Um modelo distribuído de automação de subestações baseado em tecnologia multiagentes

A distributed substation automation model based on the multi-agents technology

Klaus de Geus[1]
Flávio Milsztajn[1]
Carlos José Johann Kolb[1]
José Henrique Dometerco[1]
Alexandre Mendonça de Souza[1]
Ciro de Carvalho Braga[1]
Emerson Luis Parolin[1]
Arlenio Carneiro Frisch[1]
Luiz Kiss Fortunato Júnior[1]
Augusto Erzinger Júnior[1]
Marco Antônio Jonack[1]
Anderson Juliano Azambuja Guiera[1]

[1]**Copel – Companhia Paranaense de Energia**
klaus@copel.com
flaviomil@copel.com
kolb@copel.com
dometerc@copel.com
alexandre.mendonca@copel.com
ciro@copel.com
parolin@copel.com
arlenio@copel.com
luiz.kiss@copel.com
aerzinger@copel.com
jonack@copel.com
guiera@copel.com

Abstract: The main purpose of this paper is to analyse distributed computing technology which can be used in substation automation systems. Based on performance comparative results obtained in laboratory, a specific model for distributed substation automation is proposed considering the current model employed at Copel – Companhia Paranaense de Energia. The proposed model is based on the multi-agents technology, which has lately received special attention in the development of distributed systems with local intelligence.

Keywords: Substation Automation, Distributed Computing, Multi-agents Systems.

Resumo: Este artigo tem como objetivo fazer uma análise de tecnologias de computação distribuída que podem ser utilizadas em sistemas de automação de subestações. Com base em resultados comparativos de desempenho obtidos em laboratório, um modelo específico de automação distribuída de subestações é proposto considerando o modelo atual empregado na Copel – Companhia Paranaense de Energia. O modelo é baseado na tecnologia de multiagentes, a qual tem recebido especial atenção no desenvolvimento de sistemas distribuídos com inteligência localizada.

Palavras-Chave: Automação de Subestações, Computação Distribuída, Sistemas Multiagentes.

1 Introdução

Uma das grandes dificuldades na gestão de sistemas tradicionais de automação no contexto de sistemas elétricos consiste na carência de intercâmbio automatizado entre protocolos e plataformas distintos. Devido à natureza da comunicação provida pela internet, torna-se polêmica a viabilidade de sua utilização no contexto de automação de sistemas elétricos, por terem estes requisitos que necessitam processamento em tempo real, além de necessitarem um alto grau de segurança. Entretanto, alguns autores reivindicam que a utilização da internet nesse contexto é viável, alicerçada pelos avanços tecnológicos alcançados na área de comunicação e redes de computadores [1].

Considerando a estrutura de rede estabelecida na Copel – Companhia Paranaense de Energia, objetiva-se, por meio deste trabalho, avaliar tecnologias de computação distribuída com vistas a estabelecer uma arquitetura para operação de subestações por meio da intranet corporativa. A operação de subestações automatizadas na Copel é subdividida em diversos centros regionais, cujos sistemas de controle de operação não estão integralmente conectados entre si, o que resulta em falta de flexibilidade nas atividades de controle e operação.

Este artigo apresenta algumas arquiteturas e soluções descritas na literatura para abordar a questão de computação distribuída em sistemas elétricos, em sua grande maioria baseadas em sistemas multiagentes e em tecnologia internet. Apresentam-se também um estudo de cenários e a proposta de um modelo de sistema distribuído de automação de subestações. Além disso, um

estudo comparativo do relacionamento entre tecnologias e plataformas que podem ser utilizadas em implementações de soluções distribuídas é apresentado. Os resultados dos testes obtidos permitem importantes conclusões pertinentes a soluções computacionais no que tange às tecnologias e plataformas utilizadas.

2 Arquiteturas distribuídas em automação

Algumas abordagens de arquiteturas distribuídas são descritas na literatura científica, em várias áreas de aplicação. A maioria das aplicações SCADA (*Supervisory Control and Data Acquisition*) faz uso da tecnologia de agentes, como descrito na seqüência.

Ebata et al. [2] propõem a utilização de uma intranet como plataforma para um sistema SCADA e relatam resultados preliminares de testes. O trabalho foca em desempenho e confiabilidade em tempo real, os quais constituem fatores que implicam na solução de alguns problemas. Essas soluções, por sua vez, exigem a ampliação da arquitetura SCADA.

Buse et al. [3] descrevem uma arquitetura genérica que aplica a metodologia de sistemas multiagentes em automação de subestações. Os autores alegam que os sistemas de automação e controle de sistemas elétricos atuais, baseados no modelo SCADA, apesar de oferecerem desempenho e confiabilidade adequados, não apresentam flexibilidade no que diz respeito a acesso aberto a informações. Por sua natureza inerentemente distribuída, os sistemas elétricos são alvo de aplicação de sistemas de controle baseados em agentes, os quais provêem maior autonomia para cada parte constituinte do sistema elétrico.

