem destaque: a possibilidade das empresas com unidades distribuídas conseguirem acessar, compartilhar, analisar, e processar informações de chão de

fábrica em tempo real e com maior agilidade, e a possibilidade de terceirização de serviços técnicos ou administrativos especializados com maior grau de interação entre os parceiros, evitando assim a necessidade de especialistas em seu quadro de funcionários.(JÚNIOR, 2009)

A estrutura básica de um Laboratório Remoto é formada por um conjunto de instrumentos que de alguma forma interage com um computador conectado à web ou a uma rede interna. O computador tem como função principal controlar esses equipamentos e acessá-lo, remotamente, a partir de outros computadores ligados à Internet ou interligados a uma rede interna. O usuário remoto acessa e controla o computador do laboratório e, a partir daí, aciona equipamentos, faz observações, testa condições e coleta dados. Pode-se adicionar artifícios, como câmeras de vídeo, ao ambiente do laboratório de modo que o usuário consiga ter uma visão do que se passa no laboratório em tempo real. (MARCHEZAN, 2006)

Diante disso, este trabalho focou na interação entre um servidor OPC, que pode conter informações dos equipamentos de alguma planta existente em um laboratório, com a internet, estudando e aplicando uma melhor forma de transferir dados em tempo real a distância.

1.1 Objetivo Geral

Estabelecer uma comunicação de dados entre um servidor OPC e uma página de internet para consulta e controle de uma variável a distância em tempo real.

1.1.1 Objetivos específicos

- Definir um servidor OPC de dados;
- Criar uma interface para uma página web que mostre e envie dados a um servidor OPC em tempo real;
- Transferir informações entre o servidor OPC e o servidor web utilizando a plataforma Java EE. Será utilizada a biblioteca Utgard, que é uma biblioteca *opensource* utilizada pelo Java para transações OPC.

1.2 Justificativa

Com a grande variedade de equipamentos, surgiu a necessidade de uma padronização entre troca de dados. Nessas condições, foi criado o padrão OPC. Com a evolução das

tecnologias, o controle e acesso de dados a distância desses mesmos equipamentos, se tornou algo possível e desejado. Logo, o trabalho apresentará uma aplicação em que será mostrado a integração da funcionalidade do servidor OPC com a internet para fazer acessos remotos de equipamentos.

1.3 Estrutura do Trabalho

Este trabalho foi divido em 4 partes principais: o Capítulo 1 contém a introdução, onde é apresentado o tema e as motivações da sua escolha. O Capítulo 2 apresenta a revisão bibliográfica, ou seja, uma revisão teórica dos temas abordados e ferramentas utilizadas no trabalho. O capítulo 3 contém uma explicação do que foi feito na parte prática. É detalhado a criação da interface web, a configuração do servidor OPC e a integração entre eles. Por fim, o capítulo 4 apresenta conclusões e análises do que foi feito. Também são apresentados sugestões para trabalhos futuros.

2 REVISÃO TEÓRICA

2.1 Servidor OPC

Segundo (LEITAO, 2006), nos últimos anos o uso de *software* de controle de processos tem aumentado consideravelmente. Numa planta, em que é necessário a utilização de diversos equipamentos, muitos deles de fabricantes diferentes, precisa-se de *softwares* que se comuniquem entre si. Porém, devido a diversidade de fabricantes, cada um com seus próprios drivers e protocolos de comunicação, a integração e conectividade entre dispositivos é um procedimento complicado. Nesse ambiente, surgiu a necessidade de desenvolver um padrão de comunicação. Foi com esse pensamento que algumas empresas, junto com membros da Microsoft, se reuniram com o objetivo de desenvolver esse padrão que ficou conhecido como OPC (OLE for Process Control). Exemplificado na figura 1, o OPC surgiu para resolver os problemas de interconexão de diferentes dispositivos e foi logo aceito pela indústria.

Figura 1 – A falta de conectividade dos sistemas tradicionais
Fonte: (PUDA, 2008)

Em 1994 um grupo de fornecedores, que representam um amplo campo de disciplinas do segmento industrial, formaram o que hoje é conhecido como OPC Foundation. A OPC Foundation surgiu com o objetivo de desenvolver uma especificação única cliente/servidor que permitiria que qualquer fornecedor desenvolvesse *softwares* e aplicativos que pudessem trocar dados de forma rápida. Foi criado a primeira especificação, chamada de especificação de acesso a dados 1.0a, que foi lançado no início de 1996. Com essa especificação, os fornecedores foram capazes de desenvolver rapidamente um *software* cliente/servidor

(TECHNOLOGIES, 2016).

Com as evoluções no campo da tecnologia voltados a automação, o uso de computadores como ferramenta tem sido de muita importância. A combinação de computadores com o uso de redes de integração de dados propiciou o desenvolvimento da tecnologia OPC, que através de um serviço do sistema operacional Windows, viabilizou a comunicação de sistemas cujos drives de comunicação são disponibilizados pelo fabricante ou desenvolvidos por integradores. O resultado é a possibilidade da troca de dados em um mesmo padrão de forma satisfatória para que o usuário integre seus sistemas em uma única plataforma (RIBEIRO; LEANDRO; ANJOS, 2008).

Para muitos fornecedores, o esforço de desenvolver inúmeros drivers de comunicação compensava os esforços gastos no desenvolvimento da aplicação cliente em si. Com a adoção da tecnologia OPC, os fornecedores poderiam concentrar exclusivamente no desenvolvimento do aplicativo cliente. Vê-se na figura 2 que se as especificações forem seguidas corretamente, um fornecedor cliente sabe que qualquer servidor OPC que existe, para um dispositivo industrial, pode fornecer a conectividade necessária para o acesso a dados (TECHNOLOGIES, 2016).

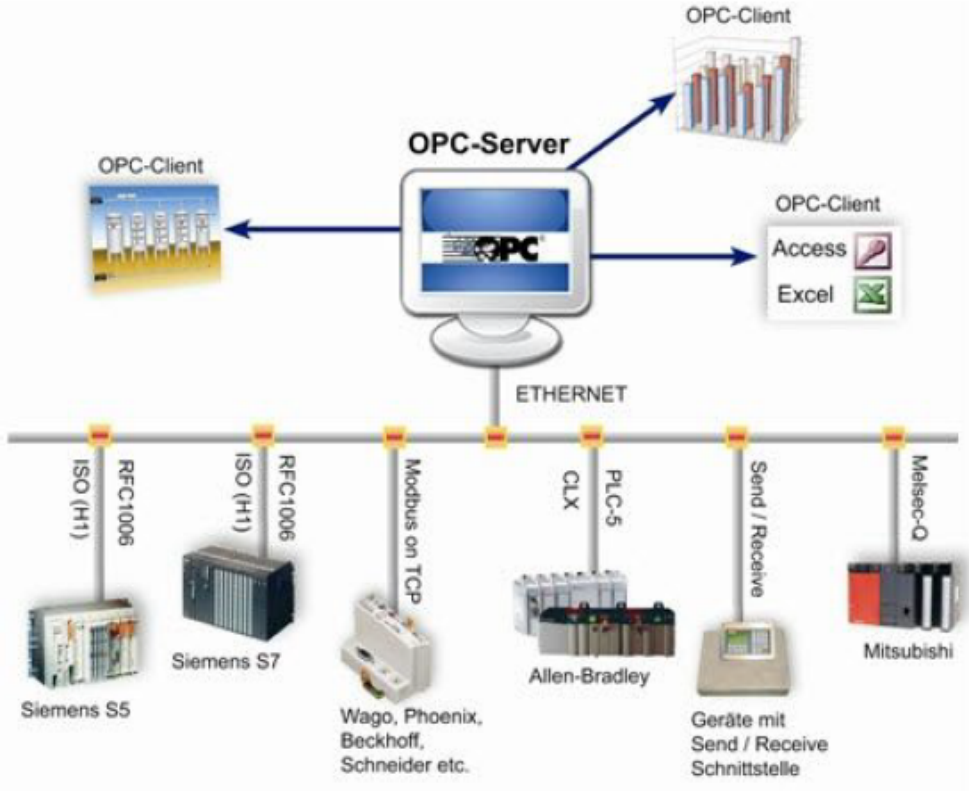

Figura 2 – Relação entre cliente e servidor OPC
Fonte: (BROCKHAUS, 2016)

