UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

OSMAR ASSIS DO NASCIMENTO FILHO

DESENVOLVIMENTO DE SERVIDORES OPC DA E OPC XML DA PARA SISTEMAS DE AQUISIÇÃO DE DADOS VIA TELEFONIA CELULAR

VITÓRIA
2005

OSMAR ASSIS DO NASCIMENTO FILHO

DESENVOLVIMENTO DE SERVIDORES OPC DA E OPC XML DA PARA SISTEMAS DE AQUISIÇÃO DE DADOS VIA TELEFONIA CELULAR

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do Grau de Mestre em Engenharia Elétrica, na área de concentração em Automação.
Orientador: Prof. Dr. Celso José Munaro, D.Sc.

VITÓRIA
2005

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Central da Universidade Federal do Espírito Santo, ES, Brasil)

Nascimento Filho, Osmar Assis do, 1977-
N244d Desenvolvimento de servidores OPC DA e OPC XML DA para sistemas de aquisição de dados via telefonia celular / Osmar Assis do Nascimento Filho. – 2005.
100 f. : il.

Orientador: Celso José Munaro.
Dissertação (mestrado) – Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Cliente/servidor (Computação). 2. Serviços na Web. 3. Automação industrial - Protocolos. 4. Telefonia celular. 5. COM/DCOM (Protocolo de rede de comunicação). I. Munaro, Celso José. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 621.3

OSMAR ASSIS DO NASCIMENTO FILHO

DESENVOLVIMENTO DE SERVIDORES OPC DA E OPC XML DA PARA SISTEMAS DE AQUISIÇÃO DE DADOS VIA TELEFONIA CELULAR

Dissertação submetida ao programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisição parcial para a obtenção do Grau de Mestre em Engenharia Elétrica - Automação.

Aprovada em 25 de novembro de 2005.

COMISSÃO EXAMINADORA

Prof. Dr. Celso José Munaro, D.Sc.
Universidade Federal do Espírito Santo – DEE
Orientador

Prof. Dr. Arlindo Gomes de Alvarenga, D.Sc.
Universidade Federal do Espírito Santo – DI

Prof. Dr. Ricardo Lüders
Universidade Tecnológica Federal do Paraná

DEDICATÓRIA

Dedico esse trabalho à minha família, aos meus amigos, e especialmente ao professor Calazans, que me auxiliou não só nos estudos de redes, mas também no aprimoramento de minha visão política e social.

AGRADECIMENTOS

Agradeço a minha família pela compreensão e auxílio nos momentos de difíceis decisões, maior dedicação e ausência.
Agradeço ao Celso, pelo constante incentivo para que eu mantivesse o foco e não me deixasse levar pelo desânimo.
Agradeço ao pessoal do LCI, pela ajuda e presença, sempre mantendo o alto astral e humanizando as pesquisas no laboratório.
Agradeço ao Marcos Fonseca que prestou apoio técnico essencial para que os sistemas propostos possuíssem o mais alto grau de qualidade de mercado.

“Nenhuma mente que se abre para uma nova idéia voltará a ter o tamanho original.”

Albert Einstein

Sumário

Lista de Figuras

Lista de Tabelas

Lista de abreviaturas e siglas

CDMA	*Code Division Multiple Access*
CLP	Controlador Lógico Programável
COM	*Component Object Model*
CRM	*Customer Relationship Management*
DLL	*Dinamic Link Library*
DCOM	*Distributed Component Object Model*
DCE	*Distributed Computing Environment*
DCS	*Distributed Control Systems*
EJB	*Enterprise Java Bean*
ERP	*Enterprise Resource Provider*
FTP	*File Transfer Protocol*
GSM	*Global System for Mobile Communications*
HMI	*Human Machine Interface*
HTML	*HyperText Markup Language*
HTTP	*HyperText Transfer Protocol*
HTTPS	*HyperText Transfer Protocol Secure*
IDL	*Interface Definition Language*
IEC	*International Electrotechnical Commission*
IP	*Internet Protocol*
MES	*Manufacturing Execution Systems*
MTA	*Multi Thread Apartment*
MTU	*Master Terminal Unit*
OASIS	*Organization for the Advancement of Structured Information Standards*
ODBC	*Open DataBase Connection*
OLE	*Object Linking and Embededding*
OPC	*OLE for Process Control*
ORPC	*Object-oriented Remote Procedure Call*
OSF	*Open Software Foundation*
OSI	*Open Systems Interconnection*
PC	*Personal Computer*
RFC	*Request For Comments*
RPC	*Remote Procedure Call*
RTS	*Real Time System*
RTU	*Remote Terminal Unit*
SCADA	*Supervisory Control and Data Aquisition*
SCM	*Supply Chain Management*
SMTP	*Simple Mail Transfer Protocol*
SOA	*Service Oriented Architecture*
SOAP	*Service Oriented Architecture Protocol*
STA	*Single Thread Apartment*
TCP	*Transmission Control Protocol*
UDDI	*Universal Description, Discovery and Integration*
URI	*Uniform Resource Identifier*
URL	*Universal Resource Locator*
UTR	Unidade Terminal Remota
W3C	*World Wide Web Consortium*

WSDL	*Web Services Definition Language*
WS-CDL	*Web Services Choreography Description Language*
WSRM	*Web Services Reliable Messaging*
WWW	*World Wide Web*
XML	*Extendable Markup Language*
XSL	*eXtensible Style Language*

Resumo

Este trabalho apresenta as informações relevantes para o desenvolvimento de servidores de dados OPC nas duas vertentes mais promissoras desse padrão: OPC DA e Web Services. É apresentada uma proposta de passos necessários para o desenvolvimento desse tipo de sistema, contextualizado com uma situação real de aquisição de dados usando telefonia celular como meio de acesso. Também são apresentados os resultados de testes de conformidade e desempenho que demonstram a capacidade de interoperabilidade fornecida pelas soluções implementadas.

Abstract

This work presents the relevant information for development of OPC data servers in the two more promising ways of this standard: OPC DA and Web Services. A proposal of necessary steps for the development of this type of system is presented, in the context of a real situation of data acquisition using cellular telephony technology as access way. The results of conformity and performance tests are presented and demonstrate the interoperability capacity supplied by the implemented solutions.

1 Introdução

A obtenção de informações de processos é uma necessidade tão crescente quanto o desenvolvimento das plantas industriais. Esta necessidade se torna ainda muito mais visível quando nos deparamos com problemas mais complexos de integração na indústria. Sistemas caros e altamente especializados criando o produto fim de uma fábrica precisam ser supervisionados e mantidos. No entanto, precisam dividir espaço e recursos com sistemas menos prioritários de gerenciamento de informação [1]. Tais sistemas, mesmo sendo menos prioritários para o produto final, apresentam-se como decisivos nas tomadas de decisão das pessoas de negócio da empresa, afetando sensivelmente os agentes da produção.

Além disso, o bom funcionamento de uma planta depende de como o sensoriamento está sendo feito na mesma. O controle preciso sobre as informações adquiridas de um sistema indica o quanto tal sistema pode evoluir, aumentando a acurácia na tomada de decisões. Manter sistemas industriais em constante comunicação pode se tornar extremamente custoso. A miscelânea de padrões em profusão no mercado vem dificultar o trabalho de supervisão ao invés de resolver os problemas de comunicação e aquisição de informação [2]. As empresas lutam para apresentar seus padrões proprietários como os mais eficientes e se especializam em soluções que atendem apenas em parte aos problemas de controle de uma planta. Com isto, a aquisição de informações fica dependente da existência de um *driver* que se comunique com o mundo de interfaces criado pela fornecedora de determinada solução de sensoriamento.

O uso de um padrão único para a implementação da comunicação entre o chão de fábrica e os sistemas de gerenciamento faz com que os custos de criação e adaptação de sistemas caiam substancialmente. A manutenção de tais sistemas também tem custo reduzido,

uma vez que o conhecimento necessário para se trabalhar com as tecnologias envolvidas no desenvolvimento de soluções em um padrão único não é demasiadamente grande e sua expansão não depende de novas curvas de aquisição de conhecimento [2].

Um importante padrão internacional que se firmou na indústria é o OPC (OLE *for Proccess Control*). Esse padrão consegue agregar a sistemas industriais uma série de vantagens que os diferenciam dos demais, como o isolamento de tráfego, a facilidade de integração entre sistemas e a interoperabilidade [3]. Criando especificações de interfaces de comunicação ao invés de tentar padronizar a operação das plantas industriais, esse padrão ganhou força na indústria. Atualmente é utilizado não só para disponibilizar dados e informações para o usuário final, mas também para expandir as possibilidades de comunicação entre os servidores de dados das empresas, servindo como uma opção mais adequada em relação aos modelos tradicionais de interconexão de protocolos [4]. Este trabalho visa apresentar as características associadas à implementação de sistemas de comunicação baseados nas especificações do padrão OPC para aquisição de dados em um canal de comunicação projetado sobre telefonia celular e posterior disponibilização na Internet. São abordados os pontos necessários para implementar servidores OPC DA (*Data Access*) e também é descrita a implementação de servidores OPC XML DA. Ambos visam fornecer uma formalização para transferência de dados para sistemas, sendo que o último vem a promover dois grandes avanços em relação ao primeiro: disponibilidade de interfaces desacopladas em memória, que permite maior controle sobre o tratamento de recursos utilizados pelo servidor; e independência nativa de plataforma, uma vez que consumidores de Web Services podem ser implementados para quaisquer dos sistemas operacionais atualmente em uso em sistemas de automação industrial [5]. Os servidores desenvolvidos agregaram aos padrões internacionais definidos pelo OPC as tecnologias de comunicação não dedicadas,

através da telefonia celular. Com isso, pretende-se apresentar uma solução que alie a interoperabilidade do padrão OPC com a versatilidade da comunicação celular para aquisição de informações e disponibilidade de acesso.

1.1 Motivação

A principal motivação para o uso de OPC é sua capacidade de uniformização de acesso a informações entre diferentes tecnologias e diferentes plataformas. Diferentes tecnologias no ponto em que serve de concentração para posterior acesso a partir de diversos tipos de software, ou seja, uma grande gama de clientes escritos em diferentes linguagens de programação. Diferentes plataformas devido à possibilidade de uso dos servidores em diferentes sistemas operacionais.

O padrão também possui a vantagem de ser internacionalmente reconhecido, tendo sido desenvolvido por um *pool* de grandes empresas desenvolvedoras de sistemas de automação industrial de diversos países, tornando-se realmente um padrão de baixa mutabilidade e fácil conformidade. Também consegue prover cerca de 80% das funcionalidades necessárias a qualquer sistema supervisório. Isto faz deste padrão o mais completo já desenvolvido dentre os utilizados nas diversas plantas industriais do mundo [1].

1.2 Escopo

Este trabalho apresenta a necessidade de padronização de acesso a informações de equipamentos de campo em sistemas SCADA, demonstrando os passos necessários para implementar servidores de dados em duas vertentes tecnológicas em plena ascensão na

indústria: OPC e Web Services. Para contextualizar a tese, os sistemas desenvolvidos utilizam telefonia celular como meio de acesso. Algumas características desse meio serão abordadas.

Por fim, são apresentados resultados de testes de conformidade e desempenho que indicam vantagens e desvantagens das duas tecnologias utilizadas.

1.3 Objetivos

Desenvolver dois servidores OPC (OLE / DCOM e XML) para um mesmo problema de aquisição de dados, baseados em sensores remotos que se comunicam com um computador via telefonia móvel. Averiguar as vantagens de desempenho e disponibilização de um servidor contra o outro. Apresentar um sistema de comunicação confiável para aquisição de informações, tornando-as disponíveis de forma padronizada.

1.4 Conteúdo

Esta tese está dividida em 7 capítulos, entre os quais este capítulo introdutório. O capítulo 2 apresenta uma série de conceitos associados aos sistemas SCADA modernos, tratados sob o enfoque da transmissão de informações e das soluções até então utilizadas. Também contextualiza os principais problemas que podem ser solucionados pelos sistemas desenvolvidos. O capítulo 3 contém detalhes tecnológicos relacionados ao paradigma cliente-servidor, base para o desenvolvimento de sistemas de informação modernos, assim como apresenta as definições necessárias para compreensão sobre o paradigma de desenvolvimento Web e as motivações para uso desse tipo de tecnologia. O capítulo 4 apresenta o padrão OPC e explicita as características comuns de implementação para quaisquer sistemas que sigam tal padronização. Fornece as informações sobre as decisões de projeto associadas ao

desenvolvimento de um servidor OPC DA com as especificidades relacionadas a sua tecnologia e explicita as necessidades e características tecnológicas importantes para o desenvolvimento de servidores OPC XML DA. O capítulo 5 apresenta o sistema desenvolvido, a tecnologia associada ao sistema de comunicação utilizado pelos servidores para a aquisição de informações, os passos necessários para associar esta camada de comunicação aos servidores e as decisões de projeto necessárias para a implementação da solução. O capítulo 6 possui as informações sobre os testes de conformidade, assim como os resultados dos testes de desempenho aplicados sobre os dois servidores. Algumas conclusões finais sobre a implementação dos sistemas são apresentadas no capítulo 7.

2 Fluxo de informações em plantas industriais

Neste capítulo os problemas relacionados à situação atual dos modelos de aquisição de informações e controle de plantas industriais serão apresentados para formar o contexto no qual a pesquisa foi realizada.

Normalmente, sistemas industriais são formados por pelo menos 4 camadas, numeradas de 0 a 3 [6]. O objetivo desta divisão é gerenciar os dados e as informações de acordo com as necessidades específicas de cada usuário que faça uso dos subsistemas e equipamentos pertencentes a cada camada. A camada 0 é a base da planta industrial. É onde se encontra todo o *hardware* relativo ao sensoriamento e atuação sobre as grandezas envolvidas na produção da indústria. Possui uma dependência muito grande com o produto final a ser produzido e com suas matérias primas, sendo seus componentes projetados para resolver uma quantidade muito pequena de problemas extremamente específicos. A camada 1 é onde se encontram as unidades remotas de concentração e, em alguns casos, tratamento crítico de informações. A maior parte da lógica de controle da planta se encontra nesta camada. Seus componentes são um pouco mais gerais do que os da camada 0, podendo ser programados de acordo com as necessidades de determinados módulos fabris. A camada 2 é formada pelos computadores de processo e sistemas de supervisão. É onde ocorre a operação da planta e o tratamento mais refinado das informações recolhidas pela camada anterior. Esta camada também fornece embasamento para alterações nos processos industriais, em prol da melhoria dos mesmos. É extremamente dependente de *drivers* de comunicação com a camada anterior, sendo dependente da tecnologia adotada na camada 1. A camada 3 apresenta os modelos de negócio e as informações que afetarão as decisões gerenciais da indústria. Esta camada é bem distinta das demais com relação à dependência tecnológica e à forma como

seus sistemas afetam a planta. Não há dependência com relação aos equipamentos utilizados na planta, nem a possibilidade de alteração direta dos parâmetros ou valores configurados nos equipamentos de campo. Em alguns casos, apenas as indicações de novos parâmetros de processos podem ser definidas nesta camada.

A tabela 2.1 apresenta algumas das principais tecnologias associadas a cada camada deste modelo.

Tabela 2.1 Tecnologias das camadas de dados de sistemas industriais

Camada	Descrição	Tecnologias disponibilização	Tecnologias comunicação
0	Atuação e sensoriamento	Sensores inteligentes, válvulas, religadores, plubiômetros, etc	FieldBus, HART, Modbus
1	Controle e supervisão	CLP, SCADA, DCS	RS232, RS485, Fieldbus Ethernet
2	Processos de produção	BD relacionais, MES	TCP/IP
3	Estratégia gerencial de negócios da indústria	ERP, CRM, SCM	TCP/IP

É verificado que não há uma padronização no formato ou na metodologia de troca de informações entre cada uma destas camadas. Há uma diferença tecnológica muito grande entre a sintaxe e a semântica dos dados utilizados para a supervisão (camada 2) e a utilizada pelos dados de sistemas de nível gerencial (camada 3). Esta diferença causa sérios problemas referentes à correta interpretação de informações, além de representar mais uma camada de implementação necessária, com gastos adicionais em tecnologia de informação.

Esta falha de integração tentou ser resolvida através do conceito de sistemas MES, que se interporiam entre as camadas de supervisão, processos e gerência, fornecendo soluções específicas para as necessidades das empresas [6].

2.1 Sistemas SCADA

Acrônimo para a expressão *Supervisory Control And Data Acquisition*, SCADA vem a definir uma série de soluções de equipamentos, comunicação e softwares para supervisão e controle de plantas onde certas limitações de qualidade e complexidade devem ser respeitadas. Dessas limitações são derivados critérios a serem avaliados antes da implantação de um sistema SCADA para impedir que as necessidades de supervisão da planta coincidam com os pontos fracos desses sistemas, restringindo as situações onde podem ser utilizados.

Todo sistema SCADA é formado por módulos que servem para cumprir as funções de buscar dados de uma planta, transportá-los para um local centralizado, tratar esses dados e disponibilizar a informação resultante para os operadores. Esses módulos são o MTU (*Master Terminal Unit*), o RTU (*Remote Terminal Unit*) e o Subsistema de Comunicação.

2.2 Aplicabilidade dos sistemas SCADA

Sistemas SCADA são adequados para uso em plantas onde três características básicas possam ser explicitadas: a planta se espalha por uma área muito grande; o controle e o monitoramento é relativamente simples; intervenções sobre a planta são necessárias, tanto sob certa regularidade quanto sob requisição imediata.

A característica de plantas espalhadas por uma vasta área geográfica vem a ser um dos principais motivadores para o uso de SCADA. É normalmente onde os maiores ganhos no uso desses sistemas são explicitados. A grande dificuldade relacionada ao monitoramento desse tipo de planta é a necessidade de pesados investimentos em estruturas que suportem a presença humana em suas instalações. Tais gastos podem se tornar ainda mais significativos se o tempo necessário para identificar uma falha dentro dessa planta for relativamente

pequeno para evitar perdas. A necessidade de supervisão desse tipo de planta exigiria organização e pessoal suficiente para elevar os custos de manutenção a ponto de inviabilizar o investimento. Nessa situação, o uso de sistemas SCADA é adequado porque o investimento nesse tipo de solução é sempre muito menor que o necessário para que pessoas façam esse trabalho manualmente. Quanto maior a distância entre o processo supervisionado ou controlado e a central de operação, maiores serão os benefícios explicitados no uso de SCADA [7].

A característica de controle e monitoramento relativamente simples já traz em si uma abertura para ampliação do leque de processos onde sistemas SCADA podem ser utilizados. O termo “relativamente simples” tem se tornado mais abrangente à medida que as tecnologias associadas a esses sistemas têm amadurecido. As melhorias alcançam todos os principais módulos constituintes desses sistemas, mas principalmente o MTU. O uso acentuado de computadores nos módulos concentradores de informação ampliou a possibilidade de expansão do sistema, permitindo que algoritmos mais complexos de controle sejam realizados por sistemas SCADA.

Esses sistemas também são apropriados para plantas onde a necessidade de interação é regular, freqüente ou mesmo sob demanda do operador, sem uma programação específica. Deve-se levar em consideração que o tempo de atraso na apresentação das informações adquiridas da planta para o operador do sistema não deve ser significativa a ponto de inviabilizar a possibilidade de ação. O sistema de supervisão mal dimensionado pode ser fonte de prejuízos para a planta e para as pessoas que a operam. Portanto, sistemas SCADA só podem ser utilizados nas plantas onde suas restrições de tempo para apresentação de informações não afetem o controle.