Nessa arquitetura, agentes são designados para diferentes funções, tais como monitorar e controlar o sistema elétrico, ler dispositivo, armazenar dados e prover interface gráfica, e se comunicam por meio de mensagens. A recuperação de informações pode ser feita por meio de agentes móveis, os quais trafegam pela rede, ou por meio da identificação de agentes responsáveis por tarefas específicas e troca de mensagens entre eles.

Hopkinson et al. [4] descrevem o desenvolvimento de um ambiente de simulação distribuído. Eles alegam que os simuladores atuais na área de sistemas elétricos de potência modelam de maneira precisa sistemas de potência do passado, os quais eram controlados como grandes parques regionais de potência sem elementos significativos de comunicação. Entretanto, uma vez que sistemas de potência estão cada vez mais adotando sistemas de controle e proteção que fazem uso de redes de computadores, tais simuladores estão cada vez menos capazes de predizer o provável comportamento da malha de potência resultante.

Similarmente, as ferramentas usadas para avaliar novos protocolos e sistemas de comunicação foram desenvolvidas sem a devida atenção aos papéis que eles podem ter em cenários de potência. Essa ferramenta utiliza múltiplos sistemas comerciais e de pesquisa para preencher a lacuna. Segundo os autores, o sistema permite que usuários encapsulem de maneira transparente o comportamento de sistemas complexos que abranjam domínios múltiplos por meio do uso de um *framework* simples baseado em agentes.

Li et al. [5] reivindicam que todos os sistemas SCADA baseados na web se defrontam com questões relacionadas ao desempenho em tempo real e à confiabilidade de controle de supervisão. Seu trabalho considera o sistema SCADA em toda a sua extensão como uma aplicação baseada na web, decompõe cada pedaço do software de aplicação em um conjunto de componentes de um website, e esquematiza o projeto de um subsistema de controle de supervisão e de um subsistema de aquisição de dados com base no modelo de camada de serviços em três níveis, o qual adiciona flexibilidade e escalabilidade ao sistema e permite que este se beneficie de toda a tecnologia internet disponível, tais como arquitetura aberta, protocolo de comunicação padrão e módulos de aplicação maduros.

Prayurachatuporn and Benedicenti [6] descrevem como o desenvolvimento de um protocolo de serviço de diretório, que utiliza tecnologia de agentes, aumentou a confiabilidade teórica do sistema SCADA. Segundo os autores, a utilização da tecnologia de agentes é adequada por permitir distribuição, a qual inerentemente promove redundância, e modularidade, a qual promove versatilidade.

As três características principais de um sistema de agentes que são vantajosos para um sistema SCADA são:

- modularidade;
- comunicação;
- mobilidade.

A utilização de agentes móveis no protocolo de um sistema de controle distribuído requer:

- um ambiente *runtime* para que os agentes possam ser executados;
- uma interface padrão para interações;
- serviços para a criação, migração e encerramento de agentes móveis;

- suporte à mobilidade e comunicação de agentes enquanto provê de segurança tanto os hosts quanto os agentes.

Agentes têm que ser identificados de maneira inequívoca no ambiente em que operam. Isso possibilita que o controle, a comunicação, a cooperação e a coordenação dos agentes ocorram. Um agente pode ser acessado onde quer que esteja, o que possibilita a implementação de um mecanismo seguro quanto a falhas, no qual múltiplas cópias de um agente podem ser ativadas em diferentes localidades, melhorando assim, de maneira efetiva, a confiabilidade do sistema.

Uma outra aplicação de tecnologia de agentes no contexto de proteção e controle de sistemas elétricos, funcionando sobre uma intranet, é descrita por Shono et al. [7]. Os autores propõem uma plataforma para agentes móveis em tempo real, a qual permite:

- sincronizações entre múltiplos agentes em um ambiente distribuído; e
- redundância para resolver problemas de comunicação.

Uma aplicação descrita nesse trabalho, a qual utiliza a plataforma de agentes em tempo real, subdivide os agentes em:

- agentes de configuração: Utilizados para configurar relés de proteção. De acordo com o comportamento do sistema elétrico, eles podem permanecer em um relé específico para mudar suas configurações de proteção e controle;
- agentes de análise: Utilizados para a coleta de dados para análise de falhas no sistema elétrico, esses agentes móveis são responsáveis também por planejar suas próprias rotas de viagem entre os dispositivos e, caso as informações coletadas sejam insuficientes para análise, executar um replanejamento de rota.
- agentes de patrulha: Utilizados para a coleta de informações sobre as condições de operação dos equipamentos.

Seki et al. [8] desenvolveram um protótipo de um sistema de processamento de informação de sistemas de energia baseado na web. Entretanto, em seu estágio inicial, o protótipo contempla apenas informações sobre falhas de energia. Sua arquitetura, que segundo os autores assegura flexibilidade e escalabilidade em sistemas de energia, consiste de:

- uma camada de aplicação baseada em web browser;
- uma camada de modelo de informação que reflete o comportamento do sistema de energia; e
- uma camada de gerenciamento de objetos distribuídos utilizando CORBA e recursos de rede disponibilizados pelo sistema operacional.