Os drives instalados nos computadores são os responsáveis por reconhecer as inter-

comunicações que ocorrem com os dados dos equipamentos. Os equipamentos de um sistema de automação possuem conexões físicas de rede e um protocolo que basicamente são regras e convenções utilizadas na comunicação da camada n de uma máquina com a camada n de outra. O driver de comunicação é um "programa" especializado que implemente o protocolo e opere a conexão física entre o equipamento e a aplicação. Um fornecedor desenvolve um driver voltado para um produto específico, nele é descrito os componentes de *software* do mesmo. O objetivo do fabricante é permitir que seu *software* comunique com esse determinado produto por meio de uma conexão de rede (RIBEIRO; LEANDRO; ANJOS, 2008). A figura 3 ilustra a melhora no envio de dados dos drivers com o servidor OPC.

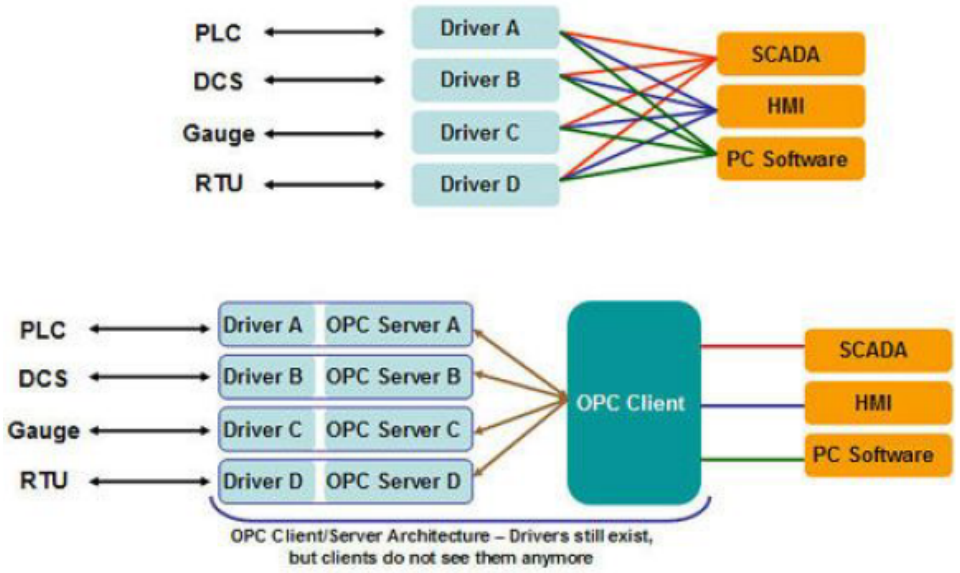

Figura 3 – Diferença entre acesso aos drivers com e sem o servidor OPC
Fonte: (BROCKHAUS, 2016)

Os computadores que possuem os drivers dos equipamentos de campo reconhecem os dados provenientes da rede de comunicação dos equipamentos da planta industrial e os "traduzem" para o padrão OPC. Os computadores responsáveis por esse reconhecimento de drives e compartilhamento dos dados por meio dos serviços do sistema operacional constituem os "servidores OPC". Os servidores são *softwares* que fornecem dados em OPC, o computador é o *hardware* convergente e disponibilizador de dados. As aplicações que "consomem" estes dados são os clientes OPCs e podem estar em quaisquer computadores conectados à rede do servidor OPC (RIBEIRO; LEANDRO; ANJOS, 2008).

De forma geral o funcionamento dos servidores OPC consiste em fazer a aquisição dos dados e deixar essa informação acessível para os clientes OPC que nele se conectarem. Esta disponibilização se dá através do que a especificação OPC chama de Itens. Um item é um dado que pode ser uma temperatura, pressão, estado de uma válvula, etc, disponível

para ser lido ou escrito por um cliente OPC. Um cliente OPC é uma aplicação a qual irá se conectar com um servidor para interagir com os itens disponibilizados. Para uma melhor organização e melhoria na transmissão de informações entre servidores/clientes, os clientes OPC devem criar grupos de itens com características semelhantes (LEITAO, 2006).

2.1.1 COM/DCOM

O avanço das ferramentas e programas nos computadores dos anos 90, fez surgir a necessidade de troca de dados entre aplicações. Em 1995 foi criado uma tecnologia de intercomunicação chamada COM (Component Object Model) que se tornou base para o desenvolvimento de aplicações orientadas ao objeto. O COM é uma tecnologia criada pela Microsoft para definições de componentes que possam ser reutilizados por outras aplicações. É por esta implementação que se tornou possível a troca de informações em tempo de execução por duas aplicações diferentes, como por exemplo, inserir tabelas do Microsoft Excel no Word, dentre outras interconexões (LEITAO, 2006).

Com o tempo a necessidade cresceu. Lançado junto com a plataforma Windows NT, o DCOM (Distributed Component Object Model) veio para suprir a necessidade da comunicação entre aplicações localizadas em computadores diferentes. O DCOM é, basicamente, um conjunto de definições para permitir a implementação de aplicações distribuídas em uma arquitetura cliente-servidor. O DCOM permite que objetos sejam instanciados de forma distribuída e seus serviços e métodos sejam acessíveis por diferentes programas (FONSECA, 2002).

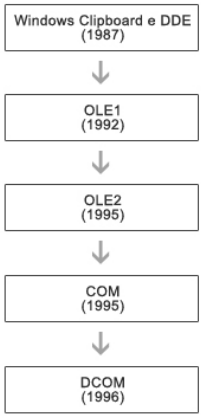

Figura 4 – Desenvolvimento histórico do COM/DCOM
Fonte: (LEITAO, 2006)

A figura 4 ilustra a evolução da ideia de interoperabilidade que surgiu desde a criação do Windows Clipboard, criada pela Microsoft, que permitia a transferência de informações entre aplicações Windows, até as tecnologias COM/DCOM que promovem o procedimento

padrão para criação de *softwares* que objetivam a integração de equipamentos. Com base nessa tecnologia, foram criadas centenas de OPC de acesso a dados tanto em servidores quanto em clientes. Logo, o OPC nada mais é do que definições de como utilizar e implementar os objetos COM/DCOM (LEITAO, 2006).