2.3 Funcionalidades dos sistemas SCADA

Os sistemas SCADA costumam ser confundidos com outras aplicações de controle moderno que prestam alguns serviços semelhantes, mas não alcançam a amplitude de funcionalidades necessárias para servirem como ponto centralizado de supervisionamento e controle ou são muito mais amplos do que o necessário.

Um dos sistemas que fornecem parte das funcionalidades dos SCADA são os DCS (*Distributed Control Systems*). Os DCS são sistemas especialistas que mantém conexões permanentes com seus itens de supervisão, causando uma maior ocupação dos canais de comunicação do que seria necessário para a supervisão. Nos sistemas SCADA, as conexões com os elementos de campo (RTU) são intermitentes; há uma varredura periódica e um formato definido para troca de mensagens.

Os sistemas de telemetria também fornecem parte da funcionalidade necessária para a supervisão remota. Mas tais sistemas apenas buscam informações da planta. São sistemas de simples leitura, nos quais a interação se limita à observação dos dados recolhidos pelo sistema. Em sua origem, esses sistemas cumpriram um papel fundamental para a observação de plantas onde a presença humana seria inviável. Mas as necessidades de alteração dos parâmetros de funcionamento do processo automaticamente ou sob a decisão daqueles que o acompanhavam causou a pressão necessária para a evolução desses sistemas para os modernos sistemas SCADA. A grande vantagem desses sistemas está no fato de possuírem comunicação em mão dupla, permitindo que a planta seja influenciada pelo controle programado na unidade central ou pela decisão de um operador remoto.

Fazem parte das características de sistemas SCADA as funções de sistemas RTS (*Real Time Systems* ou Sistemas de Tempo Real) e de sistemas *batch*. Na prática, considera-se que o

“tempo real” seja apenas representado por atrasos que não causem problemas de controle, sendo esses atrasos medidos entre a recepção de uma medida e a saída do sinal de controle. Um exemplo são os alarmes, onde a questão a ser considerada não é a de disponibilização da informação o mais rápido possível, mas sim a disponibilização de informação economicamente realizável. Os sistemas SCADA também fornecem funcionalidades de sistemas *batch* a fim de possibilitar sua utilização como ponto de programação do controle automatizado. Nesse caso, define-se a planta ou processo sob supervisão a partir de outro ponto de vista, modularizados a partir da relação de produtos e insumos que, respectivamente, fornecem ou recebem. São definidos procedimentos que, executados dentro de uma ordem pré-definida, transformarão os insumos recebidos nos produtos finais a serem entregues. Também são preparados os relatórios que indicarão o bom funcionamento da batelada de operações iniciada e gerida pelo sistema supervisório. Estas funcionalidades normalmente são fornecidas a partir da inclusão de procedimentos diretamente codificados em estruturas de execução interna nos sistemas SCADA, também conhecidos como *scripts*, que permitem estruturar o relacionamento entre os diversos módulos e procedimentos definidos.

Criados para executar controles automáticos sobre plantas industriais, os sistemas SCADA modernos estão agregando toda a funcionalidade de provedor de controle remoto. Atualmente, há uma série de funcionalidades que não podem faltar em um sistema para que o mesmo seja considerado um SCADA. São eles: aquisição de dados da planta; fornecimento de um subsistema de comunicação com os elementos de campo através de *drivers* específicos ou de padrões internacionais de comunicação; provimento de parametrização de seu funcionamento na supervisão da planta ou processo; controle automático de plantas ou processos simples, através de algoritmos de controle; apresentação de informações em tempo

hábil para que o operador tome decisões; manutenção de históricos sobre a planta ou processo supervisionado; apresentação e gerenciamento de alarmes [7].

2.4 Criticidades dos sistemas SCADA

É essencial que haja uma quantidade mínima de dados apresentados nos sistemas SCADA para ampliar o apoio à tomada de decisões tanto pelos operadores do sistema quanto pela gerência responsável pela produção. A maximização desta quantidade de informações, apesar de representar um significativo aumento na qualidade das informações para a tomada de decisões, requer o gasto adicional de recursos. Este gasto adicional pode chegar a um nível tal que ocorra o comprometimento da qualidade dos dados adquiridos, uma vez que as restrições de tempo comecem a ser alcançadas devido a limitações de recursos.

As diversas operações que devem ser realizadas pelos sistemas SCADA causam a necessidade de agregação de recursos aos sistemas computacionais. Quando se usa *drivers* específicos para realizar a aquisição, o problema pode aumentar em proporções não lineares, pois muitas vezes a única forma de fazer acesso aos elementos de supervisão é aumentando a quantidade de hardware de entrada / saída, memória e processamento do equipamento onde o sistema SCADA estiver funcionando. Com isto, chega-se a um dos grandes problemas de relação de custo x benefício das soluções para supervisão de sistemas industriais. É necessário ampliar a aquisição de dados em quantidade e qualidade, mas ao mesmo tempo reduzir o consumo dos recursos de comunicação e processamento, disponibilizando informações dentro de um padrão de qualidade economicamente viável. Essa é uma das situações mais críticas combatidas atualmente pelos sistemas SCADA e que motivam o desenvolvimento de padrões de interoperabilidade [8].

Um outro fator crítico para os atuais SCADA é a construção de sistemas mais robustos que suportem a redução da centralização. Antigamente, os centros de controle eram quase sempre representados por uma grande sala para onde convergiam os fios de transporte de informação, terminando em painéis de controle. A interface eletro-mecânica desses painéis fornecia à equipe de supervisão uma série de informações codificadas a respeito da planta sob controle. Uma vez definidas e criadas essas pesadas estruturas não podiam sofrer alterações substanciais, uma vez que os custos associados à sua modificação inviabilizavam qualquer tipo de projeto. Além disso, a idéia de controlar todo o processo a partir de um único local demonstrava ser uma solução para uma realidade onde uma parcela muito pequena de atuadores era capaz de processar os dados que estavam recuperando do processo. Os sistemas SCADA iniciais partiam desse mesmo pré-suposto, fornecendo implementações que privilegiavam essa abordagem na construção de soluções.

Com o tempo surgiram demandas para que o processamento de dados não fosse feito muito distante das áreas sob supervisão e controle, sob pena de perdas significativas na planta. Foram desenvolvidos os primeiros Controladores Lógicos Programáveis (CLPs), que permitiam regionalizar o processamento de informações e automatizavam alguns controles sobre a planta supervisionada. Também se iniciou forte trabalho para incluir capacidade de processamento local nos próprios elementos de campo. Atuadores e sensores inteligentes tornaram-se muito mais comuns nas plantas industriais, alterando o contexto onde os sistemas SCADA deveriam atuar [2 e 9]. Junto a isso, o desenvolvimento de microcomputadores permitiu descentralizar ainda mais a supervisão e o controle. Esses fatores em conjunto criaram uma demanda para o aumento da qualidade no desenvolvimento de sistemas SCADA, que deveriam ser criados sob as premissas de uma nova realidade, agregando capacidade de expansão, modularização e componentização.

3 Paradigmas para desenvolvimento de sistemas

O desenvolvimento de um software é regido por critérios e condições extremamente diferentes dos que influenciam a produção de hardware ou qualquer produto manufaturado. Ainda há muita incerteza sobre os fatores que influenciam seu desenvolvimento, tanto por causa da inexperiência em se lidar com tal tipo de tecnologia, nascida a menos de 60 anos, quanto por causa da sua inerente necessidade de abstração, uma vez que não é possível interagir fisicamente com um software de maneira comum.

A medida que os pesquisadores foram se aprimorando no conhecimento dos softwares, uma série de paradigmas foram criados para nortear seu desenvolvimento e sua manutenção. Dois desses são de extrema relevância, sendo descritos nesse capítulo.

3.1 Paradigma Cliente-Servidor

Um dos marcos associados ao uso dos modernos PCs (*Personal Computers*) foi a possibilidade de ligá-los através de um subsistema de comunicação para permitir a troca de dados e informações entre vários computadores. Com o tempo, percebeu-se que algumas funcionalidades específicas, como serviço de disponibilização de arquivos ou processamentos de algoritmos pesados, poderiam ser executadas de forma mais eficiente e a um custo menor se computadores especializados as realizassem. Nascia a idéia de servidores, equipamentos que proveriam funcionalidades específicas por requisição de uma aplicação. As aplicações que faziam uso desses servidores são denominadas clientes. Os clientes normalmente se comunicam com os servidores realizando requisições e obtendo respostas (figura 3.1) através de um subsistema de comunicação que funciona sobre uma rede. Essa é a base do Paradigma

Cliente-Servidor, ou seja, a divisão funcional dos sistemas e equipamentos sobre uma arquitetura de comunicação.

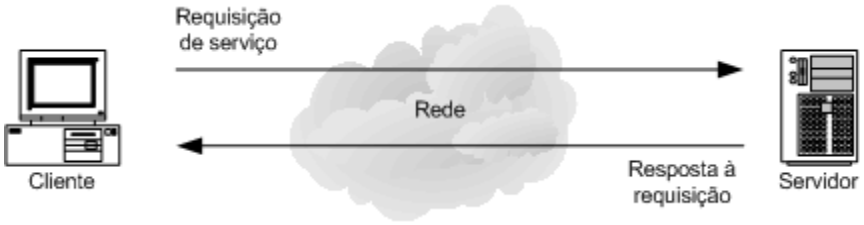

Figura 3.1Paradigma Cliente-Servidor

Nesse paradigma também é possível abstrair a divisão funcional dos equipamentos de acordo com as aplicações que utilizam. Dessa forma, um servidor de uma determinada aplicação pode ser o cliente no contexto de outra aplicação, de forma que um equipamento pode acumular as características de cliente e de servidor. Há também uma organização hierárquica baseada nos servidores de forma a facilitar a identificação dos equipamentos e de suas funções. Ganha-se com isso em escalabilidade, facilitada pela possibilidade de adição de novos servidores, e no desempenho dos sistemas, pela especialização dos equipamentos definidos para executar certos processamentos.

3.1.1 Infra-estrutura para criação de sistemas cliente-servidor

Os sistemas cliente-servidor devem se embasar em uma arquitetura que permita a comunicação entre diversos dispositivos computacionais. Para fornecer esse tipo de funcionalidade, foi definida uma série de modelos de estrutura de redes. O mais importante desses modelos é o OSI (*Open Systems Interconnection*). Este modelo divide uma rede em camadas abstratas responsáveis pela execução de serviços específicos. Uma camada só pode se comunicar com a camada subseqüente, a qual lhe fornece os serviços de que necessita para

transmitir dados de um computador para outro. A figura 3.2 mostra a divisão de camadas definida pelo modelo OSI [10].

Para cada camada do modelo há pelo menos um tipo de protocolo definido para executar os serviços especificados para a camada. Alguns desses protocolos oferecem serviços mais importantes do que outros, causando impacto direto sobre o desenvolvimento de sistemas cliente-servidor. Dentre estes, citam-se os protocolos da camada de rede (*network layer*), de transporte (*transport layer*) e de aplicação (*application layer*). Um protocolo da camada de rede, conhecido como IP (*Internet Protocol*), oferece as funções de endereçamento e de roteamento que permitem que os fragmentos de dados trafeguem de um computador para outro. Esses serviços são aproveitados pelos protocolos da camada de transporte, que se abstraem da forma de transmissão para criar conexões entre os computadores identificados como origem e destino dos dados, ou servidor e cliente de uma aplicação. Além disso, esses protocolos se preocupam em reordenar os fragmentos de dados recebidos da camada anterior para remontar o pacote completo de dados a serem entregues à próxima camada. Um protocolo muito conhecido dessa camada é o TCP (*Transmission Control Protocol*). Já na camada de aplicação, são definidos os paradigmas que orientam a programação de sistemas. Características como a segurança de acesso e autorização de um sistema, a divisão funcional e mesmo a correta utilização da infra-estrutura de comunicação são definidas nos protocolos dessa camada. É nessa camada que as características de modularização e componentização devem ser consideradas [11].

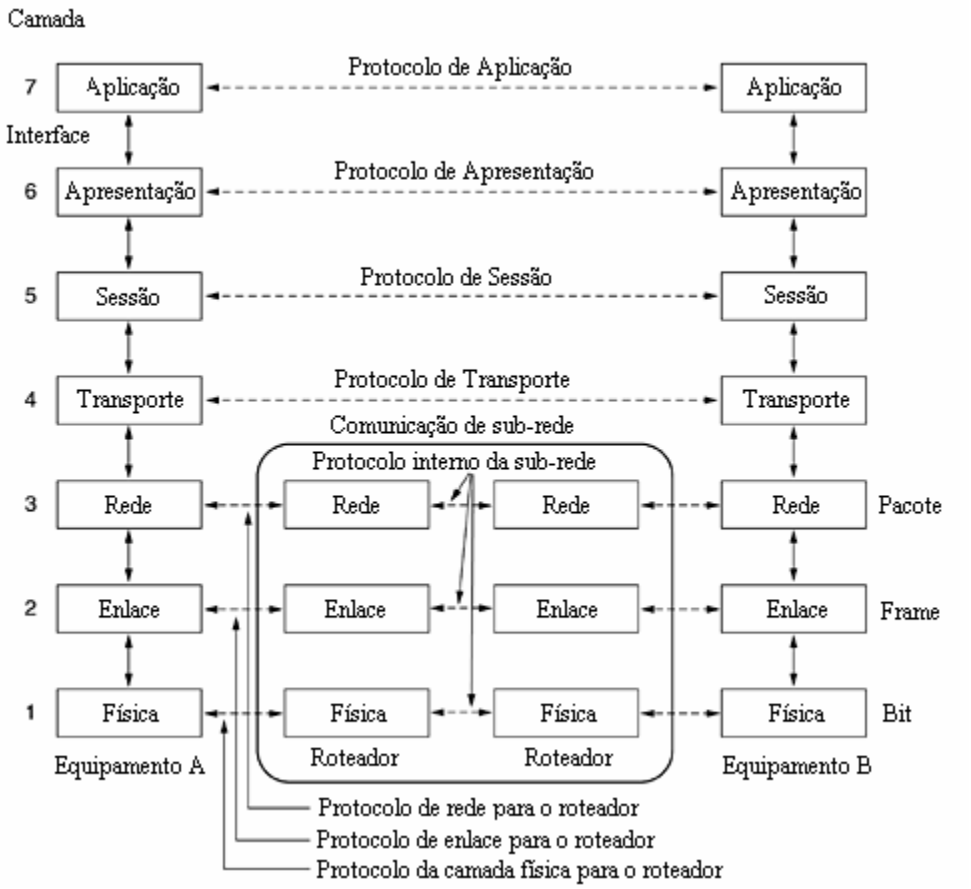

Figura 3.2 Modelo de Referência OSI

3.1.2 Modularização de sistemas

Para que as vantagens do uso do Paradigma Cliente-Servidor possam ser auferidas, é necessário que os sistemas sejam desenvolvidos a partir de uma nova ótica. Assim como ocorre a divisão funcional de servidores, os programas também devem ser divididos em pacotes especialistas, independentes entre si e que executem tarefas específicas. Os sistemas modularizados oferecem a vantagem de serem projetados por funcionalidade, o que aumenta sua manutenibilidade, ou seja, reduz o esforço associado a correções e alterações nas funcionalidades desse sistema [12].

O principal modelo de modularização de sistemas é o Modelo de Três Camadas (figura 3.3). Neste modelo, um sistema deve ser dividido em camadas para atender as necessidades específicas de apresentação, com interfaces para o usuário; necessidades de aplicação, com a execução da lógica de negócio definida para o sistema; e necessidades de gerenciamento de dados, através de acesso às informações mantidas, normalmente, em bancos de dados ou servidores de arquivos. Por muito tempo esse modelo norteou o desenvolvimento de sistemas de informação e gerou uma série de novos modelos que sofisticavam os procedimentos de interconexão entre essas camadas [2 e 12].

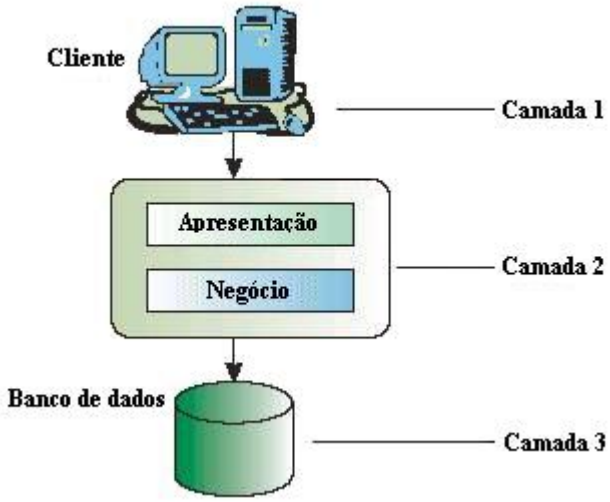

Figura 3.3 Típica divisão de sistemas de informação modularizados

No entanto, há sistemas que não se caracterizam por prover informação, mas que demandam diferentes tipos de modularização não previstos nesse modelo. Surgem então os modelos de desenvolvimento de sistemas em N camadas, onde o foco da modularização abrange todos os serviços de um sistema que possam ser separados de acordo com o tipo de recurso ou de subsistema que se trabalhe.

3.1.3 Componentização

Enquanto os sistemas evoluíam em direção à modularização, uma outra questão também teve que ser resolvida. Certos problemas computacionais devem ser resolvidos por complexos algoritmos de processamento, gerando por vezes um gasto significativo dos recursos computacionais disponíveis. Cada sistema era feito em formato monolítico, o que significa dizer que todas as funcionalidades que esse sistema poderia gerar pertenciam a um único executável. Se mais do que um sistema precisasse fornecer uma mesma funcionalidade, esta deveria ser implementada em cada um desses sistemas. O retrabalho associado a essa abordagem tornou-se crítico. A solução adotada para mitigar esse problema foram os componentes. Um componente é um trecho de programa mantido em uma estrutura disponível para uso por várias aplicações diferentes simultaneamente, dentro de um mesmo domínio de informação [13].

Os precursores dessa abordagem na plataforma Microsoft Windows foram as bibliotecas de ligação dinâmica, ou DLL na sigla em inglês. Essas bibliotecas permitiam a um sistema não possuir em seu próprio código fonte o algoritmo a ser executado, chamando esse código apenas quando fosse utilizá-lo. Vários sistemas de informação adotaram a arquitetura de componentes (figura 3.4) para resolver as questões de gasto desnecessário de recursos.

Além de auxiliar na questão de gastos de recursos computacionais, os componentes também forneciam a solução para uma necessidade crescente: a interoperabilidade. Entende-se por interoperabilidade a característica de um sistema poder interagir com outros sistemas que não estejam funcionando no mesmo escopo de memória, seja para troca de informações ou para realizar o controle programático de um sobre o outro. Há muitas razões pelas quais esse tipo de comunicação é necessário, como coordenar as ações de dois sistemas distintos de

acordo com alguma informação contida em um deles. As primeiras tentativas de fazer com que sistemas interagissem com outros sistemas tinham como objetivo apenas controlar o fluxo de vida desses sistemas, automatizando o gerenciamento dos diversos sistemas que funcionariam em um computador. A medida em que os aplicativos de escritório se popularizavam, tornava-se claro que a troca de informações entre esses aplicativos era uma funcionalidade de suma prioridade. Investimentos crescentes foram feitos para criar tecnologias que permitam essa troca de informações, de forma que hoje é possível escolher entre uma série de diferentes implementações criadas com esse fim.

Para ampliar os ganhos associados à componentização de sistemas, passou-se a agrupar componentes de acordo com a funcionalidade que forneciam. Esses componentes agrupados podem estar no mesmo computador onde a aplicação que o utiliza está, ou em um computador remoto na rede. Nesse caso, são considerados componentes distribuídos.