Yavnai [9] apresenta uma arquitetura distribuída e descentralizada para a organização, comando, controle e comunicação de um sistema multiagentes funcionando de forma cooperativa e autônoma.

A motivação para a operação cooperativa autônoma tem sua sustentação na necessidade de executar missões críticas com restrições de tempo, recursos e disponibilidade que estão além da capacidade de um único agente. No caso dos agentes estarem geograficamente distribuídos, a operação cooperativa é uma abordagem adequada, fornecendo suporte a:

- compartilhamento de informações;
- compartilhamento de recursos;
- alocação eficiente de recursos;
- respostas orientadas a contexto e situação;
- robustez e flexibilidade sob mudanças de condições;
- redundância.

Uma característica chave da arquitetura proposta é que todos os agentes são idênticos em relação à percepção, ao processamento de informação, à tomada de decisão, à capacidade de comunicação, independentemente de sua missão. Dessa forma um agente pode ser facilmente substituído no caso de falha.

3 Infra-estrutura de interconectividade

Uma arquitetura para automação distribuída de subestações, de acordo com o modelo apresentado na seção anterior, exige o estabelecimento de um arcabouço tecnológico, abrangendo desde tecnologias direcionadas à comunicação entre computadores até aquelas direcionadas a técnicas de controle e inteligência artificial.

Muitas tecnologias de interconectividade têm sido abordadas e descritas na literatura. Dentre elas, podem-se destacar Sockets, Java Remote Method Invocation (RMI), web services e tecnologia de multiagentes.

3.1 Sockets

Sockets são estruturas que permitem que funções de software se interconectem. O termo é usado para especificar uma estrutura que faz com que rotinas de software na mesma máquina ou em máquinas diferentes possam se comunicar. Sockets são implementados em várias linguagens

de programação (Assembler, C, C++, Java, dentre outras) para a maioria dos sistemas operacionais existentes. Também são utilizados como core para tecnologias mais sofisticadas de comunicação, entre elas RMI, CORBA e Web Services.

3.2 Java RMI

RMI é um sistema que permite que um objeto sendo executado em uma JVM (Java Virtual Machine*)* invoque métodos de um objeto sendo executado em outra JVM, ou seja, RMI provê um mecanismo de comunicação entre programas escritos na linguagem Java. RMI define um conjunto de interfaces remotas que podem ser utilizadas para criar objetos remotos. Diferentemente de Sockets, que disponibiliza uma API (Application Program Interface) básica para manipulação de estruturas de comunicação, a API RMI fornece uma estrutura orientada a objetos de alto nível que encapsula toda a comunicação básica para acessar métodos remotos.

3.3 Web services

De acordo com o World Wide Web Consortium (W3C), a definição de web services, que pode ser encontrada na enciclopédia eletrônica Wikipedia (www.wikipedia.org), é a seguinte: Um web service é um sistema de software projetado para prover interação entre máquinas interoperáveis em uma rede.

Uma das vantagens providas por web services é a facilidade de utilização. Entretanto, sua principal desvantagem reside no baixo desempenho quando essa tecnologia é comparada com outras, tais como RMI e CORBA, como conseqüência de seu formato baseado em texto.

3.4 Sistemas multiagentes

A tecnologia de sistemas multiagentes tem sido reconhecida como um conceito chave na construção de sistemas de controle distribuído, inteligente, auto-organizador e robusto [10].

Sistemas tradicionais centralizados são inadequados para satisfazer requisitos que dizem respeito à flexibilidade e à adaptabilidade, especialmente funcionalidades tais como detecção de falhas de uma parte do sistema e recuperação segura que minimize os impactos na funcionalidade do sistema como um todo. Adaptabilidade a mudanças no ambiente constituem outro fator importante, segundo Vrba [11].

Vrba também reivindica que, em um sistema multiagentes, cada agente individual é uma unidade autônoma e cooperativa com controle descentralizado. Sua funcionalidade de autonomia implica em que ele pode gerenciar seu comportamento sob o ponto de vista local, ou seja, ele tem conhecimento sobre seus objetivos locais, que são executados localmente sem a necessidade de controle centralizado. Entretanto, podem existir circunstâncias em que um agente não é capaz de atingir seu objetivo. Em um caso desses, o agente pode solicitar auxílio a outros agentes. A comunicação entre agentes também pode acontecer quando um replanejamento global ocorre, implicando na atribuição de novas tarefas aos agentes.

Vrba também analisa várias plataformas multiagentes, incluindo em suas avaliações plataformas tais como JADE (CSELT), FIPA-OS (Emorphia), ZEUS (British Telecom), JACK (Agent Oriented Software), GRASSHOPPER 2(IKV++ Technologies AG), ADK (Tryllian), JAS (Fujitsu, HP, IBM, Sun), AgentBuilder (IntelliOne Technologies), MadKit (MadKit Team), Comtex Agent Plataform (Communication Technologies), Bee-gent (Toshiba) e Aglets (IBM Japan). Vrba faz uma análise mais específica sobre desempenho, entre as plataformas JADE, FIPA-OS, ZEUS e JACK. De maneira geral, a plataforma JADE, em seus testes, apresentou maior desempenho com relação às outras avaliadas. Os testes consistiram de um esquema serial e outro paralelo. No esquema serial, a comunicação entre um par de agentes opera de maneira sincronizada, ou seja, o agente remetente envia a segunda mensagem apenas depois de receber a resposta do agente destinatário relativa à primeira mensagem enviada. No esquema paralelo, o agente remetente envia todas as mensagens de uma só vez. Os testes também contemplaram a comunicação entre um par de agentes e entre dez pares de agentes. Além disso, contemplaram uma configuração com apenas um servidor, com um servidor e duas máquinas virtuais Java e com dois servidores.