Toda comunicação OPC é baseada na tecnologia Microsoft COM. COM define regras para criar componentes dentro do sistema operacional do Microsoft Windows. DCOM é uma extensão do COM que gerencia a conexão entre o COM cliente e COM servidor de arquivos. Na gestão desta conexão, o DCOM realiza a triagem e autenticação das funções, assim como para determinar o protocolo de comunicação que vai ser utilizado. A fim de alcançar sucesso na comunicação entre os componentes OPC, a segurança do sistema operacional e os componentes COM devem ser devidamente configurados. Com isso, é necessário ser feito uma série de configurações de segurança no Windows para que a comunicação OPC seja permitida (OPCTRAINING, 2016). São elas:

1. OPC usa Activex COM e DCOM para comunicar, então é necessário dar permissões ao DCOM para permitir comunicações entre os objetos;
2. É necessário registrar o programa do servidor OPC como uma aplicação;
3. O DEP(*Data Execution Prevention*) é um conjunto de tecnologias de *hardware* e *software* que realiza verificações adicionais na memória para ajudar a evitar que códigos mal-intencionados executem em um sistema. O DEP também pode impedir algumas instalações de funcionar. Pode acontecer dos arquivos do servidor OPC detectar a configuração DEP e parar a instalação, com isso é necessário desativar o DEP do sistema;
4. Caso se esteja usando *workgroups*, ao invés de domínio, é necessário alterar e liberar algumas permissões nas configurações de segurança.

Segundo (BOND, 2007), devido a forte ligação entre OPC com DCOM, o OPC depende de uma série de componentes da Microsoft para configuração e operação. Como a maioria dos aplicativos do Windows, OPC e DCOM fazem uso extensivo do registro do Windows. Quando um servidor OPC é instalado, muitas vezes adiciona entradas no registro do Windows. Aqui estão algumas entradas típicas:

- Identificador de programa (ProgIO) - Esta *string* é definida como "Manufacturer.ServerName" para servidores OPC. Ao olhar entre os registros é possível ver quais aplicativos OPC estão instalados em um determinado sistema. A entrada do

programa identificador também deve conter as sub-chaves "OPC" e "CLSID (*Class identifier*)";

- Identificador de Classes (CLSID) - Este é o padrão de 128 bits GUID (*Globally Unique Identifier*) que identifica todos os objetos COM. Além disso Identificadores de Categoria (CATID) são armazenados como sub-chaves dentro do CLSID e definem quais das especificações do OPC estarão ativas. Cada fornecedor OPC tem uma CLSID e um único valor para cada aplicação.

2.2 World Wide Web

Até a década de 90, a Internet era usada basicamente por pesquisadores, acadêmicos e estudantes universitários para interligação com hospedeiros remotos, transferência de arquivos, além do envio e recebimento de correio eletrônico. Embora essas aplicações fossem, e continuem a ser, muito úteis, a Internet não era muito difundida fora das comunidades acadêmicas e de pesquisa. No entanto, no início da década de 90, surgiu a aplicação-chave da Internet: a WWW (*World Wide Web*) (BERNERS-LEE et al., 1994).

A web é a aplicação que elevou a Internet para o nível que agora ocupa, como um dos mais importantes meios de comunicação. Com o início do funcionamento da WWW, a Internet foi levada para os lares e empresas de milhões e milhões de pessoas em todo o mundo, servindo como plataforma para a habilitação e a disponibilização de centenas de novas aplicações, inclusive negociação de ações em bolsa de valores e serviços bancários on-line, além de serviços sofisticados de multimídia (BARBATO, 2008).

Segundo (FRANZINI, 2009), dentre as demais características do uso da internet, algumas se destacam, como:

1. Podem ser acessados em qualquer lugar do mundo, a qualquer hora, a qualquer dia. Ou seja, total disponibilidade, desde que se tenha acesso a web;
2. Não importa qual é a máquina ou sistema operacional, aplicativos web são implementados em cima de tecnologias padronizadas como HTTP e HTML que são reconhecidas e utilizadas em qualquer plataforma;
3. Os clientes acessam via aplicativos *browsers* (Navegadores de Internet) que são fáceis de usar e atualmente estão popularizados;
4. Não é necessário instalar nada nas máquinas-clientes para usar os sistemas;
5. Milhões de pessoas podem acessar o mesmo sistema, ao mesmo tempo;

6. Atualizações dos sistemas nas máquinas-servidores não refletem nenhuma mudança necessária nas máquinas-clientes. Quando o responsável muda o sistema, o mundo inteiro é atualizado automaticamente.

Para desenvolvimento de aplicações web, dentro das várias ofertas de linguagens de programação, o Java se destaca. Inicialmente projetado para facilitar a convergência entre um computador e equipamentos eletrônicos como os eletrodomésticos por exemplo. Depois acabou sendo adaptado para a internet e uma das grandes diferenças da linguagem de programação Java para as demais é que qualquer *hardware* ou equipamento eletrônico que possa executar uma máquina virtual conseguirá executar o Java, ou seja, o código do programa é escrito apenas uma vez e pode ser executado em qualquer equipamento eletrônico (LUCKOW; MELO, 2010).

2.2.1 Java

O enorme crescimento da *World Wide Web* levou a uma grande procura de páginas web dinâmicas e interativas, capazes de oferecer ao utilizador uma boa experiência de navegação. Para atrair a atenção de possíveis clientes, as empresas apostam cada vez mais na Internet, como forma de mostrarem ao mundo os seus produtos e/ou os seus serviços. Como forma de tornar isto possível, é importante que a página web, seja o mais agradável e intuitiva possível, possibilitando um nível de interatividade elevado e capaz de cativar todo o possível cliente (FERREIRA, 2008).

O HTML, em si, é estático, e apenas permite uma mínima interação com os utilizadores. Como consequência disto, as páginas web que quiserem ser verdadeiramente interativas e funcionais não podem utilizar somente o HTML. Face à necessidade de uma nova abordagem para introduzir alguma dinâmica num conteúdo que era estático até à data, o aparecimento da tecnologia Java veio imprimir profundas mudanças à *World Wide Web* tradicional e mudar de forma decisiva este cenário (FERREIRA, 2008).

Java é uma linguagem de programação orientada a objetos, desenvolvida por uma pequena equipe de pessoas na Sun Microsystems. Inicialmente elaborada para ser a linguagem-base de projetos de *software* para produtos eletrônicos, a grande crescida do Java aconteceu em 1995, devido ao sucesso mundial da *World Wide Web*. Esta linguagem difere das linguagens mais tradicionais, como o C ou C++, em que o código fonte é compilado diretamente para código máquina. Em Java é seguida uma filosofia diferente. Como existe código transferido pela Internet que será executado localmente numa grande variedade de plataformas clientes, o Java foi projetado para ser multi-plataforma, tanto no código fonte do programa como no ficheiro binário, depois de compilado (FERREIRA, 2008).

Uma vez que não é possível correr o mesmo ficheiro binário (por exemplo, em plataformas Windows, Linux ou MAC OS), o Java compila o seu código fonte para um ficheiro intermédio independente da plataforma, o *bytecode*. Este ficheiro será interpretado e executado por uma máquina virtual. Esta máquina virtual (*Java Virtual Machine* ou JVM) é o núcleo onde se encontra o motor responsável por executar o *bytecode* do programa Java e é específica para cada plataforma. Desta maneira, apenas trocando a JVM para o sistema operativo e arquitetura de *hardware* em causa, será possível correr a mesma aplicação sem necessidade de a recompilar numa grande variedade de plataformas. A figura 5 mostra uma sequência da criação de um programa Java.