A distribuição de componentes através da rede esbarra em problemas sérios de operabilidade, uma vez que não é possível garantir em que linguagem o componente distribuído foi escrito, em que tipo de rede ele se encontra, nem qual a arquitetura que o PC servidor possui. Para mitigar tais problemas e permitir que a troca de informações entre diferentes equipamentos em uma rede se tornasse viável, foi feita uma iniciativa de padronização de dois tópicos essenciais para a distribuição. O primeiro tópico é a transformação de parâmetros de uma representação dependente de plataforma para uma independente de plataforma, conhecida como *marshalling/demarshalling* [12]. O segundo é o endereçamento de métodos em aplicações [11]. Uma série de especificações para resolver essas questões foi desenvolvida a partir de uma iniciativa internacional conhecida como OSF (*Open Software Foundation*). Essas especificações são conhecidas como DCE RPC (*Distributed Computing Environment / Remote Procedure Call*).

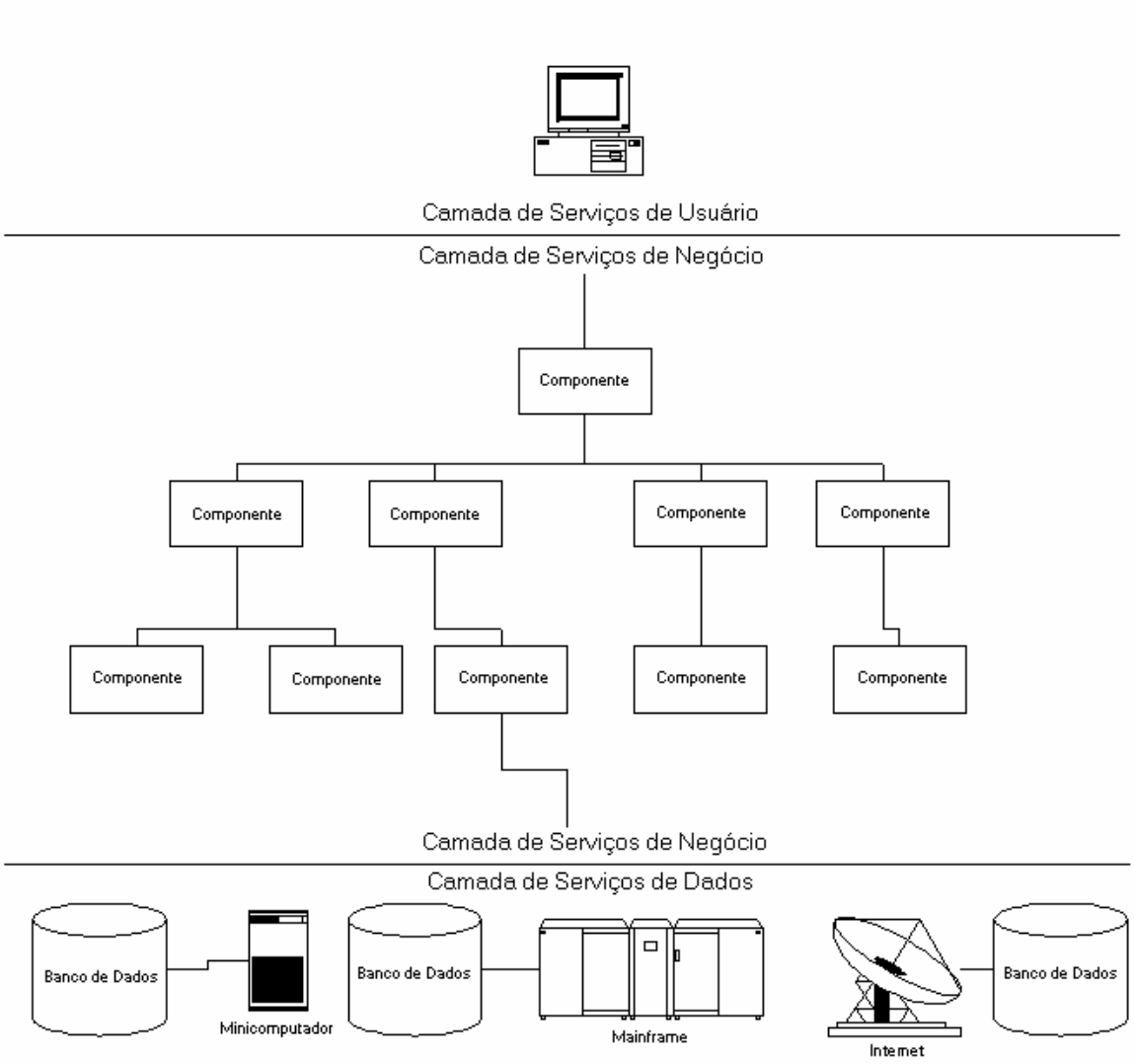

Figura 3.4 Sistema modularizado e componentizado

Algumas soluções de mercado seguiram tais especificações para gerar estruturas de desenvolvimento de softwares baseados em componentes. As mais conhecidas no meio industrial são o CORBA, o EJB e o COM/DCOM [14]. As três soluções oferecem mecanismos de modularização e de componentização requeridos para a distribuição de componentes, além de oferecerem suporte a todos os princípios requisitos que viabilizam a criação de sistemas distribuídos. A listagem abaixo apresenta os princípios básicos para a distribuição de componentes, de acordo com [14].

- *request/response*: a possibilidade de tratar requisições e enviar respostas;
- referência remota: a capacidade de identificar em uma rede o componente distribuído cuja funcionalidade será utilizada;
- interface IDL (*Interface Definition Language*): capacidade de mapear para uso em uma linguagem de programação definições criadas em outra;
- *proxy*: representação local de uma referência remota;
- *serialização*: capacidade de converter informações em um formato que possa ser transmitido através da infraestrutura de comunicação.

3.2 Disponibilização de informações na Internet

Hoje há pelo menos dois motivos que direcionam o desenvolvimento de sistemas SCADA para a Internet, fora dos canônicos modelos cliente-servidor desenvolvidos em DCOM e outras tecnologias similares. O primeiro é a impossibilidade de uso do principal padrão de desenvolvimento de componentes do mercado (COM/DCOM) em redes resguardadas por *firewalls* devido à natureza das chamadas RPC utilizadas. Desde o século passado, uma série de tentativas têm sido feitas para encapsular essa tecnologia por *wrappers* para outros protocolos livres dessas restrições [15], sem aceitação pela indústria devido ao baixo desempenho desse tipo de solução. O segundo motivo, e mais importante, é a rápida expansão de um novo paradigma de desenvolvimento de sistemas baseado em sistemas *Web* que fazem uso da Internet e dos padrões definidos para os *Web Services*.

As principais vantagens do uso da Internet para ter acesso a informações repousam nos fatores custo para obtenção de informações e facilidade de transformação dessas informações

através desse meio. Qualquer que seja a solução adotada para supervisão e controle escolhida, ela seguirá os padrões definidos para o tipo de planta supervisionada. Haverá uma sala de controle central para a qual todas as informações devem migrar em algum momento. Uma série de condições de redundância deve ser prevista e os equipamentos necessários para que a mitigação da contingência funcione devem ser alocados e subutilizados até o momento em que uma falha nos equipamentos principais obriguem seu uso. Ao se utilizar tecnologias abertas da Internet, ou seja, não proprietárias, as premissas de disponibilidade do sistema de supervisão podem ser implementadas de forma menos onerosa. Em algumas situações, o custo para tornar uma informação disponível pode indicar se um projeto pode ou não ser levado adiante. Em [16] é apresentado um exemplo claro de como a adoção de tecnologias abertas de comunicação tornou viável a pesquisa de equipes de estudos geofísicos.

Quanto à disponibilidade de acesso às informações e facilidade de transformação, há uma série de diferentes iniciativas para permitir que a Internet seja utilizada como meio de comunicação. Há tentativas de utilização de *gateways* entre antigas tecnologias e protocolos definidos para Web, como o XML, para integrar plantas fabris desde sua execução até os meios de decisão gerencial [17]. Também foram realizados trabalhos de supervisão de subsistemas hospitalares baseados em *gateways* para tecnologias SOAP [18]. A evolução dessas iniciativas passa pela estabilização da tecnologia conhecida como Web Services. Identificado pelos grupos de trabalho do OPC Foundation como uma das mais importantes tecnologias de comunicação de dados via Internet, os Web Services já alcançaram um nível de qualidade tal que permite sua aplicação em áreas críticas como a saúde pública [19 e 16], onde o acompanhamento de recursos de hospitais e até mesmo o monitoramento das condições dos pacientes já pode ser feito com segurança. Essa tecnologia se tornou também

foco de trabalhos na área de automação industrial, onde plantas e processos críticos podem ser supervisionados a partir de uma infra-estrutura mais portável e integrável.

3.3 Paradigma de sistemas Web

A evolução dos sistemas de supervisão e controle passou por dois estágios muito distintos entre si. O primeiro estágio foi a centralização total da informação em pesados sistemas altamente redundantes, específicos para um processo ou uma planta e sob restrições de operação de tempo real. As tecnologias associadas a esse estágio eram proprietárias e baseadas em ligações físicas entre os sistemas de supervisão e os sensores e atuadores de campo. O segundo estágio foi uma flexibilização desses sistemas. A centralização já era feita apenas em condições onde realmente necessária, em sistemas de baixa redundância, utilizando alguns padrões internacionais para comunicação e as restrições de tempo se transformaram em um atraso de resposta que não comprometesse o controle do processo. Há sistemas de supervisão dos dois estágios coexistindo atualmente, mas aplicados a diferentes situações de controle. Mas a evolução ocorreu em maior intensidade para os sistemas que adotaram o modelo cliente-servidor para operação.

O uso da Internet evolui o modelo cliente-servidor. Nesse modelo, os sistemas não precisam manter softwares específicos instalados nos computadores dos clientes. Devido a padronização dos protocolos de transferência de dados, exibição de informações e requisições para o servidor, reduz-se a quantidade de sistemas que devem ser mantidos no equipamento cliente. Essa característica é uma das principais responsáveis pela vantagem de uso da Internet como meio de acesso a sistemas de automação.

A Internet possui uma arquitetura baseada em canais públicos de comunicação, padrões internacionais de disponibilização de informações, além de abstração com relação ao conteúdo dos dados e informações que trafegam, gerando naturalmente um ambiente flexível e distribuído [2]. A mais conhecida forma de comunicação da Internet é a busca de páginas Web, exibidas a partir de *Web Browsers* instalados na maioria dos sistemas operacionais. O mais interessante com relação a esse paradigma é a natureza de interoperabilidade entre plataformas. Existem vários outros tipos de aplicações que podem ser disponibilizadas na Internet. Há muitos sistemas que utilizam trocas de mensagens eletrônicas (correios eletrônicos), sistemas de vendas através da Web e acesso a sistemas bancários. É crescente o aprimoramento das tecnologias nas quais essas aplicações se baseiam, uma vez que há uma tendência de institucionalização desse meio para a prestação dos mais diversos serviços, sejam eles entre empresas (*busines-to-business*) ou entre empresas e seus clientes (*business-to-costumer*). Novos padrões além dos três citados anteriormente provêm as convenções necessárias para que esse aprimoramento ocorra.

3.3.1 Organizações fornecedoras de padrões

Para evitar que o crescente uso da Internet como meio de comunicação entre aplicações se tornasse um emaranhado de especificações proprietárias que inviabilizassem a interoperabilidade, algumas organizações internacionais passaram a investir esforços para definir padrões que norteiem a criação de sistemas. As três principais são o W3C, o OASIS e a *Liberty Alliance*.

O W3C (*World Wide Web Consortium)* é um consórcio internacional com a missão de desenvolver padrões e guias para a Web a partir das organizações interessadas, do público e de uma equipe de coordenação fixa. Seus objetivos são a manutenção da Internet como uma

fonte de informação e de conhecimento que possa alcançar qualquer um, em qualquer lugar, com a segurança e confiabilidade necessárias para que mais sistemas possam ser utilizados através desse meio.

O principal foco de atuação dos produtos gerados pelo W3C é a interoperabilidade promovida pelos protocolos não-proprietários voltados para a infra-estrutura básica de funcionamento da Internet, além da arquitetura das principais tecnologias de troca de informação. Apesar de não impositivas, as recomendações apresentadas pelo Consórcio através de RFC (*Request For Comments*) são encaradas como padrões para uso em sistemas Web, assim como para outros domínios de desenvolvimento científico que se associam à Internet.

O OASIS (*Organization for the Advancement of Structured Information Standards*) também é formado por uma série de empresas que buscam a padronização de sistemas desenvolvidos para a Internet dentro de tecnologias muito mais específicas. Tratam de padrões para segurança de informação, *e-business* e aplicações de mercados específicos, tendo nos Web Services o seu principal foco.

A *Liberty Alliance* possui os mesmos objetivos do OASIS, mas formado por um grupo de empresas diferentes e observando também as necessidades de sistemas não baseados no mercado.

3.3.2 Padrões e protocolos criados para a Internet

Uma rede mundial de computadores não poderia funcionar adequadamente se não fosse criada junto a uma série de padrões que permitam a troca de informações entre si. Há uma diversidade incontável de equipamentos e sistemas que fazem uso da Internet, de forma

que a falta de padrões poderia inviabilizar a comunicação. Grande parte do sucesso da Internet se encontra na simplicidade associada aos padrões e aos protocolos criados para prover uma série de funcionalidades necessárias à transferência e à exibição de informação, como as páginas WWW (*World Wide Web*). Cada um desses padrões ou protocolos pode ser classificado de acordo com as funções que desempenham.

Sem dúvidas, os padrões criados pelo W3C são os mais respeitados no mundo da Internet. Suas definições atingem a completude das necessidades para que aplicações inteiras funcionem sobre a Rede Mundial. Apesar de mais de cinqüenta padrões ou protocolos definidos, será mantido o foco sobre os principais utilizados para a definição arquitetural de sistemas Web e para o uso de Web Services.

HTTP

Acrônimo de *HyperText Transfer Protocol*. É um protocolo de aplicação, que provê uma forma padronizada de envio de mensagens e de respostas a serem retornadas. As definições desse protocolo são feitas na RFC 2616. Encontra-se atualmente na versão 4.01.

HTML

Acrônimo para *HyperText Markup Language*. Trata-se de uma linguagem de escrita de documentos, definindo sua formatação simultaneamente à inclusão de seu conteúdo. Permite que sejam definidas ligações com outros documentos, assim como a inclusão de outros tipos de mídia em seu corpo, como figuras, outros documentos e aplicações. É o principal padrão de escrita de documentos existente na Internet e sua interpretação é obrigatória para qualquer *browser* comercial.

URI (ou URL)

URI é o acrônimo de *Uniform Resource Identifier*. Também pode ser encontrado com o nome de URL (*Universal Resource Locator*), possuindo o mesmo conceito. Protocolo que define a forma como documentos são localizados na Internet, a forma como se pode ter acesso aos mesmos e o tipo de documento representado. A formalização das ligações entre documentos também é definida por esse protocolo, que através de um endereço Web consegue indicar a informação necessária para que se encontre inequivocamente um certo documento ou serviço através da Internet.

XML e XSL

Acrônimos, respectivamente, de *eXtensible Markup Language* e *eXtensible Style Language*. Evolução do padrão semântico do HTML, essas duas tecnologias vêm oferecer uma separação entre a estrutura de conteúdo e a estrutura de formatação de um documento. Enquanto todos os dados do documento, seu conteúdo, mantêm-se no XML, no XSL ficam as definições de formatação que deverão ser aplicadas no documento. Com isso, é possível aumentar a sua própria capacidade de representação sem a necessidade de inclusão de notações proprietárias. Sua independência com relação a plataformas fornece um dos principais requisitos para permitir interoperabilidade entre sistemas de diferentes plataformas [20].

SOAP

Acrônimo de *Simple Object Access Protocol* ou de *Service Oriented Architecture Protocol* [21]. Esta tecnologia une as definições de transporte de informação via HTTP com a capacidade de representação de informação do XML. É definido um envelope padronizado

para o encapsulamento e troca de mensagens XML. Usa-se um endereço URI para identificar o destino e o serviço prestado.

Web Services

Este é o nome com que genericamente são identificadas aplicações Web que seguem os preceitos definidos pelo SOA (*Service Oriented Architecture*). Provê em sua implementação os requisitos mínimos de comunicação, apresentação de funcionalidades e independência de plataforma, ampliando a capacidade de interoperabilidade de sistemas Web.

WSDL

Acrônimo de *Web Services Definition Language*. É a forma encontrada para publicar os serviços prestados por um determinado Web Service. Mantém a informação sobre a assinatura com que uma requisição deve ser feita, assim como o resultado esperado para o serviço prestado. Por ser uma linguagem de descrição de serviços, também é independente de plataforma.

A despeito da importância de todas as especificações definidas pelas organizações internacionais, alguns padrões são básicos para prover a distribuição de aplicações através da Rede Mundial. Atualmente, os Web Services são o principal meio de prover os requisitos de comunicação necessários para que um sistema SCADA possa ser disponibilizado através da Internet.

3.3.3 Infra-estrutura necessária para uso de aplicações na Internet

Vários serviços já são atualmente fornecidos através da Internet. Esses serviços possuem seus protocolos próprios para executar as funcionalidades que se propõem a

fornecer. De modo geral, todos eles devem respeitar algumas definições arquiteturais para que funcionem sobre a Internet. Em linhas gerais, é necessário localizar o serviço, executar o transporte das informações necessárias para a prestação do serviço e combinar o padrão semântico e de formatação a ser utilizado. Um exemplo bem simples de como essas definições são utilizadas é o acesso a páginas Web. O protocolo URI é utilizado para localização da informação, o HTTP é utilizado para realizar a troca de arquivos entre o cliente e o servidor e o HTML é utilizado como padrão de escrita do arquivo disponibilizado (semântica e forma), de forma que qualquer *web browser* consiga recuperar e exibir a página requisitada pelo usuário (figura 3.5).

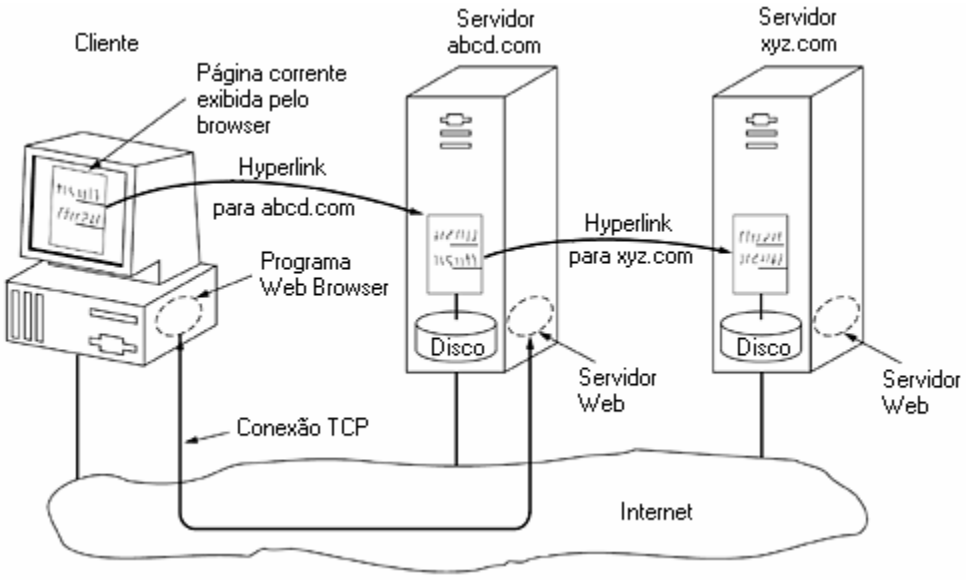

Figura 3.5 Estrutura para funcionamento de Web Browsers

Alguns novos itens passam a ser considerados para que o sistema funcione como um todo. O mais importante protocolo utilizado pela Internet é o TCP/IP (*Transmission Control Protocol Over Internet Protocol*). São os protocolos que implementam as camadas de transporte e de rede mais utilizadas pelas redes onde a Internet se distribui.