Baseado na abrangente avaliação feita no trabalho supramencionado, adotou-se neste trabalho a plataforma JADE.

JADE provê um arcabouço para a implementação de sistemas multiagentes por meio de *middleware* compatível com a especificação FIPA, que é uma organização da IEEE Computer Society responsável por promover tecnologia baseada em agentes e a interoperabilidade de seus padrões com outras tecnologias.

Uma vez que essa tecnologia é implementada inteiramente em Java, a plataforma pode ser distribuída em diversas máquinas, as quais não

precisam ter necessariamente o mesmo sistema operacional.

4 Análise de relacionamentos entre tecnologias e plataformas de automação de subestações

A tecnologia de agentes busca explorar heurística e inteligência local para, dentre outros objetivos, diminuir a necessidade de comunicação entre máquinas. Agentes podem tomar decisões e resolver problemas locais e, quando necessário, podem trocar informações e colaborar entre si para alcançar determinado objetivo. Esse esquema apresenta necessidade significativamente menor em termos de comunicação comparativamente a sistemas ingênuos ou com inteligência centralizada.

Não obstante essa diferença na abordagem intrínseca a cada tecnologia, é importante saber qual é o desempenho provido pelas tecnologias para que se possa traçar parâmetros, levando em consideração a filosofia de cada uma delas.

Para tanto, foram desenvolvidos alguns testes pertinentes à automação distribuída de subestações e às tecnologias envolvidas.

4.1 Testes com a Máquina Virtual Java

O mito de que aplicações em Java são lentas em relação àquelas desenvolvidas em linguagens de alto desempenho foi questionado nos testes de laboratório deste projeto, com o objetivo de obter uma real avaliação a respeito de seu desempenho frente a outras tecnologias, e sua pertinência a sistemas de automação.

Os testes foram realizados utilizando uma máquina Pentium 2 com 128 MBytes de memória. As máquinas virtuais Java utilizadas foram ambas desenvolvidas pela Sun Microsystems, versões 1.4.2 e 5.0. A opção “Hotspot server" e “Hotspot client” foram usadas em ambas as versões. O Hotspot server é utilizado para otimizar o desempenho no lado servidor. O Hotspot client, por sua vez, é utilizado para otimizar o desempenho no lado cliente [12].

Um conjunto abrangente de algoritmos foi utilizado para que os testes pudessem avaliar todos os principais aspectos. Alguns deles, que fazem uso de uma grande quantidade de memória, tiveram problemas na JVM 1.4.2, mas foram executados de maneira satisfatória na JVM 5.0.

Uma observação importante é que ambas as versões da JVM apresentaram desempenhos similares no sistema operacional Linux, mas a versão 5.0 apresentou desempenho bem melhor em relação à versão 1.4.2 no sistema operacional Windows.

A Tabela 1 apresenta um resumo dos testes de desempenho comparativo entre as linguagens Java e C++, com as duas versões de JDK e utilizando uma série de algoritmos.

Tabela 1: Comparativo de desempenho entre diferentes versões de JDK e entre as linguagens Java e C++, utilizando um conjunto abrangente de algoritmos (tempo medido em segundos).

	JDK 1.4.2		**JDK 5.0**		
algoritmo	**Java**	**Java Server**	**Java**	**Java Server**	**C++**
Fibo (param = 45)	39,0	29,4	51,2	30,1	59,1
hash2	24,0	19,5	24,1	19,9	0,4
hash	6,0	5,9	5,1	5,4	2,0
heapsort	51,0	46,9	46,0	46,2	47,3
matrix	51,0	35,2	53,7	39,2	28,1
methcall	38,0	4,8	33,4	4,9	26,2
nestedloop	46,0	37,6	47,7	37,8	20,0
objinst		25,8		28,8	43,3
strcat	7,0	5,4	5,3	4,7	3,4
random	63,6	39,3	69,6	46,5	21,0
sieve	25,0	22,9	26,1	20,2	19,3

O website *http://kano.net/javabench* fornece dados comparativos interessantes que foram confirmados pelos testes realizados neste trabalho. Os algoritmos utilizados aqui são os mesmos daqueles apresentados no website. A versão 5.0 da JVM não é contemplada nos testes descritos no website kano.net, mas foram realizados para enriquecer o comparativo neste trabalho e comprovar o bom desempenho da tecnologia. Os resultados, em geral, confirmam que o desempenho de Java não necessariamente é inferior ao de C++, variando de acordo com os aspectos dos algoritmos. O desempenho de Java com o HotSpot Server mostrou ser melhor do que o de Java com o Hotspot Client. Fica também evidente a discrepância de desempenho dependendo da natureza dos recursos computacionais necessários, levando à conclusão de que, quando o desempenho é uma questão chave, a natureza do tipo de processamento

específico da aplicação deve ser levada em consideração na escolha da linguagem.