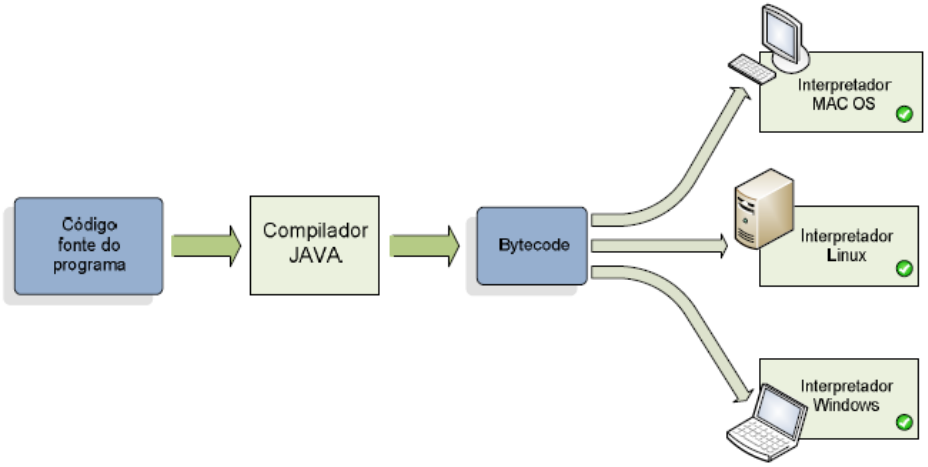

Figura 5 – Sequência de um programa Java
Fonte: (FERREIRA, 2008)

Em Java há uma distinção entre alguns tipos de aplicações, sendo os de maior importância os programas standalone, os applets e os servlets. Os programas standalone são os programas típicos de uso geral, que desempenham as mesma funções que os programas escritos numa outra linguagem de programação (FERREIRA, 2008).

Um applet é um programa Java compilado, que se integra numa página web. Ao ser realizado o acesso com um *browser* àquela página, o *bytecode* é carregado através da web, sendo, em seguida, executado na máquina local onde reside o *browser* cliente. Os applets são usados, normalmente, para fornecer interactividade a aplicações web onde o simples HTML não é capaz de o fazer por si só, como foi citado acima (FERREIRA, 2008).

Um servlet Java pode ser encarado como um applet, mas a correr do lado do servidor web. À semelhança de outras tecnologias que executam do lado do servidor, como o CGI ou o PHP, este programa Java manuseia dinamicamente pedidos e respostas tipicamente em HTML com um cliente, permitindo, assim, adicionar conteúdos dinâmicos a uma aplicação web. Um servlet Java requer um servidor de aplicação especial que proporciona o ambiente para este executar em cooperação com o servidor web. Um exemplo deste

servidor de aplicação, e que implementa as especificações definidas pela Sun Microsystems, é o Apache Tomcat desenvolvido pela Apache Software Foundation (FERREIRA, 2008).

2.2.1.1 Utgard

Utagard é uma biblioteca Java para lidar com comunicações OPC. É um distribuidor próprio, totalmente puro Java, e pode ser usado independentemente de outros projetos Openscada. Ele possui funções que possibilita conexões com um servidor OPC, atualmente suporta a interface OPC DA 2.0 (SCADA, 2016).

O Utgard não necessita mais do que o ambiente Java 1.6. sem códigos nativos, sem bibliotecas JNI (*Java Native Interface*) ou necessidade de outras bibliotecas DLLs. É uma biblioteca com o código aberto e licença livre. Por outro lado ele apenas implementa a parte do cliente do OPC DA, sendo necessário um servidor OPC (SCADA, 2016).

2.2.2 Servidor web

Os servidores web são programas responsáveis por armazenar e trocar informações com outras máquinas. Por causa disso, pelo menos dois participantes são envolvidos em cada troca de informações: um cliente, que solicita informações, e um servidor, que atende a esses pedidos (ALVES et al., 2010).

Cada lado exige também um programa especializado para negociar a troca de dados. No caso do cliente, é usado um *browser*, como o Internet Explore ou Firefox. No lado do servidor existem várias opções de *software* disponível, mais todos têm uma tarefa semelhante: negociar transferência de dados entre clientes e servidores via HTTP, o protocolo de comunicações da web (ALVES et al., 2010).

Os pedidos HTTP se referem habitualmente a páginas HTML. O processo se inicia com a conexão entre o computador onde está instalado o servidor web e o computador do cliente; como na web não é possível prever a que hora se dará essa conexão, os servidores web precisam estar disponíveis dia e noite. A partir daí é processado o pedido do cliente, e conforme as restrições de segurança e a existência da informação solicitada, o servidor devolve os dados (ALVES et al., 2010).

2.2.2.1 Apache

O servidor web Apache está dentre os servidores mais populares do mundo. Por ser um sistema de código aberto, foi muito difundido mundialmente devido às possibilidades de alteração e de melhorias em seu funcionamento. Uma grande vantagem do servidor web Apache consiste na existência de diferentes versões do programa, as quais permitem

que o servidor funcione em diferentes sistemas operacionais, de forma transparente ao usuário (ABREU; JUNIOR; BARREIROS, 2012).

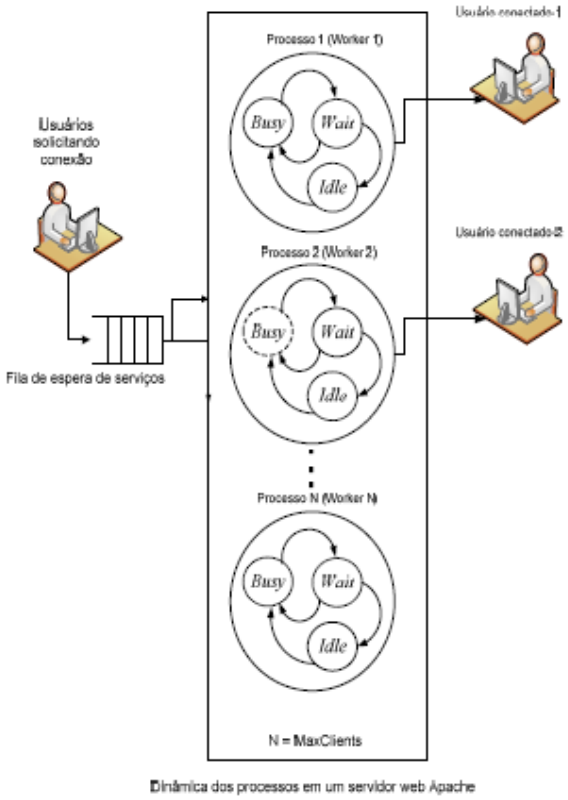

Figura 6 – Tratamento de requisição de serviços pelo servidor Apache
Fonte: (ABREU; JUNIOR; BARREIROS, 2012)

Dentre as normas de funcionamento do servidor Apache, é importante analisar-se a forma com que o servidor trata as requisições que pedem atendimento pelo sistema. O servidor Apache dispõe de processos trabalhadores (ou *threads* trabalhadoras, dependendo da configuração do servidor e do sistema operacional em questão), responsáveis pelo tratamento das requisições de serviços enviadas por clientes, vide figura 6. Estas requisições, antes de serem atendidas, entram em uma fila de requisições, que as organiza de modo que possam ser atendidas por processos que se encontrem em estado ocioso naquele momento (ABREU; JUNIOR; BARREIROS, 2012).

3 IMPLEMENTAÇÃO E ANÁLISE DOS RESULTADOS

3.1 Planta 2 Tanques Acoplados

Buscando controle por meios remotos, a ideia final é ter um laboratório remoto em que os alunos possam ter informações ou até mesmo fazer testes a distância. Este trabalho foca na parte da troca de informações entre um servidor OPC, que apresenta uma variável que representa um equipamento qualquer, com um servidor web.

Inicialmente, essa aplicação foi pensada para ser utilizada na planta "2 Tanques Acoplados" que se encontra no Laboratório de Máquinas Elétricas do prédio de automação da UFOP. Esta planta é muito utilizada em matérias de laboratórios do curso de Engenharia de Controle Automação, principalmente na matéria eletiva Controle Aplicado a Sistemas Térmicos e Fluidomecânicos onde os alunos necessitam fazer ensaios e testar valores para o controle de nível de um tanque.