Além desses protocolos, deve ser levado em consideração que os equipamentos que suprirão os serviços através da Internet devem prover tipos diferentes de aplicação, que permitam que os padrões e protocolos da camada de aplicação sejam adequadamente executados. A principal aplicação de um sistema Web é chamada Servidor Web, que disponibiliza informações através do protocolo HTTP. É a base para o uso de páginas HTML, que popularizaram o uso da Internet como é atualmente conhecida. Geralmente, os Servidores Web fazem parte de pacotes de aplicações que permitem que serviços sobre os demais protocolos também sejam fornecidos, como o FTP, para *download* de arquivos binários ou o SMTP, para troca de e-mails.

Para prover funcionalidades mais próximas ao conceito de componentização utilizado nos sistemas cliente-servidor comuns foram definidos os Web Services. Esses sistemas respeitam as definições de arquitetura Web convencionais, mas incluem mais alguns elementos para permitirem uma maior garantia de interoperabilidade. Alguns jargões são próprios dessa tecnologia. Quem fornece serviços é conhecido como provedor e aquele que faz uso dos serviços é conhecido como requerente requisitor [21]. Para que haja o correto funcionamento de um Web Service é necessário que alguns passos sejam tomados por essas entidades.

Primeiro, as duas entidades fazem-se conhecer uma pela outra. Este processo engloba o conceito de localização, mas adiciona um padrão de reconhecimento de funcionalidades providas além de indicar onde o serviço pode ser encontrado. Os protocolos utilizados para essas operações são o HTTP e o WSDL. Depois, o requisitor e o provedor concordam entre si sobre uma descrição semântica do serviço a ser utilizado. Nem sempre é necessário executar essa tarefa durante a prestação de serviços, uma vez que tais informações podem estar padronizadas com antecedência. O WSDL é utilizado como forma de realizar essa operação.

As descrições semânticas e de serviço são associadas tanto ao provedor quanto ao requisitor. Como não é feita nenhuma menção nos padrões, isso pode ser *hard coded a priori*. É importante ressaltar que sem a assimilação dessas descrições, não há como as entidades trabalharem em conjunto. O requisitor e o provedor trocam mensagens executando a requisição de serviços e a resposta de processamentos de acordo com a necessidade. O padrão utilizado para essa troca de mensagens é o SOAP [21].

Assim como para o exemplo das páginas Web, os Web Services necessitam de aplicações servidoras que forneçam a funcionalidade dos protocolos associados a sua implementação para que funcionem, ao menos junto ao provedor. Atualmente, as definições de segurança ainda não estão padronizadas, de forma que é possível adotar várias soluções diferentes para prover os requisitos necessários para garantir a segurança das aplicações que utilizam Web Services.

4 O padrão OPC

4.1 Evolução do OPC

No final da década de 80 e início da década de 90 os computadores se tornaram o principal meio a partir do qual o operador passaria a se comunicar com as plantas e processos supervisionados. Nessa época, a criação de HMI (*Human Machine Interface*) mais sofisticadas e o crescimento do número de protocolos de comunicação trouxeram consigo um novo problema, até então pouco abordado entre as empresas que forneciam sistemas de automação. A definição e a manutenção da comunicação entre seus sistemas de supervisão e controle e os diversos protocolos proprietários que se alastravam entre os equipamentos de campo faziam com que a maior parte do tempo e dos recursos dessas empresas passassem a ser gastos com *drivers* de comunicação [11].

Devido à criticidade dessa situação, um grupo de empresas especializadas em sistemas para automação industrial juntou forças para o desenvolvimento de uma solução que resolvesse esse problema. A esse grupo, juntou-se a Microsoft como fornecedora de uma tecnologia que possibilitasse a implementação das definições alcançadas, assim como suporte para a verificação de viabilidade técnica das soluções. O resultado desse trabalho foi o primórdio de uma especificação de acesso a dados em plantas industriais que originaria todos os demais padrões. Para essa nova especificação foi definido o OPC (*Object Linking and Embedding for Process Control*), e estabeleceu-se o OPC Foundation, um grupo internacional sem fins lucrativos formado atualmente por vários desenvolvedores e um *pool* de empresas, que direciona a evolução das especificações do padrão OPC.

Em sua origem, as especificações do OPC Foundation eram voltadas para sistemas SCADA e pretendiam agregar os conceitos de modularização e componentização em plena discussão à época. O grupo de trabalho escolheu a tecnologia DCOM (*Distributed Component Object Model)* como o meio de prover uma padronização que atendesse às necessidades discutidas. A medida que os trabalhos do OPC Foundation avançaram, as principais funcionalidades dos sistemas SCADA foram investigadas e modularizadas gerando especificações próprias para cada área de pesquisa vislumbrada. A figura 4.1 mostra as principais linhas de pesquisa em andamento.

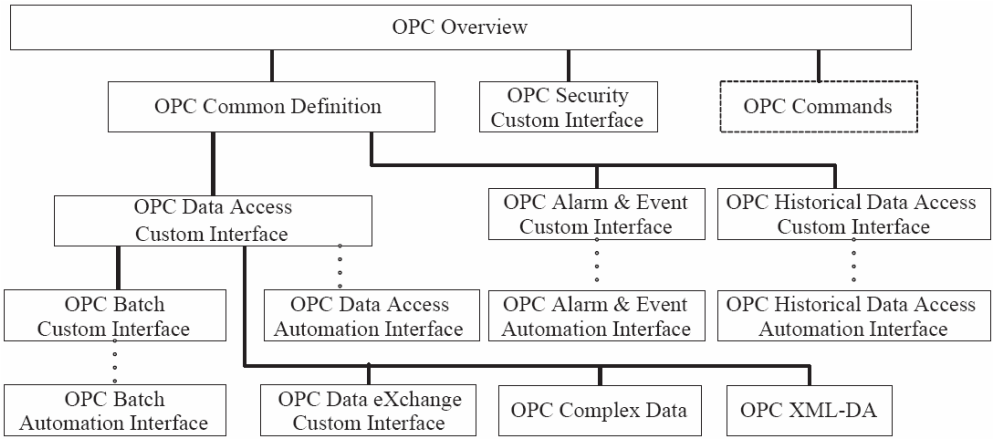

Figura 4.1 Áreas de pesquisa do OPC Foundation

Cada especificação apresenta uma padronização de mecanismos e interfaces diferentes para suprir as funcionalidades modularizadas, sendo que as especificações do OPC *Data Access* e do OPC XML *Data Access* são as principais abordadas nesse trabalho. A tabela 4.1 mostra as versões em que as especificações OPC se encontram.

Tabela 4.1 Especificações do OPC Foundation

Especificação	Versão
OPC Overview	1.00
OPC Common Definitions and Interfaces	1.00
OPC Data Access Specification	3.00
OPC Alarms and Events Specification	1.10
OPC Historical Data Access Specification	1.20
OPC Batch Specification	2.00
OPC Security Specification	1.00
OPC XMLDA Specification	1.01
OPC Data eXchange (DX) Specification	1.00
OPC Complex Data	1.00
OPC Commands Execution Interface Specification	0.29 (Draft)

Além das especificações, o OPC Foundation fornece também uma série de *redistributables*: pacotes de componentes a serem agregados aos servidores e clientes desenvolvidos usando as tecnologias padronizadas. Esses pacotes reduzem o trabalho a ser realizado pelos programadores, uma vez que oferecem as definições comuns que todo sistema deve seguir para manter a conformidade com os padrões.

4.2 Linhas de pesquisa

Os grupos de trabalho do OPC Foundation se dividem de acordo com o escopo de pesquisa. Algumas dessas linhas de pesquisa possuem maior relevância que outras quando avaliado o momento histórico ou as necessidades de um sistema supervisório. As primeiras especificações definidas foram a OPC *Common* e a OPC *Data Access*. A primeira indica as interfaces e definições de *background* que devem ser seguidas por todos os demais padrões, enquanto a segunda apresenta as interfaces necessárias para aquisição de dados instantâneos, a principal funcionalidade de um sistema SCADA.

As demais especificações apresentam características suplementares para o funcionamento de sistemas supervisórios. OPC *Alarms and Events* contém as interfaces que permitem ao usuário definir eventos a serem monitorados, classificar alterações de status e

indicar procedimentos para notificação. OPC *Historical Data Access* apresentam interfaces que permitem a definição de relatórios e análise de dados históricos do processo supervisionado. OPC *Batch* permite a definição de sequenciamento de processos e acesso a informações de batelada nos moldes do padrão IEC 61512-1 (padrão de modelos e terminologia). OPC *Security* informa uma maneira padronizada de utilizar as facilidades fornecidas pelos sistemas operacionais para garantir a integridade, autorização e autenticação dos componentes OPC desenvolvidos. OPC *Data eXchange* define interfaces que permitem a troca de informações entre dois ou mais servidores OPC. OPC *Complex Data* especifica como estruturas de dados compostas devem ser tratadas dentro do padrão OPC.

Atualmente, dois padrões estão no foco dos trabalhos do OPC *Foundation*: o OPC XML DA e o OPC *Commands*. No primeiro, são definidas as interfaces para criar um serviço de disponibilização de dados utilizando-se Web Services. No segundo, é definido como os métodos disponíveis em um servidor OPC podem ser descobertos, compreendidos e controlados, além de fornecer a interface necessária para iniciar e gerenciar a execução desses métodos. No site da OPC Foundation [22] também é possível verificar o atual trabalho na tentativa de fornecer um padrão de arquitetura unificado – OPC *Unified Architecture* – com o intuito de migrar os servidores OPC para uma arquitetura independente do DCOM. Atualmente, pelo uso do DCOM, esse padrão não é independente de plataforma.

4.3 Soluções alcançadas

A idealização dos padrões associados ao OPC nasceu da necessidade de integração de informações disponibilizadas para o usuário final a partir de diversos pontos de uma arquitetura de controle industrial. O OPC Foundation ainda mantém informações necessárias para averiguação de compatibilidade de servidores e clientes OPC desenvolvidos.

Uma importante decisão tomada pelo grupo de estudos dos padrões OPC foi a padronização das interfaces de comunicação, indicando apenas o que essas interfaces deveriam fornecer e qual o comportamento esperado de suas funções, mas sem entrar no detalhamento de como tais interfaces deveriam ser criadas. Ao se concentrarem nas interfaces entre os componentes, os padrões deixam de se preocupar com o tipo específico de comunicação dos equipamentos de campo. Dois grandes problemas são resolvidos por essa abordagem. O primeiro, caracterizado pela dependência do modelo de comunicação específico dos equipamentos sob supervisão, é eliminado pela inclusão da camada de comunicação definida pelas interfaces do padrão OPC. O segundo, relacionado à manutenção evolutiva de sistemas que utilizavam componentes, cujos métodos não podiam ter o nome alterado.

4.4 Uso dos padrões OPC hoje

Atualmente, três são os principais focos de uso de OPC. As manufaturas, as indústrias de processos e as empresas de automação predial [11]. Nas manufaturas, os padrões OPC alcançaram o mais alto nível de popularidade, sendo o foco de uso das especificações OPC Data Access. As características que delineiam esse tipo de indústria são a modularização dos passos de execução em células independentes entre si; a descentralização de controle a partir do uso de CLPs (Controladores Lógicos Programáveis); necessidades de redundância baixas, onde o custo de manutenção se sobressai pela baixa periculosidade de problemas afetarem vidas humanas. As indústrias de processos são menos modularizadas que as manufaturas, apresentando uma característica de produção baseada em procedimentos em batelada. As plantas desses tipos de indústrias são mais complexas, distribuídas por regiões geograficamente maiores e demandantes de uma quantidade muito grande de pontos de

aquisição. Necessitam de maior centralização e muito maior disponibilidade de operação, mas suas restrições de tempo não são tão críticas quanto no caso das manufaturas. Já as empresas de automação predial lidam com situações que afetam muito diretamente as pessoas em seu entorno. Necessitam também de uma grande quantidade de pontos de aquisição e controle e de redundância de operação, sendo que o tratamento de situações de alarme é sua principal função.

Os padrões OPC são adequados a esses tipos de indústria, assim como os padrões de comunicação normalmente utilizados com seus servidores. No entanto, há um tipo de automação onde esses padrões tiveram pouca penetração. Trata-se de empresas que lidam com grandes sistemas de logística ou de distribuição. Nesse tipo de organização, a complexidade das plantas supervisionadas são muito simples, com baixa quantidade de pontos de aquisição e controle e são distribuídas por extensões geográficas muito grandes e de difícil acesso. Essas plantas não oferecem risco significativo à vida humana e não possuem redundância significativa. As condições de alarme são relevantes, mas as restrições de tempo são extremamente dilatadas, podendo chegar a minutos ou horas. Encaixam-se nessa categoria grandes ferrovias, empresas de distribuição de energia e água ou de produção de insumos agro-florestais [11]. Nesses casos, o principal problema a ser resolvido é o provimento de uma forma segura e de baixo custo para supervisionamento da planta. O uso de transmissão de dados através de meio sem fio normalmente atende às necessidades de aquisição de informações desse tipo de planta a um custo muito menor que o uso de redes de campo convencionais. A tecnologia celular de comunicação tem se mostrado uma tendência ascendente quando comparada às demais tecnologias de transmissão sem fio, uma vez que sua rede de provimento de serviços vem aumentando continuamente em área de abrangência e qualidade [23]. Um estudo das características necessárias para que as especificações OPC

possam ser utilizadas com esse tipo de tecnologia fornece uma forma padronizada de utilizar esse meio de comunicação, além de possibilitar que sistemas SCADA venham a ser empregados em novos tipos de mercado.

Compete para o sucesso no uso dessa tecnologia o atual nível de maturidade das principais especificações do padrão OPC. O desenvolvimento atual está tão avançado que este padrão já está sendo utilizado na indústria como base para a definição de modelos mais sofisticados para sistemas de potência [24]. Os padrões modularizados e componentizados do OPC ainda possuem características de definição, implementação e uso que se encaixam nas necessidades atuais da indústria, seja por parte dos fornecedores ou por parte dos consumidores de sistemas de automação [25].

4.5 Servidores OPC DA

Através das especificações OPC Data Access é possível garantir interfaces bem definidas que fornecem as funcionalidades que quaisquer sistemas SCADA necessitam para buscar dados de uma planta sob supervisão. Esse padrão também fornece a estrutura básica para que o sistema de aquisição que concentrará as informações recuperadas da planta seja implementado com um alto nível de modularização e componentização, gerando a tendência para que o sistema tenha maior qualidade.

Considerado atualmente como o padrão “de facto” para comunicação de dados na indústria, o OPC DA possui muito de seu sucesso associado à aceitação que esse padrão obteve junto às empresas criadoras de sistemas supervisórios, que mantém em seus sistemas interfaces de clientes OPC DA.

4.5.1 Tecnologia associada aos Servidores OPC DA

A maior parte das empresas de tecnologia optaram pelo desenvolvimento de seus sistemas de supervisão baseados na tecnologia COM/DCOM. COM é acrônimo de *Component Object Model*, enquanto DCOM significa *Distributed Component Object Model*. Com exceção das definições de interfaces apropriadas para a distribuição em equipamentos distintos na rede, pertencente apenas aos DCOM, ambas as tecnologias oferecem uma série de características que permitem a componentização de sistemas. Primeiramente, essa tecnologia é Orientada a Objetos, ou seja, permite que sejam realizados polimorfismos, herança, abstrações e relacionamentos entre as diversas entidades criadas para representar os componentes de software, além de manter todas as especificações da OSF DCE RPC. Convencionou-se denominar ORPC a adição da Orientação a Objetos a essas especificações. Além disso, todas as funcionalidades são representadas por métodos agrupados em interfaces, que representam um certo domínio de informação. Isso provê a modularização necessária para essa tecnologia (figura 4.2). As interfaces dessa tecnologia são fortemente tipadas, o que significa que a assinatura de um método definido em uma interface terá seus parâmetros verificados para garantir que se encaixam com a representação definida no componente chamado.

Os componentes que provêm serviços são chamados de servidores DCOM. Para que haja independência entre esses componentes e os softwares clientes que fazem uso dos mesmos, o controle sobre a criação e destruição de objetos é feito diretamente pelos próprios servidores. Através de uma interface específica e de algoritmos de contagem de inicialização, o servidor DCOM sabe quando ainda há clientes executando chamadas a suas interfaces ou não, decidindo sobre sua própria remoção da memória e liberação dos recursos computacionais sob uso. Por fim, a tecnologia DCOM oferece suporte tanto para o modelo

STA (*Single Thread Apartment*) quanto ao modelo MTA (*Multi Thread Apartment*), o que garante ao desenvolvedor maior liberdade quanto à forma de programar seus sistemas [26].

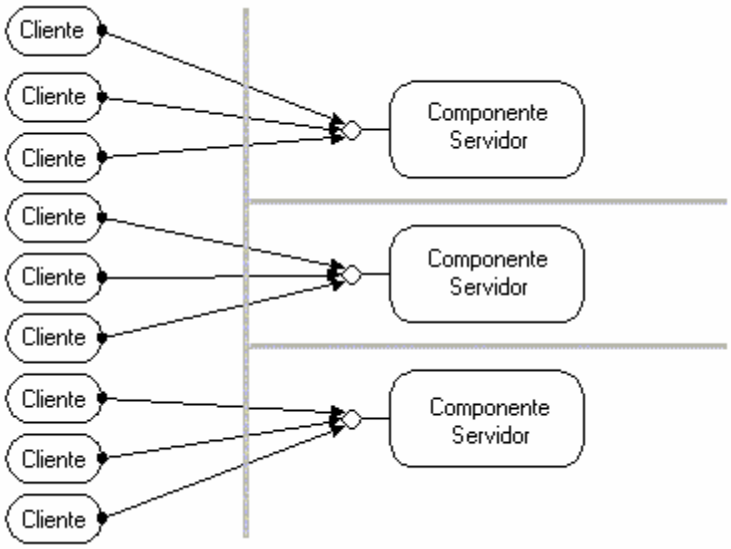

Figura 4.2 Representação de sistemas clientes chamando interfaces de componentes

4.5.2 O *framework* de desenvolvimento de Servidores OPC DA

Todo *framework* de desenvolvimento deve possuir um passo-a-passo que permita o uso mais eficiente dos padrões já definidos e impeçam que interpretações incorretas desses padrões resultem em um sistema mal feito. No caso de servidores OPC, o uso de *toolkits* provê facilidades de desenvolvimento pelo fato de permitir que os implementadores de uma solução fixem esforços no problema a ser resolvido ao invés de se preocupar com a conformidade ao padrão OPC [11]. Há uma série de diferentes *toolkits* que oferecem seus próprios meios de se alcançar a interoperabilidade fomentada pelo padrão OPC. Independente de qual for utilizado, ou até mesmo se o uso de tais ferramentas for descartada, há uma série de atividades que devem ser executadas para que se alcance a qualidade necessária no servidor desenvolvido. São elas:

Identificação dos requisitos do sistema

De acordo com as características do processo a ser supervisionado, diferentes funcionalidades devem ser providas. Como requisito relevante destacam-se a quantidade estimada de pontos a serem supervisionados (*tags*); os tamanhos das mensagens trocadas entre os sistemas remotos e o concentrador de informações; a largura de banda disponibilizada pelo subsistema de comunicação; o atraso associado ao subsistema de comunicação; a capacidade de armazenamento local dos equipamentos de campo.