4.2 Comparativo de desempenho de diferentes tecnologias

Uma das maneiras de avaliar a adequabilidade das tecnologias de computação distribuída, mencionadas neste trabalho, na questão da automação de subestações consiste em testes de desempenho.

Entretanto, deve-se ter em mente que soluções que utilizam diferentes tecnologias de distribuição podem (e devem) ter também diferentes arquiteturas. Quando se utilizam sistemas multiagentes, por exemplo, tem-se por princípio a implantação de inteligência local, a qual minimiza a necessidade de comunicação. Pode-se dizer, portanto, que a adequabilidade de sistemas multiagentes em relação a outra tecnologia não pode ser julgada apenas pelo seu desempenho apresentado, mas todo o seu paradigma deve ser considerado.

Os testes de desempenho fornecem, mesmo assim, importantes informações que podem subsidiar decisões quanto à utilização dessas tecnologias.

O estudo contempla um servidor contendo uma subestação, a qual contém interfaces para as diferentes tecnologias. Os clientes fazem requisições à subestação. Nos testes, todos os clientes implementados com as tecnologias testadas fazem a mesma requisição, a saber, a recuperação dos pontos de estado da subestação de maneira síncrona. O cliente da plataforma JADE necessita uma adaptação para funcionar de maneira síncrona, isto é, o fim da requisição se dá no momento em que o servidor responde com os dados solicitados.

Os clientes foram implementados como testes do JUnit (framework para testes de regressão) e executados pela ferramenta JMeter (aplicação desktop puramente Java que permite a execução de testes funcionais e a medição de desempenho). Testes foram executados para comparar as tecnologias levando-se em consideração também o número de requisições simultâneas a serem atendidas pelo servidor. O produto utilizado nos testes com Web services foi o Sun Java Application Server Platform Edition 8.

Os testes foram executados por duas máquinas cliente com as mesmas características de processamento e memória acessando de forma não simultânea um servidor (Pentium IV, 2,6 GHz, 512 MBytes de memória interna, com sistema operacional Windows XP). Esse procedimento foi utilizado para minimizar um possível impacto da rede corporativa nos resultados. Com esse mesmo objetivo, cada conjunto de requisições simultâneas (5, 10, 25, 50, 100) do teste foi executado três vezes, sendo computadas as médias das amostras dos dois computadores cliente.

A Figura 1 apresenta um diagrama que mostra o tempo necessário para que uma requisição seja emitida e atendida utilizando cada uma das quatro tecnologias estudadas. Cada requisição usada nos testes é constituída de um array de 10 elementos de dupla precisão (*double*). Os resultados obtidos demonstram que a tecnologia RMI apresenta melhor desempenho que as demais, inclusive Sockets. Isso ocorre porque, embora RMI seja baseada em Sockets, implementa várias otimizações que influenciam diretamente no desempenho, como um pool para a reutilização das conexões Sockets instanciadas.

Na implementação com sockets, o recurso de pool não foi utilizado, o que pode ser verificado no gráfico pela degradação dos resultados com o aumento da carga. Nesse caso, a influência no desempenho foi maior no cliente, devido ao elevado número de conexões sockets criadas pelo sistema operacional. Embora RMI adicione várias camadas de abstração sobre a tecnologia Sockets, apresenta um desempenho melhor na relação carga / desempenho devido às otimizações fornecidas em sua API.

Web services apresenta tempo de resposta adequado para poucas requisições simultâneas. Quanto maior o número de requisições simultâneas, maior o consumo de recursos de máquina, tanto no cliente como no servidor, apresentado pela tecnologia web services. A tecnologia de agentes utilizando a plataforma JADE teve desempenho melhor que web services, porém pior que RMI e Sockets. Entretanto, deve-se observar que a tecnologia de agentes, quando utilizada de acordo com sua filosofia, ou seja, considerando inteligência local, não apresenta alta demanda de comunicação.

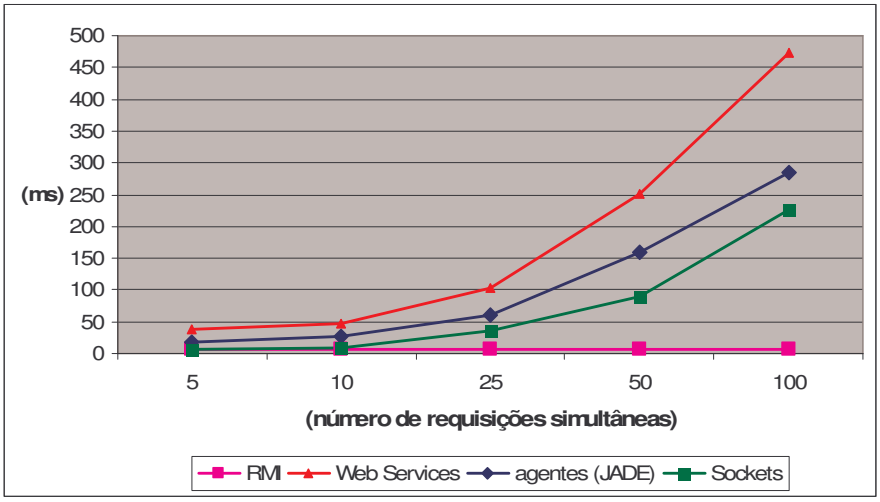

Figura 1: Testes de desempenho entre as tecnologias RMI, web services, sockets e agentes (JADE)