A planta consiste de uma bancada experimental baseada no kit "Coupled Tanks" fabricado pela empresa Quanser, figura 7. São dois tanques fixados em um painel, junto a um reservatório que possui uma bomba que bombeia água para os dois tanques. Os dois tanques deste sistema são montados de forma que o fluxo de saída do tanque superior flui para dentro do tanque inferior e o fluxo de saída do tanque inferior flui para dentro do reservatório, configurando assim um sistema fechado no qual a massa do sistema permanece constante durante o processo (COCOTA et al., 2014).

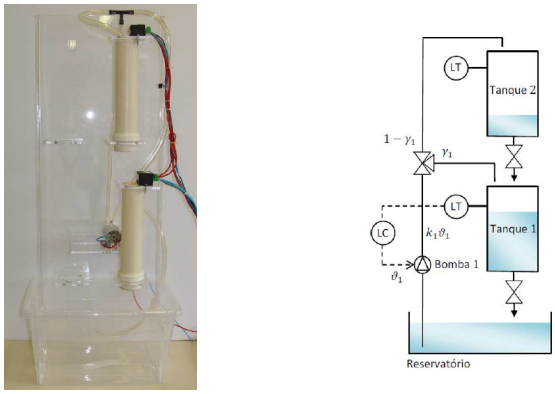

Figura 7 – Sistema de dois tanques acoplados
Fonte: (COCOTA et al., 2014)

A medição de nível dos tanques é indireta, sendo aferida pela pressão diferencial por meio de sensores. Os alunos identificam alguns parâmetros e implementam diferentes métodos de controle para comparar o desempenho dos mesmos.

Foi com base nessa planta que foi desenvolvido uma interface web para testar a transferência de dados entre o servidor OPC e o servidor web. Foi utilizado a plataforma Java EE com o ambiente de desenvolvimento Eclipse e o servidor web Apache.

3.2 Interface web

Para desenvolvimento da tela em Java foi utilizado o JSF (*Java Server Faces*). O JSF é um *framework* para programar web em Java. Foi disponibilizado pela Oracle para facilitar o desenvolvimento de aplicações web. Com ele é possível criar interfaces com textos, botões, componentes de entradas e saída, além de disponibilizar várias formas de formatação dos mesmos.

Inicialmente, foi desenvolvido um menu para escolhas e informações sobre o projeto, o resultado é visto na figura 8. Na aba projeto, pode-se escolher entre as formações que a planta pode ter, nesse caso se será utilizado 2 ou 4 tanques para fazer o controle do nível. Na aba Informações, pode ser apresentado textos explicativos sobre a planta, como: quais são e como adquirir os parâmetros da planta e os tipos de controles que são pedidos na matéria com informações de como chegar ao resultado desejado. Na aba ajuda possui o "sobre" com informações adicionais sobre o site e projeto.

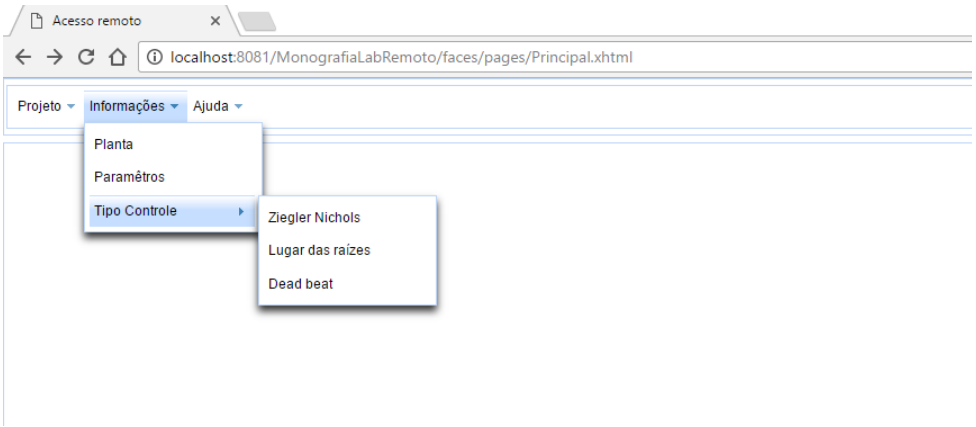

Figura 8 – Interface do menu do projeto 2 tanques acoplados

Pensando na formação com 2 tanques, foi construído uma tela rascunho apenas para teste da comunicação. Na página é mostrado a resposta em tempo real da planta, entre eles: tensão na bomba, nível do tanque superior e inferior, vide figura 9. Foi criado um botão para fazer a comunicação com o servidor OPC e, consequentemente, a aquisição de dados do mesmo.

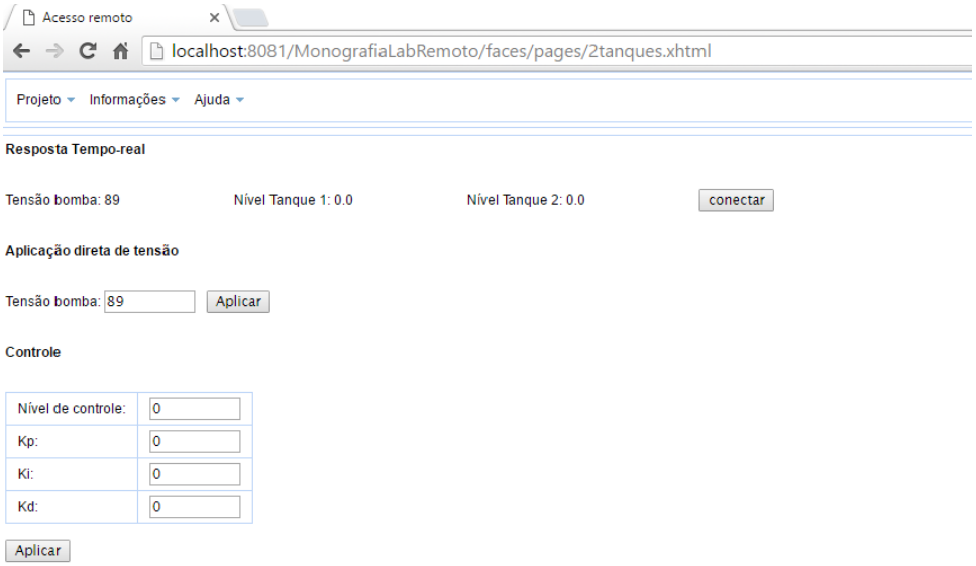

Figura 9 – Interface do projeto 2 tanques acoplados

No projeto é necessário aplicar um valor de tensão direto na bomba para que o nível real do tanque chegue próximo do valor que se deseja manter no controle. O objetivo é diminuir o erro acumulado no controlador. Nesse caso, ao invés de pegar a informação do servidor OPC, ela será levada para o mesmo, mudando o valor da variável configurada.

Foi criado uma seção para a aplicação dos valores definidos para o controle (Kp, Ki e Kd). Esse também é um caso em que os dados seriam enviados para aplicação final de controle da planta.