Além das funcionalidades básicas, é importante avaliar as necessidades específicas de cada tipo de planta. Há plantas cujas respostas a interferências são lentas, podendo ser controladas por medidas indiretas, enquanto outras plantas respondem muito rapidamente, necessitando de medidas redundantes para fornecer segurança. Nesses casos, deve-se planejar com antecedência se algoritmos de sequenciamento ou de consolidação de informações deverão ser providos.

Nesse momento também é definida a modularização a ser utilizada no sistema. O sistema é dividido em pacotes que podem ser implementados de forma independente e, de acordo com a tecnologia adotada, implantados em diferentes equipamentos. Estas informações são utilizadas para definir a arquitetura de criação do sistema, já no escopo da tecnologia a ser adotada.

Escolha do modelo arquitetural de execução

Uma importante característica do servidor é o modelo de execução. Como resultado da tecnologia adotada para as especificações OPC DA, pode-se implementar os servidores OPC em dois diferentes modelos. Essa flexibilidade visa fornecer soluções mais apropriadas para necessidades diferentes de clientes OPC e de funcionamento do sistema de aquisição de

informações, aproveitando-se das possibilidades providas pelo DCOM. O dois modelos diferentes de execução para programas que funcionam sobre os sistemas operacionais da Microsoft são o InProc e o OutProc.

No primeiro caso, o servidor OPC é implementado como uma biblioteca de ligação dinâmica (DLL, *Dinamic Link Library*). Este tipo de servidor precisa ser inicializado pelo cliente e funciona no espaço de memória deste. Esta dependência da memória interna do cliente faz com que ocorra um maior consumo de recursos dos equipamentos onde os clientes estiverem alocados. No entanto, não se perde as funcionalidades relativas ao provimento distribuído de dados pelo servidor OPC, uma vez que o próprio servidor gerencia seu ciclo de vida, deixando a memória do computador servidor apenas após o último cliente tê-lo liberado.

O outro modelo previsto é o OutProc. Neste caso, o servidor é implementado como um executável e pode ser dividido em duas novas categorias: executável comum ou serviço do Windows (*Service* NT). O executável comum funciona no espaço de memória do computador servidor, recebendo chamadas RPC às suas interfaces. Nesse caso, há uma séria desvantagem. Para que um executável comum funcione em um sistema operacional Microsoft Windows, o mesmo deve pertencer a um usuário que tenha se autenticado no sistema, ou seja, um usuário deve ter iniciado manualmente este executável. Quando esse usuário realizar o *log off*, todos os executáveis que ele por ventura tenha chamado serão finalizados. Já o *Service* NT permanece em funcionamento na memória do computador servidor independente do uso de clientes, além de não necessitar que este computador servidor esteja com um usuário autenticado e autorizado utilizando a máquina. Trata-se de um serviço do sistema operacional Microsoft Windows, que de acordo com a necessidade, pode permanecer em funcionamento por tempo indeterminado, bastando iniciar o computador.

Escolha de componentes existentes

Após a modularização do sistema, uma série de funcionalidades poderá ser mapeada para as interfaces definidas no padrão OPC. Há componentes OPC para as definições de chamadas síncronas, assíncronas, *callbacks*, manutenção do *cache* de dados, entre outros. Nesse momento são avaliadas quais funcionalidades serão implementadas e quais delas manterão apenas resultados não implementados. De acordo com essas escolhas, definem-se quais interfaces serão realmente codificadas e quais manterão apenas resultados padrões.

Componentização no OPC

Novas funcionalidades do servidor podem não estar previstas nas interfaces definidas nos padrões OPC. Caso os requisitos do sistema indiquem tal necessidade é necessário criar novos componentes, que serão agregados como interfaces específicas da solução. Como o objetivo das especificações OPC é manter um padrão de interfaces, as novas interfaces criadas deverão seguir o padrão naquilo que for aplicável, de forma a impedir um comportamento incoerente com relação às demais interfaces.

Testes dos componentes

Testes sobre as interfaces recém criadas. São os testes unitários dos componentes do servidor. Visa verificar como as interfaces respondem às requisições padronizadas que os clientes OPC realizarão. Há ferramentas específicas para realizar esse tipo de teste, com foco no tipo de resposta que uma interface fornecerá e não especificamente no comportamento do sistema de comunicação com os equipamentos de campo.

São testadas nesse momento passagens de parâmetros inválidos, chamadas a interfaces inexistentes, além de outros testes relevantes sobre a estrutura de interfaces do sistema.

Integração dos componentes

O próximo passo é montar o servidor OPC a partir de cada componente codificado e devidamente testado. Nessa fase, a ligação entre as interfaces e o sistema de aquisição e comunicação com os equipamentos de campo deve ser feita. Em uma situação ideal, o sistema aquisitor já deve ter sido modelado de acordo com os requisitos iniciais e modularizado de forma que suas funcionalidades sejam compatíveis às das interfaces do servidor OPC a ser implementado.

Testes integrados do sistema

Os testes de integração verificam o funcionamento do servidor como um todo, já considerada a lógica de aquisição de dados, o tratamento de informações no servidor e a compatibilidade das interfaces aos padrões. Há ferramentas fornecidas pelo OPC Foundation para verificar o funcionamento das interfaces. Mas para verificar o funcionamento do servidor, é necessário colocá-lo em funcionamento junto a um cliente OPC e verificar os possíveis comportamentos encontrados em uma situação real.

Esses testes devem contemplar todas as condições de eventos simultâneos, falhas de comunicação, além de testes que visem comprometer a integridade do servidor, como a verificação do gerenciamento de ciclo de vida.

A principal característica tecnológica utilizada no desenvolvimento de servidores OPC é a tecnologia COM (*Component Object Model*), que indica uma série de interfaces e especificações para componentização de trechos de programas mais complexos. Através das ferramentas existentes no próprio *framework* do COM, é possível criar as interfaces de

comunicação que encapsulam as características internas da implementação em relação à funcionalidade provida.

Todas as interfaces criadas para prover funcionalidades devem seguir um padrão de implementação originado da necessidade de distribuição de módulos codificados da forma mais independente possível. A principal estrutura desta tecnologia é a interface IUnknown, a partir da qual se gerencia o ciclo de vida de quaisquer entidades compatíveis com a tecnologia COM. Todas as demais interfaces criadas em um componente COM são herdeiras desta interface e podem prover funcionalidade pela chamada de seus métodos (ou rotinas) via RPC (*Remote Procedure Call*). Chamadas diretas a métodos feitas neste processo viabilizam o gerenciamento de exceções e a manutenção de conexões entre o componente modelado e os softwares que fazem uso de suas interfaces.

Estas características tecnológicas delineiam o projeto de um servidor OPC DA, sendo que algumas das definições apresentadas nas especificações do padrão são mais impactantes nas decisões a serem tomadas por tratarem informações arquiteturais do sistema.

Os mesmos são orientados a conexão, mantendo uma contínua troca de informações entre os clientes e seus servidores [8]. A questão do desempenho para disponibilização de informações é extremamente importante para sistemas de supervisão, pois podem determinar até que ponto haverá confiabilidade em relação à informação fornecida. O uso da tecnologia DCOM permite que o desempenho na transferência de dados seja similar ao apresentado por *drivers* proprietários de equipamentos. O fato de o padrão ser notoriamente aceito e utilizado internacionalmente também influencia na execução de extensivos testes de desempenho por parte de grandes empresas de automação [27]. Com isto, o padrão evoluiu muito rapidamente

para definições capazes de se adaptar às necessidades reais de cada tipo de aquisição, fornecendo flexibilidade suficiente para agregar soluções de melhor desempenho [27].

Um dos principais pontos de melhoria do padrão OPC encontra-se em suas restrições com relação ao meio de transporte de dados. Servidores OPC podem ser utilizados para troca de dados através de redes TCP/IP. Nestas redes é comum utilizar-se de estruturas de segurança para impedir que computadores externos venham a ter acesso aos computadores internos. A ferramenta mais utilizada para este fim é o *firewall*, solução arquitetural que envolve hardware e software específicos para reduzir a possibilidade de invasões na rede. Uma de suas principais funcionalidades é impedir a execução de código remotamente, impedindo, inclusive, que algumas operações do DCOM sejam realizadas. Isso dificulta muito o uso do padrão OPC junto a ferramentas de proteção mais sofisticadas.

As principais funcionalidades comprometidas pelo uso de *firewalls* são a procura de informações sobre os servidores existentes em determinado computador fornecido pelo componente *OPCServerBrowser*, a capacidade de instância de objetos remotos em memória de um programa cliente e a inicialização do servidor por parte do cliente.

Um dos fatores que mais consomem tempo e dinheiro na implementação da comunicação entre fontes de aquisição de dados e sistemas SCADA é a obtenção da tecnologia utilizada nesta implementação. No caso do OPC, há a necessidade de se adquirir conhecimentos sobre a tecnologia COM / DCOM para implementar ou alterar um servidor. Há uma vantagem com relação à curva de aprendizado utilizada pelo padrão OPC, pois uma vez que a tecnologia DCOM seja assimilada não há mais alterações arquiteturais que causem a necessidade de novo processo de aprendizado, situação muito comum quando se desenvolvem *drivers* específicos.

Apesar de ser apenas uma tecnologia a ser assimilada, o *framework* por trás da tecnologia DCOM é de uma complexidade considerável. Uma das principais vantagens relativas ao uso de *toolkits* para desenvolvimento de servidores OPC está no fato de poder-se saltar esta etapa de aprendizado. Aprender a utilizar um *toolkit* é muito menos custoso do que aprender todo o *framework* da tecnologia DCOM, deixando para o desenvolvedor a responsabilidade de se concentrar nas características específicas do sistema de aquisição da solução.

Este ganho de tempo não é tão efetivo quanto possa parecer, uma vez que não há padronização com relação aos próprios *toolkits*. Dessa forma, um desenvolvedor deverá sofrer nova curva de aprendizado sempre que for necessário migrar para um novo *toolkit*. Este é um dos pontos onde ainda há a possibilidade de ocorrer perda de recursos financeiros que a adoção do padrão OPC não é capaz de eliminar.

4.6 Servidores OPC XML DA

Através das especificações OPC XML Data Access são fornecidos padrões de chamadas remotas a Web Services que permitem o acesso a dados mantidos em concentradores de informação a partir da Internet. Esse padrão indica ainda algumas necessidades específicas para aplicações industriais seguindo os preceitos de supervisão e controle esperados de um sistema SCADA. O padrão possui suporte às definições criadas para os servidores OPC DA, criados com tecnologia DCOM. Também fornece definições para algoritmos de subscrição de serviço e alcance de níveis mínimos de segurança.

Uma característica própria das especificações OPC XML DA é a possibilidade de uso dessa tecnologia como uma interface de acoplamento para servidores OPC DA. Assim como

as especificações de Automation Interfaces, os Web Services criados nos padrões da OPC podem servir de camada para intercomunicação para tecnologias Web.

4.6.1 Tecnologia associada aos servidores OPC XML DA

A evolução atual das tecnologias baseadas em XML supera as definições criadas para comunicação entre aplicações, como a tecnologia OLE, ampliando a possibilidade de que aplicações conversem entre si através da Internet.

Uma das principais vantagens é a possibilidade de expandir sua própria representação para qualquer domínio de informação. O HTML, apesar de extremamente genérico, não é capaz de se adaptar a todo tipo de informação existente. Quando necessário aumentar sua expressividade necessita-se implementar características proprietárias que dificultam a integração entre diferentes sistemas. Já o XML se baseia em definições de sintaxe, de forma que, para atender ao padrão basta que um documento XML corresponda a um determinado esquema, representado em outro arquivo. As características de representação, de conversão e de processamento são mantidas em arquivos XSL.

Para que fosse possível aproveitar essas definições na troca de informações entre aplicações foram criados os Web Services. No seu princípio, os Web Services tinham como missão permitir que empresas trocassem informações entre si de forma padronizada pela Internet, mas sem muita preocupação com a segurança da informação trocada. À medida que maiores investimentos passaram a ser feitos nessa tecnologia de intercomunicação, foi necessário aumentar a abrangência dos protocolos utilizados e a complexidade das funcionalidades oferecidas [28].

A primeira apresentação de padrões voltados para Web Services do W3C em 2003 [21], continha definições focadas em funcionalidades centrais, como o protocolo de publicação de serviços e o protocolo de transporte e representação de objetos. Outras organizações internacionais aprofundaram-se em diferentes focos de padronização, como o OASIS, que privilegiam funcionalidades de mais alto nível, como segurança, autenticação, processos de negócio e garantia de entrega de mensagens [29 e 30]. Apesar de ainda não conflitantes, há uma falta de definição internacional sobre o direcionamento desses esforços de padronização. Há organizações trabalhando na padronização dos mesmos temas de forma independente, o que pode causar problemas de interoperabilidade e impedir que o principal objetivo da criação dos Web Services seja alcançado. A tabela 4.2 apresenta algumas das principais especificações direcionadas ao mercado.

Tabela 4.2 Iniciativas internacionais de padronização por organizações

Tema	Especificação	Descrição	A quem foi submetida
Processos de negócio	WS-CDL	Descreve o funcionamento integrado de componentes de um mesmo domínio de negócios.	W3C
Entrega segura de mensagens	WS-Reliability	Promove eliminação de replicações	OASIS
	WSRM	Define um modelo de ordenamento e não repúdio de mensagens	-
Segurança	Basic Security Profile	Definição de cenários de segurança e como tratá-los	-
	SAML 2.0	Definição de assertivas utilizadas na linguagem de representação para garantir segurança	OASIS
	WS-Federation	Integra o SAML 2.0 a conceitos de estrutura de implantação de Web Services.	-

O padrão OPC XML DA utiliza apenas os padrões de infra-estrutura definidos pelo W3C. Com isso, há questões ainda não padronizadas com o uso do OPC. Uma dessas questões é a segurança do Web Service desenvolvido. A única referência sobre a segurança adotada no padrão diz respeito ao padrão de comunicação, utilizando HTTPS [31]. Não são

levados em consideração problemas importantes associados à autenticação e à autorização de uso dos serviços, uma vez que não há um padrão de fato definido internacionalmente.

Web Services podem ser focalizados a partir de diferentes modelos de perspectiva. Cada modelo explicita algumas importantes facetas da arquitetura. Há quatro modelos possíveis: o Modelo Orientado a Mensagens; o Modelo Orientado a Serviços; o Modelo Orientado a Recursos e o Modelo de Políticas [21]. A partir dessas visões, é possível elencar definições arquiteturais mais relevantes para o funcionamento dos Web Services. Uma dessas definições é a prestação de serviços em ambientes distribuídos, que disciplina o uso de orientação a mensagens. Isso causa o isolamento da implementação do Web Service, permitindo assim que sistemas legados sejam adaptados aos requisitos de distribuição de aplicações. Uma outra definição é a neutralidade de plataforma, que justifica o uso de XML como padrão de semântica dos Web Services, uma vez que sua implementação independe da plataforma sobre a qual o sistema está funcionando. Com isso, é possível indicar situações onde o uso de Web Services se torna mais apropriado do que outras tecnologias:

- Sistemas que necessitem operar sobre a Internet;
- Sistemas com fraca restrição de garantia e confiabilidade;
- Componentes de sistemas distribuídos sobre diferentes plataformas;
- Integração de sistemas legados

4.6.2 Framework de desenvolvimento de Servidores OPC XML DA

Há duas possibilidades de desenvolvimento de um serviço no padrão OPC XML DA. A primeira é a utilização dessa tecnologia como camada adicional para um sistema pré-

existente, de forma a encapsular esse legado e permitir acesso a suas funcionalidades através de um ambiente padronizado e de fácil comunicação via Internet. A segunda possibilidade é o desenvolvimento de novos sistemas já seguindo o padrão de desenvolvimento e as tecnologias disponíveis para criação de Web Services. Nesse caso, todo o sistema é implementado baseando-se em *framework* específico, apesar deste não ser tão diferente de um desenvolvimento de sistemas de informação modularizados e componentizados.

Quando desenvolvido independentemente de sistemas pré-definidos, o OPC Foundation indica que as definições do OPC XML DA devem ser as mais próximas possíveis do padrão OPC DA, visando manter a compatibilidade com os estudos realizados até então. Com isso, boa parte dos passos a serem realizados para desenvolver um tipo de servidor é aplicável para o outro tipo de servidor. As principais distinções se encontram na escolha de interfaces relevantes. Além disso, dois pontos foram identificados onde as atividades são exclusivas para o desenvolvimento de um Web Service. Segue a descrição dos passos.

Definição do *Minimum Delay Polling Period* (*Holdtime*) fornecido pelo servidor.

Abaixo desse valor, não será possível verificar por alterações nos itens, pois o aquisitor não terá condições de buscar tais informações. Caso um valor menor seja escolhido, perdem-se *round-trips* entre o cliente e o servidor, que certamente não terá a informação solicitada. Este tempo é dependente de parâmetros dependentes da forma de recuperação de dados pelo aquisitor.

Definição do tempo de espera sem alterações (*Waittime*)

Valor dentro do qual qualquer alteração nos itens sob supervisão deve ser imediatamente informada. Este tempo não é dependente do servidor, mas sim do algoritmo de

polling do cliente, de forma que a única restrição a ser verificada é com relação à forma de manutenção dos *buffers* e dos *caches*.

Há uma série de diferenças entre sistemas que utilizam troca de mensagem ao invés de chamadas diretas às interfaces de métodos. Destas, duas são bastante impactantes e são a principal fonte de medição de qualidade no comparativo entre um servidor OPC DA e um servidor OPC DA XML.

A primeira diferença se encontra no consumo de recursos do equipamento onde o produto estiver funcionando. Os sistemas que utilizam troca de mensagens não mantêm conexões fixas entre o cliente e o servidor de dados, sendo muito mais escaláveis. Reduz-se dessa forma o consumo de processamento e aumenta-se a robustez do sistema.

A segunda diferença é em relação à restrição do tempo de resposta. Isto porque o uso de troca de mensagens contém uma sobrecarga associada ao encapsulamento e tráfego de informações em um nível mais alto que a troca de informações via DCOM. Com isto, os sistemas baseados em troca de mensagem perdem em velocidade para levar informações entre dois ou mais aplicativos quando comparados a sistemas desenvolvidos com DCOM.

Para aproveitar ao máximo as vantagens relativas à escalabilidade evitando ao máximo o *overhead* associado, foi definido um padrão de troca de mensagens com um encapsulamento mínimo suficiente para reconhecimento pelos pontos de comunicação. O padrão SOAP fornece uma estrutura bem definida que fornece essas vantagens (ver figura 4.3).