Deve-se considerar também que o desenvolvimento na plataforma JADE demanda maior elaboração por se tratar de um modelo assíncrono de comunicação e por possuir uma estrutura própria de desenvolvimento baseada em comportamentos.

5 Modelo distribuído de automação de subestações

A modernização dos sistemas de automação de subestações nas concessionárias de energia elétrica implica na adoção de uma abordagem específica no tocante à infra-estrutura legada. A substituição dessa infra-estrutura por outra que seja moderna, distribuída e que traga todos os benefícios em termos de flexibilidade, segurança e desempenho é possível, mas constitui uma tarefa penosa, além de incorrer em alguns riscos que devem ser analisados cuidadosamente.

Na Copel, as subestações estão conectadas, por meio de um protocolo específico do sistema de automação, a cinco centros de operação de distribuição (CODs) e a nove centros de operação de transmissão (COEs), espalhados pelas regiões elétrico-geográficas do Estado do Paraná. Esses centros de operação não estão necessariamente conectados entre si, exigindo que cada subestação seja operada exclusivamente por meio do centro de operação ao qual pertence.

O estudo de possíveis arquiteturas para o sistema de automação distribuída foi então baseado em um ambiente constituído de quatro camadas. A primeira camada é a plataforma do usuário ou operador. A segunda camada diz respeito ao núcleo de interação com o usuário, a qual basicamente consiste de um módulo servidor (seja centralizado ou distribuído) que pode estabelecer conexão com os outros componentes do sistema. A terceira é constituída dos centros de operação, e a quarta das subestações controladas pelos centros de operação. Essa situação é ilustrada na Figura 2.

Os estudos demonstraram que, como primeira alternativa, é possível mesclar as camadas 2 e 3, concebendo um servidor distribuído com unidades servidoras nos centros de operação, não necessariamente em todos.

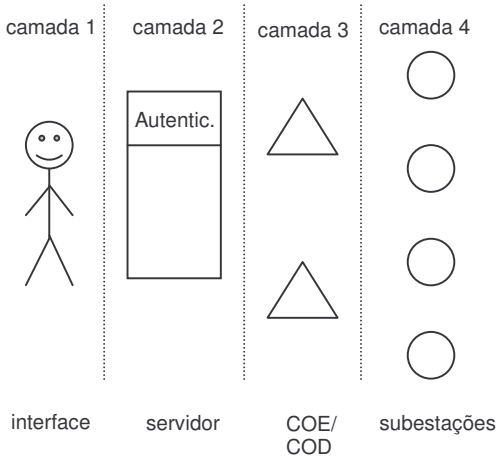

Figura 2: Modelo de quatro camadas de abstração de um ambiente de automação de subestações

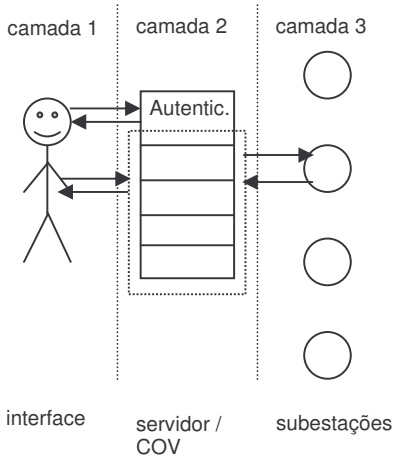

Figura 3: Modelo de três camadas de abstração de um ambiente de automação de subestações

A alternativa derivada dos estudos de cenários aponta para uma arquitetura na qual as camadas 2, 3 e 4 estão presentes em cada subestação, contemplando um servidor que permita a comunicação entre subestações, ou seja, sem a presença de um servidor central. Entretanto, essa arquitetura, extremamente flexível, implica em significativa complexidade de gerência.

Dessa forma, um modelo abstrato adequado que contemple um servidor distribuído aponta para a primeira alternativa, ou seja, uma arquitetura com três camadas, como ilustrado na Figura 3. A camada 2 é constituída de um ou mais servidores para prover redundância e anular problemas causados por falhas em um dos servidores. O papel exercido pelos centros de operação (COEs e CODs) é, neste modelo, exercido pelos centros de operação virtuais (COVs), os quais independem de localização física.