Também foi utilizado outros dois *frameworks*: PrimeFaces e Facelets. São ambos *frameworks* para projetos JSF, criados para desenvolver e facilitar a construção das interfaces JSF utilizadas em aplicações sofisticadas para empresas ou no desenvolvimento de sites padrões. Com eles é possível aumentar a produtividade do desenvolvedor e a experiencia do usuário com a aplicação, pois torna menos árduo criar uma aplicação que seja exibida corretamente na maioria dos dispositivos, sem contar que é muito flexível e personalizável, com uma grande opção de componentes para os mais diversos fins. Eles são acrescentados como bibliotecas para o JSF e suas funções e ferramentas são chamadas através de comandos.

```
<ui:composition xmlns="http://www.w3.org/1999/xhtml"
        xmlns:h="http://java.sun.com/jsf/html"
        xmlns:f="http://java.sun.com/jsf/core"
        xmlns:p="http://primefaces.org/ui"
        xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
        template="/templates/modeloProjeto.xhtml">
        <ui:define name="menu">
          <ui:include src="/includes/menu.xhtml"/>
        </ui:define>
        <ui:define name="conteudo">
          <h:outputLabel value="Resposta Tempo-real"
          style="font-weight:bold"/>
          <h:form >
                  <h:outputLabel
                  value="Tensao bomba:#{MBVariaveis.variaveis.vbomba}"/>
                  <p:spacer width="100" height="10" />
                  <h:outputLabel
                  value="Nivel Tanque1:#{MBVariaveis.variaveis.niveltq1}"/>
                  <p:spacer width="100" height="10"/>
                  <h:outputLabel
                  value="Nivel Tanque2:#{MBVariaveis.variaveis.niveltq2}"/>
                  <p:spacer width="100" height="10"/>
                  <h:commandButton value="conectar"
                  actionListener="#{MBVariaveis.comunicar()}"/>
          </h:form>
        </ui:define>
</ui:composition>
```

No trecho acima, da linha de código utilizada no trabalho, é possível observar que os *frameworks* são definidos por letras, que ao serem chamadas com suas funções, trazem as ferramentas escolhidas pelo desenvolvedor. O comando do Facelets, identificadas com as letras "ui", separa seções da tela. Neste caso, a superior contendo o menu e o inferior o conteúdo da página. Os demais são comandos, HTML e primeface, que são identificados respectivamente com as letra "h"e "p". Estes são utilizados para formatação da escrita, dos espaços entre textos e configurações de entrada e saída das variáveis.

3.3 Servidor OPC

Na procura de um servidor OPC, o servidor OPC Matrikon surgiu como uma das aplicações mais comuns e com mais informações. Ele permite que seja configurado drivers de comunicação para diversas redes e protocolos de diferentes fabricantes. Os programas Matrikon possuem duração de 1 mês grátis, mas possui um servidor OPC apenas para

simulações e este é gratuito. O MatrikonOPC for Simulation cria um servidor OPC e fornece opções de variáveis para teste de comunicação. Uma vez que a comunicação funcionar com o simulador, ela também funcionará para os demais servidores OPC do mercado.

Lembrando do que já foi dito, para utilizar a comunicação OPC em um sistema operacional Windows é necessário configurar o DCOM para que a aceite. Uma vez terminado a configuração é necessário instalar o MatrikonOPC for Simulation junto com MatrikonOPC Explorer. O MatrikonOPC Explorer é um cliente OPC que oferece ao usuário a funcionalidade de visualizar em tempo real as variáveis presentes no servidor OPC. Ele identifica os servidores presentes no computador e oferece a opção de conexão. Quando a conexão estiver pronta ele possibilita enxergar as variáveis contidas no servidor escolhido.

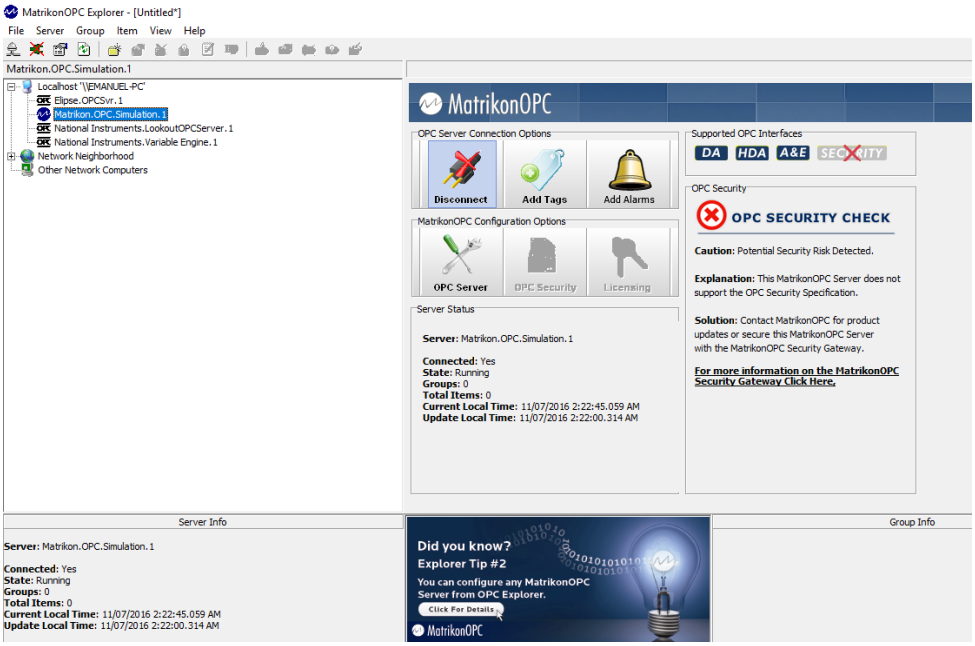

Figura 10 – Interface inicial do MatrikonOPC Explorer

No MatrikonOPC Explorer, ao selecionar o servidor MatrikonOPC for Simulation e conectar-se a ele, aparecerá a opção de adicionar tags, como mostrado na figura 10. Acionando essa opção, será aberto uma nova janela. A figura 11 mostra que possui uma aba onde aparecem os itens que estão contidos no servidor. O programador irá selecionar qual variável ele irá necessitar. Ao selecioná-la, ela aparecerá na tela principal do Explorer mostrando qual a sua situação e qual é o valor que ela está carregando. No caso do servidor para simulação, será apresentado uma variedade de itens para simulações. Ao selecionar uma dessas variáveis, a mesma seguirá para tela principal e será possível alterar o seu valor e ver como ela está se comportando, vide figura 12. Concluído essa etapa, a variável estará disponível no MatrikonOPC Explorer para acompanhar se a mesma será buscada ou alterada, corretamente, pela biblioteca Utgard do Java.