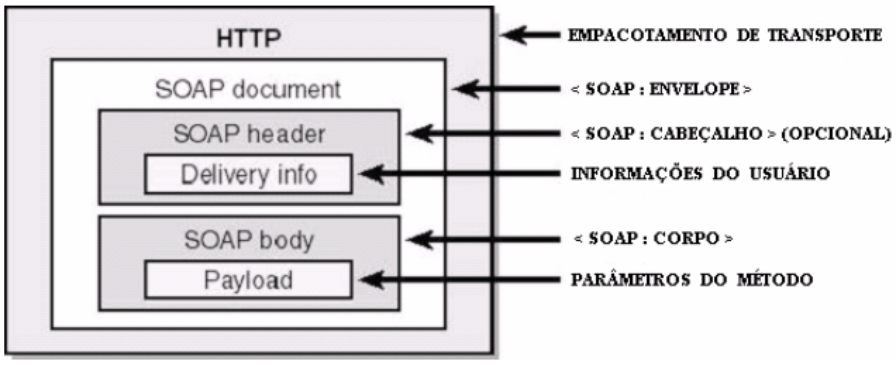

Figura 4.3 Envelope SOAP

A troca de mensagens ocorre assincronamente, de forma que é necessária a definição de um *buffer* entre o fornecedor de informações via mensagens SOAP e o receptor destas mensagens. O uso de filas para gerenciamento destas trocas de mensagens cria a necessidade de um intermediador entre o sistema de disponibilização de informações e o aquisitor de dados. Também é necessária a definição de manipulação do *poll* (consulta) de informações, de forma a controlar a comunicação eminentemente assíncrona desta arquitetura.

Uma vez definidas as filas para o envio e recepção de mensagens é necessária a definição de uma política de envio e de leitura destas mensagens. A política *subscribe polling-pull* é a aplicada. São criadas filas de envio de mensagens que recebem requisições de serviços a serem prestados pelo servidor. As filas de recepção guardam as respostas geradas pelo servidor. Os serviços requisitados devem estar publicados para serem reconhecidos pelo servidor e poderem ser executados. Um servidor sabe quais serviços pode prestar a partir das informações mantidas em um arquivo de configuração contendo a assinatura dos métodos que representam cada serviço. Tal arquivo é um WSDL (*Web Service Definition Language*) [32], seguindo as definições de um XML.

4.6.3 Especificidades do padrão OPC XML DA

A especificação atual para implementação de servidores XML baseia-se na versão 1.0 da especificação [31]. As interfaces implementadas pertencem aos tipos leitura, escrita, submissão e cancelamento de submissão de informação. Além disto, há a necessidade de publicação do servidor e suas funcionalidades, já que soluções de registro não se aplicam a esta tecnologia. Com isto, verificam-se três características que delineiam o desenvolvimento de um servidor OPC XML DA.

A primeira é a identificação do servidor OPC. Como eles são disponibilizados como *web services*, a forma adotada de publicação de suas informações baseia-se no protocolo UDDI – *Universal Description, Discovery and Integration* [32 e 31]. Este protocolo indica a localização dos servidores baseado em pesquisas realizadas em diretórios replicados na Internet. Através desse protocolo, consegue-se identificar até mesmo o IP do servidor, que não é obrigando a utilizar uma porta específica para disponibilização dos serviços. Estes, por sua vez, são apresentados a partir da inclusão de informações relevantes no WSDL do servidor. O WSDL é um descritor que informa a todos os que entrarem em contato com o *web service* que operações poderão ser realizadas a partir deste.

Uma vez identificado o servidor e seus serviços, a segunda característica a ser considerada é a troca de informações de *locale* (internacionalização). O padrão OPC XML DA não é voltado à conexão, mas sim à subscrição de requisições e ao *polling* por resposta. Assim sendo, não são mantidas informações intermediárias entre uma conexão criada para requisição e outra conexão criada para busca de resposta. Cada nova conexão deve possuir toda a informação de *locale* do servidor OPC para que a interpretação dos dados recuperados

não seja prejudicada. No entanto, tal situação aumenta substancialmente o overhead relativo à inteligibilidade dos dados trocados entre o cliente e o servidor.

A terceira característica está relacionada ao algoritmo de *polling* a ser adotado pelo servidor (ver figura 4.4). Isto afeta a confiabilidade do serviço prestado pelo sistema e sua viabilidade em aplicações que necessitam de dados em tempo real com alta taxa de atualização. Para isto, um sistema de resposta controlada é obrigatoriamente implementado para realizar acesso ao servidor, baseado nas interfaces de requisição de serviço e de resposta. Neste caso, as interfaces são publicadas no WSDL do servidor e obrigatoriamente seguem as assinaturas de leitura (*Read* e *ReadResponse*), escrita (*Write* e *WriteResponse*), conexão (*Subscribe* e *SubscribeResponse*), *polling* (*SubscriptionPolledRefresh* e *SubscriptionPolledRefreshResponse*) e desconexão (*SubscriptionCancel*, *SubscriptionCancelResponse*).

Duas características dos *Web Services* fazem desta tecnologia mais apropriada para o transporte de informações de grande densidade. A primeira é a simplicidade relativa à serialização de informações com o uso de XML, permitindo que quaisquer níveis de informação possam ser representados e transportados rapidamente. A segunda é a assincronicidade inerente à característica *subscribe-polling*, que fornece os meios para que haja menor consumo de recursos do sistema quando quantidades muito grandes de dados forem transmitidas.

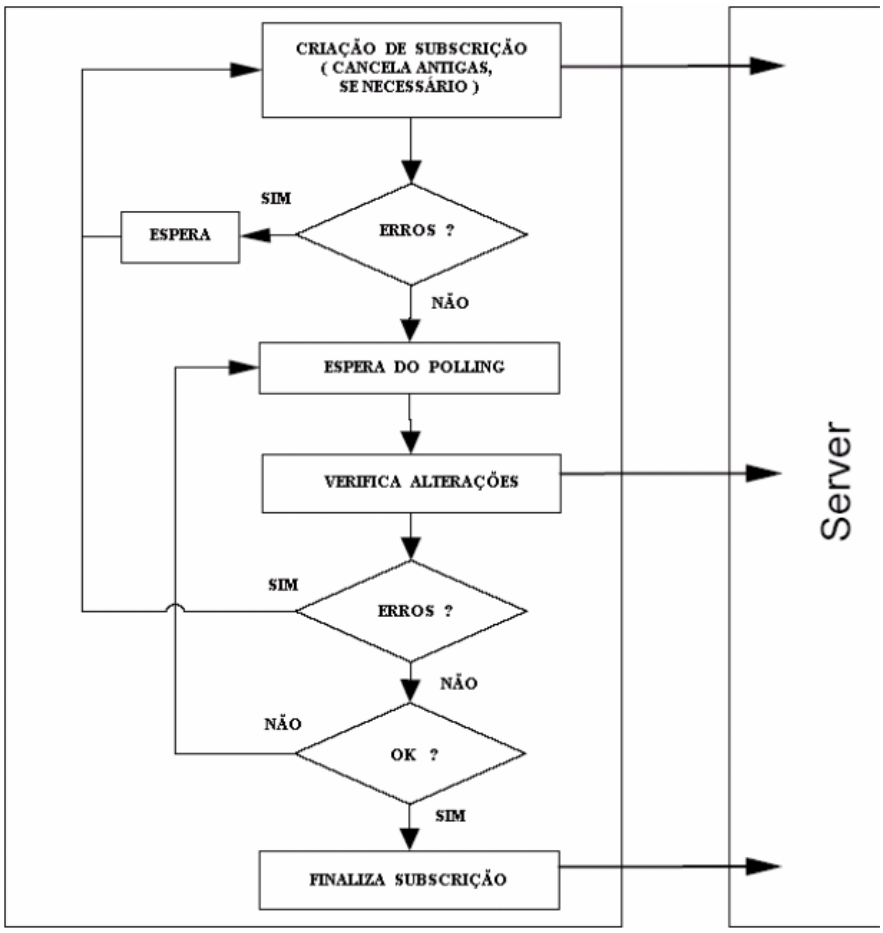

Figura 4.4 Algoritmo subscribe-polling

A despeito da alta capacidade de transferência de informação pela simplicidade do meio utilizado, os *Web Services* apresentam uma grande diferença de desempenho quando comparados com soluções proprietárias ou mesmo com servidores OPC criados com a tecnologia DCOM. Isto se dá pelo uso da troca de mensagens entre sistemas em execução, de desempenho muito inferior à instanciação em memória do código a ser executado. Isto faz com que o *overhead* da estrutura criada para a troca de mensagens seja representativo, caso a quantidade de informações a serem transmitidas seja muito pequena.

5 Sistemas desenvolvidos

O objetivo dos sistemas criados é possibilitar que plantas distribuídas por regiões geográficas distantes e de difícil acesso sejam supervisionadas por qualquer tipo de software SCADA que ofereça a funcionalidade de cliente OPC. Para resolver a situação de comunicação exeqüível para tais tipos de plantas foi utilizada tecnologia de comunicação via telefonia celular. Para permitir a integração com sistemas SCADA, o sistema de aquisição foi criado usando os padrões OPC, tanto com interfaces DCOM quanto com Web Services. Com isso espera-se abrir espaço para o uso das especificações OPC em uma nova fronteira, onde ainda não alcançaram a notoriedade de padrão industrial; além de expandir as possibilidades de supervisão de plantas convencionais com o uso de um novo meio de comunicação.

5.1 Uso de celulares para comunicação de dados

Há uma série de tecnologias associadas ao uso de celulares. As principais no Brasil são a GSM (*Global System for Mobile Communications*) e a CDMA (*Code Division Multiple Access*), ambas aplicáveis a celulares de segunda geração (celulares digitais). O CDMA é a de maior abrangência no Estado do ES, sendo escolhida para uso na implementação dos servidores OPC. A característica de abrangência é importante pois determinará o tipo de funcionalidade que seu canal de comunicação poderá ou não oferecer ao seu sistema. Através de qualquer das tecnologias é possível trocar informações usando o próprio canal de voz ou outro tipo de protocolo. Optou-se pelo uso do próprio canal de voz, que possui características de segurança adequadas para a situação desejada [23], além de permitir a supervisão de regiões mais abrangentes, uma vez que apenas a existência de cobertura celular passa a ser requerida.

A transmissão de dados via celular é muito vantajosa quando comparada às outras soluções de transmissão sem fio. Mas é necessário aproveitar uma de suas peculiaridades para se obter os principais ganhos de custo. Trata-se da não manutenção de conexão dedicada entre o emissor e o receptor. Isso significa que o tempo necessário para iniciar uma conexão, transmitir a informação e finalizar a conexão deve ser considerado em praticamente todas as trocas de dados entre o concentrador e os equipamentos de campo.

Uma outra condição de contorno é que um aparelho celular convencional, utilizando os recursos oferecidos em maior abrangência geográfica, não é capaz de manter mais de uma conexão simultânea com outros aparelhos. Logo, o meio de comunicação determina a forma como a rede de comunicação entre celulares deverá ser gerenciada. Para impedir que haja problemas de perda de informação ou atrasos nas transmissões, deve-se manter o canal de comunicação livre a maior parte do tempo possível.

5.2 O sistema de aquisição de dados

O sistema de aquisição proposto é baseado em premissas diferentes de supervisão com relação aos sistemas normalmente tratados com uso do OPC. Os sistemas de supervisão associados a plantas distribuídas por regiões geográficas de proporções metropolitanas ou ainda que atinjam a extensão de um estado brasileiro carecem de formas de comunicação diferenciadas e algoritmos específicos de controle. O sistema é formado por dois servidores OPC; um servidor OPC DA e um OPC XML DA. Esses servidores funcionam em máquinas denominadas concentradores, para as quais são enviados todos os dados de uma série de equipamentos de campo, denominados remotas. A figura 5.1 apresenta uma visão geral do sistema.

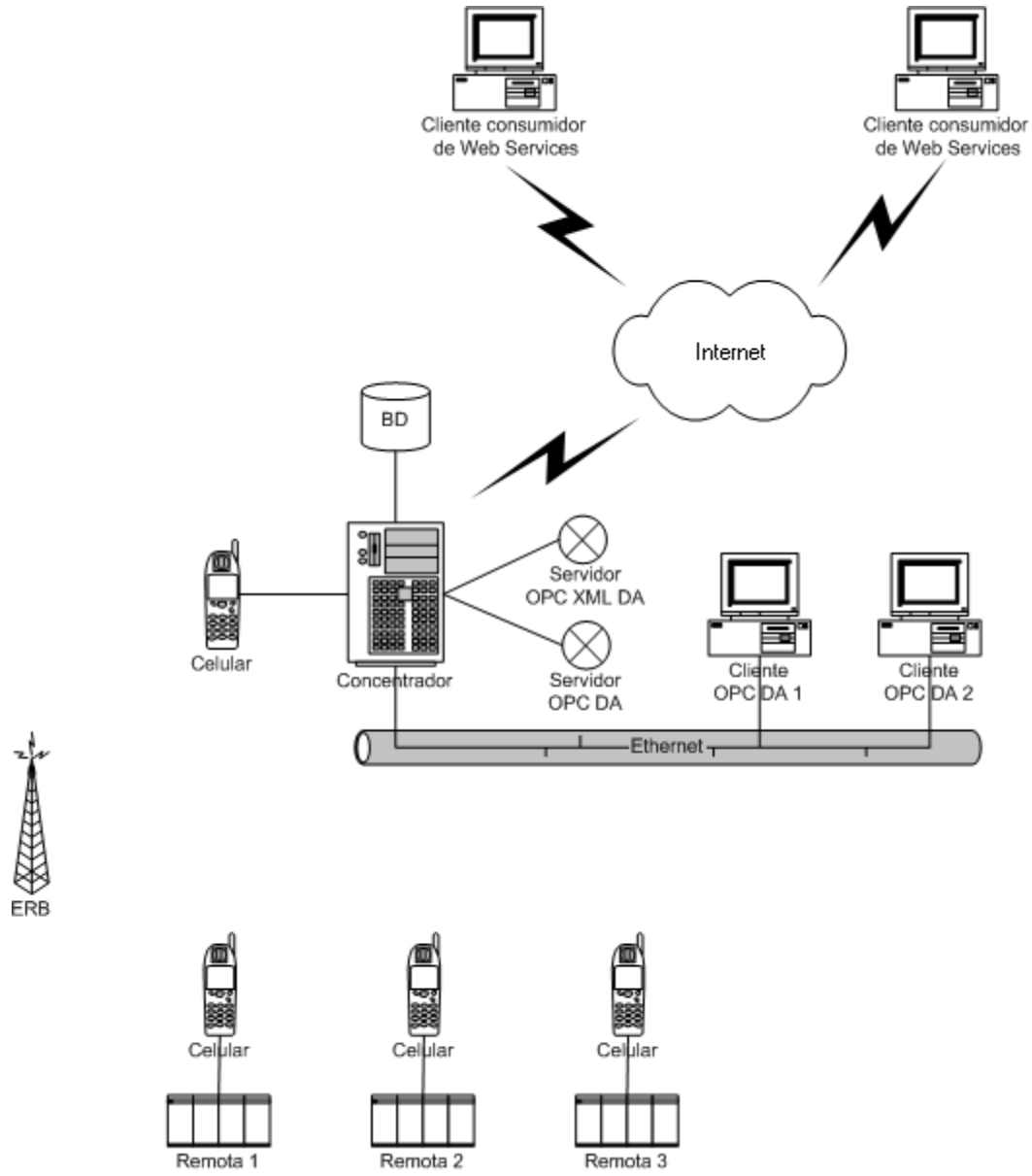

Figura 5.1 Visão geral do sistema

Através dessa solução é possível recuperar dados analógicos ou digitais, assim como receber informações mais sofisticadas, uma vez que é possível que as unidades remotas realizem chamadas ao concentrador quando eventos parametrizáveis ocorrerem. Um PC concentra o recebimento dos dados e informações, estando ligado a um aparelho celular. Um único PC pode conter tanto o módulo de comunicação do sistema, junto ao modem, quanto a base de dados e os servidores de dados no padrão OPC. O computador concentrador possui

um sistema de aquisição de dados modularizado de forma a apresentar, simultaneamente, um subsistema de persistência de informações consolidadas, um subsistema de gerenciamento de acesso às remotas e de conexões e um subsistema de disponibilização dos dados para sistemas clientes.

5.2.1 Subsistema de persistência

O subsistema de persistência de informações é formado por um banco de dados e um conjunto de classes definidas para encapsular o acesso ao banco de dados. A figura 5.2 apresenta o modelo lógico do banco de dados onde são mantidos os dados recuperados das remotas. A partir desse banco de dados também é montado o *Namaspace* dos servidores OPC, usando as informações mantidas na entidade “Item”.

As entidades do banco de dados foram definidas de acordo com a funcionalidade apoiadas pelos dados que armazenam. As entidades azuis possuem os dados estáticos, que representam o negócio da solução e não se alteram pelo uso do sistema. As entidades verdes possuem os dados relacionados às leituras realizadas pelo sistema. Já as entidades cinzas possuem os dados relacionados à estrutura de *hardware* que está sob supervisão, sua configuração e forma com que se relacionam.

Além dessas distinções, também foram tomadas algumas decisões com relação à estrutura do banco de dados. As entidades que representam informações de configuração foram mantidas isoladas das entidades que representam os próprios itens. O objetivo dessa opção foi facilitar a reutilização da mesma base de dados para os mais diversos tipos de *hardware* de aquisição, cujas informações podem ser mantidas e disponibilizadas a partir desse sistema. A funcionalidade de agrupamento do padrão OPC foi representada a partir de

uma auto-referência na tabela que representa os itens, de forma que a integridade com relação à natureza do grupo de itens não fique desvirtuada com relação aos parâmetros de acesso.

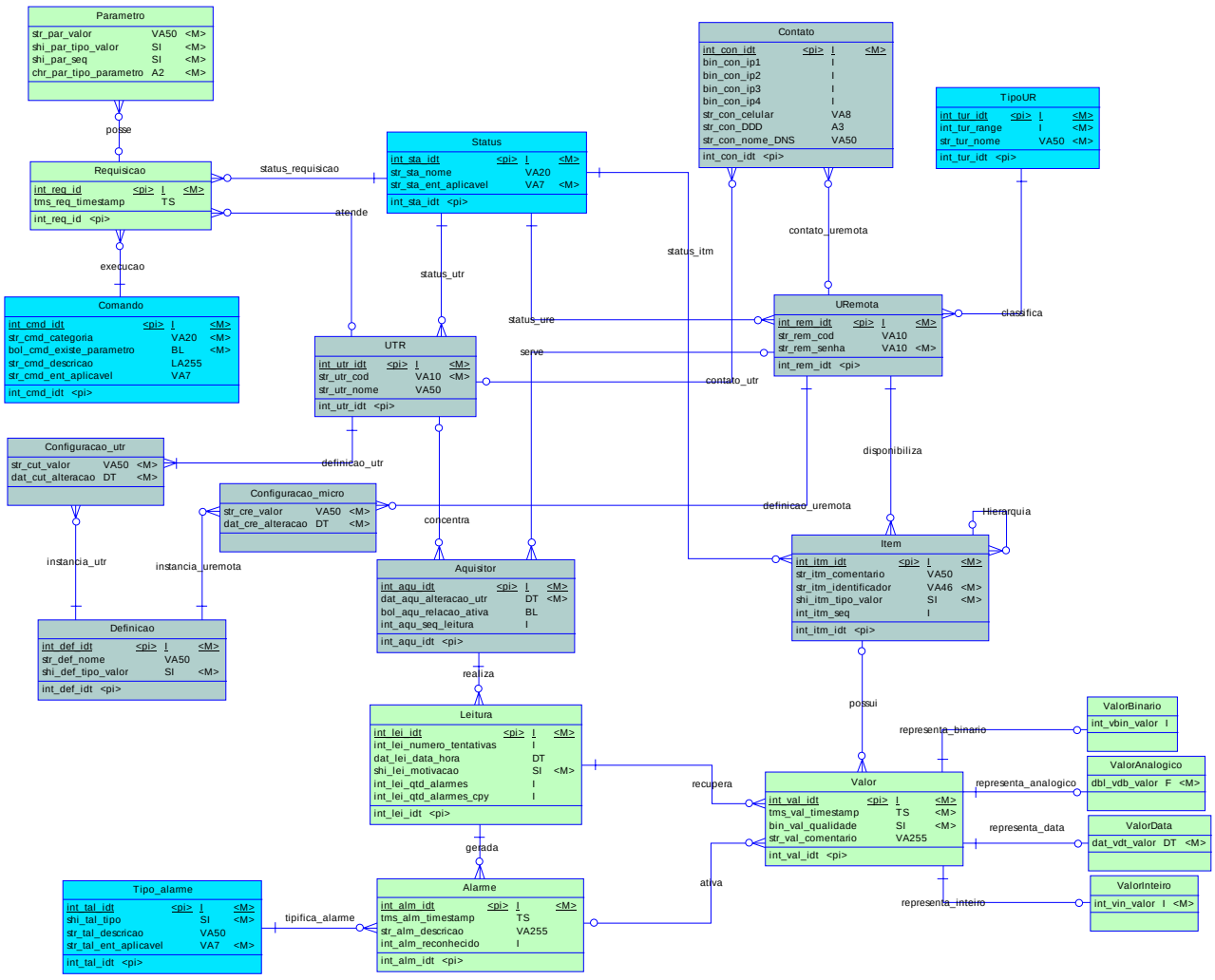

Figura 5.2 Modelo lógico do banco de dados do sistema de aquisição

Para permitir que um banco de dados fosse utilizado como forma de persistência para o sistema, algumas características básicas foram delineadas de forma a impedir que houvesse perda de desempenho associada ao seu uso. As classes de acesso ao banco de dados foram criadas segundo o padrão de espelhamento, com uma classe por entidade do banco de dados o acesso dessas classes ao banco de dados é feito através de ODBC (*Open DataBase Connection*).