O COV é uma sessão de operação de um conjunto de subestações por parte de uma equipe de operadores. A operação de uma subestação específica pode ser negociada entre COVs, uma vez que a subestação só pode ser controlada em um determinado instante por um único centro de operação. Isso significa dizer que os centros de operação físicos do modelo anterior são substituídos por centros de operação virtuais dinâmicos. Igualmente, um conjunto de operadores pode obter acesso a mais de um COV, por exemplo, numa situação emergencial em que o operador vinculado a um determinado COV necessite acesso a uma subestação vinculada a outro COV.

A Figura 4 ilustra o modelo proposto de maneira mais detalhada. É relevante observar que a plataforma multiagentes engloba as camadas 2 e 3. Na camada 3, cada subestação possui um container que pode abrigar vários agentes, que se comunicam com um container principal na camada 2, na qual é feita a comunicação entre a plataforma multiagentes e os operadores por meio do servidor de aplicação. Esse servidor provê serviços básicos, tais como HTTP e autenticação.

Este modelo embute em sua arquitetura uma estrutura de negociação entre os agentes que representam os COVs. Dessa maneira, um COV pode negociar a possessão temporária de uma subestação com seu atual proprietário, por meio de regras pré-estabelecidas e regras baseadas no *status* do sistema. Por exemplo, uma das regras prioritárias consiste no fato de que nenhuma subestação pode ficar desacoplada de um agente operador e monitorador em qualquer instante.

As negociações também levam em consideração os perfis dos operadores envolvidos, estabelecendo regras de prioridade que provêem determinado operador de maior ou menor poder em determinado instante. Por exemplo, a solicitação de controle de uma determinada subestação por parte de um operador de um COV pode ter maior ou menor prioridade de acordo com seu perfil.

Esta arquitetura distribuída possibilita que nos agentes das subestações existam funções automatizadas que atuam independentemente do restante do sistema e que decisões que dependam de mais de uma subestação fiquem a cargo dos COVs.

O importante nesse caso é a capacidade que as subestações têm de fornecer as informações necessárias para a tomada de decisão, de forma a não centralizar tarefas especificas que não requerem uma monitoração global. Isso também pode representar uma menor carga nos servidores centrais, diminuindo a necessidade de comunicação, de acordo com a filosofia de sistemas multiagentes.

Neste trabalho, o desenvolvimento de um protótipo tem como objetivo validar os conceitos embutidos no modelo. Cada COV é representado por um agente registrado no serviço de diretório da plataforma JADE. Sempre que uma nova subestação é incorporada ao sistema, ela se anuncia aos COVs por meio do serviço de diretório. O processo de negociação é iniciado assim que a subestação faz uma chamada por propostas (*Call for proposals – CPF*) aos COVs disponíveis. Essa transação é feita de acordo com o protocolo FIPA (*FIPA-Contract-Net*) [13]. Seu objetivo é estabelecer, por meio de regras pré-estabelecidas e do estado atual do sistema, o COV mais adequado para assumir o controle da subestação.

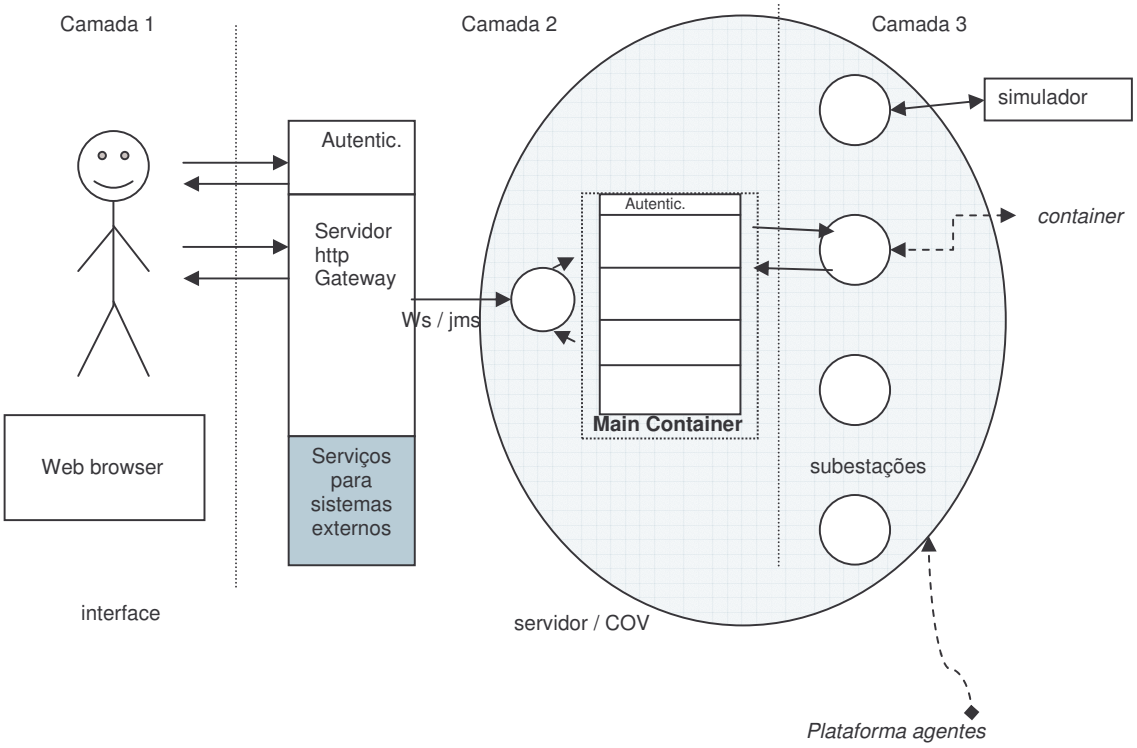

Figura 4: Diagrama detalhado do modelo de três camadas proposto,
Incluindo o conceito de Centro de Operação Virtual (COV).