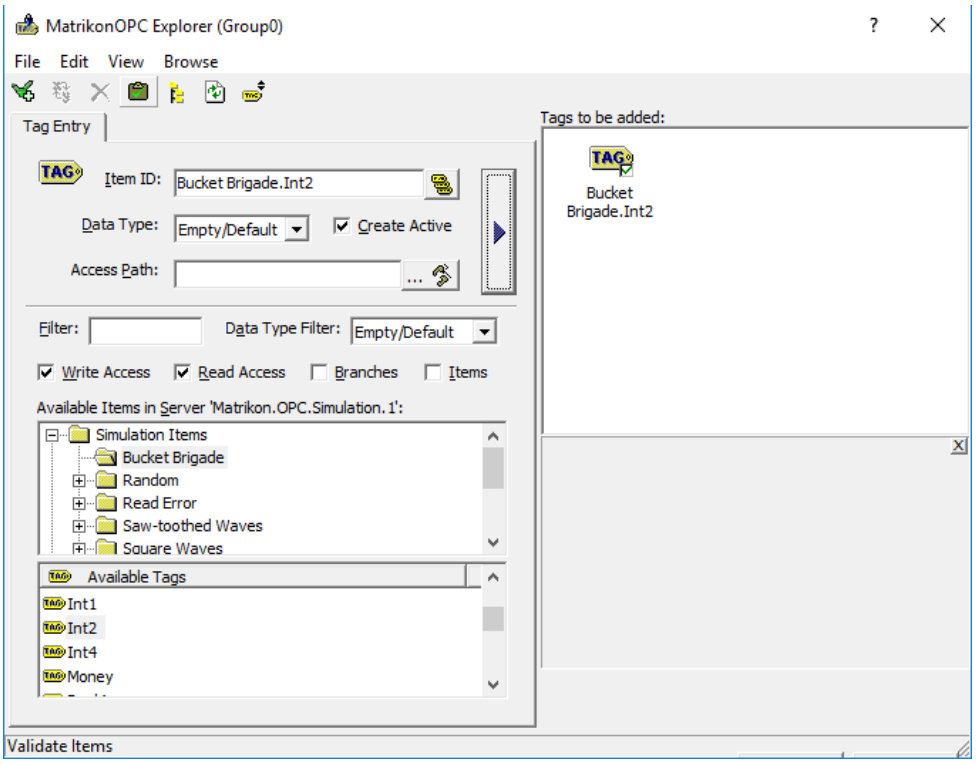

Figura 11 – Quadro de seleção de variáveis do servidor MatrikonOPC for Simulation

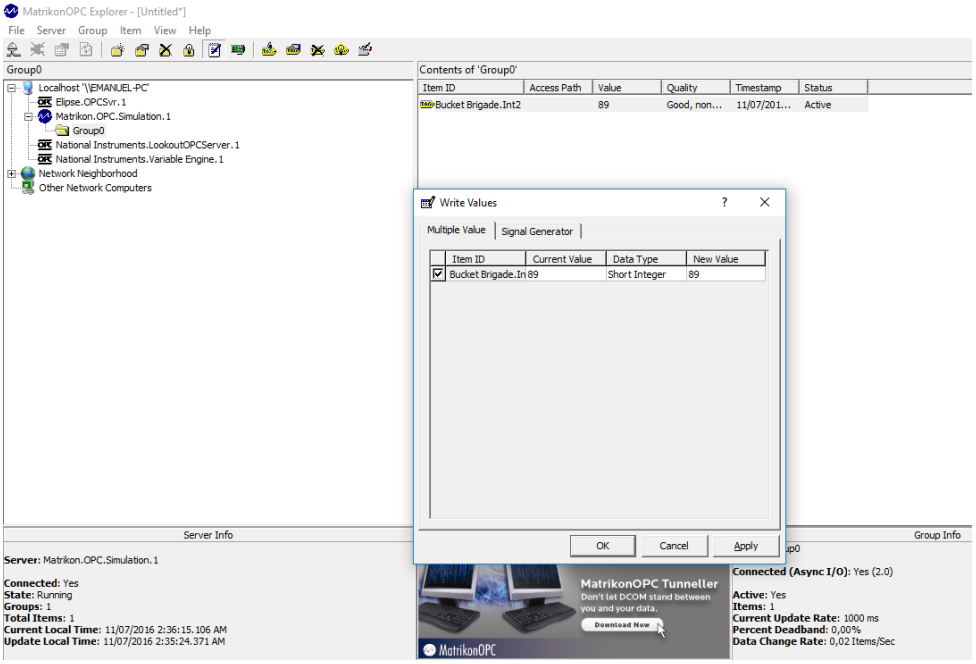

Figura 12 – Visão da situação das variáveis selecionadas no servidor MatrikonOPC for Simulation

3.4 Comunicação entre Java e servidor OPC usando a biblioteca UTGARD

A plataforma Eclipse permite que o programador trabalhe, separadamente, a parte web da parte de programação normal do ambiente Java. Com a parte web pronta, resta resolver a parte da programação que irá tratar as variáveis e as funções que irão ser usadas no programa web.

Inicialmente, na parte da programação Java, deve-se declarar todas as variáveis que serão mostradas ou recolhidas na interface web. Aqui também é onde será utilizado a biblioteca Utgard para fazer a conexão com o servidor OPC para a troca de informações. Intermediando a parte Java da parte web, o JSF utiliza a classe Java managed bean. É nessa classe que ocorre a interligação entre as variáveis e métodos de botões feitos na programação Java com a interface web.

Normalmente é prudente fazer a programação das classes escalonadas, separando partes da programação que tem funções diferentes. Ajuda na simplificação do código e principalmente na procura de erros. Mas como o objetivo do trabalho é ver a funcionalidade da biblioteca Java Utgard comunicando com um servidor OPC, o programa será reduzido para teste de apenas uma variável, não necessitando de uma programação extensa. Logo, as funções ficaram todas na classe maneged beans.

```
package br.com.remoto.bean;
import javax.faces.bean.ManagedBean;
import javax.faces.bean.ViewScoped;
import br.com.remoto.domain.Variaveis;
import java.util.concurrent.Executors;
import org.jinterop.dcom.core.JIVariant;
import org.openscada.opc.lib.common.ConnectionInformation;
import org.openscada.opc.lib.da.Group;
import org.openscada.opc.lib.da.Item;
import org.openscada.opc.lib.da.Server;
@ManagedBean(name = "MBVariaveis")
@ViewScoped
public class VariaveisBean {
        private Variaveis variaveis;
        public Variaveis getVariaveis() {
                if (variaveis == null){
                        variaveis = new Variaveis();}
                return variaveis;       }
public void comunicar() throws Exception {...}
public void aplicar() throws Exception {...}
```

Na classe managed bean, como podemos ver no trecho de código acima, deve-se inicialmente importar todas as bibliotecas que serão utilizadas e definir um nome de acesso que será usado como ligação pelo programa web para conectar variáveis e botões com as funções da classe, linha 11. Deve-se, também, dar acesso à classe com as variáveis Java e iniciar as mesmas caso sejam requisitados na interface web.

A leitura e escrita de variáveis no servidor serão feitas na interface web após eventos como cliques de botões, atualização de página ou outros. Foi utilizado os cliques de botões na página web para chamar as funções contidas na classe managed bean. No programa, foi definido "comunicar" para quando o operador, na interface web, deseja saber qual é o valor da variável que está no servidor naquele momento, e "aplicar" para quando o operador deseja enviar um valor para a variável escolhida. A variável a ser alterada será Vbomba, que representa a tensão na bomba. No servidor OPC ela estará representada por "Bucket Brigade.Int2".

```
public void comunicar() throws Exception {
        final ConnectionInformation ci = new ConnectionInformation();
        ci.setHost("localhost");
        ci.setUser("emanuel");
5       ci.setPassword("balduino12");
        ci.setClsid("F8582CF2-88FB-11D0-B850-00C0F0104305");
        final Server server = new Server(ci,
         Executors.newSingleThreadScheduledExecutor());
        server.connect();
```

Contando que todos as bibliotecas estejam instaladas e devidamente declaradas na classe. Tratando inicialmente da função "comunicar", deve-se identificar qual servidor o programador gostaria de acessar. Deve-se informar o Host, usuário e senha de quem está conectado na rede, junto com o nome do PrgID ou CLSID do servidor. Com essas informações o programa tentará achar e conectar com o servidor OPC determinado pelo programador.

```
1       Group group = server.addGroup();
        Item item = group.addItem("Bucket Brigade.Int2");
        JIVariant valorvar = item.read(false).getValue();
        short valorconvertido = valorvar.getObjectAsShort();
        variaveis.setVbomba(valorconvertido);
6       server.dispose();
```