Essas classes representam o domínio da informação, mas não as interações realizadas para executar determinada operação de negócio específico do sistema de aquisição de dados. Por isso, todas essas classes ficam abaixo de uma classe que segue o padrão de *Session Facade* [12]. A classe CPersistencia não possui atributos, mas apenas métodos que referenciam as classes de acesso ao banco de dados de acordo com as necessidades do sistema. A figura 5.3 apresenta um trecho de código que exemplifica a seqüência de uso das classes de acesso ao banco de dados. Algumas das funcionalidades implementadas nessa classe permitem que seja definida a ordem de acesso das microremotas quando da leitura temporizada.

```
// Verifica a quantidade de comandos assincronos não reconhecidos da base de dados.
// Retorna informação necessária para atender ao comando definido como mais prioritário na base de dados.
int CPersistencia::VerificarComandoAssincrono(int * pIntParIdRequisicao, int * pIntParIdComando, CStringList * pLstParParametros)
{
    // Retorno da operação
    int intResultado = 0;

    CString strMsgErro = "";

    // Indica se encontrou a requisição para alteração de status
    bool bolRequisicaoEncontrada = false;

    // Instancia um objeto da classe CRequisicaoEntity
    if (!pDatabaseEscelsa->IsOpen()) pDatabaseEscelsa->OpenEx(_T("DSN=escelsa_bd;UID=root"), CDatabase::useCursorLib);
    CRequisicaoEntity requisicao(pDatabaseEscelsa);

    // Abre a tabela na base de dados
    if (requisicao.Open(CRecordset::snapshot, NULL, CRecordset::none))
    {
        // Verifica se a tabela requisicao não está vazia
        if (!requisicao.IsBOF())
        {
            requisicao.MoveFirst();

            // Recupera a microremota a partir do código de identificação
            while (!requisicao.IsEOF())
            {
                // Comparação que motiva a procura
                if (requisicao.m_INT_STA_IDT == _STA_ABERTA_ID)
                {
                    // Incrementa quantidade de comandos a serem retornados
                    intResultado++;

                    // Verifica se é o primeiro comando não atendido encontrado na base de dados
                    if (!bolRequisicaoEncontrada) // intResultado == 1
                    {
```

Figura 5.3 Trecho de código de recuperação de informações no banco de dados

5.2.2 Subsistema de comunicação

O subsistema de aquisição de dados desenvolvido faz uso de algumas definições de equipamentos e protocolos utilizados em um sistema de remotas desenvolvido no LCI (Laboratório de Controle e Instrumentação) do DEE (Departamento de Engenharia Elétrica) da UFES. Esse sistema consiste em uma série de remotas criadas para permitir a supervisão e controle automático de religadores e medidores de alta tensão em linhas de transmissão elétrica [33]. Esses equipamentos serviram para realizar os testes dos servidores desenvolvidos, para validar o encapsulamento entre os diversos subsistemas criados e delinear um *Namespace* real a ser usado nos testes dos servidores desenvolvidos. A partir das remotas que fazem uso do religador é possível recuperar da planta quatro informações analógicas: três medidas de corrente e a tensão da alimentação. É possível recuperar também oito entradas digitais que definem o estado do religador. Já a partir das remotas que fazem uso do medidor de tensão é possível recuperar quatro medidas de tensão, dentre as quais a tensão da alimentação, além de uma informação digital que indica se houve invasão da caixa da microremota.

Nesse subsistema foram criadas classes que realizam o tratamento da comunicação através de aparelhos celulares usando tecnologia CDMA. Sobre o canal de voz dessa tecnologia foi implementado um protocolo de comunicação específico, criado a partir de uma simplificação do Modbus. As classes desse subsistema são CModComm e CModProtocol.

A classe CModComm encapsula a implementação da comunicação entre o concentrador e o celular no qual está conectado. A comunicação entre o computador e o celular é feita utilizando o protocolo RS232 através da porta serial, sendo que as funcionalidades de gerenciamento desse recurso tiveram que ser implementadas. Também

nessa classe foi implementado o gerenciamento de acesso às remotas e de conexões através de algoritmos de tratamento de tempo e de atrasos. Essas informações são configuráveis, de forma que é possível determinar tempos de leitura programada e de manutenção da conexão entre os equipamentos.

Foi criado um algoritmo de gerenciamento de conexões para gerenciar o canal de comunicação enquanto o servidor OPC estiver funcionando. Esse algoritmo é formado por uma série de estados finitos associados à identificação de respostas do modem, conforme apresentado na figura 5.4.

Inicialmente, haviam duas máquinas de estados para tratar paralelamente de conexões iniciadas pela UTR e de conexões iniciadas pelas microremotas. No entanto, foi verificado que a unificação das máquinas de estado reduziam um fator de imprevisibilidade do sistema, o que agregou maior robustez à solução implementada. O algoritmo representado por essa máquina de estados é executado aproximadamente a cada 333 milisegundos. Enquanto está em um determinado estado, o sistema realiza ações de manutenção do canal de comunicação e repassa dados para a classe de tratamento de protocolo, que verificará a necessidade de alteração de estados. Este algoritmo prioriza a manutenção do canal livre a maior parte do tempo, portanto em cada estado há verificações de situação da conexão, levando para o estado de reconexão ou de início sempre que problemas forem identificados.

É necessário tratar situações de atraso das conexões porque a abertura e o fechamento do canal de comunicação não são determinísticos. Portanto, há situações em que o tempo para realizar uma conexão se torna muito longo, podendo chegar a falhas de conexão, em situações extremas. Alguns *times outs* foram definidos no sistema para impedir que essas situações extremas viessem a causar o travamento de todo o sistema de aquisição de dados. Os dois

principais *times outs* são o de ociosidade da conexão, de 120s quando no estado “conectado”, e o de recepção de resposta do modem, de 20s quando no estado “verifica resposta modem”.

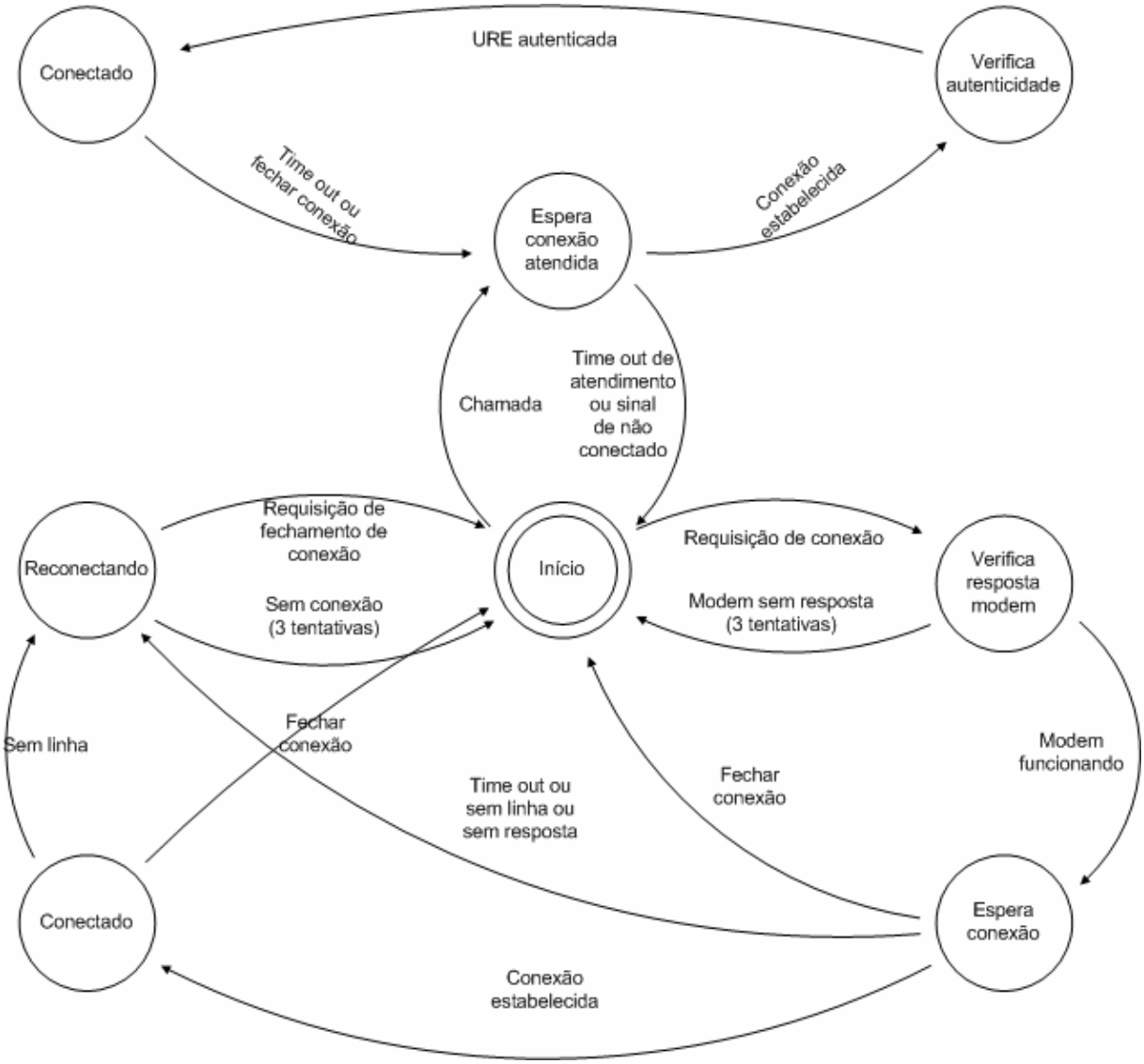

Figura 5.4 Máquina de Estados de Conexão a classe CModComm

Apesar de serem assíncronas, as conexões são tratadas pela máquina de estados através de interrupções de tempo definidas pela função de *timer* da plataforma .NET [34]. Para cada tipo de operação, há um timer diferente, sendo que nenhum é superior a 8s.

Nesse subsistema também está implementado o tratamento do protocolo de transmissão de dados, criado de forma a aumentar a garantia de segurança sobre as informações transferidas para o concentrador. O tratamento do protocolo é feito pela classe CModProtocol. Há métodos definidos nessa classe para lidar com as necessidades específicas do protocolo de comunicação utilizado sobre o canal, onde são implementados os métodos que permitem verificar a validade das mensagens trocadas, além de montar as mensagens a serem passadas para as remotas. Os principais métodos dessa classe fornecem para os demais subsistemas os dados recuperados a partir das remotas. Internamente, essa classe também fornece as funcionalidades de reconhecimento dos *frames* do protocolo e de checagem de erro, retornando para a classe CModComm determinações para que o sistema mude de estado.

5.2.3 Subsistema de disponibiliziação de dados

Nesse ponto são mais importantes as especificações OPC. É nesse subsistema que todos os demais estão inseridos, modularizados de acordo com o tipo de funcionalidade a ser provida aos clientes OPC. A modularização do sistema permite que todos os componentes do subsistema de aquisição de dados possam funcionar independentemente da interface escolhida. A vantagem dessa abordagem é o uso do mesmo sistema de aquisição para os dois servidores criados. Dentro das classes criadas para os servidores OPC são feitas chamadas às classes dos demais subsistemas de acordo com a necessidade de operação a ser realizada.

Para garantir o máximo de disponibilidade do sistema foi seguida a especificação 3.00 do OPC Data Access e 1.00 do OPC XML DA. As peculiaridades relacionadas a cada servidor são tratadas nos tópicos que seguem.

5.3 Decisões de projeto para o Servidor OPC DA

De acordo com a solução de aquisição de dados proposta, algumas das características dos servidores OPC DA tiveram que ser escolhidas para a implementação da solução. Os passos definidos no capítulo 4 foram seguidos e delinearam as escolhas realizadas.

Uma série de requisitos foi identificada para prover o acesso via telefonia celular. O primeiro requisito foi a quantidade de pontos a serem supervisionados. Cada remota é capaz de mapear até seis condições de alarme, quatro leituras analógicas e duas digitais. Para permitir que toda a informação necessária seja transferida em uma única ligação telefônica, as mensagens possuem tamanho de 376 bytes.

Verificou-se que a taxa de transmissão e a largura de banda do sistema de comunicação usando o canal de voz eram muito superiores que as necessidades para troca de informações entre o concentrador e as microremotas. Mas o tempo de conexão entre o celular do concentrador e o celular de qualquer uma das remotas é muito maior do que o geralmente encontrado em sistemas industriais. Não é possível estabelecer uma conexão com menos de 30s. Isso pode causar impacto no funcionamento do sistema e será melhor discutido no tópico da restrição de tempo. Não há *buffer* de leitura nos equipamentos de campo, a não ser para situações de alarme. Nenhum tipo de processamento necessita ser feito para tratamento de informações trazidas pelo sistema de aquisição, de forma que os dados, uma vez armazenados no banco de dados podem ser imediatamente disponibilizados para os clientes.

Existe a necessidade de manter livre o canal de comunicação o maior tempo possível para reduzir a característica de custo associado à comunicação e reduzir a probabilidade de choques de tentativa de uso do canal pelas microremotas. Para satisfazer tal necessidade, o principal algoritmo implementado nesse sistema é o de gerenciamento de conexões

implementado na classe CModComm. Apesar de não representar um sequenciamento de operações entre os módulos do sistema, representa o principal consumidor de tempo de processamento do sistema.

O sistema foi modularizado de forma a separar as funcionalidades específicas que permitem a adequada comunicação com os equipamentos de campo, o armazenamento dos dados recolhidos e a disponibilização das informações.

Dentre os modelos de execução disponíveis para implementação de um servidor OPC, escolhido foi o *Service* NT. O principal motivo para essa escolha é o fato de as remotas poderem retornar para o concentrador informações independentes de requisição devido a eventos pré-configurados. Nesse caso, optou-se por manter na base de dados as informações relacionadas aos eventos, independente do fato de existirem ou não clientes buscando essas informações. Privilegiar a manutenção dessas informações agrega ao sistema a possibilidade de acompanhar os acontecimentos sobre a planta, mesmo que em um momento posterior à ocorrência dos eventos.

Nenhuma nova interface precisou ser criada para prover as funcionalidades esperadas desse sistema. Para possibilitar o uso de clientes OPC de mercado, optou-se por manter dentro das interfaces já existentes nas especificações OPC todas as operações que podem ser feitas sobre a remota.

5.3.1 O problema da restrição de tempo

Um problema que imediatamente se mostrou no desenvolvimento do servidor é o tempo de busca de uma informação no equipamento, muito maior do que o normalmente utilizado por um servidor OPC DA. Além disso, esse tempo não é determinístico, de forma

que o atraso da conexão com certo equipamento em um momento não garante que em um momento seguinte a conexão tenha o mesmo atraso. Isso gerou uma limitação física para a quantidade de pontos que a solução pode supervisionar, determinada pelo atraso de gerenciamento das conexões e da influência do limite de liberação do canal.

O atraso para realizar uma conexão é significativo e, de acordo com o tipo de processo sob supervisão, proibitivo. A média é de 20s, mas não preciso. Em algumas situações, é possível inclusive que a conexão não seja feita, de forma que o algoritmo de discagem prevê até três tentativas simultâneas de conexão antes de considerar que a tentativa de contato com a remota falhou.

Também não há garantias de disponibilização de comunicação em todo momento que se deseja. Dessa forma, podem ocorrer situações em que um evento que deveria ser transferido da remota para o concentrador tenha que esperar por um tempo muito maior que o simples tempo de abertura de conexão. O canal pode estar ocupado e, em condições de conflito de uso do canal entre muitas remotas, a prioridade de aquisição do canal não seria controlada, possibilitando que uma determinada remota jamais conseguisse enviar suas informações (*starvation*). Isso gerou a necessidade de adotar soluções que evitem que o canal fique ocupado por mais tempo que o necessário e mitigam a possibilidade de conflito entre as remotas. Portanto, os *timeouts* de conexão utilizados na máquina de estado de conexões são da mesma ordem da média de abertura de conexão, 60s. Além disso, foi adicionada nas remotas a funcionalidade de desconexão imediata após transmissão de eventos quando a conexão for iniciada por elas e não pelo concentrador. Para reduzir a possibilidade de conflito, foram inseridos atrasos diferentes de rediscagem em cada remota quando identificado canal ocupado. O tempo programado foi sempre inferior a 120s e a diferença de valores das remotas foi de 2s.

Essas escolhas favoreceram a disponibilidade do canal, limitando a quantidade de remotas que podem ser supervisionadas. A limitação não foi testada exaustivamente, mas algumas estimativas foram verificadas nos testes do sistema.

5.4 Decisões de projeto para o Servidor OPC XML DA

De acordo com a solução de aquisição de dados proposta nesse trabalho, algumas das características dos servidores OPC DA tiveram que ser escolhidas para a implementação da solução. Como se trata do mesmo sistema de aquisição, valem para esse servidor exatamente as mesmas escolhas realizadas para o servidor OPC DA. As únicas ressalvas são quanto aos pontos específicos na implementação de um servidor OPC XML DA.

A primeira alteração substancial foi no algoritmo de *polling*. Este algoritmo determina a freqüência com que um cliente pode verificar se sua requisição de informações do servidor já foi processada. No padrão OPC, o sistema cliente é quem define com que freqüência fará uso das interfaces de subscrição e conferência de resultado. Mas fica a critério do servidor definir o período mínimo de *polling* para impedir trocas de mensagens desnecessárias entre o cliente e o servidor. O sistema manteve sempre o valor de 40s para responder ao cliente, de forma que esse valor foi definido como período mínimo de atraso antes de uma nova requisição de resposta ao servidor.