A partir do momento que a subestação estiver sob supervisão de um determinado COV, ela se torna visível aos operadores a ele conectados.

Da mesma maneira, operadores podem, por meio de seus agentes, negociar a transferência de controle de uma determinada subestação com outros ligados ao mesmo COV. Vale salientar que, se a negociação ultrapassar os limites de um COV, quem executa a negociação são os COVs envolvidos. No caso da desconexão de um operador, por exemplo, as subestações sob sua supervisão serão distribuídas entre os demais operadores do COV por meio de regras. A decisão final sobre a transferência de cada subestação está a cargo do próprio COV.

6 Conclusões

Este trabalho apresenta as tendências tecnológicas de sistemas distribuídos de automação no contexto de sistemas elétricos. Como pode ser visto na revisão de literatura, grande esforço tem sido feito na tecnologia de multiagentes. Apesar de muitos trabalhos apresentarem apenas protótipos, há indícios de que a tecnologia está madura o suficiente para ser utilizada de forma efetiva.

Um modelo específico para a automação de subestações é proposto, o qual apresenta uma estrutura virtual dinâmica de controle e monitoração. Este modelo ainda deve ser avaliado de forma criteriosa, considerando-se os requisitos existentes no sistema de automação de subestações atual e os que são necessários à sua evolução frente às tecnologias disponíveis atualmente.

Além disso, é apresentada uma avaliação das potencialidades de utilização de algumas tecnologias de distribuição com resultados práticos obtidos em laboratório. Tais resultados ratificam a tendência verificada na literatura, apontando para a utilização da tecnologia de multiagentes e, como conseqüência, a inclusão de inteligência localizada.

7 Referências

[1] QIU, B. et al. Internet-based SCADA display system. **IEEE Computer Applications in Power**, January, 2002, p. 14-19.

[2] EBATA, Y. et al. Development of the Intranet-based SCADA (supervisory control and data acquisition system) for power system. In: Power Engineering Society Winter Meeting, 2000. IEEE v. 3, 23-27 Jan. 2000, p. 1656 – 1661.

[3] BUSE, D. P. et al. Agent-Based Substation Automation. In: Proc. IEEE Power & Energy, v. 1, n. 2, March/April 2003, p. 50-55.

[4] HOPKINSON, K.M. et al. EPOCHS: integrated commercial off-the-shelf software for agent-based electric power and communication simulation. In: Simulation Conference, 2003. Proceedings of the 2003 Winter, v. 2, December 2003, p.1158 – 1166.

[5] LI, D.; SERIZAWA, Y.; KIUCHI, M. Concept design for a Web-based supervisory control and data-acquisition (SCADA) system. In: Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES, v. 1, October 2002, p. 32 – 36.

[6] PRAYURACHATUPORN, S.; BENEDICENTI, L. Increasing the reliability of control systems with agent technology. **ACM SIGAPP Applied Computing Review**, v. 9, n. 2, July 2001.

[7] SHONO, T. et al. A Remote Supervisory System for a Power system Protection and Control Unit Applying Mobile Agent Technology. In: Transmission and Distribution Conference and Exhibition 2002: Asia Pacific, IEEE/PES v. 1, Oct. 2002. p. 148-153.

[8] SEKI, T. et al. Power systems information delivering system using Internet technology. In: Transmission and Distribution Conference and Exhibition 2002: Asia Pacific. IEEE/PES, v. 1, October 2002, p. 12 - 17.

[9] YAVNAI, A. Distributed decentralized architecture for autonomous cooperative operation of multiple agent system. In: AUV '94 – Proceedings of the 1994 Symposium on Autonomous Underwater Vehicle Technology. July, 1994. p. 61 – 67.

[10] VAN LEEUWEN, E. H.; NORRIE, D. Intelligent manufacturing: holons and holarchies. **Manufacturing Engineer**, v. 76, n. 2, 1997, p. 86–88.

[11] VRBA, P. JAVA-Based Agent Platform Evaluation. In: HoloMAS 2003, LNAI 2744, 2003, p.47–58.

[12] SUN MICROSYSTEMS. **Java SE HotSpot at a glance**. Disponível em: http://java.sun.com/javase/technologies/hotspot.jsp. Acesso em: 12 jun. 2006.

[13] BELLEFEMINE, F. et al.. **JADE Programmers Guide**. Disponível em: http://jade.tilab.com. Acesso em: 14 abr. 2005.