Uma vez conectado ao servidor OPC, nele as variáveis são separadas em grupos. Logo, é necessário acioná-los para então procurar as variáveis dentro do mesmo. O termo "Item"

é como as variáveis são identificadas dentro do servidor, estas serão achadas como foram configuradas no servidor OPC. O Item buscado é do tipo JIVariant, logo é necessário criar uma variável do mesmo tipo para o receber. Como a interface web não consegue interpretar uma variável JIVariant, é necessário convertê-la para um tipo mais comum, no caso, "short". Após convertido, o valor buscado no servidor está pronto para ser atribuído à variável "Vbomba", criado na programação Java, que será enviado para interface web quando for acionada.

```
        Group group = server.addGroup();
        Item item = group.addItem("Bucket Brigade.Int2");
        short valorescrito = variaveis.getVbomba();
4       JIVariant teste = new JIVariant(valorescrito);
        item.write(teste);
        server.dispose();
```

Quando a função "aplicar" é acionada, o cliente na interface web deseja enviar um valor a uma variável do servidor. Neste caso, inicia-se o mesmo processo de identificação do servidor e busca da variável a ser alterada. O que muda, no entanto, é que ao invés do comando de leitura no servidor, será feito um de escrita. Mas antes é necessário que a variável que carrega o valor a ser enviado seja convertido para JIVariant.

Concluídas estas duas funções é possível ver a interação entre uma página de internet com um servidor OPC. Mais variáveis podem ser adicionadas ao programa para serem analisadas ou alteradas remotamente, uma vez comprovado que a troca de informações, utilizando a biblioteca Java Utgard, é eficiente.

4 CONCLUSÕES

Foi demonstrado que a ferramenta Utgard é eficiente na interconexão entre o servidor OPC e a plataforma Java. Unindo a sua funcionalidade com o desenvolvimento de uma interface web, utilizando a mesma plataforma Java, é possível a integração para fazer acessos remotos de equipamentos. O acesso ao servidor, tanto para escrita quanto leitura de dados, é eficiente e rápido, uma vez que a biblioteca executou a comunicação pela primeira vez.

Apesar de utilizar um simulador de servidor OPC nas operações, tudo o que foi testado com ele também é válido para os demais servidores existentes. Como o mercado possui inúmeras empresas para inúmeros equipamentos, o servidor OPC é uma excelente ferramenta de integração. Logo, uma biblioteca como a Utgard que permite a comunicação entre Java e OPC é de grande utilidade pra quem precisa fazer controle de equipamentos a distância.

Para trabalhos futuros é possível testar o funcionamento do acesso remoto utilizando uma planta real. Melhorar a interface web para acessar diversos equipamentos e mostrar resultados em gráficos e tabelas. Criar um sistemas de câmeras para poder ver a ação dos equipamentos no laboratório.

REFERÊNCIAS

ABREU, T. W. M.; JUNIOR, W. B.; BARREIROS, C. T. C. J. J. A. L. Estratégias de identificação paramétrica aplicadas à modelagem dinâmica de um servidor web apache. **Revista Controle and Automação**, v. 23, n. 1, p. 38–48, 2012.

ÁLVARES, A. J.; FERREIRA, J. C. E. Metodologia para implantação de laboratórios remotos via internet na área de automação da manufatura. In: **2o Congresso Brasileiro de Engenharia de Fabricação (COBEF)**. Uberlândia: [s.n.], 2003. v. 18.

ALVES, M. C. S. et al. Desenvolvimento de um sistema de aquisição e monitoramento de dados à distância para processos de usinagem. In: ABCM. **VI Congresso Nacional de Engenharia Mecânica**. Campina Grande, PB, 2010.

BARBATO, A. K. **Políticas para Servidores Web Baseados em Sessões Visando Qualidade e Diferenciação de Serviços**. Monografia (Dissertação de mestrado em Ciências de Computação e Matemática Computacional) — Instituto de Ciências Matemáticas e de Computação, USP, São Carlos, 2008.

BERNERS-LEE, T. et al. The world-wide web. **Commnunications of the ACM**, v. 37, n. 8, p. 76–82, 1994.

BOND, D. **OPC Security White Paper 1**. 2007. Disponível em: <https://scadahacker.com/library/Documents/OPC_Security/OPC%20Security%20-%20Understanding%20OPC.pdf>. Acessado: Nov 7, 2016.

BROCKHAUS, G. **OPC Connectivity Using Java**. 2016. Disponível em: <http://pt.slideshare.net/mbohnen/opc-connectivity-using-java>. Acessado: Nov 9, 2016.

COCOTA, J. A. N. et al. Análise de diferentes controladores para o processo de dois tanques acoplados. **Congresso Brasileiro de Educação em Engenharia**, 2014.

FERREIRA, J. A. O. **Interface Homem-Máquina para Domótica baseado em tecnologias Web**. Monografia (Dissertação de mestrado em Engenharia Eletrotécnica e de Computadores) — Faculdade de Engenharia da Universidade do Porto, FEUP, Porto, 2008.

FONSECA, M. O. Comunicação opc - uma abordagem prática. **VI Seminário de Automação de Processos**, 2002.

FRANZINI, F. **Aplicações Web com Java**. 2009. Disponível em: <http://imasters.com.br/artigo/13463/java/aplicacoes-web-com-java?trace=1519021197&source=single>. Acessado: Nov 8, 2016.

JÚNIOR, R. F. F. **Identificação Remota de Plantas Industriais Utilizando Tecnologias OPC E Cyberopc**. Dissertação (Dissertação de mestrado em Engenharia Elétrica) — Engenharia de São Carlos Elétrica e de Computação, USP, São Carlos, 2009.

LEITAO, G. B. P. **Arquitetura e Implementação de um cliente OPC para aquisição de dados na indústria do petróleo**. Monografia (Trabalho de final de curso em Engenharia de Computação) — Departamento de Engenharia de Computação e Automação, UFRN, Natal, 2006.

LUCKOW, D. H.; MELO, A. A. **Programação Java para a Web**. São Paulo : Novatec Editora: Novatec, 2010.

MARCHEZAN, A. R. **Ferramentas aplicadas no desenvolvimento de laboratório remoto e/ou presencial no ensino de engenharia eletrônica**. Dissertação (Dissertação de mestrado em Engenharia Elétrica) — Faculdade de Engenharia Elétrica e de Computação, UNICAMP, Campinas, 2006.

OPCTRAINING, I. **OPC and DCOM: 5 Things You Need to Know**. 2016. Disponível em: <https://scadahacker.com/library/Documents/ICS_Protocols/OPC%20Training%20Inst%20-%20OPC-DCOM%20-%205%20Things%20You%20Need%20to%20Know.pdf>. Acessado: Nov 7, 2016.

PUDA, A. P. Padronização da comunicação através da tecnologia opc. **SoftBrasil Automação Ltda**, 2008.

RIBEIRO, D. F.; LEANDRO; ANJOS, R. Utilização do drive na integração de sistemas de automação industrial. In: **3o Semnário Nacional de Sistemas Industriais e Automação**. Belo Horizonte: [s.n.], 2008.

SCADA, O. **Utgard**. 2016. Disponível em: <http://openscada.org/projects/Utgard/>. Acessado: Nov 8, 2016.

TECHNOLOGIES, K. **OPC Interoperability: Open Connectivity Through Open Standards**. 2016. Disponível em: <https://www.kepware.com/en-us/products/kepserverex/opc-interoperability/>. Acessado: Nov 7, 2016.