5.4.1 Segurança para disponibilização de informação via *Web Services*

Uma importante consideração para a segurança dos servidores implementados é o protocolo de acesso quando se utiliza o OPC XML DA. Pelo fato de poder utilizar-se o *firewall* para impedir acesso às portas do servidor, este servidor agrega maior valor associado

à segurança. No entanto, os envelopes SOAP trafegam sobre o protocolo HTTP, não existindo garantia de origem válida das requisições direcionadas a um servidor OPC XML DA.

Para impedir tais situações, é mantida uma tabela de *hosts* que podem ser aceitos para acesso ao *Web Service*. Caso o *host* não seja identificado, será necessário que ele forneça a informação de número de celular que está sendo utilizado para buscar a informação. Se o número não for condizente com o listado em uma tabela do banco de dados, não será inicializada a comunicação, impedindo que o sistema seja utilizado indevidamente.

Esta solução é portável e independente de plataforma, servindo de paliativo à falta de padronização de segurança das especificações OPC. Com esta solução, o sistema já se encontra preparado para ser disponibilizado na Internet.

Antes de entrar na descrição do ambiente de desenvolvimento dos sistemas, apresenta-se um resumo das decisões de projeto na tabela 5.1.

Tabela 5.1 Resumo de decisões de projeto

Questão de projeto	Decisão	Onde se aplica
Tecnologia	CDMA (canal de voz)	Subsistema de comunicação
Tamanho das mensagens	376 bytes	
Modelo de execução	Service NT e Web Services	Subsistema de disponibilização de dados
Modularização	*Session Facade*	Subsistema de persistência
Leituras	6 alarmes; 4 analógicas; 2 digitais	Remotas
Armazenamento local	Somente para alarmes	

5.5 O ambiente de desenvolvimento

Para criar os sistemas foi necessário o uso de ferramentas de desenvolvimento de alto nível. No caso do desenvolvimento das classes modularizadas nos subsistemas de comunicação e tratamento de protocolo, assim como no acesso ao banco de dados foi utilizado o MS .NET [34]. Essa ferramenta oferece possibilidade de programação na linguagem C++, essencial para o desenvolvimento de cada módulo utilizado nos servidores, além de se integrar com a ferramenta automatizada de geração de interfaces OPC.

O banco de dados utilizado para o armazenamento e a definição do *Namespace* foi o MySQL 5.0 [35].

5.5.1 Toolkit de criação de interfaces

Sistemas que sigam o padrão OPC podem ser criados tanto diretamente por programadores experientes nas tecnologias adotadas por esses padrões quanto através de ferramentas que automatizam parte da codificação. Sem que se perca a característica de generalidade dos passos definidos para o desenvolvimento de servidores OPC, um *toolkit* foi utilizado para gerar as interfaces OPC necessárias.

Vários *toolkits* foram utilizados, dentre os quais o SLIKDA, da Northern Dynamic. Assim como os demais, esse *toolkit* deve ser instalado sobre o sistema operacional Microsoft Windows, onde já se encontre instalado anteriormente o MS Visual Studio ou o MS .NET. Após sua instalação, é adicionada a opção de criação de servidores OPC entre as soluções possíveis para implementação. Caso essa opção seja escolhida, o *toolkit* apresenta um *wizard* para a definição de algumas características relevantes na implementação de um servidor OPC. Essas telas requerem que algumas decisões sejam tomadas sobre o tipo de servidor OPC que

se deseja implementar. Todo o código de interfaces é gerado automaticamente, com seus métodos vazios. Dessa forma, pode-se partir para o passo de “escolha de componentes existentes” com o foco na remoção daqueles que não se mostrarem necessários.

Apesar de fornecidos por empresas que se esmeram na confecção de seus sistemas, os *toolkits* não oferecem garantia de conformidade com o padrão OPC. Dessa forma, é necessário que se usem ferramentas de teste de conformidade independente do uso ou não de *toolkits* na geração de um servidor OPC.

6 Testes dos servidores

Para garantir que os servidores realizem as funções propostas no padrão OPC, é necessário verificar sua correta adequação às especificações. É necessário realizar testes que validem as interfaces de comunicação através de ferramentas específicas, desenvolvidas pelo OPC Foundation. Também é necessário garantir a qualidade do desenvolvimento realizado com o fim de recuperar dados e enviar comandos para os equipamentos de campo. Nesse caso, não é suficiente o uso de ferramentas automatizadas, mas sim a o uso de sistemas supervisórios comerciais junto à solução, de forma a averiguar se os resultados alcançados podem ser utilizados em ambiente industrial.

6.1 Ambiente de testes

Os testes foram realizados utilizando-se dois computadores ligados em rede. Os servidores OPC, o banco de dados, o celular de comunicação com as remotas e as ferramentas de verificação de conformidade foram instalados em um desses equipamentos. No outro computador foi instalado o sistema de supervisão Elipse SCADA, utilizado para supervisionar os resultados provenientes do servidor OPC DA. Nesse computador também foi instalado o programa consumidor de Web Services, desenvolvido para os testes uma vez que não foram encontradas aplicações comerciais de uso na indústria que pudessem utilizar essa tecnologia na época dos testes.

6.2 Testes de conformidade

Os testes de conformidade (*compliance test*) fornecem a segurança sobre a aderência do sistema com relação ao padrão OPC. Isso garante que o sistema se comunicará adequadamente com sistemas supervisórios que ofereçam as funcionalidades de cliente OPC. Como essa camada de comunicação normalmente não possui preocupação de fornecer o correto funcionamento da aquisição de informações, tais testes não demonstrarão se os algoritmos de busca de dados ou de execução de *callbacks* funcionam adequadamente. O objetivo é averiguar se as interfaces criadas funcionam para disponibilizar dados do concentrador em um sistema SCADA.

Para realização destes testes, é necessária uma série de ferramentas fornecidas pela OPC Foundation. Essas ferramentas trabalham sobre o servidor criado, permitindo averiguar quais interfaces não responderam de acordo com o padrão definido de respostas.

Durante os testes, verificou-se que duas funcionalidades de duas interfaces geradas não funcionavam adequadamente. As interfaces com problemas foram IOPCBrowse, do objeto Servidor, e IOPCAsyncIO2, do objeto Grupo. A falha identificada não era de isolamento binário, mas sim de lógica na definição do resultado, o que causava, em algumas circunstâncias, um comportamento inesperado e o travamento do servidor. Para sanar os problemas, foi necessário alterar a implementação das interfaces através da inclusão de um atraso na execução do servidor. Com isso, o caminho normal definido pelo padrão passou a ser seguido.

6.3 Testes com clientes

O subsistema de persistência foi o primeiro a ser implementado, permitindo que os testes com os servidores fossem feitos inicialmente a partir do banco de dados. A massa carregada nesse banco de dados foi gerada durante os testes de unidade do subsistema de aquisição de dados, utilizando-se as remotas de campo. Estes testes não explicitavam o problema de restrições de tempo de uso de canal não dedicado, mas serviram para refinar a lógica de tratamento das interfaces que fazem uso de *cache*. Através dessas simulações, detectou-se um comportamento inadequado da disponibilização de dados pelos servidores. Apesar de estar em conformidade com o padrão OPC, o tipo de informação que deveria ser fornecido estava falho. O erro sempre ocorria com itens que não sofriam alterações dentro do período de leituras testadas. Apesar de o padrão ser seguido e o *timestamp* associado ao dado estar correto, a informação de qualidade era incorretamente apresentado como “ruim”. Este problema foi resolvido por programação e era causado por um erro na leitura dos dados da entidade Valor, no banco de dados.

Os clientes foram instalados em um PC na rede diferente do utilizado com os servidores para verificar a funcionalidade de identificação do servidor OPC DA e de descobrimento de serviços do servidor OPC XML DA. Não foram encontrados problemas relacionados a essas funcionalidades.

Nos testes realizados sem simulação, houve problemas associados à restrição de tempo de conexão em relação aos dois servidores. Esses problemas foram resolvidos conforme indicado no tópico 5.3.1.

6.4 Comparativo entre os servidores

Os cenários de testes propostos para os servidores possuíam duas vertentes principais. A primeira era sobre os resultados ideais de funcionamento junto ao canal de comunicação não dedicado. O segundo era de *stress* sobre os limites de disponibilização de dados pelos servidores.

Antes de iniciar o teste de *stress*, foi necessário definir qual o tempo limite de funcionamento do algoritmo de conexões, ou seja, qual a diferença de tempo mínima necessária entre o início de uma conexão para uma remota e o início de uma conexão para outra remota. Indiretamente, essa informação indica a quantidade de remotas que podem ser supervisionadas por um concentrador sem que haja perda de informação relevante para o controle e monitoramento do sistema. Foi verificado que era possível realizar conexões a cada 60s.

Para alcançar a situação de *stress*, dois clientes foram configurados para fazerem uso de uma série de funcionalidades em diferentes interfaces em tempos de requisição próximos ao limite identificado para funcionamento do algoritmo de gerenciamento de conexões. O aumento da freqüência de requisições foi observado em duas condições diferentes. Primeiro, com as informações de simulação do banco de dados, onde o comportamento dos dois servidores foi muito semelhante, sem deixar de atender nenhuma requisição. Já com o uso de remotas, não foi possível reduzir o tempo entre duas requisições para remotas para um valor menor do que 120s, sob pena de atrasar todas as demais requisições exponencialmente. Nessa periodicidade de *polling*, os dois servidores se comportaram de forma idêntica.

A tabela 6.1 apresenta os principais resultados alcançados durante os testes do sistema e que viabilizaram a instalação em campo de todo o sistema construído.

Tabela 6.1 Resumo dos resultados dos testes

Característica medida	Valor
Timeouts de conexão (ambiente de comunicação)	60s
Atraso de conexão	30s
Atraso de execução de comandos	60s
Holdtime	Mínimo de 60s
Rediscagem da remota	120s (- 2s em cada remota adicionada)

7 Conclusões

Segundo Mattila, o uso de servidores OPC DA ou OPC XML DA deve ser avaliada de acordo com os seguintes critérios [36]:

1- Conexões de alto nível, entre aplicações em sistemas operacionais diferentes.

2- Conexões de longa distância, onde puder se utilizar a Internet.

3- Sistemas locais, desde que o desempenho não seja um fator crítico.

Os testes sobre os sistemas revelaram que para as situações onde é aplicável o uso de telefonia celular como meio de comunicação para sistemas supervisórios, quaisquer dos dois tipos de servidores podem oferecer as funcionalidades necessárias. Os tempos necessários para a realização de conexão são muito maiores que os que restringiriam o uso de Web Services.

Sistemas que fazem aquisição de informações a partir de telefonia celular já possuem uma restrição de tempo baseado na incerteza de conexão das redes de telefonia e no tempo necessário para conexão. Com isso, sistemas que demandem tempos de resposta a requisições inferiores a 5 minutos e garantia de comunicação em níveis maiores do que 98 % não podem usar a telefonia celular como meio de acesso a dados de sua planta. No entanto, as restrições de utilização de servidores OPC XML DA são muito inferiores a este valor, de forma que sua utilização junto à tecnologia de telefonia celular não apresenta perdas de desempenho quando comparados aos servidores OPC DA.

Ainda há uma grande vantagem no desenvolvimento de servidores OPC DA devido à maturidade da tecnologia utilizada e da grande quantidade de *toolkits* disponíveis para o

desenvolvimento. Mas há uma expectativa de que o uso de servidores baseados em Web Services venha a tomar o lugar dos baseados em DCOM em todos os pontos de aplicação, uma vez que a própria OPC Foundation escolheu a tecnologia Web Service como base de sua *Unified Architecture* [22]. As vantagens associadas à interoperabilidade dos Web Services, junto à criação de novas ferramentas de apoio ao desenvolvimento, uma melhor definição das questões de segurança farão desse padrão a melhor solução para uso na indústria. Uma dependência não tecnológica, mas sim de mercado, é o desenvolvimento de clientes para esses Web Services. À medida que mais empresas especializadas em automação industrial projetarem sistemas SCADA com clientes de Web Services, maior será a penetração dessa tecnologia entre os servidores de dados e informações.

O uso de quaisquer dos dois tipos de servidores oferece a vantagem de redução de custos na alocação de equipamentos de comunicação. Como o sistema possui a capacidade de lidar com a comunicação de um número crescente de equipamentos sem perda de desempenho nem da qualidade da aquisição, é possível reduzir o custo de montagem de uma rede industrial, já que todo o projeto de cabeamento para transmissão de dados da planta pode ser substituído por pontos de comunicação baseados em sistema de telefonia celular.

Um ponto de partida para pesquisas posteriores seria aperfeiçoar o algoritmo de gerenciamento de conexões através de técnicas de otimização de sistemas. Como essa é uma área em franca expansão nas ciências da computação, os resultados de seus estudos sobre minimização / maximização poderiam ser aplicados sobre o subsistema de comunicação, ampliando a quantidade de equipamentos que poderiam ser supervisionados.

Um outro ponto a ser considerado é a evolução do sistema de comunicação utilizando tecnologias 2.5G ou 3G da telefonia celular, como o 1xRTT. Essa tecnologia permite que

conexões com os equipamentos de campo se tornem permanentes, não necessitando de algoritmos de gerenciamento de conexões. Nessa situação, a limitação na quantidade de pontos supervisionados não dependeria do sistema de algoritmos definidos sobre o meio de comunicação, tornando o sistema mais compatível às estruturas de controle das redes de computadores.

8 Bibliografia

[1] Iwanitz, F., Lange, J., “OPC Fundamentals, Implementation and Application”, Second reviewed edition, Hüthig GmbH & KG Heidelberg, 2002.

[2] Riedl, M., Thron, M.; Hadlich, T., “DriveServer - significantly reduce in engineering expense”, IECON'01: The 27th Annual Conference of the IEEE Industrial Electronics Society, 2001.

[3] Nascimento Filho, O. A., Munaro, C. J., “Considerations about implementation and applications of OPC Data Access and OPC XML Data Access standards in industrial automation”, VI INDUSCON, out. 2004.

[4] Hao, X., Hou, S., “OPC DX and Industrial Ethernet glues fildbus together”, *8th International Conference on Control, Automation, Robotics and Vision Kunming*, China, dec. 2004.

[5] Wollschlaeger, M., Diedrich, C., Simon, R., “Web Integration of Factory Communication Systems using a XML Framework”, *Proceedings of the 2002 IEEE International Symposium on*, Volume: 1 , 8-11 Julho – 2002.

[6] Manetti, J., "How technology is transforming manufacturing", *Production and Inventory Management Journal*, Alexandria, V. 42, First Quater 2001.

[7] Boyer, S. A., “SCADA: Supervisory Control and Data Acquisition”, ISA Books, 2004.

[8] “OLE for Process Control – Common Definitions and Interfaces V 1.0”, disponível em http://www.opcfoundation.org. Acesso em 15 jun. 2003.

[9] Souza, A. J., Oliveira, L. C., “Automação Industrial”, UFRN, mai. 2003, disponível em http://www.dca.ufrn.br/~ajdsouza/producao.html. Acesso em 10 mar. 2004.

[10] Tanenbaum, A. S., “Computer Networks”, 3ª Ed., Prentice Hall, 17 mar. 2003.

[11] Iwanitz, F., Lange, J., “OPC Fundamentals, Implementation, and Application”, 2ª Ed., Hüthig Verlag Heidelberg, 2002.

[12] Gamma, E., Helm, R., Johnson, R., Vlissides, J., “Design Patterns Elements of Reusable Object-Oriented Software”, 1ª Ed., Addison-Wesley Pub Co, 15 jan. 1995.

[13] Abdmouleh, A.; Spadoni, M.; Vernadat, François; “Distributed client/server architecture for CIMOSA-based enterprise components”, *Computers in Industry*, 2004, Issue 55, pg 239-253.

[14] Plásil, F., Stal, M., “An architecture view of distributed objects and components in CORBA, Java RMI and COM/DCOM”, *Software Concepts & Tools*, n. 19, p. 14 – 28, 1998.

[15] Levy, M., “COM Internet Services”, Microsoft Corporation MSDN, 23 abr. 1999, disponível em http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndcom/html/cis.asp. Acesso em 20 nov. 2004.

[16] Nance, K.L., Hay, B., “Automatic transformations between geoscience standards using XML”, *Computers & Geosciences*, Volume 31, Issue 9, p. 1165-1174, nov. 2004.

[17] Wollschlaeger, M., Diedrich, C., Simon, René, “Web Integration of Factory Communication System using an XML Framework”, IEEE, 2002.

[18] Kapsalis, V., Koubias, S., Papadopoulos, G., “OPC-SMS: a wireless gateway to OPC-based data sources”, *Computer Standards & Interfaces*, n. 24, p. 437 - 451, 2002.

[19] Anzböck, R., Dustdar, S., “Modeling and implementing medical Web services”, *Data & Knowledge Engineering*, Volume 55, Issue 2, p. 203-236, nov. 2005.

[20] Tao, Y., Hong, T., Sun, S., “A XML Implementation process model for enterprise applications”, *Computers in Industry*, n. 55, p. 181-196, jun. 2004.

[21] “Web Services Architecture”, W3C Work Group Note, 11 fev. 2004, disponível em http://www.w3c.org. Acesso em 14 out. 2004.

[22] “OPC Unified Architecture V 1.00”, disponível em http://www.opcfoundation.org. Acesso em 05 out. 2005.

[23] Cuadros, M. A. S. L., “Implementação de um sistema de supervisão sem fio via telefonia celular”, dissertação de mestrado apresentada no PPGEE da UFES em junho de 2004.

[24] Xu Hong, Wang Jianhua, “An extendable data engine based on OPC specification”, *Computer Standards & Interfaces*, n. 26, p. 515 - 525, 2004.

[25] Hong, X., Jianhua, W., “Using Standard components in automation industry: a study on OPC Specification”, *Computer Standards & Interfaces*, 2005.

[26] “OLE for Process Control – Data Access Standard V 3.00”, disponível em http://www.opcfoundation.org. Acesso em 15 jun. 2003.

[27] Fonseca, M. O., “Comunicação OPC – Uma abordagem prática”, *VI Seminário de Automação de Processos, Associação Brasileira de Metalurgia e Materiais*, 2002.

[28] Leavitt, N., “Are Web Services Finally Ready to Deliver ?”, IEEE Computer Magazine, Vol. 37, n 11, nov. 2004, disponível em www.computer.org. Acesso em 10 dez. 2004.

[29] Moitra, D., Ganesh, J., “Web services and flexible business processes: towards the adaptive enterprise”, *Information & Management*, Volume 42, Issue 7, p. 921-933, out. 2005.

[30] Muehlen, M., Nickerson , J. V., Swenson, K. D., “Developing web services choreography standards—the case of REST vs. SOAP”, *Decision Support Systems*, Volume 40, Issue 1, p. 9-29, jul. 2005.

[31] “OLE for Process Control – OPC XML – DA Pre-release Specification V 1.0”, disponível em http://www.opcfoundation.org. Acesso em 02 set. 2003.

[32] Colgrave, J., Januszewski, K., “Using WSDL in a UDDI Registry, V. 2.0”, disponível em: http://www.uddi.org. Acesso em 05 set. 2003.

[33] Munaro, C. J., Bastos, L. P., Vivaqua, R. D, Nascimento Filho, O. A., Cuadros, M. A. L., Tozi, W. J., Azeredo, E. F., “Sistema de monitoramento de redes de distribuição de média tensão via telefonia celular”, VII INDUSCON, abr. 2006.

[34] MICROSOFT Visual Studio .NET : development software. [S.1.] Microsoft Corporation, 2002.

[35] MANUAL MySQL Reference Manual for version 5.0.1-alpha. Mar. 2004.

[36] Mattila, M., “OPC XML”, disponível em http://www.opcfoundation.org. Acesso em 02 set. 2003.